A Review of Automatic Drum Transcription

Chih-Wei Wu[1], Christian Dittmar[2], Carl Southall[3], Richard Vogl[4,5], Gerhard Widmer[4], Jason Hockman[3], Meinard Müller[2], *Senior Member, IEEE*, and Alexander Lerch[1] *Member, IEEE*

***Abstract*—In Western popular music, drums and percussion are an important means to emphasize and shape the rhythm, often defining the musical style. If computers were able to analyze the drum part in recorded music, it would enable a variety of rhythm-related music processing tasks. Especially the detection and classification of drum sound events by computational methods is considered to be an important and challenging research problem in the broader field of Music Information Retrieval. Over the last two decades, several authors have attempted to tackle this problem under the umbrella term Automatic Drum Transcription (ADT). This paper presents a comprehensive review of ADT research, including a thorough discussion of the task-specific challenges, categorization of existing techniques, and evaluation of several state-of-the-art systems. To provide more insights on the practice of ADT systems, we focus on two families of ADT techniques, namely methods based on Non-negative Matrix Factorization and Recurrent Neural Networks. We explain the methods' technical details and drum-specific variations and evaluate these approaches on publicly available datasets with a consistent experimental setup. Finally, the open issues and under-explored areas in ADT research are identified and discussed, providing future directions in this field.**

***Index Terms*—Music Information Retrieval, Automatic Music Transcription, Automatic Drum Transcription, Machine Learning, Matrix Factorization, Deep learning.**

I. Introduction

In music information retrieval (MIR), the task of Automatic Music Transcription (AMT) is considered to be one of the most challenging research problems [1]. In simple terms, transcription can be understood as the reverse of music making. Instead of having musicians perform with their instruments according to a notated sheet music, AMT aims at deriving such symbolic notation from previously recorded music. If computers were able to fulfill this task with high accuracy, this would enable diverse applications in music education, music production, musicology, and other areas. In the MIR literature, many authors focus on transcribing the pitch, onset time, and duration of note sequences that are either played by melodic instruments such as piano and guitar, or performed by human singing voice [2]. Fewer authors have proposed algorithms for Automatic Drum Transcription (ADT), where the equivalent of discerning musical pitches is the detection and classification of drum sound events. On the one hand, ADT systems focus on the detection and recognition of highly transient and impulsive events, which could be similar to other audio signal processing problems such as audio-surveillance [3] and acoustic event detection [4]. On the other hand, the musically organized drum events and the underlying vocabulary resemble well-studied problems in speech or language processing [5]. The combination of both makes ADT a unique research problem that might be of interest to the general audio signal processing community. In an effort to reflect and facilitate the progress in ADT, FitzGerald and Paulus [6] provided a coherent summary of early approaches. However, due to the lack of comparability of results, a detailed quantitative comparison among the reviewed systems is hard to achieve.

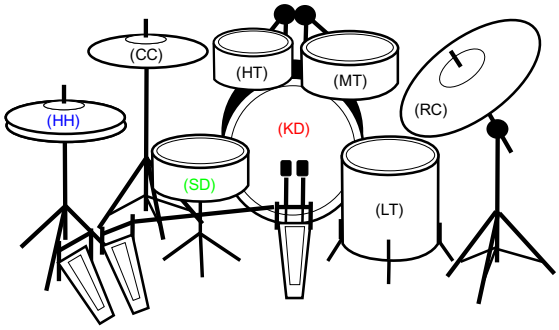


Fig. 1: The most important parts of a drum kit as used in Western popular music. The color-coding and abbreviations are used throughout the article.

In this overview article, we want to provide a comprehensive review and categorization of existing approaches to ADT and related tasks. ADT generally covers a wide spectrum of percussive instruments that can be found in both Western and non-Western music (e.g., drum kits, Tabla [7], or Beijing opera percussion ensemble [8]). In this paper, we focus on the transcription of drum kits in the context of Western music. To this end, we discuss a variety of techniques and applications, pointing out the differences, commonalities, benefits and limitations of the respective approaches and highlight open issues and challenges. Beyond the literature survey, we present a systematic comparison of state-of-the-art approaches on two publicly available corpora of drum recordings. After discussing the experimental findings, we highlight and indicate

[1]Chih-Wei Wu and Alexander Lerch are with the Center for Music Technology, Georgia Institute of Technology, Atlanta, GA, USA.

[2]Christian Dittmar and Meinard Müller are with the International Audio Laboratories Erlangen, Germany.

[3]Carl Southall and Jason Hockman are with the Digital Media Technology Lab, Birmingham City University, UK.

[4]Richard Vogl and Gerhard Widmer are with the Department of Computational Perception, Johannes Kepler University Linz, Austria.

[5]Richard Vogl is also affiliated with the Institute of Software Technology & Interactive Systems, Vienna University of Technology, Austria.

Manuscript received xyz, 2017; revised xyz, 2017.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2018.2830113, IEEE/ACM Transactions on Audio, Speech, and Language Processing

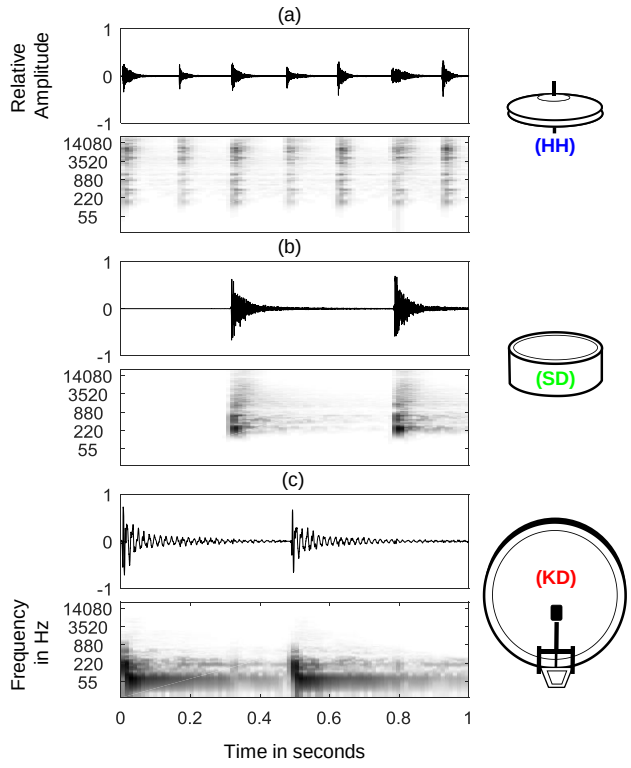


Fig. 2: Illustration of typical drum sound events of **(a)** HH, **(b)** SD, and **(c)** KD. The panels show the time-domain signal in black and the corresponding spectrogram representation, with darker shades of gray encoding higher energy. For the sake of visibility, the spectrograms are given with a logarithmically spaced frequency axis and logarithmically compressed magnitude values.

the potential directions for future research.

A. Introduction to Drum Kits

The drum kit plays an important role in many Western music genres such as rock, pop, jazz, and dance music. The traditional role of drums in these music genres is to emphasize the rhythmical structure as well as to support the segmentation of the piece into different parts. Generally speaking, the sound characteristics of drum instruments (unpitched, percussive, and transient) differ in many aspects from pitched instruments which constitute the melodic and harmonic foundations of music. It should be noted that there are exceptions to this tendency. For example, there are pitched percussion instruments such as the vibraphone. Moreover, certain instruments such as piano and guitar also comprise transient sound components.

Fig. 1 introduces the parts of a basic drum kit with their abbreviations and color coding as used throughout this article. The different drum instruments can be roughly classified into the two classes membranophones and idiophones. The Kick Drum (KD), Snare Drum (SD), and High /Mid /Low Toms (HT, MT, LT) are typical examples of membranophones, which have vibrating membranes spanned over cylindrical bodies. In contrast, the Hi-Hat (HH), Crash Cymbal (CC), and Ride Cymbal (RC) are typical examples of idiophones, whose metallic body vibrates as a whole.

Fig. 2 illustrates the typical sound events produced by the three drum instruments KD, SD, and HH. The KD is played via a foot pedal, generating sounds with low indefinite pitch. In Figure 2c, this can clearly be seen by the concentration of energy in the lower frequency region. In the frequency band around 55 Hz, the initial transient is followed by a slow decay spread over several hundred milliseconds. Depending on the music style and recording conditions, the presence of such tonal components within drum sounds is not an uncommon phenomenon. The SD often acts as the rhythmic counterpart of the KD. It has snare wires stretched across the lower drum head. When striking the upper head with a drum stick, the lower head's vibrations excite the snares, generating a bright sound. In Fig. 2b, it can be seen that the SD tends to decay faster than the KD and usually covers the middle frequency range. The sound of a HH can be influenced by opening or closing it with a foot pedal. When closed, it produces a quickly decaying clicking sound. When open, it produces a standard cymbal sound exhibiting many inharmonic partials. As shown in Fig. 2a, the HH's sound components are usually concentrated in the higher frequency regions.

Acoustic drum instruments can produce a wide variety of drum sounds depending on many influences (e.g., the striking position and velocity). Professional drummers may use this freedom for artistic expression. In contrast, drum sampler softwares usually feature a limited number of prerecorded drum sounds per instrument. To emulate the variability of acoustic drums, it is common to switch between different samples of the same drum, either based on velocity levels or random selection.

B. Challenges and Particularities

As already indicated, drum instruments are quite different from pitched instruments. Hit with sticks or mallets, drums usually start with transient-like sound components exhibiting broadband, noise-like spectra. Tonal components may also occur for certain drum types and playing techniques. Contrasting pitched instruments, the tonal elements are usually not structured like partials in a harmonic series. Instead, their frequency relationship can range from inharmonic to chaotic. Due to these characteristics, certain algorithms tailored to pitched instruments (e.g., fundamental frequency estimation) are not applicable for ADT. As shown in Fig. 2, the magnitude spectrograms of drums do not show a clear harmonic structure as occurring for many pitched instruments. Moreover, in comparison to singing voice, for example, there is less variability within a single drum instrument. For that reason, template-based approaches are often used in ADT.

In music recordings, drum sounds are usually superimposed on top of each other, i.e., different drum instruments are played simultaneously. To illustrate the implications, we show the spectrogram of a funk drum pattern played with KD, SD, and HH in Fig. 3a. At first sight, one can observe a fairly complex mixture of different drum sound events. As emphasized by the color-coding in Fig. 3b, there is a strong overlap between KD, SD, and HH in both time and frequency. This can lead to

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2018.2830113, IEEE/ACM Transactions on Audio, Speech, and Language Processing

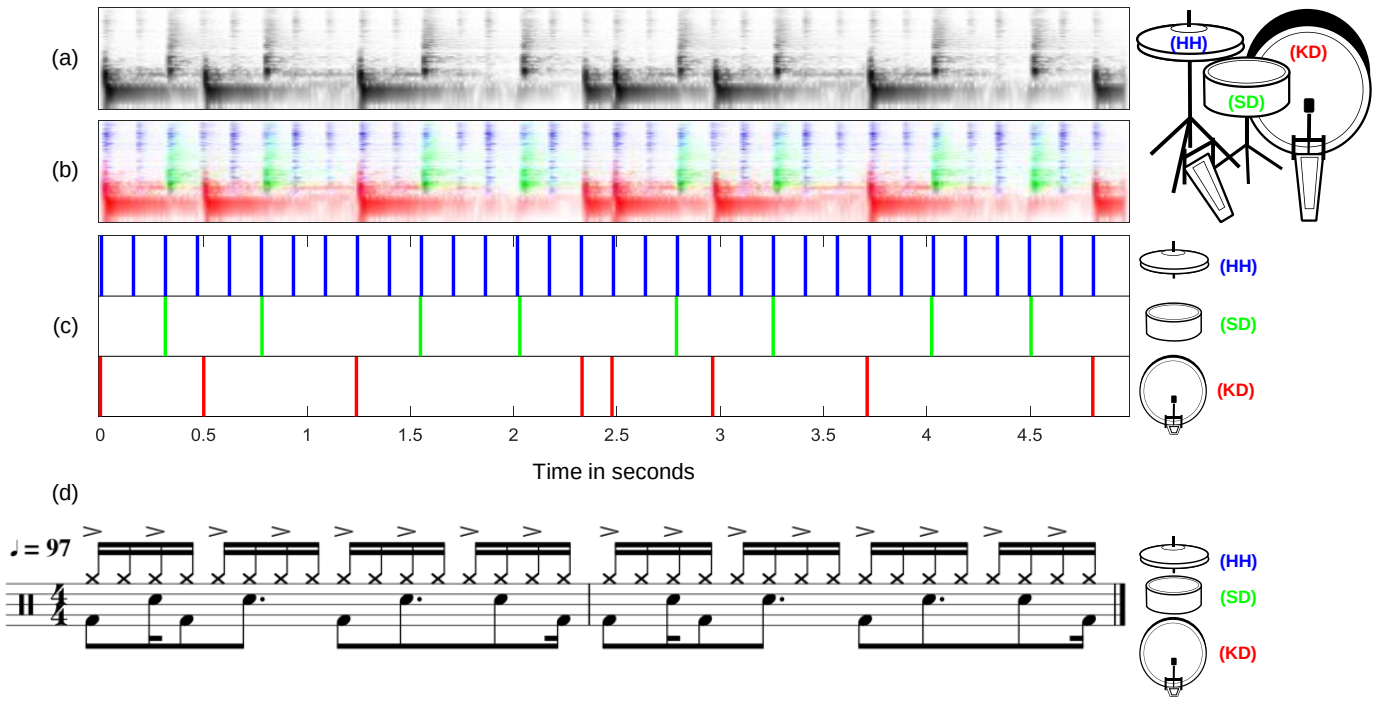


Fig. 3: Illustration of drum transcription in drum-only recordings. **(a)** Example drum recording in a spectrogram representation with logarithmic frequency spacing and logarithmic magnitude compression. Darker shades of gray encode higher energy. **(b)** The same spectrogram representation with color-coded contributions of individual drum instruments. **(c)** Target onsets displayed as discrete activation impulses. **(d)** Drum notation of the example signal. The note symbols have been roughly aligned to the time axis of the figures above.

ambiguous situations where it is hard to automatically classify drum sound events or combinations thereof. This challenge is further intensified when drums are mixed with other instruments (see Sect. III-A). There are, however, other properties of the drum signals that can be exploited. For instance, drums are usually played in a locally periodic and repetitive fashion in order to shape the rhythm. Thus, many onset events of the same drum instrument can be observed throughout a recording, often repeating a rhythmic pattern. This can be utilized by methods that inherently capture these quasi-periodic characteristics.

Going beyond the recognition of drum sound events, the recognition of special playing techniques, which create a variety of spectral and temporal variations of the drum sounds, poses an additional challenge (see Sect. III-B).

As will be shown in the later sections, these challenges not only highlight the unique aspects of drum sound analysis, but also play an important role in advancing the development of the state-of-the-art ADT systems.

C. Task Definition

The various tasks that are typically considered as ADT are introduced in the top-most rows of Table I. In its most basic form, Drum Sound Classification (DSC) aims at automatic instrument classification of recordings of isolated drum sounds. A related task is Drum Sound Similarity Search (DSSS), where the aim is to quantify how similar isolated drum sounds are to each other. Drum Technique Classification (DTC) goes beyond that, paying extra attention to recognizing special playing techniques.

As opposed to isolated drum events, typical drum recordings are sequences of drum sounds. One special case of transcribing a sequence of non-overlapping drum events is voice percussion transcription (VPT), which involves the detection of the percussion sounds produced during beat boxing (a vocal technique to mimic drum rhythms). While some skilled beatboxing artists are capable of producing simultaneous sounds mimicking two or more drum instruments, voice percussion is usually monophonic due to limitations of the human vocal tract. Therefore, VPT is often considered as a simplified ADT task. Drum Transcription of Drum-only recordings (DTD) is a well-studied task that is addressed in many publications. As with VPT, the task is to recognize different drum sounds exclusively played on drum instruments. In contrast to VPT, different instruments may occur simultaneously, making it more difficult to unambiguously discern multiple drum instruments. A typical output of DTD is shown in Fig. 3c, where the discrete onsets (i.e., the physical time when a certain drum is hit) are encoded as impulse-like activations.

Drum Transcription in the presence of Percussion (DTP) allows that additional percussion instruments besides the targeted ones may be played. Clearly, this is a more complex scenario which typically leads to more erroneously detected onsets.

Finally, Drum Transcription in the presence of Melodic instruments (DTM) aims at detecting and classifying the

TABLE I: List of acronyms and abbreviations used in this article. We have grouped the diverse terms into main categories for better structuring.

Category	Acronym	Abbreviation for
Drum Transcription Task	DSC	Drum Sound Classification
	DSSS	Drum Sound Similarity Search
	DTC	Drum Technique Classification
	DTD	Drum Transcription of Drum-only Recordings
	DTP	Drum Transcription in the Presence of Additional Percussion
	DTM	Drum Transcription in the Presence of Melodic Instruments
	OD	Onset Detection
	VPT	Voice Percussion Transcription
Feature Representation	AVF	Audio-Visual Features
	BPF	Bandpass Filterbank
	CQT	Constant-Q Transform
	DWT	Discrete Wavelet Transform
	HPSS	Harmonic-Percussive Source Separation
	LLF	Low-Level Audio Features
	LSF	Line Spectral Frequencies
	MLS	Mel-Scale Log Magnitude
	MFCC	Mel-Frequency Cepstral Coefficients
	STFT	Short-Time Fourier Transform
	WAV	Waveform
	ZCR	Zero-Crossing Rate
Method for Activation Function and Feature Transformation	AdaMa	Template Adaptation and Matching
	FDA	Fisher Discriminant Analysis
	ICA	Independent Component Analysis
	ISA	Independent Subspace Analysis
	LDA	Linear Discriminant Analysis
	MDS	Multi Dimensional Scaling
	MPSC	Matching Pursuit Using Sparse Coding Dictionary
	NSP	Noise Subspace Projection
	NMF	Non-Negative Matrix Factorization
	NMFD	Non-Negative Matrix Factor Deconvolution
	NNICA	Non-Negative ICA
	PCA	Principal Component Analysis
	PFNMF	Partially-Fixed NMF
	PLCA	Probabilistic Latent Component Analysis
	PSA	Prior Subspace Analysis
	SANMF	Semi-Adaptive NMF
Classifiers for Frame-Wise Processing	ALC	Alternate Level Clustering
	ABT	AdaBoost
	CRF	Correlation Function
	DNN	Deep Neural Network
	DT	Decision Tree Classifier
	HCA	Hierarchical Cluster Analysis
	KNN	K-Nearest Neighbor Classifier
	SVM	Support Vector Machine
Classifiers Exploiting Temporal Context	BLSTM	Bidirectional LSTM
	BRNN	Bidirectional RNN
	CNN	Convolutional Neural Network
	GRU	Gated Recurrent Unit
	HMM	Hidden Markov Model
	LSTM	Long-Short Term Memory
	RNN	Recurrent Neural Network
	CNN	Convolutional Neural Network
	CRNN	Convolutional Recurrent Neural Network

occurrences of different drum sounds in full-mixture music such as pop, rock, or jazz recordings.

We would like to point out that we use the term transcription in a rather loose way, as is common in the MIR literature. A complete transcription would require to fit the recognized drum onsets into a rhythmical grid (e.g., a direct mapping to symbolic representation as done in [9] or detecting bar position as in [10], [11]) in order to generate a musical score in drum notations. Additionally, other meta data included in sheet music, (e.g., instructions for playing techniques, embellishments, indications for tempo and dynamics changes) may be regarded to be part of full transcripts. This is illustrated in Fig. 3d, where we show the ground-truth drum notation of the example signal. To make the correspondences more obvious, we roughly aligned the musical time axis to the physical time axis of the panels above.

Having a complete symbolic representation as the output of ADT systems is usually beneficial in terms of applicability and accessibility for human musicians. However, this requires the integration of various MIR systems, which adds another layer of complication to the core research problem. As a result, much of the ADT research focuses on extracting drum onset times as the output representation. For the sake of consistency with prior work, this overview paper uses the term *drum transcription* to cover the detection and classification of drum sound events.

D. Application Scenarios

ADT entails some challenging audio processing problems: in general, it is an instance of detecting unexpected, sparsely-distributed events, which can be related to broader applications such as detecting impulsive and transient sounds from audio streams. In the following, we introduce music-related application scenarios which benefit the most from ADT research.

Music Education: Music education software and video games such as RockSmith[1] or Songs2See[2] could potentially benefit from automatic drum transcription. Very few educational applications offer the possibility to practice drums by using electronic drum pads that output MIDI signals. None of the existing applications allow users to practice on acoustic drum kits. In this context, the goal would be to monitor the players while they are practicing and provide automatic performance assessment, ideally in real-time.

Music Production: In professional music production, drum parts are usually recorded using multiple microphones. Post-processing typically includes equalization, reverberation, dynamics processing, or even drum replacement using specialized plug-ins.[3] It is difficult to properly set up drum microphones and engineer the microphone signals to minimize cross-talk (leakage). In [12], an approach for drum leakage suppression was proposed (which later went into the product Drumatom[4]). With the availability of affordable, easy-to-use, and high-quality drum sample software, it becomes more and more common in music productions to use both sampled

[1]http://rocksmith.ubi.com/, last accessed 2017/10/02
[2]http://www.songs2see.com/, last accessed 2017/10/02
[3]http://www.drumagog.com/, last accessed 2017/10/02
[4]http://drumatom.com/, last accessed 2017/10/02

drums and recorded acoustic drums with extracted triggers. Having a reliable ADT method at hand would facilitate both drum leakage suppression as well as drum replacement applications. Additionally, with the growing size of drum loop databases, ADT would enable content-based approaches for retrieving these samples, improving the efficiency of computer-aided music composition or even automatic music accompaniment systems.

Music Remixing: Dittmar and Müller showed that reliable drum transcription is beneficial for decomposing monaural drum recordings into single drum hits for the purpose of remixing in [13]. In this context, a score-informed audio-aligned transcription is used for initialization of an audio-decomposition method. Recently, the music software Regroover,[5] whose main feature is a similar source separation technology, was released. For certain tasks, this software still requires a lot of intervention by the user, which could be alleviated when having a reliable ADT algorithm at hand.

Music Information Retrieval: More generally speaking, ADT is a useful preprocessing step for obtaining higher-level music content descriptions in MIR. First, transcription of the drums is an important prerequisite for determining the rhythm patterns. This information can be valuable for structuring large corpora of popular music [14] as well as electronic music [15]. Moreover, music recommender systems could use the data to better rate the danceability and rhythmic similarity between different songs. Going more into musicological research, there is a high interest in determining microrhythmic properties such as swing, shuffle and groove [16]–[18] inherent in music recordings. Robust ADT in conjunction with precise estimation of onset times (see discussion in Sect. II-F) can be beneficial in that regard as well.

E. Structure of the Paper

The remaining parts of the paper are organized as follows: Sect. II to Sect. III focus on the comprehensive review of prior work. Sect. II discusses and categorizes previous publications on ADT. This includes an extensive literature review and a general introduction of commonly used datasets and evaluation metrics. Next, we discuss current challenges of ADT systems in Sect. III.

Sect. IV to Sect. VIII are dedicated to the evaluation of state-of-the-art systems: Sect. IV introduces the mathematical notations and the systems' commonalities, followed by the detailed description of two specific algorithmic paradigms: Non-Negative Matrix Factorization (NMF) in Sect. V and Recurrent Neural Networks (RNNs) in Sect. VI. In Sect. VII, we explain the datasets and evaluation strategies that we used to compare NMF-based and RNN-based ADT methods in systematic experiments using two publicly available datasets. In Sect. VIII, we present the most important findings from our experiments in condensed form. Finally, in Sect. IX, we conclude with recommendations and a list of the identified important directions for future work.

[5]http://regroover.com/, last accessed 2017/10/02

II. General Trends in Drum Transcription

A. General Design Patterns

In this section, important directions in ADT research are presented. Table I provides a reference for acronyms and abbreviations used throughout this paper. Table II provides an exhaustive listing and categorization of the reviewed publications comprised in our literature review. In earlier works on ADT, FitzGerald and Paulus [6] proposed to categorize the systems into two types, namely the pattern recognition and separation-based approaches. Later on, a more refined grouping of four categories was proposed [19], [20]. These are:

1) *Segment and Classify Approach*,
2) *Separate and Detect Approach*,
3) *Match and Adapt Approach*,
4) *HMM-based Recognition Approach*.

Considering the increasing amount of ADT research published, we found it difficult to draw clear boundaries between separate categories, and the classic categorization might not accurately reflect the advances in ADT in recent years. As an alternative, we propose to distinguish between methods according to their constituent building blocks. Specifically, we identify six generic design patterns that are used in most methods, see Fig. 4 for an overview. Before we briefly introduce each of these patterns in the following paragraph, we first want to emphasize that they can be used like items from a toolbox, interchangeably and in no particular order. Second, the distinction between the patterns is sometimes vague, and the particular technical tool implementing each of these patterns may vary depending on the method. And third, the patterns are often not specific to drums, but very generic, e.g., inspired from research in speech, language, and general multimedia processing. For a general introduction to these processing steps, please refer to [21], [22].

Feature Representation (FR): Apart from the time-domain waveform, discretized audio signals can also be converted into feature representations that are better suited for certain processing tasks. A natural choice are Time-Frequency (TF) transforms (e.g., Short Time Fourier Transform, STFT), or Low-Level Features (LLF) derived from them. These representations are beneficial for untangling and emphasizing the important information hidden in the audio signal. Into this pattern, we also subsume processing steps intended to emphasize the target drum signal in an audio mixture. These can either be based on spectral characteristics (e.g., band-pass filters, BPF, with predefined center frequencies and bandwidths) or based on TF characteristics (e.g., Harmonic-Percussive Source Separation, HPSS [23]).

Event Segmentation (ES): The main goal of this design pattern is to detect the temporal location of musical events in a continuous audio stream before applying further processing. This usually consists of computing suitable novelty functions (e.g., Spectral Flux) and identifying locations of abrupt change. A typical procedure would be to extract local extrema by

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2018.2830113, IEEE/ACM Transactions on Audio, Speech, and Language Processing

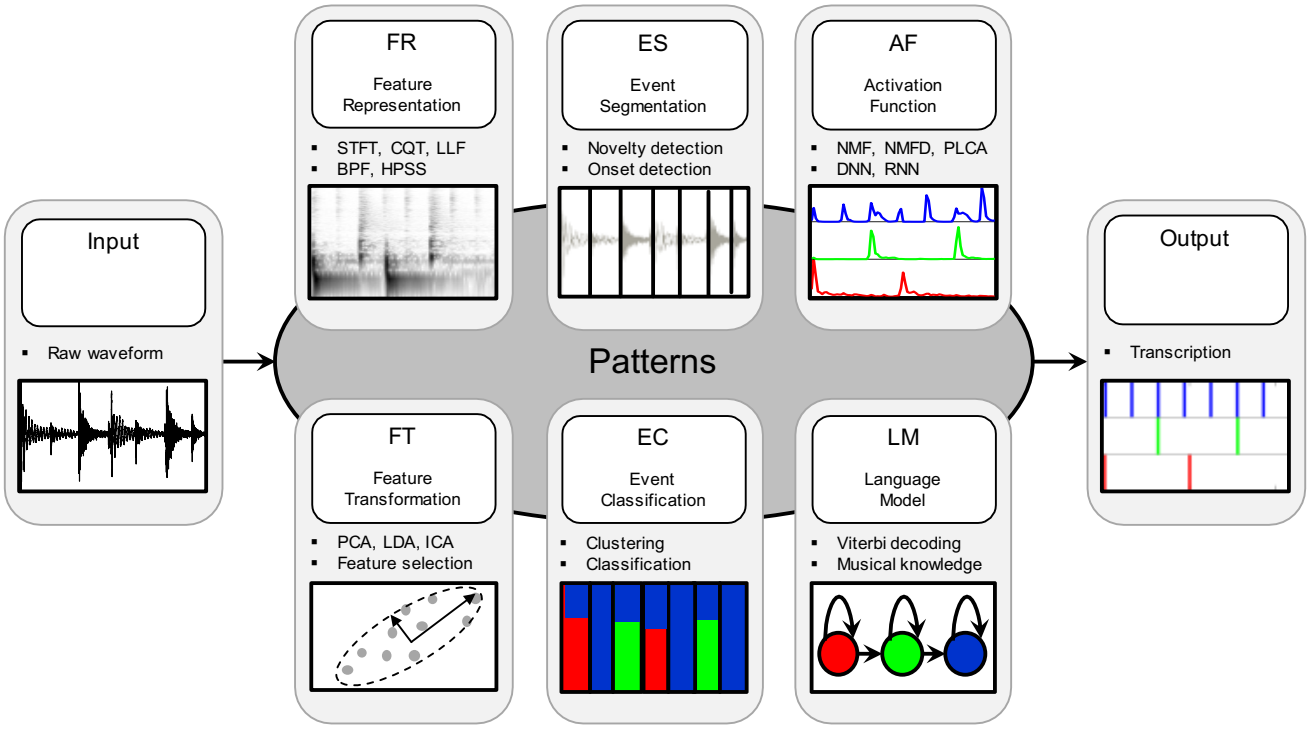


Fig. 4: Our proposed grouping of design patterns that are relevant for ADT.

applying a suitable peak-picking strategy, often referred to as OD in MIR research. Recently, learned feature representations have been shown to yield superior performance compared to hand-crafted ones for event segmentation.

Activation Function (AF): This design pattern seeks to map feature representations into activation functions, which indicate the activity level of different drum instruments. Different techniques such as NMF, Probabilistic Latent Component Analysis (PLCA) or Deep Neural Neworks (DNNs) are commonly used for deriving the activation functions.

Feature Transformation (FT): This design pattern provides a transformation of the feature representation to a more compact form. This goal can be achieved by different techniques such as feature selection, Principal Component Analysis (PCA), or Linear Discriminant Analysis (LDA). It should be mentioned that there is a strong overlap between the patterns **FT** and **AF**.

Event Classification (EC): This processing step aims at associating the instrument type (e.g., KD, SD, or HH) with the corresponding musical event. In the majority of papers, this is achieved through machine learning methods (e.g., Support Vector Machines, SVM) that can learn to discriminate the target drum instruments (or combinations thereof) based on training examples. Inexpensive alternatives include clustering (e.g., Alternate Level Clustering, ALC) and cross-correlation.

Language Model (LM): This pattern takes the sequential relationship between musical events into account. Usually this is achieved using a probabilistic model capable of learning the musical grammar and inferring the structure of musical events. **LM**s are based on classical methods such as Hidden Markov Models (HMM) or more recent methods such as RNNs.

While these design patterns represent essential building blocks, usually only a subset of them are used in specific ADT approaches. Different methods comprise selected patterns with additional minor adaptations.

The following sections will discuss various combinations and cascades of the introduced patterns in more detail. In each of the section headings, the typical cascade of patterns (e.g., **FR**, **ES**, **EC**) is given with the abbreviations introduced in Fig. 4. Note that these combinations are not exhaustive as new methods emerge constantly. However, with this flexible framework, it is possible to characterize future studies with different sets of cascaded patterns.

*B. Segmentation-Based Methods (**FR**, **ES**, **EC**)*

This type of approach centers around the Event Segmentation **ES** concept and generally uses a cascade of Feature Representation **FR** and **ES** with occasional inclusion of Event Classification **EC**. Since most of the drum events are percussive and transient in nature, it is intuitive to apply a simple **ES** method (e.g., OD) on the input signal for segmenting and detecting such events. The rationale is to first emphasize the drum sound events within an audio mixture through various Feature Representation **FR** operations (e.g., HPSS [23], BPF), and perform **ES** on the resulting feature representations.
One of the earliest systems in this category was presented

TABLE II: Overview of previously proposed methods for ADT. The properties in the columns highlight the most important algorithmic details. For a reference to the acronyms, please refer to Table I.

Year	Author(s)	Reference(s)	ADT Task	Design Patterns	FR	FT	ES	EC	AF	LM
1985	Schloss	[24]	DTD	**FR**, **ES**	WAV	–	OD	–	–	–
2000	Gouyon et al.	[25]	DSC	**FR**, **EC**	LLF	–	–	FDA	–	–
2002	FitzGerald et al.	[26]	DTD	**FR**, **AF**, **ES**	STFT	–	OD	–	ISA	–
2002	Herrera et al.	[27]	DSC	**FR**, **ES**, **FT**, **EC**	LLF	CFS	OD	KNN	–	–
2002	Zils et al.	[28]	DTM	**FR**, **ES**, **EC**	WAV	–	OD	CRF	–	–
2003	Eronen	[29]	DSC	**FR**, **ES**, **FT**, **EC**	MFCC	ICA	OD	HMM	–	–
2003	FitzGerald et al.	[30]–[32]	DTD	**FR**, **AF**, **ES**	STFT	–	OD	–	PSA	–
2003	Herrera et al.	[33]	DSC	**FR**, **ES**, **FT**, **EC**	LLF	CFS	OD	DT/MDS	–	–
2004	Dittmar & Uhle	[34]	DTM	**FR**, **AF**, **ES**	STFT	–	OD	–	NNICA	–
2004	Gillet & Richard	[35]	DTD	**FR**, **ES**, **EC**	LLF	–	OD	HMM/SVM	–	–
2004	Herrera et al.	[36]	DSC	**FR**, **ES**, **EC**	LLF	–	OD	DT/KNN	–	–
2004	Nakano et al.	[37]	VPT	**FR**, **LM**	MFCC	–	–	–	–	HMM
2004	Sandvold et al.	[38]	DTM	**FR**, **ES**, **FT**, **EC**	LLF	CFS	OD	DT/ABT	–	–
2004	Steelant et al.	[39], [40]	DSC	**FR**, **EC**	LLF	–	–	SVM	–	–
2004	Tindale et al.	[41]	DTC	**FR**, **ES**, **EC**	LLF	–	OD	SVM/KNN	–	–
2004	Yoshii et al.	[42]–[44]	DTM	**FR**, **ES**, **EC**	STFT	–	OD	AdaMa	–	–
2005	Degroeve et al.	[45]	DSC	**FR**, **EC**	LLF	–	–	SVM	–	–
2005	Gillet & Richard	[46]	DTM	**FR**, **ES**, **FT**, **EC**	BPF/NSP	PCA	OD	SVM	–	–
2005	Gillet & Richard	[47]	DTM	**FR**, **ES**, **FT**, **EC**	AVF	PCA	OD	SVM	–	–
2005	Hazan	[48]	VPT	**FR**, **ES**, **EC**	LLF	–	OD	DT/KNN	–	–
2005	Paulus & Virtanen	[49]	DTD	**FR**, **AF**, **ES**	STFT	–	OD	–	NMF	–
2005	Tanghe et al.	[50]	DTM	**FR**, **ES**, **EC**	LLF	–	OD	SVM	–	–
2005	Tzanetakis et al.	[51]	DTM	**FR**, **ES**	DWT/BPF	–	OD	–	–	–
2006	Bello et al.	[52]	DTD	**FR**, **ES**, **EC**	LLF	–	OD	HCA	–	–
2007	Gillet & Richard	[53]	DTM	**FR**, **ES**, **LM**	Symbolic	–	OD	–	–	N-gram
2007	Moreau & Flexer	[54]	DTM	**FR**, **ES**, **EC**	NMF/LLF	–	OD	KNN	–	–
2007	Roy et al.	[55]	DSC	**FR**, **ES**, **FT**, **EC**	LLF	IGR	OD	SVM/KNN	–	–
2008	Gillet & Richard	[19]	DTM	**FR**, **ES**, **FT**, **EC**	MFCC	PCA	OD	SVM	–	–
2008	Pampalk et al.	[56]	DSSS	**FR**, **EC**	MLS	–	–	MNSR	–	–
2009	Alves et al.	[57]	DTM	**FR**, **AF**, **ES**	STFT	–	OD	–	NMF	–
2009	Paulus & Klapuri	[20], [58]	DTM	**FR**, **FT**, **LM**	MFCC	LDA	–	–	HMM	
2010	Scholler & Purwins	[59]	DSC	**FR**, **EC**	MPSC	–	–	DT	–	–
2010	Spich & Zanoni	[60]	DTM	**FR**, **FT**, **ES**	STFT	–	OD	–	PSA	–
2011	Şimşekli et al.	[61]	DTD	**FR**, **LM**	STFT	–	–	–	–	HMM
2012	Battenberg	[62], [63]	DTD (RT)	**ES**, **FR**, **AF**	STFT	–	OD	–	NMF	–
2012	Kaliakatsos et al.	[64]	DTD	**FR**, **ES**	WAV/BPF	–	OD	–	–	–
2012	Lindsay-Smith et al.	[65]	DTD	**FR**, **AF**, **ES**	STFT	–	OD	–	NMFD	–
2013	Miron et al.	[66], [67]	DTD (RT)	**ES**, **FR**, **EC**	LLF	–	OD	KNN	–	–
2014	Dzhambazov	[10]	DTM	**FR**, **LM**	LLF	–	–	–	–	HMM
2014	Benetos et al.	[68]	DTM	**FR**, **AF**, **ES**	CQT	–	OD	–	SIPLCA	–
2014	Dittmar & Gärtner	[69]	DTD (RT)	**FR**, **AF**, **ES**	STFT	–	OD	–	SANMF	–
2014	Thompson & Mauch	[9]	DTM	**FR**, **ES**, **EC**	MFCC	–	OD	SVM	–	–
2015	Röbel et al.	[70]	DTM	**FR**, **AF**, **ES**	STFT	–	OD	–	NMFD	–
2015	Souza et al.	[71]	DSC, DTC	**ES**, **FR**, **EC**	MFCC/LSF	–	OD	SVM	–	–
2015	Rossignol et al.	[72]	DTM	**FR**, **EC**, **ES**	LLF	–	OD	ALC	–	–
2015	Wu & Lerch	[73], [74]	DTD, DTM	**FR**, **AF**, **ES**	STFT	–	OD	–	PFNMF	–
2016	Gajhede et al.	[75]	DSC	**ES**, **FR**, **EC**	MLS	–	OD	CNN	–	–
2016	Vogl et al.	[76], [77]	DTD, DTM	**FR**, **AF**, **ES**	CQT	–	OD	–	RNN	–
2016	Southall et al.	[78]	DTD, DTM	**FR**, **AF**, **ES**	STFT	–	OD	–	BRNN	–
2016	Wu & Lerch	[79]	DTC	**FR**, **AF**, **EC**	STFT/LLF	–	–	SVM	PFNMF	–
2017	Vogl et al.	[11]	DTM	**FR**, **AF**, **ES**	CQT	–	OD	–	CNN/CRNN	–
2017	Southall et al.	[80]	DTM	**FR**, **AF**, **ES**	STFT	–	OD	–	CNN	–

by Schloss [24]. The system estimates the envelope of the waveform and determines the attack with a threshold on the envelope-slope. Additionally, the decay time-constant is characterized by model fitting. By combining this information, the resulting system is able to detect basic strokes from drum-only recordings. Zils et al. [28] proposed a method starting with initial drum sound templates created from band-pass-filtered impulses. Next, the calculation of correlation between the time-domain signal and the initial templates, followed by a peak-quality assessment, is used as the event classification **EC** step. Finally, the templates are updated with the averaged time-domain signals of the detected events. This process is repeated until the number of detected events stops changing. While this *analysis by synthesis* approach has the advantage of requiring minimum prior knowledge, it has some potential issues due to its focus on time-domain signals, such as the confusion between high-pitched percussive sounds and singing voice, simultaneous events, and mismatches between initial template and the target drum sounds. These issues may become severe when the complexity of the audio mixture increases.

Another method of this category was proposed by Tzanetakis et al. [51]. The **FR** emphasises the characteristic frequency ranges of KD (30-280 Hz) and HH (2.7k-5.5k Hz) via BPF based on Discrete Wavelet Transform (DWT). Next, the **ES** and **EC** for each drum was achieved by OD on the extracted envelope of the time-domain sub-band signal. Since this method

relies heavily on the selection of the frequency ranges of the filters, its generalization to other types of drum sounds can be problematic.

Kailakatsos-Papkostas et al. [64] proposed a similar method with a focus on the real-time performance. First, multiple band-pass filters are applied followed by suitable amplifiers. Instead of using predefined frequency ranges, an iterative process is used to estimate optimal filter parameters (e.g., filter passband, stopband, onset detection threshold) by minimizing an objective function. Once the training is completed, a threshold is used to decide whether a drum is active or inactive. This method provides an alternative solution to the selection of characteristic frequency ranges of drums.

Generally speaking, the simplicity of the above mentioned methods has several advantages. First, the direct use of waveforms in the processing pipeline provides good interpretability of the results; this allows users with limited or minimal technical background to gain better control over the systems. Additionally, simple **FR** methods (such as BPF) and **EC** methods (such as cross-correlation or thresholding) can be implemented very efficiently, therefore enabling real-time applications, e.g., in the context of live music performances. However, such systems also have downsides. First, the robustness to additional sound components (e.g., coming from melodic instruments) might be insufficient. Since the systems typically use a simple **FR** step such as BPF to highlight the presence of drum events, they are susceptible to the interference of additional sounds. Second, these systems mainly use time-domain signals in favor of the fast processing speed. This potentially limits their capability of extracting more detailed information of the musical content, compared to other signal representations. Finally, the basic **EC** methods incorporated in this type of approach might not be able to differentiate subtle timbral variations created by various playing techniques.

C. Classification-Based Methods (*FR, ES, FT, EC*)

This type of approach builds around the Event Classification **EC** concept that differentiates different drum sounds using classifiers. *Classification-based methods* and *Segmentation-based methods* may look similar in terms of their cascaded patterns, but they are quite different in nature; *Segmentation-based methods* emphasize the efficiency and interpretability, whereas *Classification-based methods* focus on getting better performances with more sophisticated algorithms. There are many papers implementing this strategy; the basic idea is to extract Feature Representations **FR** from the audio signal, find the location of the potential events using Event Segmentation **ES**, refine the features with Feature Transformation **FT**, and then determine the instrument class of the events using **EC** Event Classification. Since this processing pipeline is based on the standard pattern recognition paradigm, many different systems using different choices of **FR**, **FT**, and **EC** have been proposed. The most commonly used input representations are combinations of spectral features (e.g., centroid, flux, flatness), temporal features (e.g., zero crossing rate, local mean energy, RMS, envelope descriptors), and Mel-Frequency Cepstral Coefficients (MFCCs) [19], [25], [27], [33], [36], [38]–[41], [45], [48], [50], [52], [66], [67], [72]; other features, such as NMF derived features [54] and learned features [55], were also found useful in drum sound classification and drum transcription, respectively. To derive spectral features, mainly the STFT was used as **FR**; variants such as Constant-Q Transform (CQT) [52], [72], Line Spectral Frequencies (LSF) [71], and Mel-scale Log magnitude Spectrogram (MLS) [75] have been shown to be viable options as well. Besides audio features, Gillet and Richard [47] proposed to use audio-visual features (AVF), which included features derived from video recordings of the drum performances. In contrast to the input representations, **FT** methods are optional and thus more situational. Techniques that were adopted in previous systems include Principal Component Analysis (PCA) [47], Information Gain Ratio [55], Recursive Feature Elimination [19], Correlation-based Feature Selection (CFS) [27] and Sparse Coding Matching Pursuit (SC-MP) [59], [81].

In terms of classifiers, basic models such as K-Nearest Neighbors (KNN) were often selected for their simplicity and interpretability [27], [33], [36], [41], [54], [66]. To account for non-linear relationships of the extracted features, SVMs with different kernel functions were used extensively in various systems [9], [19], [35], [39]–[41], [45], [47], [50], [55], [71], [81]; ensemble methods, such as Adaboost [39] and Random Forest (RF) [59], were often included in comparative studies for their effectiveness. Recently, successful models from other applications of machine learning, such as Convolutional Neural Networks (CNNs), have also been applied for drum sound classification [75]. In addition to the above mentioned supervised approaches, unsupervised methods were also applied for **EC**. For example, algorithms such as K-means [25], [52], [67] and ALC [72] were adopted to solve different ADT sub-tasks.

In Eronen's work on musical instrument recognition, a slightly different approach using a probabilistic model in the **EC** stage for classifying the drum sounds was presented [29]. Eronen proposed to use an HMM to model the temporal progression of features within an isolated audio sample. MFCC and the first derivative of MFCC were extracted as the features, followed by a **FT** step using Independent Component Analysis (ICA) that transforms the features into statistically independent representations.

Another system that falls implicitly into this category is the AdaMa-appraoch proposed by Yoshii et al. [42]–[44]. The general concept is to start with an initial guess for the drum sounds (sometimes called templates) that are iteratively refined to match the drum sounds that actually occur in the target recording. The refinement is based on alternating between drum onset detection with the latest drum template estimate and updating the template with an averaged model of several, trustworthy onset instances of the drum sound. Unlike the system proposed by Zils et al. [28], AdaMa uses an STFT-based **FR** instead of raw waveforms, and an **EC** step based on a customized distance measure between the target event and the templates.

To summarize, the *Classification-based methods* have the following advantages. First, the general processing flow

inherited from the pattern recognition paradigm allows an efficient and automated search of suitable settings. For instance, different classifiers or feature selection methods can be easily introduced in a modular fashion. Second, the possibility of adding various features during the **FR** step ensures the flexibility of incorporating expert knowledge in this type of system. However, since this type of system relies on a robust **ES** step to detect the musical events, any potential errors made in this stage are propagated through the system. Furthermore, to be able to handle simultaneous events (e.g., HH + SD, HH + KD), more classes are needed during the training phase. Thus, the number of class combinations will increase drastically as more instruments (e.g., HT, MT, LT, RC, and CC) are considered. Finally, *Classification-based methods* might have difficulties to recognize drum sound events in the presence of other melodic instruments that have never been presented to the system at training time, as the trained features are usually susceptible to the interference of the melodic instruments.

*D. Language-Model-Based Methods (**FR**, **FT**, **LM**)*

After applying Feature Representation **FR** and Feature Transformation **FT** patterns, *Language-model-based methods* typically rely on a final processing stage, which involves the deployment of a Language Model **LM** to account for the temporal evolution of events on a higher hierarchical level. Instead of detecting drum sound events directly from input representations, *Language-model-based methods* infer the underlying drum sound events by considering neighboring events and their probability as an entire sequence. This step is usually implemented using probabilistic models such as HMMs, where emission and transition probabilities are estimated from the temporal context of the training data.

One of the earliest works in this category was presented by Nakano et al. [82], which focused on VPT (i.e., beatboxing). The proposed system first extracts MFCCs from the given audio recording. Next, the acoustic features are decoded into sequences of onomatopoeic expressions using the Viterbi algorithm. Finally, the onomatopoeic expressions are mapped to drum sequences by retrieving the drum patterns with highest similarity from the predefined database. Another work that applies HMMs to model drum sequences was proposed by Paulus and Klapuri [20], [58]. In the **FR** step, the system uses a *sinusoids-plus-residual* model to suppress the harmonic components in the audio mixtures. Next, MFCCs are extracted as the feature representation, followed by an **FT** step using Linear Discriminant Analysis (LDA). Finally, the Viterbi algorithm and trained HMMs are used to determine the underlying drum sequences. Similarly, Şimşekli et al. [61] also use HMMs for detecting percussive events such as clapping and drum hits; with additional parameters, the model can be adjusted for the trade-off between accuracy and latency. The authors report good performances on the specific datasets, however, their generalizability on other datasets still needs to be further investigated.

In addition to decoding the underlying drum sequences, language models can also be used as post-processing tool. Gillet and Richard proposed to apply N-gram models on the symbolic data in order to fine-tune the detected onsets from the ADT systems in [53]. Their system first aligns the detected onsets to the tatum grid (a grid based on the smallest time unit inferred from the perceived musical events). Next, the probability of a particular sequence can be estimated using a smoothed probability distribution of various sequences in the training corpus, as presented in [10]. Both supervised and unsupervised training schemes are evaluated, and the experiment results show a general performance gain of these methods. Nevertheless, the error from the preceding step (i.e., drum onset detection) may propagate through and reduce the overall performance.

The above mentioned methods are based on statistical estimation of the most likely drum sequences, and are hence aware of the musical context. In other words, these systems try to make predictions that are musically meaningful. For example, an unusual hit after certain sequences might be ignored due to the low probability of the resulting drum hit sequence.

LMs are not commonly used in modern ADT systems and are usually limited to basic methods. This is due to the fact that the application of **LM**s in the context of ADT, and more general in music related tasks bears several challenges. First, the application of **LM**s commonly used in Automatic Speech Recognition (ASR) on music data is only viable to a certain degree. The different properties of speech and language require a reformulation of the basic underlying assumptions. In ASR, **LM**s usually model lengths of phonemes and identify words, and on a second level may be used (e.g.,LSTMs) to model the grammar and typical sentences of a language. These concepts do not translate to music, while in ASR durations and pauses are of little concern, these factors are essential for music, especially drums. Also, music generally does not follow strict rules compared to the grammar of a language. Attempts at using **LM**s for music in the context of chord recognition showed that the adaptation is far from trivial [83]. Furthermore, training of valid **LM**s usually requires large amounts of training data, which are available in the case of ASR, but are lacking for ADT tasks.

*E. Activation-Based Methods (**FR**, **AF**, **ES**)*

Activation-based systems often comprise a cascade of Feature Representation **FR**, Activation Function **AF**, and Event Segmentation **ES** steps. The defining factor of this approach is the concept **AF**, which generates the activity of a specific instrument over time. With the activation functions for every drum instrument, the **ES** step can be as simple as finding local maxima of those activation functions by means of suitable peak-picking algorithms.

There are basically two families of algorithms for deriving activation functions. The first one uses magnitude spectrograms as **FR** and applies matrix factorization algorithms as **AF** in

order to decompose the spectrogram into basis functions and their corresponding activation functions. Early systems used methods such as Independent Subspace Analysis (ISA) [26], Prior Subspace Analysis (PSA) [30]–[32], [60], and Non-Negative Independent Component Analysis (NNICA) [34]. The basic assumption of these algorithms is that the target signal is a superposition of multiple, statistically independent sources. Already for drum-only recordings, this assumption is problematic since the activations of the different drum instruments are usually rhythmically related. When the signal contains both drums and melodic instruments, this assumption may be more severely violated. Recently, more and more systems opted for NMF, which has less strict statistical assumptions about the sources. In NMF, the only constraint is the non-negativity of the sources, which is naturally given in magnitude spectograms. NMF-based ADT systems include basic NMF [49], [57] as well as related concepts such as Non-negative Vector Decomposition (NVD) [62], [63], Non-Negative Matrix Deconvolution (NMFD) [65], [70], Semi-Adaptive NMF [69], Partially-Fixed NMF [73], [74], [79], and Probabilistic Latent Component Analysis (PLCA) [68]. Most of these factorization-based methods require a set of predefined basis functions as prior knowledge; when this predefined set does not match well with the components in the target signal, the resulting performance may decrease significantly. In Sect. V, we will provide an in-depth description of the technical details and peculiarities of NMF-based ADT approaches.

The second family of algorithms which can be used to generate activation functions are based on Deep Neural Networks (DNN). In general, DNNs are a machine learning architecture that allow to learn non-linear mappings of arbitrary inputs to target outputs based on training data. They are usually constructed as a cascade of layers consisting of learnable, linear weights and simple non-linear functions. The learning of the weight parameters is performed by variants of *gradient descent* [84]. In recent years, RNNs, a special form of DNNs designed to work on time series data, have been applied successfully for ADT. The use of bidirectional RNNs [78], RNNs with label time shift [76], as well as RNNs with Gated Recurrent Units (GRUs) and Long Short-Term Memory cells (LSTMs) [77], [80], showed comparable results to state-of-the-art systems. It is important to note that RNNs can in principle also perform sequence modeling, similar to the more classic methods such as HMM (see Sect. II-D). However, the lack of large amounts of training data and the applied training methods, prohibit this behavior in the related work, so far. Recently, promising first attempts to apply CNNs and CRNNs to the task of ADT have been made [11], [80], showing the possibilities of adopting different architectures in addition to RNNs.

In Sect. VI, we will provide an in-depth description of the technical details and peculiarities of RNN-based ADT approaches.

Overall, *Activation-based methods* have the advantage of producing intermediate output representations that are easy to interpret. Some of the factorization-based approaches can also be used to reconstruct the magnitude spectrogram of drum sources and serve as source separators. In addition, this type of approach takes care of simultaneous events without the need of introducing combined classes during training (see Sect. II-C). However, when the multiple sources overlap in the spectral domain, cross-talk between activation functions will appear and degrade the performance. For instance, the activation function of a KD may also contain the interference from a bass guitar. Furthermore, the use of magnitude spectrograms neglects the phase, which could potentially strip away critical information.

F. Datasets and Metrics

In addition to the combinations of design patterns, the data used to train and evaluate ADT systems plays an important role. Furthermore, there are commonly accepted procedures for assessing ADT performance.

Public Datasets: In Table III, we present an overview of existing datasets. These are often associated with different ADT tasks and contain different types of recordings. For example, 200 Drum Machines [85] features a collection of electronic drum sounds, whereas as MDLib2.2 [86] only features acoustic drum sounds. As a result, the choice of dataset may have significant impact on the generalization capabilities of the resulting system.

Among these publicly available datasets, IDMT-SMT-Drums [69] and ENST-Drums [87] are two of the most commonly used datasets in recent ADT studies. IDMT-SMT-Drums [69] comprises solely drum recordings containing the major drum instruments (i.e., HH, SD, KD). Each item in the dataset has a ground-truth transcription and comes with training audio files, which contain the used drum sounds in isolation. This dataset can be used for DSC and DTD tasks.

ENST-Drums [87] comprises recordings of full drum kits, including instruments such as CC, RC, HT, MT, and LT (see Fig. 1). Again, each item in the dataset has a corresponding ground-truth transcription available. These recordings are played by three different drummers on their own drum kits. Additionally, some of the drum recordings have corresponding accompaniment recordings, allowing the creation of complete mixtures. The accompaniments contained in ENST-Drums are partly played on real instruments (e.g., bass, guitar, saxophone, clarinet) and partly on synthesizers. All are temporally aligned to the drum recordings, since the drummers were asked to play along to the backing tracks. This dataset can be used for DTD, DTP, and DTM tasks. These datasets, while being limited in certain aspects (see Sect. III-D for a detailed discussion), provide a great starting point for most ADT tasks. Therefore, both of the datasets are currently considered as benchmark datasets for ADT research.

Common Metrics: As discussed in Sect. I-C, ADT studies cover a variety of tasks, and their evaluation metrics differ from each other. For tasks such as DSC and DTC, many previous studies (e.g., [25], [27], [41]) performed cross-validation on the collection of isolated drum sounds and reported the classification accuracy per instrument. This accuracy is usually calculated as the ratio between number of correct samples and number of total samples.

For tasks such as DTD, DTP, and DTM, the main focus is

TABLE III: An overview of the existing annotated datasets for ADT tasks. * indicates the dataset that is not freely available

Dataset	Suited for ADT task	Size (Duration)	Audio avail.
ENST-Drums [87]	DTD/DTP/DTM	316 files (10-90 s each)	Y
IDMT-SMT-Drums [69]	DTD	560 files (5-20 s each)	Y
DREANSS [88]	DTD/DTM	22 files (10 s each)	N
Tindale et al. [41]	DTC	1264 files (<1 s each)	Y
200 Drum Machines [85]	DSC	7371 files (<1 s each)	Y
MDLib2.2 [86]	DTC	10624 files (1-2 s each)	Y
RWC-POP* [89]	DTM	100 files (3-5 min each)	Y
Drum PT [79]	DTC	30 files (30-90 s each)	N

to extract onset times of different drum instruments from a continuous audio stream. In this case, the metrics for assessing onset detection algorithms, namely Precision, Recall, and F-measure, are commonly used in several studies [58], [69], [74], [76], [78]. A detected onset is counted as a *true positive (TP)* if its deviation from the corresponding ground-truth annotation is less than a pre-determined tolerance window. If a detected onset does not coincide with any annotated drum event, it is counted as a *false positive (FP)*; alternatively, if an annotated drum event does not coincide with any detected onset, it is counted as a *false negative (FN)*. These three quantities define the standard Precision $P = \mathrm{TP}/(\mathrm{TP} + \mathrm{FP})$, Recall $R = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN})$, and F-measure $F = 2 \cdot \mathrm{TP}/\left(2 \cdot \mathrm{TP} + \mathrm{FP} + \mathrm{FN}\right)$.

Tolerance Window: The size of the pre-determined tolerance window is dependent on the ADT application. For example, if the desired output is a musical score or tabulature, the time resolution for the detected onsets can be relatively coarse as onset times are quantized based on the smallest increment within a metrical grid (i.e., 16th or 32nd notes), for these kind of representations. For other applications, like analysis of (un-)intentional variation of onset timings by human performers (e.g.,for humanization), musicological studies of micro-timing [16]–[18], or extraction of a precise symbolic representation (e.g.,MIDI files), a greater precision is required. Perceptionally, the lower bound of the time-gap which allows humans to identify two click sounds as separate is somewhere in the range of 8 -10 ms [90]. This suggests that for precise reproduction for a human ear, a tolerance of up to 20 ms should be acceptable. Common tolerance window sizes used in existing ADT literature include 50 ms [69], [74], [78], 30 ms [65], and 20 ms [11], [77].

The choice of tolerance window also depends on the precision of the available training and evaluation data's annotations. To be able to use low tolerance windows, the annotations must also conform to these high standards. With synthetic (i.e. generated) datasets it is easy to achieve annotations with high precision, but with human annotated datasets a high level of quality insurance is necessary. This might be a reason why in tasks like ADT and onset detection, traditionally, relatively high tolerance windows (e.g.,50 ms) are commonly used.

III. Current Challenges

In the following section, we highlight the challenges that are commonly encountered in current and previous ADT research.

A. Interference of Multiple Instruments

The major challenge of state-of-the-art ADT systems usually comes from the interference of other instruments. The superposition of various instruments (e.g., guitar, keyboard, vocal, or drums) makes the recognition of a specific instrument difficult due to the overlaps in both spectral and temporal domain. Typically, the challenges arise in the presence of the following types of instruments:

Percussive Instruments: A basic drum kit, as introduced in Sect. I-A, includes drums of different sizes and well-distinguishable timbral characteristics. However, in a more advanced setup for studio recordings, similar drums with subtle variations in timbre often appear, resulting in sounds that are harder to differentiate. This problem is more severe when these sounds occur simultaneously. In previous work, this problem is mostly addressed as a DSC task, in which the sounds are presented as isolated audio samples, and Classification-Based Methods (**FR**, **ES**, **FT**, **EC**) tend to achieve a reasonably high classification accuracy. For example, in [33], a classification task for 33 different percussive instruments was performed; in [71], an attempt was made to classify different cymbals, such as china, crash, hi-hat, ride and splash. However, in a more realistic setting such as DTP, the perfect isolation of each drum sound is hard to achieve. Thus, the classification accuracy can be expected to decrease.

Melodic Instruments: Despite the fundamental difference between percussive and melodic instruments, the wide range of sounds produced from a drum kit can potentially coincide with sound components of many melodic instruments (e.g., the KD may overlap with bass guitar or SD may overlap with guitar and piano). As a result, DTM is considered much more challenging than DTP and DTD. Among all the methods in Table II, only less than half of the systems were evaluated under the DTM setting, and most of them reported a noticeable drop in performance compared with DTD and DTP. Preprocessing steps intended to suppress the melodic content of the audio signals have been proposed in [19], [44], however, the improvement has not been substantial so far.

B. Playing Techniques

Playing techniques are an important aspect of expressive musical performances. For drum instruments, these techniques include basic rudiments (e.g., roll, paradiddle, drag, and flam) as well as timbral variations (e.g., ghost note, brush, cross stick, and rim shot). In spite of being an essential part of

performances, most of the systems only focus on transcribing basic strikes, and the effects of different playing techniques are largely overlooked.

In an early attempt to transcribe playing techniques, Tindale et al. [41] presented a study on the automatic identification of timbral variations of the snare drum sounds induced by different excitations. A classification task is formulated to differentiate sounds from different striking locations (center, halfway, edge, etc.) with different excitations (strike, rim shot, and brush). Similarly, Prockup et al. [86] explored the discrepancy between more expressive gestures on a larger dataset with combinations of different drums, stick heights, stroke intensities, strike positions, and articulations. In addition to membranophones percussive instruments, Souza et al. [71] thoroughly investigated different playing techniques for cymbal sounds. They differentiated either by the position where the cymbal is struck (bell, body, edge), how a hi-hat is played (closed, open, chick), or other special effects such as choking a cymbal with the playing hand. All of these studies showed promising results in classifying the isolated sounds, however, when the classifier is applied to the real-world recordings, as pointed out in [79], the performance dropped drastically. Another attempt to retrieve playing techniques was proposed by Hochenbaum and Kapur through the use of both audio and accelerometer data [91]. However, the extra requirement of attaching the sensors to the performer's hands might impact the playing experience and deviate from the real playing gestures.

C. Recording Conditions and Post Production

In musical terms, the drum recordings contained in these corpora exhibit different degrees of rhythmic complexity. With respect to their acoustic properties, both corpora feature clean recordings that allow for controlled transcription experiments. However, in real-world drum recordings, there might be additional problems that are not reflected well in this data yet.

In practice, it is likely that we have to deal with convolutive, time-variant, and non-linear mixtures instead of linear superpositions of single drum sounds. First, the acoustic conditions of the recording room and the microphone setup lead to reverberation effects that might be substantial. Furthermore, the recording engineer will likely apply equalization and filtering to the microphone signal. Mostly, the resulting signal alterations can be modeled as convolution with one or more impulse responses. Second, non-linear effects such as dynamic compression and distortion might be applied to the drum recordings. Especially dynamics processing is considered to be one the most important post-processing steps that recording engineers use to modify drum sounds.

Not having these aspects covered in our datasets has two consequences. First, any methods involving machine-learning might deteriorate if the "closed world" of the training data does not match the "open world" of some target data. A typical example is found in speech processing where systems trained with clean speech often fail under noisy or reverberant conditions. Second, any methods involving decomposition based on a linear mixture model might be affected when the observed drum mixtures do violate these basic assumptions. A possible strategy to counter these challenges might be data augmentation. In our case the amount of training data could be greatly enhanced by applying diverse combinations of audio processing algorithms including reverberation, distortion and dynamics processing.

D. Insufficient Real-world Datasets

As summarized in Table II, many of the existing ADT systems are based on data-driven machine learning approaches. However, with the complexity of music, the difficulty of generating labels, and the restrictions of intellectual property laws, building and sharing annotated datasets becomes a non-trivial task; many of the commonly used datasets are thus limited in different aspects. A closer look at the existing datasets shown in Table III reveals the following limitations:

Size: The most common issue of all the existing drum transcription datasets is the insufficient amount of data. Overall, the datasets that contain only audio samples with a single drum hit (Tindale et al. [41], 200 Drum Machines [85], and MDLib2.2 [86]) have more files, whereas the datasets that contain entire drum sequences (ENST-Drums [87], IDMT-SMT-Drums [69], DREANSS [88], RWC-POP [89] and Drum PT [79]) have less files. However, the total duration of each dataset is usually less than a few hours and might not be representative for the immense amount of real-world music. Furthermore, since these datasets are created under very different conditions, they cannot be easily integrated into one large entity. Recently, an early attempt to address this challenge by utilizing unlabeled music data was presented in [92], but the insufficient amount of labeled data still remains to be an open problem.

Complexity: The existing datasets have the tendency of over-simplifying the ADT problem. For example, in datasets containing isolated drum hits (i.e., Tindale et al. [41], 200 Drum Machines [85], and MDLib2.2 [86]), the transcription problem is reduced to the classification of different drum sounds; In IDMT-SMT-Drums [69], only the drum sequences with basic patterns are presented in the dataset. The lack of complexity results in datasets that are unrealistic for the real-world use cases.

Diversity: Most of these datasets do not cover a wide range of music genre and playing style. For instance, RWC-POP [89] only covers Japanese pop music, IDMT-SMT-Drums [69] only covers basic patterns and playing techniques for pop and rock music, and ENST-Drums [87] only features playing styles from 3 drummers. The limitation in terms of diversity can hinder the system's capability of analyzing a wider range of music pieces. Particularly, the lack of any singing voice in the corpora ENST-Drums and IDMT-SMT-Drums indicates their insufficiency. Tests on tracks containing singing voice revealed that this poses a big problem, especially for RNN-based ADT methods.

Homogeneity: The problem of homogeneity usually originates from the creation of the dataset. Since each dataset is most likely to be generated under fixed conditions (i.e., recorded in the same room by the same group of performers), the audio files within the same dataset tend to have high similarities. This is very different from real-world scenarios, where the drum recordings come from different musicians, different drum kits, and different recording and processing conditions (as discussed in Sect.III-C). This limitation in homogeneity can potentially lead to an overfitting issue in the resulting ADT systems.

With these general remarks on challenges in ADT research, we conclude our literature overview. In the following sections, we focus on evaluating state-of-the-art ADT systems. In the process, many of the aforementioned difficulties will become relevant again. Over the past few years, activation-based ADT systems have achieved the state-of-the-art results, which lead to the in-depth discussion and evaluation in the following sections. However, with the introduction of general design patterns in Sect. II-A, we hope to encourage the discovery of un-explored combinations and inspire future ADT research

IV. Commonalities of State-of-the-art Systems

Following the general overview of ADT approaches and challenges, we now want to focus on ADT systems that are currently defining the state-of-the-art. According to their constituent design patterns, these systems can all be categorized as activation-based methods. However, based on how the activation functions are derived, they can be further categorized into two families, namely the NMF-based and RNN-based approaches. The next four sections will provide a more detailed discussion on these techniques. We will start by introducing their commonalities in **FR** and **ES** with consistent mathematical notations. Next, we will provide detailed description on **AF** with both NMF-based and RNN-based approaches. Finally, a comprehensive evaluation will highlight the strengths and weaknesses of the different approaches. Not all aspects of the systems are covered and so the reader is referred to the original papers if further information is desired. In order to put both in an unified perspective, we will start with the introduction of a common signal model.

A. Common Notation

The following mathematical notation will be used throughout the remainder of this paper. Uppercase italic letters such as K will be used to denote fixed scalar parameters, while lowercase italic letters such as k are used to denote running variables or indices. We denote integer intervals as $[1:K] := \{1, 2, \ldots, K\}$. Uppercase non-italic letters such as X usually denote matrices, while lower-case non-italic letters such as x denote column vectors. The operation $\mathrm{X}^\top$ denotes transposition. Rounded brackets are used to refer to elements of vectors and matrices, e.g., $\mathrm{X}(k,t)$ refers to the element located at the k^{th} row and the t^{th} column of matrix X. The colon is a short notation for taking slices along a certain dimension of a matrix, e.g., $\mathrm{X}(:,t)$ denotes the t^{th} column. For notational convenience, we also introduce the superscript notation $\mathrm{x}^t := \mathrm{X}(:,t)$ and the subscript notation $\mathrm{x}_k := \mathrm{X}(k,:)$. In Sect. VI, we will make extensive use of that notation, also for the sake of compatibility with previous work. Other notational conventions will be explained in the respective paragraphs.

B. Feature Representation

Both families of ADT systems considered here belong to the activation-based methods. As such, they are all based on the signal model assumption that the given drum recording is approximately a linear mixture of constituent drum sound events. Let $\mathrm{X} \in \mathbb{R}_{\geq 0}^{K \times T}$ be the signal's magnitude spectrogram from the STFT, with $\mathrm{X}(k,t)$, representing the non-negative, real-valued TF magnitude coefficient at the k^{th} spectral bin for $k \in [0:K]$ and the t^{th} time frame for $t \in [1:T]$. The number of frequency bins is determined by the window size N as $K = N/2$. The number of spectral frames T is determined by the available signal samples. Our objective is to map X to an activation representation $\mathrm{G} \in \mathbb{R}_{\geq 0}^{R \times T}$. Here, the number of rows $R \in \mathbb{N}$ is usually equal to the number of distinct drum instruments (e.g., $R = 3$ for KD, SD, HH). As G encodes the activations of a certain drum instrument over time, $\mathrm{G}(r,t)$ should be large if instrument r has an onset at time t and otherwise small. Ideally, the activations should be impulse-like as shown in Figure 3c.

C. Event Segmentation (Peak-Picking)

The detection of candidate onset events is typically approached by picking the peaks in the activation function $\mathrm{G}(r,:)$ for each $r \in [1:R]$. This process is similar to the *Peak-Picking* step in generic onset detection methods [93], and different adaptive thresholding techniques may be chosen for further optimization. However, since the activation functions of the evaluated systems in this paper are different in nature, no specific optimization has been done. Instead, we employ a very simple procedure consistently for all evaluated methods instead of the different peak-picking approaches used in the original works. This is done in order to easier identify the differences focusing on the extraction of activation functions. First, a dynamic threshold $\Delta \in \mathbb{R}_{\geq 0}^{R \times T}$ is calculated for each considered drum instrument and each frame using

$$\Delta(r,t) = \frac{1}{2\Gamma + 1} \sum_{n=t-\Gamma}^{t+\Gamma} \mathrm{G}(r,n). \tag{1}$$

In this context, $\Gamma \in \mathbb{N}$ determines the window used to calculate the average (we assume suitable zero padding at the boundaries). Second, we introduce a binary-valued output matrix $\mathrm{O} \in \mathbb{B}^{R \times T}$ with $\mathbb{B} := \{0, 1\}$. The elements of O encode onset candidates and are defined as follows:

$$\mathrm{O}(r,t) = \begin{cases} 1, & \mathrm{G}(r,t) = \max\left(\mathrm{G}(r, t-\Omega : t+\Omega)\right) \\ & \text{and} \quad \mathrm{G}(r,t) > \Delta(r,t) \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

Here, $\Omega \in \mathbb{N}$ determines the window used for local maximum search. In other words, a candidate peak is only accepted if it exceeds the dynamic threshold Δ as well as its local neighborhood. If the criterion in Eq. (2) also is true for two peaks within a certain distance, the weaker of both is discarded. The output matrix O will become important again in the context of evaluation metrics in Sect. VII-C.

V. NMF-BASED ADT SYSTEMS

In this section, we provide more details of the different ADT systems employing variants of NMF. Fig. 5 depicts the basic of decomposing the mixture spectrogram X into spectral basis functions $\mathrm{B}(:, r)$ (called templates), and corresponding time-varying gains $\mathrm{G}(r, :)$ (called activations). Intuitively speaking, the templates comprise the spectral content of the mixture's constituent components, while the activations describe when and with which intensity they occur.

The entries of template matrix B can be interpreted as averaged spectra of the corresponding drum instruments KD, SD, and HH. The KD, in red, occupies the lowest frequency region, the SD, in green, occupies the mid-region, and finally the HH, in blue, has most of its energy in the upper frequency region. In G, the corresponding drum onset events occur as peaks with quickly rising attacks. They are followed by exponentially decaying slopes that correspond to the natural decay of the drum sound events.

In both B and G we also inserted hatched regions. These shall express that we might add additional components modeling sound events in the mixture that do not originate from KD, SD, or HH. We will return to this concept in Sect. V-C. Furthermore, we discuss a convolutive extension to NMF in Sect. V-E.

A. Basic NMF Model

Mathematically, NMF is based on iteratively computing a low-rank approximation $\tilde{\mathrm{X}} \in \mathbb{R}_{\geq 0}^{K \times T}$ of the mixture spectrogram X. Specifically, $\tilde{\mathrm{X}}$ is defined as the linear combination of the templates $\mathrm{B} \in \mathbb{R}_{\geq 0}^{K \times R}$ and activations $\mathrm{G} \in \mathbb{R}_{\geq 0}^{R \times T}$ such that $\mathrm{X} \approx \tilde{\mathrm{X}} := \mathrm{B} \cdot \mathrm{G}$. Note that $\tilde{\mathrm{X}}$ always uses the latest available version of all parameters.

NMF typically starts with a suitable initialization of matrices B and G. Subsequently, these matrices are iteratively updated to approximate X with respect to a cost function $\mathcal{L}$. A standard choice is the generalized Kullback-Leibler Divergence [94], given as

$$\mathcal{L} = D_{\mathrm{KL}}(\mathrm{X} \mid \tilde{\mathrm{X}}) = \sum \left(\mathrm{X} \odot \log \left(\frac{\mathrm{X}}{\tilde{\mathrm{X}}} \right) - \mathrm{X} + \tilde{\mathrm{X}} \right). \quad (3)$$

The symbol $\odot$ denotes element-wise multiplication; the division is to be performed element-wise as well. The sum is to be computed over all KT elements of X. To minimize this cost, an alternating scheme with multiplicative updates is used [94]. The respective update rules are given as

$$\mathrm{B} \leftarrow \mathrm{B} \odot \frac{\frac{\mathrm{X}}{\tilde{\mathrm{X}}} \cdot \mathrm{G}^{\top}}{\mathrm{J} \cdot \mathrm{G}^{\top}}, \quad (4)$$

$$\mathrm{G} \leftarrow \mathrm{G} \odot \frac{\mathrm{B}^{\top} \cdot \frac{\mathrm{X}}{\tilde{\mathrm{X}}}}{\mathrm{B}^{\top} \cdot \mathrm{J}}, \quad (5)$$

where the symbol $\cdot$ denotes the matrix product. Furthermore, $\mathrm{J} \in \mathbb{R}^{K \times T}$ denotes a matrix of ones. Since this is an alternating update scheme, it should be noted that Eq. (4) uses the latest update of G from the previous iteration. In the same vein, Eq. (5) uses the latest update of B. These update rules are typically applied for a limited number of iterations L, with the iteration index $\ell \in [1 : L]$.

B. Fixed-Bases NMF (FNMF)

When using NMF for ADT, it is essential to choose a suitable number of components $R \in \mathbb{N}$ for the approximation and to provide good initializations for B. One popular choice (see for example [20], [49], [63], [69]) is to set R to the number of distinct drum instruments and to initialize individual $\mathrm{B}(:, r)$ with averaged spectra of isolated drum sound events. The rationale is to let the NMF component updates start from a point in the parameter space that is already close to a meaningful optimum.

In this context, some authors [63], [69] also propose to keep the initialized $\mathrm{B}(:, r)$ fixed throughout the NMF iterations, i.e., not to apply Eq. (4), which makes the optimization problem convex. Although this is a very appealing and simple approach, fixed NMF bases may be problematic in cases where the mixture consists of other components than the previously trained drum sounds. Intuitively speaking, the NMF updates rules will try to model the observed X as accurate as possible given the fixed prior basis vectors, possibly leading to spurious activations that resemble cross-talk between the different drum sounds.

C. Partially-Fixed NMF (PFNMF)

In addition to the fixed bases, additional templates in B can also be initialized randomly in order to model the harmonic part of the mixtures. In PFNMF [74], the matrices B and G are further split into the matrices $\mathrm{B_D}$ and $\mathrm{B_H}$, as well as $\mathrm{G_D}$ and $\mathrm{G_H}$, respectively. The matrix $\mathrm{B_D}$ is initialized as described in Sect. V-A and is fixed during the factorization process, while the matrices $\mathrm{B_H}$, $\mathrm{G_H}$, and $\mathrm{G_D}$ are initialized randomly and are updated iteratively. The number of components R_{D} in $\mathrm{B_D}$ and $\mathrm{G_D}$ depends on the number of templates (i.e., instruments) provided, and the number of additional templates R_{H} is a free parameter. The total number of components is $R = R_{\mathrm{D}} + R_{\mathrm{H}}$. To further emphasize the drum components, both the $\mathrm{B_D}$ and $\mathrm{B_H}$ can be weighted inside the loss function by scaling factors γ and δ, respectively. These scaling factors are set to be $\gamma = (R_{\mathrm{D}} + R_{\mathrm{H}})/R_{\mathrm{D}}$ for each drum template and $\delta = R_{\mathrm{H}}/(R_{\mathrm{D}} + R_{\mathrm{H}})$ for each harmonic template. This setting strengthens drum templates and attenuates harmonic templates

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2018.2830113, IEEE/ACM Transactions on Audio, Speech, and Language Processing

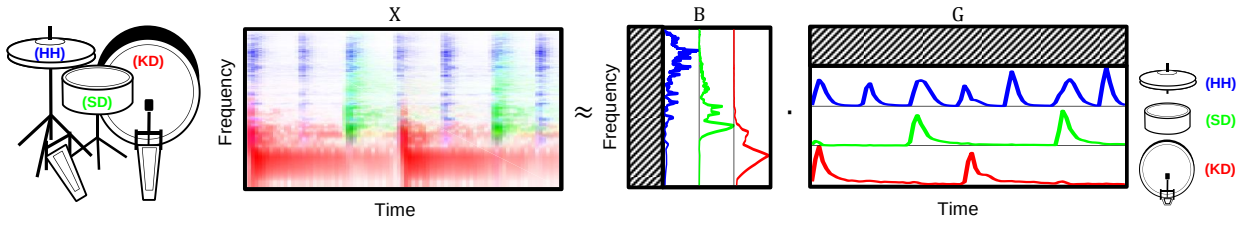


Fig. 5: Illustration of an NMF-based ADT system. The individual drum instruments appear in the same order as in Fig. 3.

when R_{H} is larger than R_{D}. The modified NMF cost function is defined as:

$$\mathcal{L} = D_{\mathrm{KL}}(\mathrm{X} \mid \gamma\tilde{\mathrm{X}}_{\mathrm{D}} + \delta\tilde{\mathrm{X}}_{\mathrm{H}}). \tag{6}$$

The matrices B_{H}, G_{H}, and G_{D} will be updated according to the following update rules:

$$\mathrm{B}_{\mathrm{H}} \leftarrow \mathrm{B}_{\mathrm{H}} \odot \frac{\frac{\mathrm{X}}{(\gamma\tilde{\mathrm{X}}_{\mathrm{D}}+\delta\tilde{\mathrm{X}}_{\mathrm{H}})} \cdot \mathrm{G}_{\mathrm{H}}^{\top}}{\mathrm{J} \cdot \mathrm{G}_{\mathrm{H}}^{\top}}, \tag{7}$$

$$\mathrm{G}_{\mathrm{D}} \leftarrow \mathrm{G}_{\mathrm{D}} \odot \frac{\mathrm{B}_{\mathrm{D}}^{\top} \cdot \frac{\mathrm{X}}{(\gamma\tilde{\mathrm{X}}_{\mathrm{D}}+\delta\tilde{\mathrm{X}}_{\mathrm{H}})}}{\mathrm{B}_{\mathrm{D}}^{\top} \cdot \mathrm{J}}, \tag{8}$$

$$\mathrm{G}_{\mathrm{H}} \leftarrow \mathrm{G}_{\mathrm{H}} \odot \frac{\mathrm{B}_{\mathrm{H}}^{\top} \cdot \frac{\mathrm{X}}{(\gamma\tilde{\mathrm{X}}_{\mathrm{D}}+\delta\tilde{\mathrm{X}}_{\mathrm{H}})}}{\mathrm{B}_{\mathrm{H}}^{\top} \cdot \mathrm{J}}. \tag{9}$$

Note that the algorithm reduces to the FNMF approach as described in Sect. V-B when $R_{\mathrm{H}} = 0$.

To further improve the pre-defined drum dictionary B_{D}, two template adaptation methods are introduced in [74]. In the first method (referred to as AM1), the drum dictionary B_{D} is updated based on evaluating the cross-correlation between the activations G_{H} and G_{D}. PFNMF starts by randomly initializing B_{H} with R_{H} components. Although B_{H} tends to adapt to the harmonic content, it may still absorb spectral magnitude of the drum sounds, which leads to unwanted cross-talk artifacts between G_{H} and G_{D}, generating less pronounced activations in G_{D}. However, these harmonic templates may also provide complementary information to the original drum templates. To identify these entries, the normalized cross-correlation between G_{H} and G_{D} for each individual drum is computed as:

$$\rho_{x,y} = \frac{\langle x, y\rangle}{\|x\|_2 \cdot \|y\|_2}, \tag{10}$$

where $\langle\cdot,\cdot\rangle$ denotes the inner product and $\|\cdot\|_2$ is the Euclidean norm. Furthermore, x and y represent different pairs of activation vectors (e.g., $x = \mathrm{G}_{\mathrm{D}}(r_1, :)$, $y = \mathrm{G}_{\mathrm{H}}(r_2, :)$, with $r_1 \in [1 : R_{\mathrm{D}}], r_2 \in [1 : R_{\mathrm{H}}]$). A threshold ρ_{thr} is defined for identification of relevant entries, and each drum template $\mathrm{B}_{\mathrm{D}}(:, r)$ is updated after Eq. (7) via:

$$\mathrm{B}_{\mathrm{D}}(:, r) \leftarrow (1-\alpha)\mathrm{B}_{\mathrm{D}}(:, r) + \alpha\frac{1}{|S|}\sum_{i\in S}^{S}(\rho_{x,y}(i)\mathrm{B}_{\mathrm{H}}(:, i)). \tag{11}$$

Here, $S \subset [1 : R_{\mathrm{H}}]$ denotes the subset of component indices whose corresponding activations fulfill $\rho_{x,y} \geq \rho_{\mathrm{thr}}$ and $|S|$ is the cardinality of this subset. In other words, the right-most term in this equation represents a weighted combination of templates from the harmonic dictionary that potentially contribute to the drums. A high threshold ρ_{thr} leads to minimal adaptation of the initial $\mathrm{B}_{\mathrm{D}}(:, r)$, whereas a low threshold leads to strong adaptation. The amount of adaptation also depends on the blending parameter $\alpha = \frac{1}{2^{\ell}}$, which decreases as the iteration index ℓ increases.

In [74], the second method (referred to as AM2) allows adaptation of the drum templates B_{D} by alternatively fixing B_{D} and G_{D} during the decomposition process. The adaptation process starts by fixing B_{D}, and PFNMF will try to fit the best activation G_{D} to approximate the drum part in the music. Once G_{D} is determined, a new iteration of PFNMF is started by fixing G_{D}, while B_{D}, B_{H} and G_{H} are updated. This modification will guide the algorithm to fit better drum templates based on the detected activation G_{D}. The update rule for B_{D} is as follows:

$$\mathrm{B}_{\mathrm{D}} \leftarrow \mathrm{B}_{\mathrm{D}} \odot \frac{\frac{\mathrm{X}}{(\gamma\tilde{\mathrm{X}}_{\mathrm{D}}+\delta\tilde{\mathrm{X}}_{\mathrm{H}})} \cdot \mathrm{G}_{\mathrm{D}}^{\top}}{\mathrm{J} \cdot \mathrm{G}_{\mathrm{D}}^{\top}} \tag{12}$$

Both methods have the same criterion to stop iterating when the error between two consecutive iterations changes by less than 0.1% or the number of iterations exceeds 20. In our experiments, the adaptation process typically converges after 5–10 iterations.

D. Semi-Adaptive NMF (SANMF)

An alternative approach for combining meaningful initialization with adaptability is to allow the spectral bases in B to deviate from their initial shape with increasing iteration count. Dittmar and Gärtner [69] proposed to enforce this behavior by blending between the latest update of B obtained from Eq. (4) and the fixed initial dictionary denoted this as $\overline{\mathrm{B}}$:

$$\mathrm{B} \leftarrow (1-\alpha) \cdot \overline{\mathrm{B}} + \alpha \cdot \mathrm{B}. \tag{13}$$

The blending parameter α depends on the ratio of the current iteration count ℓ to iteration limit L taken to the power of β:

$$\alpha = \left(\frac{\ell}{L}\right)^{\beta}. \tag{14}$$

Thus, only small adaptations of the NMF components are allowed early on, whereas stronger adaptation are allowed in later iterations. The larger the parameter β, the longer one attenuates the influence of Eq. (4) on B.

Note that both Eq. (11) and Eq. (13) are ad-hoc updates and the convergence is not always guaranteed. However, in practice, these update rules generally converge in a reasonable number

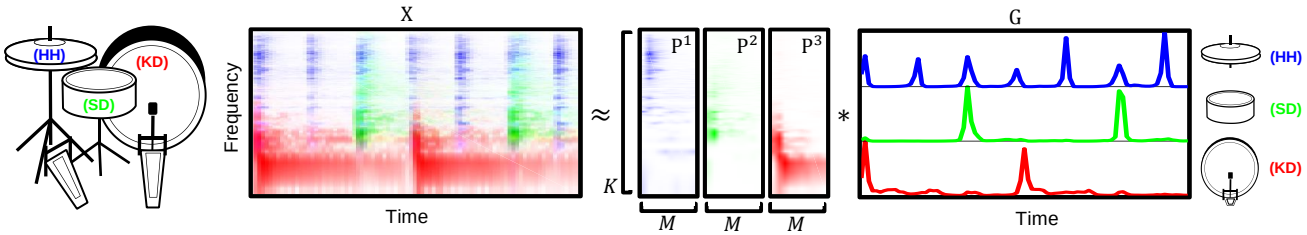


Fig. 6: Illustration of an NMFD-based ADT system.

of iterations. More details and experimental behaviors can be found in the original papers [69], [74]
.

E. Non-Negative Matrix Factor Deconvolution (NMFD)

The different NMF methods presented so far assumed that one template per drum instrument is sufficient to model temporal dynamics of drum sounds. However, we indicated already in Sect. I-A that certain drum instruments may generate complex, time-varying patterns when being struck. This is in line with the findings of [62], [63], where separate NMF templates for attack and decay of a drum sound event are used. As an alternative to that, previous works (such as [13], [65], [70], [95], [96]) successfully applied NMFD, a convolutive version of NMF, for drum transcription and drum sound separation.

As has been discussed in the above-mentioned publications, the NMFD model assumes that all drum sound events occurring in the mixture can be explained by a prototype event that acts as an impulse response to some impulse-like activation (e.g., striking a particular drum). In Figure 6, we illustrate this by introducing $R = 3$ prototype magnitude spectrograms $\mathrm{P}^r \in \mathbb{R}_{\geq 0}^{K \times M}$. Each P^r can be directly interpreted as a spectrogram pattern consisting of $M \ll T$ spectral frames. Each pattern is convolved with the corresponding row of G, yielding a convolutive approximation of X.

Mathematically, this can be formalized by grouping the above-mentioned patterns into a pattern tensor $\mathrm{P} \in \mathbb{R}_{\geq 0}^{K \times R \times M}$. In short notation, the slice of the tensor which refers to the r^{th} pattern is $\mathrm{P}^r := \mathrm{P}(:, r, :)$; whereas $\mathrm{P}_m := \mathrm{P}(:, :, m)$ refers to the m^{th} frame index simultaneously in all patterns. The convolutive spectrogram approximation $\mathrm{X} \approx \tilde{\mathrm{X}}$ is modeled as:

$$\tilde{\mathrm{X}} := \sum_{m=0}^{M-1} \mathrm{P}_m \cdot \overset{m\rightarrow}{\mathrm{G}}, \tag{15}$$

where $\overset{m\rightarrow}{(\cdot)}$ denotes a frame shift operator (explained in [95]). Similar to NMF, both P and G are suitably initialized. Subsequently, they are iteratively updated to minimize a cost function between the convolutive approximation $\tilde{\mathrm{X}}$ and X. According to [95], the update-rules extending Eq. (4) and Eq. (5) are given by:

$$\mathrm{P}_m \leftarrow \mathrm{P}_m \odot \frac{\frac{\mathrm{X}}{\tilde{\mathrm{X}}} \cdot \left(\overset{m\rightarrow}{\mathrm{G}}\right)^\top}{\mathrm{J} \cdot \left(\overset{m\rightarrow}{\mathrm{G}}\right)^\top} \tag{16}$$

$$\mathrm{G} \leftarrow \mathrm{G} \odot \frac{\mathrm{P}_m^\top \cdot \overset{\leftarrow m}{\left[\frac{\mathrm{X}}{\tilde{\mathrm{X}}}\right]}}{\mathrm{P}_m^\top \cdot \mathrm{J}} \tag{17}$$

for $m \in [0 : M-1]$.

As can be seen in Fig. 6, the NMFD-based activations in G exhibit a more spiky, impulse-like shape compared to the ones resulting from NMF in Fig. 5. As said before, this is a desirable property since it alleviates the **ES** step. The peaks are more concentrated since the P^r have the capability to better model the decay part of the drum sound events, thus attenuating the level of the activations during the decay phase. However, if the pattern length M is set too high, the increased expressiveness is also a potential drawback of NMFD. As discussed in [65], it may happen that the NMFD fails to untangle the underlying drum sounds, and instead captures sequences of drum strokes. For reasonable M (see Table IV), the learned P^r typically resemble spectrogram snippets averaged from all instances of the target drum sound occurring in the signal (as shown in the center panel of Fig. 6).

Note that NMFD is conceptually similar to the classic AdaMa method [42]–[44]. The typical alternation between drum detection and drum template refinement used by AdaMa is also entailed in the update rules for NMFD activations and templates. In contrast to AdaMa, no explicit decision making about the acceptance of drum sound candidates is required during NMFD updates, so that hard decisions can be omitted.

VI. RNN-Based ADT Systems

In this section, we provide more details of the different ADT systems based on recurrent variants of DNNs, called RNNs. Fig. 7 illustrates the basic concept behind ADT with RNNs. In contrast to the NMF-based systems, the mixture spectrogram X is processed as a time-series in a frame-wise fashion, i.e., we insert each individual spectral frame x^t sequentially into a trained RNN. If an input frame corresponds to the start of a drum sound event, it should ideally lead to a spiky, impulse-like activation at the RNNs' output as shown in Fig. 7e. In order to explain the necessary training steps enabling this desired input-output behavior, a few basics of DNN training are first reviewed.

A. DNN Training

As briefly explained in Sect. II-E, DNNs are networks consisting of linear, learnable parameters (weights and biases) and fixed non-linearities. These essential building blocks are usually organized in layers. For our concrete ADT tasks, we use spectral slices x^t of X as input to the first layer. Processing the input data in the first layer is interpreted as transformation to a more abstract representation, which in turn is used as input to the subsequent layer. Ideally, when the data has been processed by all layers, the neurons in the network's output layer should generate activation functions of the assigned drum instruments, as shown in Fig. 7. This is achieved by training the network with pairs of typical input data and target output data and automatically adjusting the learnable parameters towards the desired behavior. In our ADT scenario, the target output is typically generated from ground-truth transcriptions of the training data. For each of the considered drum instruments, frames corresponding to the start of a drum sound event are labeled as 1 and the remaining frames as 0 (as shown in Fig. 3c). The trained DNN should then produce similar activation functions when provided with the spectrogram input data of previously unseen drum mixtures.

Mathematically, the input-output behavior of a single network-layer can be formalized as

$$\mathrm{h} = \phi\left(\mathrm{W} \cdot \mathrm{x} + \mathrm{b}\right), \tag{18}$$

where $\mathrm{W} \in \mathbb{R}^{D \times K}$ is the weight matrix and $\mathrm{b} \in \mathbb{R}^D$ is the bias vector. The non-linearity $\phi(\cdot)$ is applied in an element-wise fashion to yield the layers' output $\mathrm{h} \in \mathbb{R}^D$. A variety of non-linearities are used in the literature, the most common ones being hyperbolic tangent ($\tanh$), sigmoid (σ), and rectified-linear units (ReLU). The meta-parameter $D \in \mathbb{N}$ determines the number of neurons per layer and is also referred to as layer width. Sticking to our ADT example of detecting KD, SD, HH sound events using just a single network layer, $D = 3$ would be a natural choice.

In accordance to the literature, we denote the entirety of network parameters as the set Θ, such that $\mathrm{W} \subseteq \Theta$ and $\mathrm{b} \subseteq \Theta$. During training, the parameter set is adapted so that the DNN produces the desired input-output behavior as specified by the training data. In the following, we denote the ground-truth target output as y and the output delivered by the network as $\hat{\mathrm{y}}$. For example, one has $\hat{\mathrm{y}} = \mathrm{h}$ for the simple, one-layer network presented above.

The parameters Θ need to be suitably initialized and can then be iteratively optimized by *gradient descent* [97]. For the optimization, one needs a cost function (often called loss function) $\mathcal{L}$ that measures the deviation between the network output $\hat{\mathrm{y}}$ and the target output y. A popular choice is cross-entropy:

$$\mathcal{L} = \frac{1}{D} \sum_{d=1}^{D} \left(\mathrm{y}_d \log \hat{\mathrm{y}}_d + (1 - \mathrm{y}_d) \log(1 - \hat{\mathrm{y}}_d)\right). \tag{19}$$

From this, the gradient $\mathcal{G}$ of the cost function with respect to the network parameters Θ needs to be determined. Then, the update of the network parameters is given by

$$\Theta \leftarrow \Theta - \mu \cdot \mathcal{G}. \tag{20}$$

The meta-parameter μ, a small positive constant, is called learning rate. As with the NMF-based ADT methods, the parameter updates are iterated for L epochs.

In contrast to our simplified example, DNNs are usually a cascade of many layers with individually trainable weights and biases. Although this seems to complicate the derivation of the gradient $\mathcal{G}$, the layered architecture of DNNs allows the use of the *backpropagation* algorithm [97] to efficiently calculate gradients for the parameters. In practice, this is usually achieved by using automatic differentiation libraries (e.g.,Theano, TensorFlow, etc.).

There are different approaches to utilize training data in this process: using the full dataset (Batch Gradient Descent, BGD), a single data point (Stochastic Gradient Descent, SGD), or a small portion of data points (Mini-Batch Gradient Descent, MBGD) for one update. To accelerate the convergence of gradient descent and to avoid getting stuck in local minima, several modifications have been proposed. Momentum approaches use past update values of the gradient to speed up convergence in problematic areas of the loss function $\mathcal{L}$ (e.g., SGD with momentum [97] and Nesterov accelerated gradient [98]). Adaptive learning rate methods adjust the parameter μ according to the history of past gradients (e.g., Adagrad [99], Adadelta [100], RMSprop [101], and Adam [102]).

B. Basic RNN Model (RNN)

In the following sections, four RNN-based ADT systems proposed in the literature [76]–[78], [103] will be discussed in detail. Their differences with respect to network configuration, cell architecture, and training strategy will be explained in the corresponding subsections.

RNNs represent an extension of DNNs featuring additional recurrent connections within each layer. The recurrent connections provide the single layers with the previous time step's outputs as additional inputs. The diagram of Fig. 7b visualizes this concept by a feedback connection from a neuron's output to its input. The equation for the output of an RNN layer at time step t is given by

$$\mathrm{h}^t = \phi\left(\mathrm{W} \cdot \left[\mathrm{x}^t, \mathrm{h}^{t-1}\right] + \mathrm{b}\right), \tag{21}$$

where $[:,:]$ denotes concatenation. Furthermore, W and b represent the appropriately sized weight matrix and biases vector, while x^t is the current input to the layer and h^{t-1} is the output from the previous time step of the same layer. In case of RNNs with several hidden layers, the output h^t is interpreted as input to the next hidden layer. The feedback of the outputs within the hidden layer acts as a simple form of memory and makes RNNs suitable for dealing with time series such as the sequence of spectral frames x^t in our spectrogram X.

An algorithm called *Back-Propagation Through Time (BPTT)* [104] is utilized to train RNNs, during which the

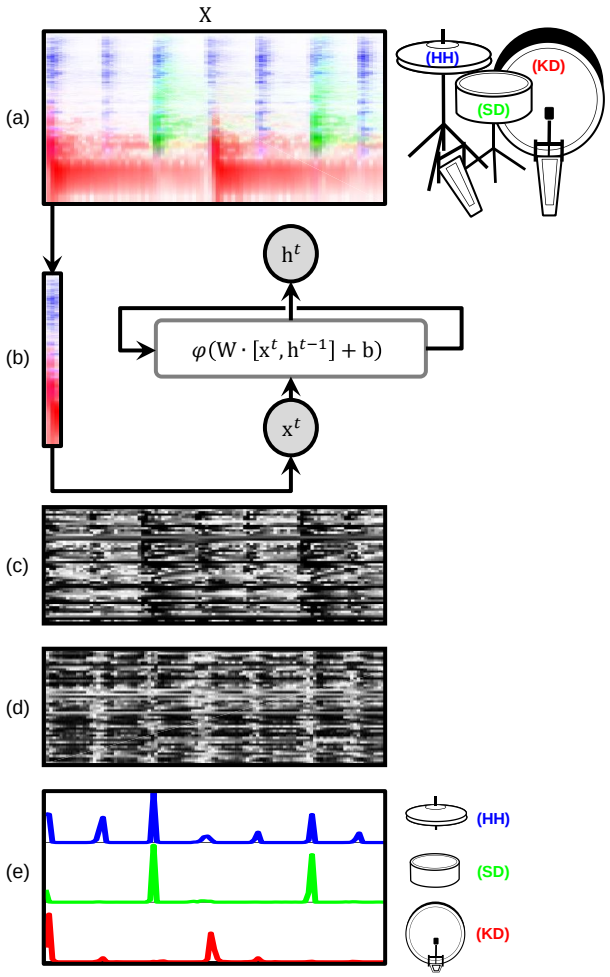


Fig. 7: Illustration of an RNN-based ADT system. **(a)** Spectrogram of the drum mixture. **(b)** Spectrogram frames are sequentially used as input features for a pretrained RNN. **(c)** Activations of the first hidden layer. **(d)** Activations of the second hidden layer. **(e)** Activations of the output layer.

network is thought of being unfolded in time for the length of the time series sequence. Unfolded RNNs become very deep networks, depending on the sequence length used for training. Since deep networks are harder to train, often only subsequences of the time series data are used for training.

In Fig. 7c and Fig. 7d, we show the hidden layer activations in a trained RNN. Darker shades of gray encode higher absolute activation. On closer inspection, some structure is visible as the activations tend to be stronger simultaneously to drum sound events occurring in the input. Finally, Fig. 7e displays the output activations according to our example drum recording. The output activations nicely indicate the onset times of drum sound events. For our example signal, the RNN-based activations are even more pronounced and spiky than the ones obtained via NMFD (cf. Fig. 6).

For the evaluation in Sect. VII, we use a simple baseline RNN, similar to the plain RNNs in [76], [78]. The meta-parameters used in our experiments are given in Table IV.

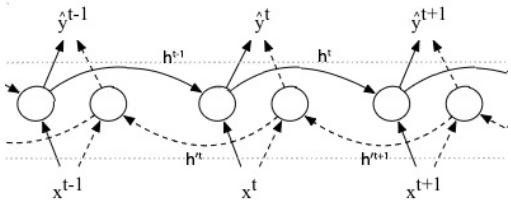


Fig. 8: An overview of an unfolded bidirectional RNN. The solid (forward) connections are also found in a standard RNN while the bidirectional RNN contains additional backward connections (dashed arrows). x^t and $\hat{y}^t$ are the inputs and outputs at time step t, with the circles representing the layers of the network.

C. Bidirectional RNNs (tanhB)

Southall et al. [78] introduced a system based on Bidirectional RNNs (BRNN) [105] for ADT. BRNN layers consist of two RNN sub-layers, one with recurrent connections in forward direction ($t-1 \rightarrow t$) and the other with recurrent connections in backward direction ($t+1 \rightarrow t$) as shown in Fig. 8. These allow the network to take past as well as future information into consideration for the output at time step t, which has been shown to be beneficial for many different tasks. As a downside of BRNNs, the entire sequence to be processed must be available in advance, making them generally unsuitable for real-time applications. By using small subsequences of the input stream it is possible to partly circumvent this issue.

The network configuration for the BRNNs used in [78] is given in Table IV. Each drum instrument under observation is treated as an independent classification problem using separate neural networks with softmax output layers. This approach allows to easily remove and add additional observed drum instrumentation.

D. RNNs with Label Time-Shift (ReLUts)

Vogl et al. [76] confirmed that BRNNs perform better than RNNs, but also showed that equal results can be achieved with RNNs using a label time-shift (25 ms). For this, all drum instrument annotation labels are shifted in time +25 ms (for a more detailed explanation see [76]). This shift allows an RNN to access information before and after the true start of drum sound events. One major benefit of using time shifts (instead of BRNNs) is that the method enables online application (with only a short delay). The network transcribes all three drum instruments using a sigmoid output layer with three neurons. This approach exploits the advantages of Multi-Task Learning (MTL) [106] by using a common model for different tasks which can improve overall performance. The meta-parameters of the network configuration are given in Table IV.

E. Long Short-Term Memory (LSTM) (lstmpB)

In addition to recurrent connections, LSTM cells [107] feature an internal memory (in the following denoted as c), which allows the network to learn long-term dependencies. The internal memory is accessed and updated using three gates (input gate i, forget gate f, and output gate o) controlled by

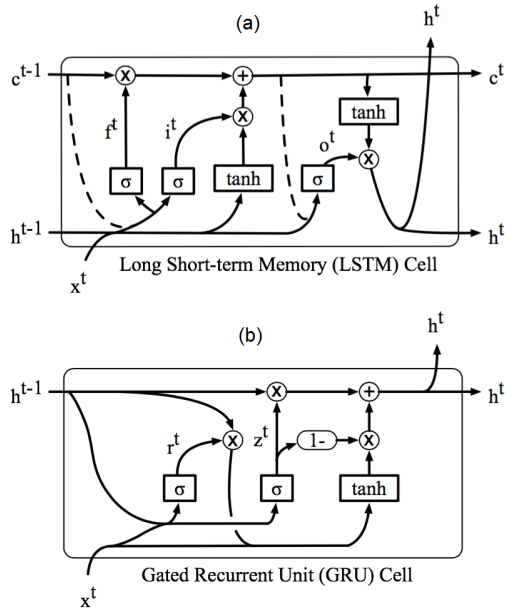


Fig. 9: Overview of LSTMP (a) and GRU (b) cell architectures. Converging connections represent concatenation of the respective data. Diverging connections represent copies of the same matrix. Dashed lines in the LSTM cell represent peephole connections for LSTMPs. The application of weights and biases is omitted for simplicity and the output arrows show connections to both the next layer and time step.

the input x^t, the hidden state h^{t-1} and, in case of LSTMs with peephole connections (LSTMPs), the cell memory c. The inclusion of c as a gate input allows the long-term dependencies stored within the cell memory to influence the flow of information through the gates. The model for an RNN layer with LSTMP architecture is specified as follows (see also Fig. 9a):

$$\mathrm{i}^t = \sigma(\mathrm{W}_\mathrm{i} \cdot \left[\mathrm{x}^t, \mathrm{h}^{t-1}, \mathrm{c}^{t-1}\right] + \mathrm{b}_\mathrm{i}), \tag{22}$$

$$\mathrm{f}^t = \sigma(\mathrm{W}_\mathrm{f} \cdot \left[\mathrm{x}^t, \mathrm{h}^{t-1}, \mathrm{c}^{t-1}\right] + \mathrm{b}_\mathrm{f}), \tag{23}$$

$$\tilde{\mathrm{c}}^t = \tanh(\mathrm{W}_\mathrm{c} \cdot \left[\mathrm{x}^t, \mathrm{h}^{t-1}, \mathrm{c}^{t-1}\right] + \mathrm{b}_\mathrm{c}), \tag{24}$$

$$\mathrm{c}^t = \mathrm{f}^t \odot \mathrm{c}^{t-1} + \mathrm{i}^t \odot \tilde{\mathrm{c}}^t, \tag{25}$$

$$\mathrm{o}^t = \sigma(\mathrm{W}_\mathrm{o} \cdot \left[\mathrm{x}^t, \mathrm{h}^{t-1}, \mathrm{c}^t\right] + \mathrm{b}_\mathrm{o}), \tag{26}$$

$$\mathrm{h}^t = \mathrm{o}^t \odot \tanh(\mathrm{c}^t). \tag{27}$$

In these equations, the subscripts are used to denote to which of the internal gates the weights and biases are associated to. In the work of Southall et al. [80] bidirectional LSTMs with peephole connections (BLSTMP) are used in an architecture similar to [78]. The corresponding meta-parameters of the network configuration are given in Table IV.

F. Gated Recurrent Unit (GRU) (GRUts)

Similar to LSTMPs, Gated Recurrent Units (GRU) [108] can be seen as a modification of standard LSTMs. GRUs have a significantly lower number of parameters compared to LSTMs. This is achieved by reducing the number of gates, using only an update gate z and a reset gate r, as well as merging the memory and the hidden state (h^{t-1}). The model for an RNN layer with GRU architecture is specified in the following equations (see also Fig. 9b):

$$\mathrm{z}^t = \sigma(\mathrm{W}_\mathrm{z} \cdot \left[\mathrm{x}^t, \mathrm{h}^{t-1}\right] + \mathrm{b}_\mathrm{z}), \tag{28}$$

$$\mathrm{r}^t = \sigma(\mathrm{W}_\mathrm{r} \cdot \left[\mathrm{x}^t, \mathrm{h}^{t-1}\right] + \mathrm{b}_\mathrm{r}), \tag{29}$$

$$\tilde{\mathrm{h}}^t = \tanh(\mathrm{W}_\mathrm{h} \cdot \left[\mathrm{x}^t, \mathrm{r}^t \odot \mathrm{h}^{t-1}\right] + \mathrm{b}_\mathrm{h}), \tag{30}$$

$$\mathrm{h}^t = \mathrm{z}^t \odot \mathrm{h}^{t-1} + (1 - \mathrm{z}^t) \odot \tilde{\mathrm{h}}^t. \tag{31}$$

In [77], Vogl et al. implement RNNs using GRUs combined with label time-shift (30 ms). The corresponding meta-parameters of the network configuration are given in Table IV.

VII. Evaluation

In this section, we provide the details of the evaluation we conducted with the state-of-the-art ADT systems introduced in the last two sections. Specifically, we implemented ten systems from publications within the last five years (cf. Table II) in order to assess and compare their capabilities in a unified experimental framework. The selected algorithms are listed in Table IV, where we refer the reader to the original papers as well as the corresponding paragraphs in this article. Whenever implementational details are omitted, they are equivalent to the descriptions in the original works. The source code of the implemented systems can be found online.[6,7,8]

A. Evaluation Datasets

As indicated earlier, we used two publicly available corpora of drum recordings for our experiments. We processed and partitioned the available corpora in such a way that they directly correspond to the three most relevant ADT tasks introduced in Sect. I-C. In particular, these are Drum Transcription of Drum-only recordings (DTD), Drum Transcription in the presence of Percussion (DTP), and Drum Transcription in the presence of Melodic instruments (DTM). Table V gives an overview of the content of these datasets; additional information is provided in the following paragraphs.

D-DTD: This dataset is intended to evaluate DTD performance, i.e., transcription of recordings containing only the three drum instruments KD, SD, HH. A real-world application scenario for this task would be the transcription of single track drum recordings in a studio. This dataset uses the latest version of the IDMT-SMT-Drums corpus [69].

D-DTP: This dataset is intended to assess DTP performance, i.e., transcription of recordings containing other percussion instruments in addition to the drum instruments under observation. A user aiming to transcribe recordings of a large

[6]https://github.com/cwu307/NmfDrumToolbox, last accessed:10/02/2017
[7]https://github.com/CarlSouthall/ADTLib, last accessed:10/02/2017
[8]https://github.com/richard-vogl/dt_demo, last accessed:10/02/2017

TABLE IV: Overview of all implemented systems included in our evaluation.

Type	Abbrev.	Reference	Sect.	Parameters
NMF-based	SANMF	Dittmar and Gärtner [69]	V-D	$R = 3, L = 30, \beta = 4$
	NMFD	Lindsay-Smith et al. [65]	V-E	$R = 3, L = 30, M = 10$
	PFNMF	Wu and Lerch [74]	V-C	$R_{\mathrm{D}} = 3, R_{\mathrm{H}} = 10$ (DTD), $R_{\mathrm{H}} = 50$ (DTP & DTM), $L = 20$
	AM1	Wu and Lerch [74]	V-C	$R_{\mathrm{D}} = 3, R_{\mathrm{H}} = 10$ (DTD), $R_{\mathrm{H}} = 50$ (DTP & DTM), $L = 20$
	AM2	Wu and Lerch [74]	V-C	$R_{\mathrm{D}} = 3, R_{\mathrm{H}} = 10$ (DTD), $R_{\mathrm{H}} = 50$ (DTP & DTM), $L = 20$
RNN-based	RNN	Vogl et al. [76] Southall et al. [78]	VI-B	1 hidden layer, $D = 200$, tanh, RMSprop with initial $\mu = 0.005$, sigmoid outputs, bias init 0, mini-batch size = 8 sequences of length 100, weight init uniform ± 0.01
	tanhB	Southall et al. [78]	VI-C	2 hidden layers, $D = 50$, tanh, Adam with initial $\mu = 0.05$, softmax outputs, bias init 0, mini-batch size = 10 sequences of length 100, weight init uniform ± 1, dropout rate 0.25
	ReLUts	Vogl et al. [76]	VI-D	1 hidden layer, $D = 100$, ReLU, RMSprop with initial $\mu = 0.001$, sigmoid outputs, bias init 0, mini-batch size = 8 sequences of length 100, weight init uniform ± 0.01, dropout rate 0.2
	lstmpB	Southall et al. [80]	VI-E	2 hidden layers, $D = 50$, BLSTMP, Adam with initial $\mu = 0.05$, softmax outputs, bias init 0 mini-batch size = 10 sequences of length 100, weight init uniform ± 1, dropout rate 0.25
	GRUts	Vogl et al. [77]	VI-F	2 hidden layers, $D = 50$, GRU, RMSprop with initial $\mu = 0.007$, sigmoid outputs, bias init 0, mini-batch size = 8 sequences of length 100, weight init uniform ± 0.1, dropout rate 0.3

TABLE V: Overview of the three datasets used for our evaluation.

Dataset	Reference	Total #onsets	KD #onsets	SD #onsets	HH #onsets	Total #items	Avg. Dur.	Subset 1 Origin (#items)	Subset 2 Origin (#items)	Subset 3 Origin (#items)
D-DTD	IDMT-SMT-Drums [69]	8722	2309	1658	4755	104	15 s	D-DTD-1 RealDrum (20)	D-DTD-2 TechnoDrum (14)	D-DTD-3 WaveDrum (70)
D-DTP	ENST-Drums minus-one [87]	22391	6451	6722	9218	64	55 s	D-DTP-1 Drummer1 (21)	D-DTP-2 Drummer2 (22)	D-DTP-3 Drummer3 (21)
D-DTM	ENST-Drums accompanied [87]	22391	6451	6722	9218	64	55 s	D-DTM-1 Drummer1 (21)	D-DTM-2 Drummer2 (22)	D-DTM-3 Drummer3 (21)

drum kit but only being interested in a subset of the drum instruments is a real-world example of this scenario. Therefore, we use all items contained in the ENST-Drums minus-one dataset [87]. In order to use this information for DTP evaluation, we only consider the annotations for KD, SD, and HH for our performance metrics (see Sect. VII-C). In contrast to D-DTD, this set does not have training audio of isolated drum sound events for each recording, but only for the three different drum kits that have been used in the recordings. More detailed information about the content of this dataset is provided in the second row of Table V.

D-DTM: This set is intended to evaluate DTM performance, i.e., transcription of polyphonic music recordings containing a variety of melodic instruments in addition to the drum instruments under observation. This scenario represents transcription of full song recordings, which is the most demanding task but also the one with highest applicability to real-world music data. Again, we use all items contained in the ENST-Drums minus-one dataset. We combined accompaniment and drum tracks using a mixing ratio of $1/3$ and $2/3$, respectively. This ratio is chosen for consistency with prior work [19], [58], and is reasonable as confirmed by listening experiments. We can readily re-use the ground-truth transcriptions of D-DTP since the underlying drum recordings stay the same. We again focus on KD, SD, HH and interpret the melodic accompaniment and the additional percussion as interference making the DTM task the most challenging in our performance comparison.

As shown in the three rightmost columns of Table V, all three datasets come with a natural split into three subsets. For the IDMT-SMT-Drums corpus, the subsets correspond to the different origins of the drum recordings, namely acoustic drum kits (RealDrum), drum computers (TechnoDrum), and drum sampler software (WaveDrum). For the ENST-Drums corpus, the subsets correspond to three different session drummers, each one playing an individual acoustic drum kit. As layed out in Table V, we denote the individual subsets with the respective dataset name, followed by the suffix -1,-2, and -3. As an example, the subset named D-DTP-2 refers to the set of all drum recordings played by the second drummer in the ENST-Drums corpus. In the next section, we will explain why these different subsets are important for our evaluation.

B. Evaluation Strategies

The goal of our evaluation is to compare the attainable ADT performance of NMF-based and RNN-based systems within a common evaluation framework. As explained in Sect. V, all ADT systems employing NMF-variants require informed initialization of their spectral bases with averaged drum sound spectra. This step is essential and can be interpreted as some sort of training stage.

Similarly, all ADT systems employing RNN-variants require a training stage (see Sect. VI), where a large number of input feature vectors and target output vectors are presented to the network to adjust the internal parameters. Moreover, both families of algorithms belong to the cluster of Activation-Based Methods (**FR**, **AF**, **ES**), whose output activations have to undergo an **ES** stage, which we realize via peak picking. As

described in Sect. IV-C, the identification of peak candidates also depends on meta-parameters that have to be optimized.
In our evaluation, we follow the established standards used for evaluating machine learning algorithms. First and foremost, that means we have to partition the entirety of our data into disjoint sets used for training, validation, and testing. The training data is used to optimize the internal parameters of the selected ADT systems, the validation data is used to optimize hyper-parameters (i.e., the meta-parameters for peak-picking) and to prevent overfitting of the DNN models, while the test data is used to measure the performance on unseen data. Note that parameters of DNNs (i.e., number of neurons, number of layers, and activation functions) are kept the same as in their original publications and are thus not optimized during the process.
We pursue three evaluation strategies explained in the following paragraphs. In Table VI, we illustrate how the three strategies apply to the dataset D-DTD. The same principle then applies for the remaining two datasets D-DTP and D-DTM, the only difference being that the datasets need to be swapped.

Eval Random: This strategy evaluates the ADT performance within the "closed world" of each dataset D-DTD, D-DTP, and D-DTM individually. In order to maximize the diversity of the data, all items (regardless of the subset partitions) are randomly split into non-overlapping training, validation and testing set.

Eval Subset: This strategy also evaluates the ADT performance within the "closed world" of each dataset but using a three-fold subset cross-validation. To this end, each of the three subsets (see Table V) is evenly split into validation and testing sets. The union of all items contained in the remaining two subsets serves as training data. A single subset is used for the validation and testing set in order to maintain sufficient training data.

Eval Cross: This strategy evaluates ADT performance within the "open world" and the generalization capabilities of the systems across the different datasets. To this end, each of the datasets (in full) is used as the testing data for the systems trained, using the other two corresponding datasets, in the Eval Random evaluation strategy.

C. Parameters and Performance Metrics

The **FR** considered in our evaluation is computed via STFT with a blocksize of $N = 2048$ and a hopsize of $\frac{N}{4} = 512$. Since all items have a sampling rate of 44.1 kHz, the frequency resolution of the STFT is approximately 21.5 Hz and the temporal resolution is approx. 11.6 ms. As window function, we use a symmetric Hann-window of size N.
For performance metric, we use the standard F-measure as discussed in Sect. II-F with a tolerance window of 50 ms. This choice of tolerance window is consistent with many previous studies on ADT [69], [74], [78] and onset detection [93] (see Sect. II-F for more discussions on tolerance window). A reduction of the tolerance window, as shown in [65], generally leads to a degradation in performance.

TABLE VI: Summary of the three evaluation strategies applied to the dataset D-DTD (the same principle also applies for D-DTP and D-DTM by swapping them). The given percentages denote random selection of items contained in the respective dataset or subset. The curly brackets denote the union of the enclosed subsets.

Evaluation Strategy	Training	Validation	Testing
Eval Random	70% D-DTD	15% D-DTD	15% D-DTD
Eval Subset	{D-DTD-2, D-DTD-3}	50% D-DTD-1	50% D-DTD-1
	{D-DTD-1, D-DTD-3}	50% D-DTD-2	50% D-DTD-2
	{D-DTD-1, D-DTD-2}	50% D-DTD-3	50% D-DTD-3
Eval Cross	70% D-DTP	15% D-DTP	100% D-DTD
	70% D-DTM	15% D-DTM	100% D-DTD

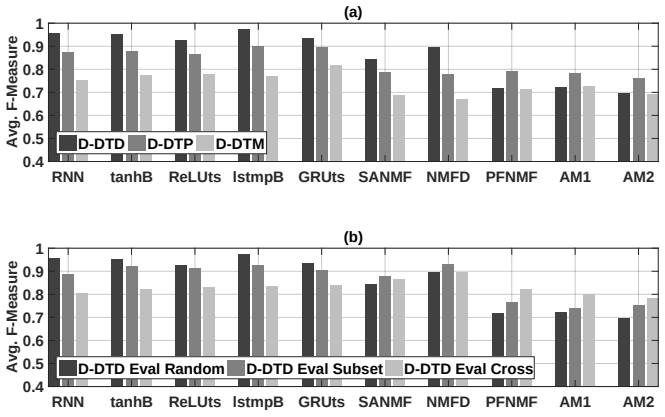


Fig. 10: Summary of our evaluation. **(a)** F-measure for the ADT task for different datasets and different algorithms using the *Random* scenario. **(b)** The F-measure similar to (a). This time, however, different evaluation strategies are used with D-DTD dataset only.

VIII. Results and Discussions

To highlight the essence of our evaluation, Sect. VIII-A yields a top-down summary of the main findings. Sect. VIII-B and Sect. VIII-C provide a more detailed discussion. For the sake of completeness and reproducibility, the table with all evaluation results can be found on our complementary website[9].

A. Results Summary

In Fig. 10a, we assess how well the selected systems can cope with ADT tasks of increasing complexity. To this end, we show the average F-measure across our three datasets in the evaluation scenario Eval Random. This evaluation scenario provides the most ideal case, in which the training data is likely to be representative of the test data. As expected, the

[9]http://www.audiolabs-erlangen.de/resources/MIR/2017-DrumTranscription-Survey/, last accessed 2017/10/02

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2018.2830113, IEEE/ACM Transactions on Audio, Speech, and Language Processing

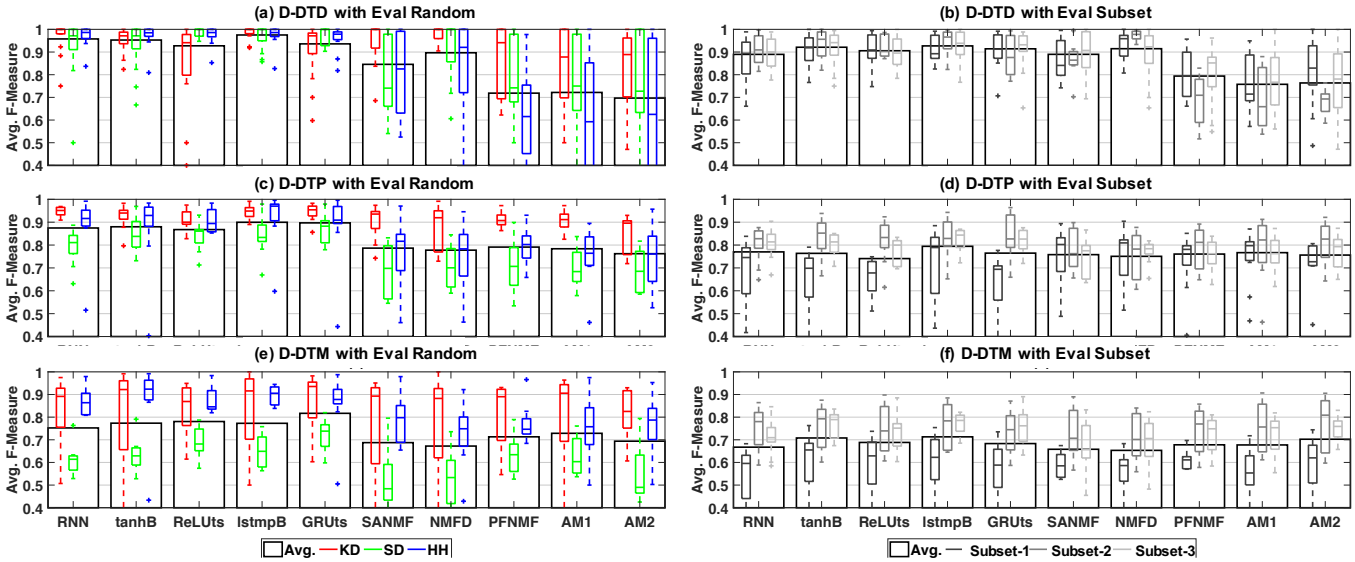


Fig. 11: Evaluation results (Eval Random and Eval Subset) of dataset (a)(b) D-DTD (c)(d) D-DTP (e)(f) D-DTM

highest results are achieved with the least complex dataset D-DTD. From the family of RNN-based methods, **lstmpB** is the best-performing system with approximately 0.97 F-measure, i.e., almost perfectly solving the DTD task. From the family of NMF-based methods, **NMFD** scores best but falls short of all RNN-based systems. For the more challenging dataset D-DTP, the performance of all systems drops, except for **PFNMF** variants. Although they do not surpass the RNN-systems, they seem to have an advantage when dealing with additional percussion instruments. Finally, for the most challenging D-DTM dataset, **GRUts** is the only system that surpasses 0.8 F-Measure. Once again, the performance of all other systems deteriorates. Only the **PFNMF**-variants can partly compensate for the performance drop, with **AM1** scoring best among the NMF-methods.

In Fig. 10b, we assess the generalization capabilities of the evaluated systems. To this end, we stay with the dataset D-DTD and sweep through our evaluation scenarios. This dataset is the simplest among the three, which gives the measure of the best case scenario. We observe that the RNN-based systems are quite susceptible to mismatches in the training data. Performing RNN-training on the Eval Subset data already leads to a slight decrease. The performance drop is even more pronounced when the training is based on the Eval Cross data. In contrast, the NMF-based methods either stay stable or improve their performance through the different training scenarios. This can be attributed to the adaptivity inherent to NMF.

It should be noted that we present here the averaged results, i.e., the Eval Subset training results are averaged over the test splits of D-DTD-1, D-DTD-2, and D-DTD-3. Likewise, the Eval Cross training results are averaged over training with D-DTP and D-DTM. More detailed results are provided in Fig. 11 to Fig. 12.

Based on the above results, the following trends can be concluded: First, RNN-based systems generally outperform NMF-based systems. Even the basic RNN system (included as a baseline) performs on a par with the other systems in most cases. Since RNNs exploit the temporal dependencies in the input data, they have the potential to learn the underlying structure and temporal context. However, for less challenging data, NMF-based system may provide competitive results without requiring a computationally expensive training session. Second, the margin between the strongest and weakest systems decreases as the signals get increasingly difficult. This result indicates the typical vulnerability against the interference of other instruments that is common for all state-of-the-art systems. Third, the differences between different training strategies are less pronounced for NMF-based systems, whereas for RNN-based systems, the performance drop from Eval Random over Eval Subset to Eval Cross is noticeable. Since Eval Random offers more diversity (i.e., more training examples similar to the ones in the test set), it is expected to be more advantageous for RNNs. On the contrary, when the test data contains unseen examples, RNNs become less reliable.

B. Eval Random vs. Eval Subset Results

In Fig. 11a to Fig. 11f, we depict the F-measure scores achieved across all three datasets. The results obtained via Eval Random are always presented in the left panels. In that case, the box plots summarize the statistics of individual results of KD, SD, and HH. The results obtained via Eval Subset are presented in the right panels, with the box plots summarizing the statistics of different subsets.

In Fig. 11a and Fig. 11b, it can be found that the two families of algorithms react differently under the different evaluation strategies. In Eval Random the best performing system is **lstmpB**; in Eval Subset the best performing system is **NMFD**. Additionally, for RNN-based systems, switching from Eval Random to Eval Subset decreases the overall performances; for NMF-based systems, however, the result is the exact opposite. In Fig. 11c, the best performing systems are **GRUts** and **lstmpB**. Similar to the D-DTD dataset, switching from Eval

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TASLP.2018.2830113, IEEE/ACM Transactions on Audio, Speech, and Language Processing

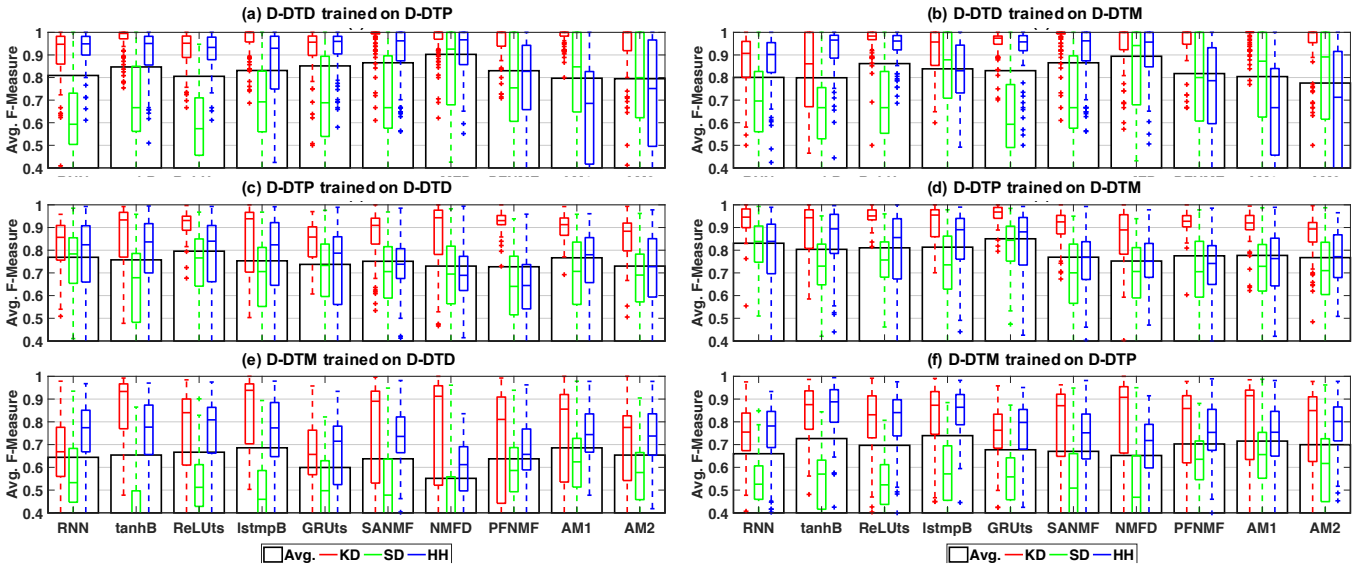


Fig. 12: Evaluation results (Eval Cross) of dataset (a)(b) D-DTD with systems trained on D-DTP and D-DTM (c)(d) D-DTP with systems trained on D-DTD and D-DTM (e)(f) D-DTM with systems trained on D-DTD and D-DTP

Random to Eval Subset , as shown in Fig. 11d, introduces a noticeable drop in the overall performances for RNN-based systems; for NMF-based systems, the discrepancy between the two evaluation strategies is relatively small. An interesting phenomenon is the steep performance-drop of the RNN-systems for subset D-DTP-1. This is possibly caused by the special sound characteristic of the drum kit in that subset, which is not well reflected in the other two subsets; this may imply the tendency of overfitting with RNN-systems. NMF-systems, on the other hand, adapt better on D-DTP-1. This is possibly due to their ability to separate superimposed sound sources.

In Fig. 11e, the results generally follow the same trend in Fig. 11c with a slightly inferior performance for all systems. Note that in Fig. 11f, the combination of dataset D-DTM and Eval Subset training is used, which represents a challenging evaluation scheme that is common in previous work [58], [74], [76], [78]. In this case, the best performing system is **lstmpB**. However, the gap between the best performing system and the others is marginal. Specifically, the NMF-based system **AM2** achieved similar performance as the RNN-based system **lstmpB**. Also, the performance drop for D-DTM-1 can be observed from all systems, showing that additional harmonic sounds are problematic to both RNN and NMF systems. All of the systems tend to achieve the highest performance on KD, may be due to its distinctive frequency range. On the other hand, all systems have difficulties with SD, this can be explained by the large spectral overlap between SD and the melodic instruments in the dataset D-DTM.

C. Eval Cross Results

In Fig. 12a to Fig. 12f, the results for our cross evaluation strategy are shown. By using each of the datasets D-DTD D-DTP and D-DTM as test data once, this evaluation strategy indicates the capability of the evaluated systems to generalize across different datasets. The error bar represents the standard deviation across different instruments.

Results using test data from the D-DTD dataset, is shown in Fig. 12a and Fig. 12b. The best performing system based on the averaged F-measure is **NMFD** for both training datasets D-DTP and D-DTM). Additionally, the differences between the two training scenarios seem to be small for most of the systems.

Fig. 12c and Fig. 12d are based on test data from the D-DTP dataset. When training with D-DTD the best performing system is **ReLUts**. When training with D-DTM the best performing system is **GRUts**. Comparing these two training datasets, D-DTM seems to lead to better performances for most of the systems.

Fig. 12e and Fig. 12f show the results when using test data from the D-DTM. Not surprisingly, using training data from D-DTP achieves slightly better results since the drum kits are the same in both the test and training dataset.

Based on the results, the following observations can be made. First, while RNN-based systems outperform NMF-based systems in many cases, the margin becomes small. In the most challenging case (D-DTM), NMF-based systems actually achieve a performance comparable to RNN-based methods, although on a low level. This finding is consistent with the results in Fig. 11f, in which the RNN and NMF-based systems performed similarly under the most challenging combination of evaluation scenario and test data. This indicates the advantage of the NMF-based systems, which is the generality for unseen data. Second, most of the systems tend to perform better when the test data is less complex than the training data. This result shows the benefits of having data with higher complexity (i.e., real-world data of polyphonic music), and it also implies the need for more representative datasets in order to make further progress in ADT research (see Sect. III-D). Third, the performance drop from D-DTD to D-DTP and D-DTM

indicates that all of the systems suffer from the presence of additional sounds, which could be due to the superimposed percussive sounds or harmonic sounds in the background. Further comparison of results between D-DTP and D-DTM confirms the influence from the harmonic sounds, and the gap between D-DTD and D-DTM shows that there is still plenty of room for improvements for all ADT systems.

IX. Conclusion

In this survey paper, we provided an up-to-date review of research in the field of automatic drum transcription over the last 15 years. This fills up the gap that existed since the previous survey [6] that had been published a decade ago, and it also contextualizes modern ADT systems that are based on the novel matrix factorization and deep learning approaches. Furthermore, we conducted a systematic evaluation of state-of-the-art systems on ADT. This evaluation yields a detailed analysis and comparison between various systems under well-controlled experimental conditions.

Based on our experiments, RNN-based methods seem to be the most promising approaches, and they are recommended when a large and diverse training dataset with high-quality annotations is available. NMF-based methods, on the other hand, provide decent performance with only little training data required; suitable for cases when large training datasets are not available. Generally speaking, reliable performances can be expected from the state-of-the-art systems for the DTD task; for DTP and especially DTM tasks, however, there is still plenty of room for future improvement.

In the following sections, we identify and summarize promising future directions in ADT research.

A. More Data

As highlighted in Sect.III-D, having a substantial collection of high-quality and representative data is the key to the success of data-driven approaches. ADT research, as one of many research areas that rely on publicly available data, is also in need of more data for making further progress. Having more annotated music available would provide the necessary diversity and complexity for training models that generalize well for real-world music recordings. Since creating human-annotated datasets is a labor-intensive task, an organized and distributed effort within the ADT research community should be highly encouraged. Also, as it is a common practice to record drums into multiple tracks, building multi-track drum datasets and exploiting the isolated drum information can be another interesting direction for future ADT research.

B. Public Evaluation

In addition to publicly available datasets, the research community also benefits from an open evaluation forum for sharing the latest technological advances, as exemplified by the Music Information Retrieval Evaluation eXchange (MIREX) [109]. Despite the continued success of MIREX, ADT is still a relatively underrepresented task. Recently, ADT research has seen a steady growth in the MIR community, and efforts have been made to revive the ADT MIREX task. However, active participation from the community is vital for the success of these efforts.

C. More Instruments

So far, most published approaches focus on only the three main drum instruments, namely the HH, SD, and KD. For certain applications, a wider range of instruments in the drum kits (e.g., tom-tom drums, cymbals, or electronic drum sounds), as well as other drum instruments (e.g., tablas, congas, or other percussive sounds) would be desirable. In the state-of-the-art systems evaluated in this paper, such as NMF-based and RNN-based methods, the extension is conceivable by adding more templates or neurons to account for extra instruments. Nevertheless, the viability of the existing methods for these instruments needs to be further assessed. Also, suitable datasets would be required in any case, which remains to be an open-ended issue at this moment.

D. More Dynamic Details

One of the shortcomings shared by most of the state-of-the-art systems is the ignorance of dynamics of the drum events. That is, the intensity (or loudness) of a drum event is usually ignored in favor of the simple and robust binary representation of the onsets. Activation-based methods provide curves which tend to be interpreted as onset intensities, but this information is usually not encoded in the output of the transcription. Since dynamics has a strong connection to playing techniques (as described in Sect.III-B) and expressivity, it would be a reasonable next step for ADT research.

E. Pre/Post-processing Strategies

Intuitively, ADT tasks should benefit from preprocessing techniques that suppress the irrelevant components and enhance the target drum sounds. In that regard, source separation methods (e.g., HPSS [23]) would be an ideal inclusion that might lead to better suited **FR** and overall performance. An example for such techniques is given in [44], where performance improvements for the AdaMa algorithm could be achieved when using *Harmonic Structure Suppression* to attenuate the influence of pitched instruments on the detection of KD and SD. However, other studies incorporating similar ideas report inconclusive results [19]. A common problem is that suppression of pitched instruments might lead to additional artifacts that can have a detrimental effect on the ADT performance.

Additionally, existing ADT systems including NMF-based (see Sect. V) and RNN-based (see Sect. VI) approaches implicitly perform source separation during the optimization process which reduces the need for such preprocessing. Nevertheless, with the latest developments in source separation techniques such as the contributions in Signal Separation Evaluation Campaign for Music (SiSEC MUS[10]), new strategies that

[10]https://www.sisec17.audiolabs-erlangen.de, last accessed 2018/04/10

are optimal for ADT tasks could be worth exploring. For post-processing, using **LM**s in ADT seems to be promising and currently under-explored, but the limitation regarding the availability of symbolic data should be taken into consideration (see next section).

F. Integration of Music Language Models

Current state-of-the-art ADT systems mainly focus on extracting the onset times of the drum events without taking into account the musical context. Specifically, most of the state-of-the-art systems are activation-based methods with a simple peak-picking process as the final step. While achieving decent results, these approaches do not benefit from high-level musical information. The integration of **LM**s (as mentioned in Sect.II-D) into ADT systems has been proposed in previous work [53]. However, results so far are below current systems without **LM**s. Furthermore, new types of **LM**s (e.g.,LSTMs) have not been tested for ADT. This is mainly due to the fact that the application of common **LM**s from the automatic speech recognition domain is not trivial, and large datasets for both audio and symbolic data for drums are not publicly available (as mentioned in Sect.II-D). Although the lack of large training datasets as well as the adaptation of ASR methods for music are a challenge, the integration of **LM**s in modern ADT approaches might be another direction that can potentially lead to a breakthrough in ADT.

G. Towards Full Transcripts

To obtain a complete transcription in the format of sheet music, more information, such as tempo, dynamics, playing styles, or time signatures are required in addition to onset times. This implies the importance of integrating various MIR systems to the processing chain of ADT systems in order to achieve the ultimate goal of full transcriptions. The research along this direction is still relatively sparse, however, the importance of this subject will increase as the MIR systems mature.

ADT is a research topic that is crucial to the understanding of rhythmic aspects of music, and has potential impact on broader areas such as music education and music production. We hope that this paper may serve as reference for continued research in the field of automatic drum transcription and automatic music transcription in general, leading towards the realization of intelligent music systems in the near future.

ACKNOWLEDGMENT

The authors would like to thank Telecom ParisTech for making the ENST-Drums dataset publicly available and Fraunhofer IDMT for making the IDMT-SMT-Drums dataset publicly available. Christian Dittmar and Meinard Müller are supported by the German Research Foundation (DFG-MU 2686/10-1). The International Audio Laboratories Erlangen is a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and the Fraunhofer-Institut für Integrierte Schaltungen IIS.
This work has been partly funded by the Austrian FFG under the BRIDGE 1 project *SmarterJam* (858514).

REFERENCES

[1] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri, "Automatic music transcription: challenges and future directions," *Journal of Intelligent Information Systems*, vol. 41, no. 3, pp. 407–434, 2013.
[2] J. Salamon, E. Gómez, D. P. W. Ellis, and G. Richard, "Melody Extraction from Polyphonic Music Signals: Approaches, applications, and challenges," *IEEE Signal Processing Magazine*, no. February 2014, pp. 118–134, 2014.
[3] G. Valenzise, L. Gerosa, M. Tagliasacchi, F. Antonacci, and A. Sarti, "Scream and gunshot detection and localization for audio-surveillance systems," in *IEEE Intl. Conf. on Advanced Video and Signal Based Surveillance (AVSS)*, London, UK, 2007, pp. 21–26.
[4] A. Mesaros, T. Heittola, A. Eronen, and T. Virtanen, "Acoustic event detection in real-life recordings," in *European Signal Processing Conf. (EUSIPCO)*, Aalborg, Denmark, 2010, pp. 1267–1271.
[5] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.
[6] D. FitzGerald and J. Paulus, "Unpitched percussion transcription," in *Signal Processing Methods for Music Transcription*, A. Klapuri and M. Davy, Eds. Springer, 2006, pp. 131–162.
[7] P. Chordia, "Segmentation and recognition of tabla strokes," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, 2005, pp. 107–114.
[8] M. Tian, A. Srinivasamurthy, M. Sandler, and X. Serra, "A study of instrument-wise onset detection in Beijing opera percussion ensembles," in *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2014, pp. 2178–2182.
[9] L. Thompson, S. Dixon, and M. Mauch, "Drum transcription via classification of bar-level rhythmic patterns," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, Taipei, Taiwan, October 2014, pp. 187–192.
[10] G. Dzhambazov, "Towards a drum transcription system aware of bar position," in *Proc. Audio Engineering Society Conf. on Semantic Audio (AES)*, London, UK, Jan 2014.
[11] R. Vogl, M. Dorfer, G. Widmer, and P. Knees, "Drum transcription via joint beat and drum modeling using convolutional recurrent neural networks," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, Suzhou, CN, Oct 2017, pp. 150–157.
[12] E. Kokkinis, A. Tsilfidis, T. Kostis, and K. Karamitas, "A new DSP tool for drum leakage suppression," in *Proc. Audio Engineering Society Convention (AES)*, New York, NJ, USA, 2013.
[13] C. Dittmar and M. Müller, "Reverse Engineering the Amen Break – Score-informed Separation and Restoration applied to Drum Recordings," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 9, pp. 1531–1543, 2016.
[14] M. Prockup, A. F. Ehmann, F. Gouyon, E. M. Schmidt, and Y. E. Kim, "Modeling musical rhythm at scale with the music genome project," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, New York, USA, Oct 2015.
[15] M. Leimeister, D. Gärtner, and C. Dittmar, "Rhythmic classification of electronic dance music," in *Proc. Audio Engineering Society Conf. on Semantic Audio (AES)*, London, UK, Jan 2014.
[16] M. Davies, G. Madison, P. Silva, and F. Gouyon, "The effect of microtiming deviations on the perception of groove in short rhythms," *Music Perception*, vol. 30, no. 5, pp. 497–510, 2013.
[17] C. Dittmar, M. Pfleiderer, and M. Müller, "Automated estimation of ride cymbal swing ratios in jazz recordings," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, Malaga, Spain, October 2015.
[18] C. Dittmar, M. Pfleiderer, S. Balke, and M. Müller, "A swingogram representation for tracking micro-rhythmic variation in jazz performances," *Journal of New Music Research*, vol. 47, no. 2, pp. 97–113, 2017.
[19] O. Gillet and G. Richard, "Transcription and separation of drum signals from polyphonic music," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 3, pp. 529–540, 2008.
[20] J. Paulus, "Signal processing methods for drum transcription and music structure analysis," Ph.D. dissertation, Tampere University of Technology, Tampere, Finland, 2009.

[21] A. Lerch, *An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics*. John Wiley & Sons, 2012.

[22] M. Müller, *Fundamentals of Music Processing*. Springer Verlag, 2015.

[23] D. FitzGerald, "Harmonic / percussive separation using median filtering," in *Proc. Intl. Conf. on Digital Audio Effects (DAFx)*, Graz, Austria, September 2010, pp. 246–253.

[24] W. A. Schloss, "On the Automatic Transcription of Percussive Music - From Acoustic Signal to High-Level Analysis," Ph.D. dissertation, Stanford University, 1985.

[25] F. Gouyon, F. Pachet, and O. Delerue, "On the use of zero-crossing rate for an application of classification of percussive sounds," in *Proc. Intl. Conf. on Digital Audio Effects (DAFx)*, Verona, Italy, 2000.

[26] D. FitzGerald, B. Lawlor, and E. Coyle, "Sub-band independent subspace analysis for drum transcription," in *Proc. Intl. Conf. on Digital Audio Effects (DAFx)*, Hamburg, Germany, 2002, pp. 65–69.

[27] P. Herrera, A. Yeterian, and F. Gouyon, "Automatic classification of drum sounds: A comparison of feature selection methods and classification techniques," in *Proc. Intl. Conf. on Music and Artificial Intelligence (ICMAI)*, Edinburgh, Scotland, UK, 2002, pp. 69–80.

[28] A. Zils, F. Pachet, O. Delerue, and F. Gouyon, "Automatic extraction of drum tracks from polyphonic music signals," *Proc. Intl. Conf. on Web delivering of Music (WEDELMUSIC)*, 2002.

[29] A. Eronen, "Musical instrument recognition using ICA-based transform of features and discriminatively trained HMMs," in *Proc. Intl. Symposium on Signal Processing and Its Applications (ISSPA)*, vol. 2, Paris, France, July 2003, pp. 133–136.

[30] D. FitzGerald, R. Lawlor, and E. Coyle, "Prior subspace analysis for drum transcription," in *Proc. Audio Engineering Society Convention (AES)*, March 2003.

[31] D. FitzGerald, B. Lawlor, and E. Coyle, "Drum transcription in the presence of pitched instruments using prior subspace analysis," in *Proc. Irish Signals and Systems Conf. (ISSC)*, Limerick, Ireland, July 2003.

[32] D. FitzGerald, "Automatic drum transcription and source separation," Ph.D. dissertation, Dublin Institute of Technology, Dublin, Ireland, 2004.

[33] P. Herrera, A. Dehamel, and F. Gouyon, "Automatic labeling of unpitched percussion sounds," in *Proc. Audio Engineering Society Convention (AES)*, Amsterdam, Netherlands, March 2003.

[34] C. Dittmar and C. Uhle, "Further steps towards drum transcription of polyphonic music," in *Proc. Audio Engineering Society convention (AES)*, Berlin, Germany, 2004.

[35] O. Gillet and G. Richard, "Automatic transcription of drum loops," in *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 4, Montreal, Quebec, Canada, May 2004, pp. 269–272.

[36] P. Herrera, V. Sandvold, and F. Gouyon, "Percussion-related semantic descriptors of music audio files," in *Audio Engineering Society Conf.: Metadata for Audio (AES)*, London, UK, June 2004, pp. 69–73.

[37] T. Nakano, J. Ogata, M. Goto, and Y. Hiraga, "A drum pattern retrieval method by voice percussion," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, 2004, pp. 550–553.

[38] V. Sandvold, F. Gouyon, and P. Herrera, "Percussion classification in polyphonic audio recordings using localized sound models," *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, pp. 537–540, 2004.

[39] D. V. Steelant, K. Tanghe, S. Degroeve, B. D. Baets, M. Leman, J.-P. Martens, and J. P. Martens, "Classification of percussive sounds using support vector machines," in *Proc. Annual Machine Learning Conf. of Belgium and The Netherlands (BENELEARN*, 2004, pp. 146–152.

[40] D. V. Steelant, K. Tanghe, S. Degroeve, B. D. Baets, M. Leman, and J.-P. Martens, "Support vector machines for bass and snare drum recognition," in *Classification?the Ubiquitous Challenge*. Springer, 2005, pp. 616–623.

[41] A. Tindale, A. Kapur, G. Tzanetakis, and I. Fujinaga, "Retrieval of percussion gestures using timbre classification techniques," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, 2004.

[42] K. Yoshii, M. Goto, and H. G. Okuno, "Automatic drum sound description for real-world music using template adaptation and matching methods," *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, 2004.

[43] ——, "Adamast: A drum sound recognizer based on adaptation and matching of spectrogram templates," *Annual Music Information Retrieval Evaluation eXchange (MIREX)*, 2005.

[44] ——, "Drum sound recognition for polyphonic audio signals by adaptation and matching of spectrogram templates with harmonic structure suppression," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 1, pp. 333–345, 2007.

[45] S. D. Sven, K. Tanghe, B. D. Baets, M. Leman, and J.-P. Martens, "A simulated annealing optimization of audio features for drum classification," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, 2005, pp. 482–487.

[46] O. Gillet and G. Richard, "Drum track transcription of polyphonic music signals using noise subspace projection," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, London, UK, 2005.

[47] ——, "Automatic transcription of drum sequences using audiovisual features," in *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 3, Philadelphia, Pennsylvania, USA, March 2005, pp. 205–208.

[48] A. Hazan, "Towards automatic transcription of expressive oral percussive performances," in *Proc. Intl. Conf. on Intelligent User Interfaces*, San Diego, California, USA, 2005, pp. 296–298.

[49] J. Paulus and T. Virtanen, "Drum transcription with non-negative spectrogram factorisation," in *Proc. European Signal Processing Conf. (EUSIPCO)*, Antalya, Turkey, 2005.

[50] K. Tanghe, S. Degroeve, and B. D. Baets, "An algorithm for detecting and labeling drum events in polyphonic music," in *Proc. first MIREX*, London, UK, 2005.

[51] G. Tzanetakis, A. Kapur, and R. I. McWalter, "Subband-based drum transcription for audio signals," in *Proc. Workshop on Multimedia Signal Processing*, Shanghai, China, 2005.

[52] J. P. Bello, E. Ravelli, and M. B. Sandler, "Drum sound analysis for the manipulation of rhythm in drum loops," in *Proc. IEEE Intl. Conf. on Acoustics Speech and Signal Processing (ICASSP)*, vol. 5, Toulouse, France, May 2006.

[53] O. Gillet and G. Richard, "Supervised and unsupervised sequence modelling for drum transcription," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, Vienna, Austria, September 2007, pp. 219–224.

[54] A. Moreau and A. Flexer, "Drum transcription in polyphonic music using non-negative matrix factorisation," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, Vienna, Austria, September 2007, pp. 353–354.

[55] P. Roy, F. Pachet, and S. Krakowski, "Improving the classification of percussive sounds with analytical features: A case study," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, Vienna, Austria, September 2007, pp. 229–232.

[56] E. Pampalk, P. Herrera, and M. Goto, "Computational models of similarity for drum samples," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, pp. 408–423, 02/2008 2008.

[57] D. S. Alves, J. Paulus, and J. Fonseca, "Drum transcription from multichannel recordings with non-negative matrix factorization," in *Proc. European Signal Processing Conf. (EUSIPCO)*, Glasgow, Scotland, UK, Aug 2009, pp. 894–898.

[58] J. Paulus and A. Klapuri, "Drum sound detection in polyphonic music with hidden markov models," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2009, no. 14, 2009.

[59] S. Scholler and H. Purwins, "Sparse coding for drum sound classification and its use as a similarity measure," in *Proc. Intl. Workshop on Machine Learning and Music (MML)*, 2010, pp. 9–12.

[60] A. Spich, M. Zanoni, A. Sarti, and S. Tubaro, "Drum music transcription using prior subspace analysis and pattern recognition," in *Proc. Intl. Conf. on Digital Audio Effects (DAFx)*, Graz, Austria, 2010.

[61] U. Şimşekli, A. Jylhä, C. Erkut, and A. T. Cemgil, "Real-time recognition of percussive sounds by a model-based method," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2011, 2011.

[62] E. Battenberg, V. Huang, and D. Wessel, "Live drum separation using probabilistic spectral clustering based on the Itakura-Saito divergence," in *Proc. Audio Engineering Society Conf. on Time-Frequency Processing in Audio (AES)*, Helsinki, Finland, 2012.

[63] E. Battenberg, "Techniques for machine understanding of live drum performances," Ph.D. dissertation, University of California at Berkeley, 2012.

[64] M. A. Kaliakatsos-Papakostas, A. Floros, M. N. Vrahatis, and N. Kanellopoulos, "Real-time drums transcription with characteristic bandpass filtering," in *Proc. Audio Mostly: A Conference on Interaction with Sound*, Corfu, Greece, 2012.

[65] H. Lindsay-Smith, S. McDonald, and M. Sandler, "Drumkit transcription via convolutive NMF," in *Proc. Intl. Conf. on Digital Audio Effects (DAFx)*, York, UK, September 2012.

[66] M. Miron, M. E. P. Davies, and F. Gouyon, "An open-source drum transcription system for pure data and max MSP," in *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Vancouver, BC, Canada, May 2013, pp. 221–225.

[67] ——, "Improving the real-time performance of a causal audio drum transcription system," in *Proc. Sound and Music Computing Conf. (SMC)*, Stockholm, Sweden, 2013, pp. 402–407.

[68] E. Benetos, S. Ewert, and T. Weyde, "Automatic transcription of pitched and unpitched sounds from polyphonic music," in *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Florence, Italy, May 2014, pp. 3107–3111.

[69] C. Dittmar and D. Gärtner, "Real-time transcription and separation of drum recordings based on NMF decomposition," in *Proc. Intl. Conf. on Digital Audio Effects (DAFx)*, Erlangen, Germany, September 2014, pp. 187–194.

[70] A. Röbel, J. Pons, M. Liuni, and M. Lagrange, "On automatic drum transcription using non-negative matrix deconvolution and itakura saito divergence," in *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Brisbane, Australia, April 2015, pp. 414–418.

[71] V. M. A. Souza, G. E. A. P. A. Batista, and N. E. Souza-Filho, "Automatic classification of drum sounds with indefinite pitch," in *Proc. Intl. Joint Conf. on Neural Networks (IJCNN)*, Killarney, Ireland, Jul 2015, pp. 1–8.

[72] M. Rossignol, M. Lagrange, G. Lafay, and E. Benetos, "Alternate level clustering for drum transcription," in *Proc. European Signal Processing Conf. (EUSIPCO)*, Nice, France, August 2015, pp. 2023–2027.

[73] C.-W. Wu and A. Lerch, "Drum transcription using partially fixed non-negative matrix factorization," in *Proc. European Signal Processing Conf. (EUSIPCO)*, 2015.

[74] ——, "Drum transcription using partially fixed non-negative matrix factorization with template adaptation," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, Malaga, Spain, October 2015, pp. 257–263.

[75] N. Gajhede, O. Beck, and H. Purwins, "Convolutional neural networks with batch normalization for classifying hi-hat, snare, and bass percussion sound samples," in *Proc. Audio Mostly: A Conference on Interaction with Sound*, Norrköping, Sweden, 2016, pp. 111–115.

[76] R. Vogl, M. Dorfer, and P. Knees, "Recurrent neural networks for drum transcription," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, New York City, United States, August 2016, pp. 730–736.

[77] ——, "Drum transcription from polyphonic music with recurrent neural networks," in *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, New Orleans, Louisiana, USA, Mar 2017, pp. 201–205.

[78] C. Southall, R. Stables, and J. Hockman, "Automatic drum transcription using bi-directional recurrent neural networks," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, New York City, United States, August 2016, pp. 591–597.

[79] C. Wu and A. Lerch, "On drum playing technique detection in polyphonic mixtures," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, New York City, United States, August 2016, pp. 218–224.

[80] C. Southall, R. Stables, and J. Hockman, "Automatic drum transcription for polyphonic recordings using soft attention mechanisms and convolutional neural networks," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, Suzhou, CN, Oct 2017, pp. 606–612.

[81] S. Scholler and H. Purwins, "Sparse approximations for drum sound classification," *Journal of Selected Topics Signal Processing*, vol. 5, no. 5, pp. 933–940, 2011.

[82] T. Nakano, M. Goto, J. Ogata, and Y. Hiraga, "Voice drummer: a music notation interface of drum sounds using voice percussion input," in *Proc. Annual ACM Symposium on User Interface Software and Technology (UIST)*, 2005, pp. 49–50.

[83] F. Korzeniowski and G. Widmer, "On the futility of learning complex frame-level language models for chord recognition," in *Proc. AES Intl. Conf. on Semantic Audio*, Erlangen, DE, Jun 2017.

[84] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Aistats*, vol. 9, 2010, pp. 249–256.

[85] "200 Drum Machines Dataset Web Presence." [Online]. Available: http://www.hexawe.net/mess/200.Drum.Machines/

[86] M. Prockup, E. M. Schmidt, J. Scott, and Y. E. Kim, "Toward Understanding Expressive Percussion Through Content Based Analysis," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, 2013.

[87] O. Gillet and G. Richard, "Enst-drums: an extensive audio-visual database for drum signals processing," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, 2006.

[88] R. Marxer and J. Janer, "Study of regularizations and constraints in NMF-based drums monaural separation," in *Proc. Intl. Conf. on Digital Audio Effects (DAFx)*, 2013, pp. 1–6.

[89] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC Music Database: Popular, Classical and Jazz Music Databases." in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, vol. 2, 2002, pp. 287–288.

[90] R. Y. Litovsky, H. S. Colburn, W. A. Yost, and S. J. Guzman, "The precedence effect," *The Journal of the Acoustical Society of America*, vol. 106, no. 4, pp. 1633–1654, 1999.

[91] J. Hochenbaum and A. Kapur, "Drum Stroke Computing: Multimodal Signal Processing for Drum Stroke Identification and Performance Metrics," in *Proc. Intl. Conf. on New Interfaces for Musical Expression (NIME)*, 2011.

[92] C.-W. Wu and A. Lerch, "Automatic drum transcription using the student-teacher learning paradigm with unlabeled music data," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, 2017.

[93] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. B. Sandler, "A tutorial on onset detection in music signals," *IEEE Transactions on Audio, Speech, and Language Processing*, pp. 1–13, 2005.

[94] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Proc. Neural Information Processing Systems (NIPS)*, Denver, CO, USA, 2000, pp. 556–562.

[95] 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 (ICA)*, Grenada, Spain, 2004, pp. 494–499.

[96] C. Laroche, H. Papadopoulos, M. Kowalski, and G. Richard, "Drum extraction in single channel audio signals using multi-layer non negative matrix factor deconvolution," in *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, New Orleans, Louisiana, USA, Mar 2017, pp. 46–50.

[97] R. S. Sutton, "Two problems with backpropagation and other steepest-descent learning procedures for networks," in *Proc. Annual Conf. of the Cognitive Science Society*. Erlbaum, 1986, pp. 823–831.

[98] Y. Nesterov, "A method for unconstrained convex minimization problem with the rate of convergence o (1/k2)," in *Doklady an SSSR*, vol. 269, no. 3, 1983, pp. 543–547.

[99] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[100] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[101] T. Tieleman and G. Hinton, "Lecture 6.5—rmsprop: Divide the gradient by a running average of its recent magnitude," http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf, October 2012.

[102] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[103] C. Southall, R. Stables, and J. Hockman, "ADT with BLSTMP," in *Under Review*, 2017.

[104] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

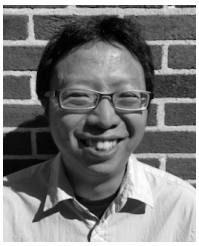
[105] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[106] R. Caruana, "Multitask learning," in *Learning to learn*. Springer, 1998, pp. 95–133.

[107] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, November 1997.

[108] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "Learning phrase representations using rnn encoderdecoder for statistical machine translation," in *Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, October 2014.

[109] J. S. Downie, X. Hu, J. H. Lee, K. Choi, S. J. Cunningham, and Y. Hao, "Ten Years of MIREX: Reflections, Challenges, and Opportunities," in *Proc. Intl. Society for Music Information Retrieval Conf. (ISMIR)*, 2014.



Chih-Wei Wu received the M.Sc. degree in engineering and system science from National Tsing Hua University (NTHU), Hsinchu, Taiwan, in 2009, and the M.Sc. degree in sound and music innovative technologies from National Chiao Tung University (NCTU), Hsinchu, Taiwan, in 2012. Currently he is a Ph.D. candidate at Georgia Institute of Technology, Center for Music Technology. His research interests include music information retrieval, sound analysis and synthesis, and audio signal processing.

Christian Dittmar received the Diploma degree in electrical engineering from the Jena University of Applied Sciences, Jena, Germany, in 2003. Since summer 2014, he has been working toward the Ph.D. degree in the research group of Meinard Müller, International Audio Laboratories Erlangen, Germany. Before that, he had lead the Semantic Music Technology Research Group at the Fraunhofer Institute for Digital Media Technology (IDMT), Ilmenau, Germany. Since 2012, he has been also the CEO and co-founder of the music technology start-up Songquito. He authored and coauthored a number of peer-reviewed papers on music information retrieval topics. His recent research interests include music information retrieval, audio signal processing, and music education applications.

Carl Southall received the B.Sc (Hons) degree in 2015 in sound engineering and production from Birmingham City University (BCU). He joined the BCU Digital Media Technology Lab in 2015 and is currently working towards a Ph.D. in music information retrieval. His main areas of interest are machine learning, automatic music transcription and automatic music generation.

Richard Vogl received his M.Sc. degree in computer science with focus on machine learning in 2009 from Johannes Kepler University, Linz, Austria. After working at a Linz based start-up, he joined the Department of Computational Perception at JKU in 2014 to continue his work in music information retrieval and machine learning, pursuing a Ph.D. degree. He is currently involved in a project in the context of MIR and recommender systems at Technische Universität Wien, Vienna. His research interests include deep learning, rhythmical analysis of music, and automatic drum transcription.

Gerhard Widmer is Professor and Head of the Department of Computational Perception at Johannes Kepler University, Linz, Austria, and Head of the Intelligent Music Processing and Machine Learning Group at the Austrian Research Institute for Artificial Intelligence (OFAI), Vienna, Austria. His research interests include AI, machine learning, and intelligent music processing, and his work is published in a wide range of scientific fields, from AI and machine learning to audio, multimedia, musicology, and music psychology. He is a Fellow of the European Association for Artificial Intelligence (EurAI), has been awarded Austrias highest research awards, the START Prize (1998) the Wittgenstein Award (2009), and currently holds an ERC Advanced Grant for research on computational models of expressivity in music.

Jason Hockman is a Senior Lecturer in audio engineering at the School of Computing and Digital Technology in Birmingham City University (BCU), United Kingdom. He stuided sociology at Cornell University, USA, before earning a Masters of Music from New York University, USA in 2007 and a doctorate in Music Research from McGill University, Canada in 2014. In 2015, he joined the Digital Media Technology Labratory (DMT Lab) at BCU, where he conducts research in the field of music information retrieval with a focus on computational meter and rhythm description and digital audio effects.

Meinard Müller received the Diploma degree in mathematics and the Ph.D. degree in computer science from the University of Bonn, Germany. In 2007, he finished his Habilitation in the field of multimedia retrieval. From 2007 to 2012, he was a Member of the Saarland University and the Max-Planck Institut für Informatik. Since 2012, he holds a professorship for Semantic Audio Processing at the International Audio Laboratories Erlangen. His recent research interests include music processing, music information retrieval, audio signal processing, multimedia retrieval, and motion processing. He was a Member of the IEEE Audio and Acoustic Signal Processing Technical Committee form 2010 to 2015 and is a Member of the Board of Directors, International Society for Music Information Retrieval since 2009. Meinard Müller has coauthored more than 100 peer-reviewed scientific papers, wrote a monograph titled Information Retrieval for Music and Motion (Springer-Verlag, 2007) as well as a text-book titled *Fundamentals of Music Processing* (Springer-Verlag, 2015, www.music-processing.de).

Alexander Lerch is Assistant Professor at the Georgia Institute of Technology, Center for Music Technology, where his research areas include Music Information Retrieval, Audio Content Analysis, and Intelligent Audio Processing. He studied electrical engineering at the Technical University Berlin and Tonmeister (Music Production) at the University of the Arts Berlin. He received his Ph.D. on algorithmic music performance analysis from the Technical University Berlin. In 2001, he co-founded the company zplane.development, a research-driven technology provider for the music industry. His book *An Introduction to Audio Content Analysis* was published in 2012 by Wiley/IEEE Press.