DRUM TRANSCRIPTION FROM POLYPHONIC MUSIC WITH RECURRENT NEURAL NETWORKS

Richard Vogl,[1,2] *Matthias Dorfer,*[1] *Peter Knees* [2]

[1] Dept. of Computational Perception, Johannes Kepler University Linz, Austria
[2] Institute of Software Technology & Interactive Systems, Vienna University of Technology, Austria
richard.vogl@jku.at, matthias.dorfer@jku.at, knees@ifs.tuwien.ac.at

ABSTRACT

Automatic drum transcription methods aim at extracting a symbolic representation of notes played by a drum kit in audio recordings. For automatic music analysis, this task is of particular interest as such a transcript can be used to extract high level information about the piece, e.g., tempo, downbeat positions, meter, and genre cues. In this work, an approach to transcribe drums from polyphonic audio signals based on a recurrent neural network is presented. Deep learning techniques like dropout and data augmentation are applied to improve the generalization capabilities of the system. The method is evaluated using established reference datasets consisting of solo drum tracks as well as drums mixed with accompaniment. The results are compared to state-of-the-art approaches on the same datasets. The evaluation reveals that F-measure values higher than state of the art can be achieved using the proposed method.

Index Terms— Drum transcription, neural networks, deep learning, automatic transcription, data augmentation

1. INTRODUCTION

The goal of automatic drum transcription (ADT) systems is to create a symbolic representation of the drum instrument onsets contained in monaural audio signals. A reliable ADT system has many applications in different fields like music production, music education, and music information retrieval. Good transcription results can be achieved on simple solo drum tracks [1], but for complex mixtures in polyphonic audio, the problem is still not solved satisfactorily. In this work, a robust method to transcribe solo drum tracks using RNNs [2] is further extended to be applicable on polyphonic audio.

As in other work (e.g. [3, 1, 4, 5, 6]), the transcribed instrument classes are limited to the three main instruments used in most drum kits: bass drum, snare drum, and hi-hat. This is a reasonable simplification as these three classes usually suffice to capture the main rhythm patterns of the drum track [3] and cover most of the played notes in full drum kit recordings.[1]

[1] >80% in the case of the ENST-Drums [7] dataset, see sec. 4.1.

2. RELATED WORK

The majority of ADT methods can be categorized into three classes: *i. segment and classify*, *ii. match and adapt*, and *iii. separate and detect* methods (cf. [9]).

Segment and classify methods first segment the audio signal using, e.g., onset detection and then classify the resulting fragments regarding contained drum instruments [8, 9]. Miron et al. use a combination of frequency filters, onset detection and feature extraction in combination with a k-nearest-neighbor [10] and a k-means [8] classifier to detect drum sounds in a solo drum audio signal in real-time.

Match and adapt methods use temporal or spectral templates of the individual drum sounds to detect the events. These templates are iteratively adapted during classification to better match the events in the input signal. Yoshii et al. [11] present an ADT system based on template matching and adaptation incorporating harmonic suppression.

The *separate and detect* methods utilize source separation techniques to separate the drum sounds from the mix. Subsequently the onsets for the individual drums are detected. The most successful technique in this context is non-negative matrix factorization (NMF). Dittmar and Gärtner [1] use an NMF approach for a real-time ADT system of solo drum tracks. Their approach utilizes training instances for the individual drum instrument of each track.

Additionally there are methods which combine techniques from these categories. Hidden Markov Models (HMMs) can be used to perform segmentation and classification in one step. Paulus and Klapuri [3] use HMMs to model the development of MFCCs over time and incorporate an unsupervised acoustic model adaptation. Decoding the most likely sequence yields activation curves for bass drum, snare drum, and hi-hat and can be applied for both solo drum tracks as well as polyphonic music. Wu and Lerch [5] use an extension of NMF, the so-called partially fixed NMF (PFNMF), for which they also evaluate two different template adaptation methods.

Artificial neural networks represent a powerful machine learning technique which is being successfully applied in many different fields. Recurrent neural networks (RNNs) are neural networks with additional connections (recurrent connections) in each layer, providing the outputs of the same

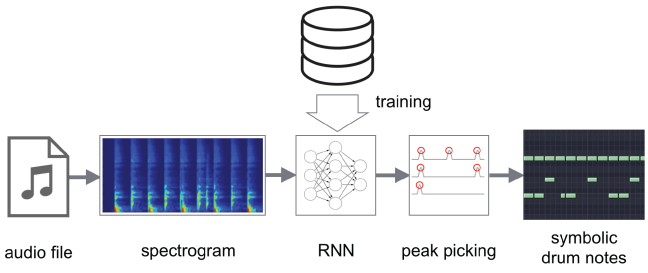

Fig. 1. Overview of the proposed method. The extracted spectrogram is fed into the trained RNN which outputs activation functions for each instrument. A peak picking algorithms selects appropriate peaks as instrument onset candidates.

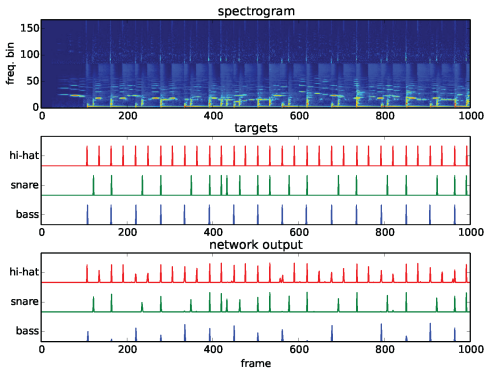

Fig. 2. Spectrogram of a drum track with accompaniment (top) and target functions for bass drum, snare drum, and hi-hat (middle). The target functions have a value of 1.0 for the frames which correspond to the annotations of the individual instruments and 0.0 otherwise. The frame rate of the target function is 100Hz, the same as for the spectrogram. The third plot (bottom) shows the output of a trained RNN for the spectrogram in the top plot.

layer from the last time step as additional inputs. These recurrent connections serve as a kind of memory which is beneficial for tasks with sequential input data. For example, RNNs have been shown to perform well for speech recognition [12] and handwriting recognition [13].

RNNs have several advantages in the context of automatic music transcription. As shown in the context of automatic piano transcription by Böck and Schedl [14], RNNs are capable of handling many different classes better than NMF. This becomes particularly relevant when classifying pitches (typically up to 88) [14] or many instruments. Southall et al. [15] apply bidirectional RNNs (BDRNNs) for ADT and demonstrate the capability of RNNs to detect snare drums in polyphonic audio better than state of the art. In [2], we show that time-shift RNNs (tsRNNs) perform as well as BDRNNs when used for ADT on solo drum tracks, while maintaining online capability and also demonstrate the generalization capabilities of RNNs in the context of ADT. In the present work, this method is further developed into an online-capable ADT system for polyphonic audio, which further improves the state of the art.

3. METHOD

Fig. 1 shows an overview of the proposed method. First, the input features derived from the power spectrogram of the audio signal are calculated. The result is frame-wise fed into an RNN with three output neurons. The outputs of the RNN provide activation signals for the three drum instruments considered (bass drum, snare drum, and hi-hat). A peak picking algorithm then identifies the onsets for each instrument's activation function, which yields the finished transcript. The next sections will cover the individual steps of the method in detail.

3.1. Feature Extraction

As input, mono audio files with 16 bit resolution at 44.1kHz sampling rate are used. The audio is normalized and padded with 0.25 seconds of silence at the start to avoid undesired artifacts resulting from onsets which occur immediately at the beginning. A logarithmic power spectrogram is calculated using a 2048-samples window size and a resulting frame rate of 100Hz. The frequency axis is transformed to a logarithmic scale using twelve triangular filters per octave over a frequency range from 20 to 20,000 Hz. This results in a total number of 84 frequency bins. Additionally the positive first-order-differential over time of this spectrogram is calculated. The resulting differential-spectrogram-frames are stacked on top of the normal spectrogram frames, resulting in feature vectors with a length of 168 (2x84) values.

3.2. Recurrent Neural Network

In this work, a two-layer RNN architecture with label time shift is used (tsRNN). It has been shown that these networks perform as well as BDRNNs on solo drum tracks, while having the advantage of being online capable [2]. The RNN features a 168 node input layer which is needed to handle the input data vectors of the same size. Two recurrent layers, consisting of 50 gated recurrent units (GRUs [16]) each, follow. The connections between the input and the recurrent layers, the recurrent connections, as well as the connections between the recurrent layer and the next layer are all realized densely (every node is connected to all other nodes). A so-called dropout layer [17] is situated between the recurrent and the output layer. In this layer, connections are randomly disabled for every iteration during training. This helps preventing overfitting to the training data. The amount of disabled connections is controlled by the dropout rate, which was set to $r_d = 0.3$. The output layer consists of three nodes with sigmoid transfer functions, which output the activation functions for the three instrument classes defined earlier.

Label time shift refers to the process of shifting the orig-

inal annotations. After transcription, the detected onsets are shifted back by the same time. In doing so, the RNN can also take a small portion of the sustain phase of the onset's spectrogram into account. The used delay of $30ms$ (corresponds to three spectrogram frames) in this work is still sufficiently small for certain applications like score following and other visualizations, while it can be tuned to meet the demands of other applications.

3.3. Peak Picking

The neurons of the output layer generate activation functions for the individual instruments (see fig. 2). The instrument onsets are identified using the same peak picking method as in [2]: A point n in the function $F(n)$ is considered a peak if these terms are fulfilled:

1. $F(n) = max(F(n-m), \cdots, F(n))$,
2. $F(n) \geq mean(F(n-a), \cdots, F(n)) + \delta$,
3. $n - n_{lp} > w$,

where δ is a variable threshold. A peak must be the maximum value within a window of size $m+1$, and exceeding the mean value plus a threshold within a window of size $a+1$. Additionally, a peak must have at least a distance of $w+1$ to the last detected peak (n_{lp}). Values for the parameters were tuned on a development dataset to be: $m = a = w = 2$.

3.4. RNN Training

When fed with the features at the input nodes, the RNN should reproduce the activation functions of the individual instruments at the output neurons. During training, the update function adapts parameters of the network (weights and biases of neurons) using the calculated error (loss) and the gradient through the network. As update function, the *rmsprop* method is used [18].

As loss function, the mean of the binary cross-entropy between outputs of the network and target functions is used (see fig. 2). Snare drum and hi-hat onsets are considered more difficult to transcribe than bass drum [3, 5, 15]. Due to this fact, the loss functions of the output neurons for bass drum (1.0), snare drum (4.0), and hi-hat (1.5) are weighted differently. This way, errors for snare drum and hi-hat are penalized more, which forces the training to focus on them.

RNN training using *rmsprop* involves so-called mini-batches. In this work, a mini-batch consists of eight training instances. The training instances are obtained by cutting the extracted spectrograms into 100-frame-segments with 90 frames overlap. The order of the segments for training is randomized. To further increase generalization, data-augmentation [19] is used. The training instances are randomly augmented using pitch shift (-5 to $+10$ frequency bins) and time stretching (scale factors: 0.70, 0.85, 1.00, 1.30, 1.60).

Training is structured into epochs, during which the training data is used to optimize the parameters of the network. At the end of an epoch a validation set (25% excluded from the training set) is used to estimate the performance of the trained network on data not used for training. The training of the RNN is aborted as soon as the resulting loss on the validation set has not decreased for 10 epochs. The initial learning rate was set to $r_l = 0.007$, the learning rate is reduced to a fifth every 7 epochs.

All hyperparameters like network architecture, dropout rate, augmentation parameters, and learning rate were chosen accordingly to experiments on a development dataset, experience, and best practice examples.

4. EVALUATION

The well-known metrics precision, recall, and F-measure are used to evaluate the performance of the presented system. True positive, false positive, and false negative onsets are identified by using a $20ms$ tolerance window. It should be noted that state-of-the-art methods for the ENST-Drums dataset [3] as well as for the IDMT-SMT-Drums dataset [1], use less strict tolerance windows of $30ms$ and $50ms$, respectively, for evaluation. However, listening experiments showed that distinct events with a delay of $50ms$ are already perceivable. Therefore, in this work, $20ms$ windows are used.

4.1. Datasets

For evaluation, two well-known datasets are used. The IDMT-SMT-Drums [1] contains recorded (*RealDrum*), synthesized (*TechnoDrum*), and sampled (*WaveDrum*) drum tracks. It comprises 560 files of which 95 are simple drum tracks (of approx. 15sec). The rest are single-instrument training tracks.

As second dataset the ENST-Drums set [7] is used. The dataset consists of real drum recordings of three drummers performing on three different drum kits. The recordings are available as solo instrument tracks and as two mixtures (dry and wet). For a subset, accompaniment tracks are included (minus-one tracks). The total length of the recorded material is roughly 75 minutes per drummer. In this work, the wet mixes of the minus-one tracks plus accompaniment of all three drummers were used. Since the ENST-Drums dataset contains more than the three main instruments, only the snare, bass, and hi-hat annotations were used. 81.2% of onsets are annotated as snare drum, bass drum, and hi-hat while the remaining 18.8% cover other cymbals and tom-tom drums.

4.2. Experiments

The proposed method was evaluated in four different experiments. These were performed using *i.* the drum tracks of the IDMT-SMT-drums dataset (SMT solo), *ii.* the minus-one tracks of the ENST-drums dataset without accompaniment (ENST solo), and *iii.* the minus-one tracks mixed with accompaniment of aforementioned (ENST acc.). In the experiments, on *SMT solo* a three-fold cross validation on the

F-measure [%] for individual methods on datasets

Method	SMT solo		ENST solo	ENST acc.
NMF [1]	—	(95.0)	—	—
PFNMF[5]	81.6	(—)	77.9	72.2
HMM [3]	—	(—)	81.5	74.7
BDRNN [15]	83.3	(96.1)	73.2	66.9
tsRNN	**92.5**	**(96.6)**	**83.3**	**75.0**

Table 1. Top four rows show results of state-of-the-art algorithms. Highest values were achieved at peak picking thresholds of 0.10 and 0.15 (*ENST solo*, *SMT solo opt.* cf. fig. 3). Values in brackets represent results for optimized models (*SMT solo opt.* see sec. 4.2).

three splits (*RealDrum*, *TechnoDrum*, and *WaveDrum*) of the dataset was performed (comparable to the *automatic* experiment in [15] and [5]). Additionally a six-fold cross validation on six randomized splits was performed (*SMT solo opt.*). This task is comparable to the *semi-automatic* experiments in [15], and [1]—it is arguably a even harder task, since in a model is trained on more than the training data of just one track. In both cases the corresponding splits of the training tracks are additionally used only for training.

In the case of *ENST solo* and *ENST acc.* the dataset was split into three parts consisting of the tracks of one drummer and a three-fold cross validation was performed. Training for each fold was performed on all tracks of two drummers while testing was done on the minus-one tracks (without and with accompaniment resp.) of the third drummer and thus on unseen data. This is consistent with the experiments performed in [3, 5, 15].

5. RESULTS

Tab. 1 summarizes the results of the presented method and state-of-the-art methods on the used datasets. It can be seen that the F-measure values for the tsRNN approach are higher than the state of the art for *SMT solo* and *ENST solo*, and on the same level for *ENST acc.*Since for training, both tracks with and without accompaniment were used, the same models are applied to *ENST solo* and *ENST acc.* splits, which further demonstrates the capability of the presented method to generalize well. Fig. 3 shows F-measure and precision-recall curves for the cross-validation results on the individual datasets. For these curves the threshold level for peak picking was varied in the range 0.05 to 0.95 using steps of 0.05. It can be seen that the highest F-measure values are found for threshold values of 0.10 and 0.015, which is lower than the expected value of around 0.5 (target functions range is 0–1). This is due to the fact that the output of the RNN does not contain much noise (see fig. 2), which implies that the trained RNN is capable of effectively filtering accompaniment.

Since the target functions contain little noise while strong peaks are present for instrument onsets, only little time was

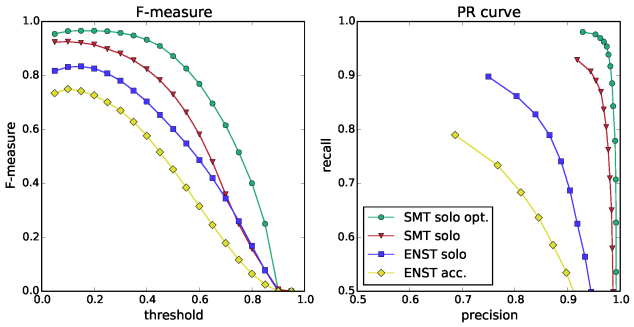

Fig. 3. Results of the evaluation on the individual datasets. Left plot shows F-measure curve, right plot precision-recall curves for different threshold levels (δ) for peak picking. Best results were achieved at thresholds of 0.10 and 0.15.

invested optimizing peak picking. Noticeable improvements to [2] were achieved by using data augmentation and GRUs instead of RNN units for the network.

6. CONCLUSION

In this work, an approach for drum transcription from solo drum tracks and polyphonic music was introduced. The proposed method uses an RNN with two recurrent layers consisting of GRUs in combination with label time shift and introduces loss function weighting for the individual instruments to increase transcription performance. Additionally dropout and data augmentation are successfully applied to overcome overfitting to the individual drum sounds in the different dataset splits. The presented system is online capable with a latency of $30ms$ introduced by the label time shift.

In contrast to hand-crafted systems and features, where the architecture is often difficult to adapt when shortcomings are detected, RNNs have shown to be more flexible. A major advantage of such a technique is that the system can be focused on training instances on which the model previously failed. In table 1 it can be seen that RNNs are capable of learning to filter accompaniment and perform well also on polyphonic music. It has been shown that the transcription F-measure performance of the proposed method is higher than the results of state-of-the-art approaches, even when using a more stringent tolerance window for evaluation.

7. ACKNOWLEDGMENTS

This work has been partly funded by the EU's seventh Framework Programme FP7/2007-13 for research, technological development and demonstration under grant agreement no. 610591 (*GiantSteps*) and by the Austrian FFG under the BRIDGE 1 project *SmarterJam* (858514), We gratefully acknowledge the support of the NVIDIA Corporation by donating one Titan Black GPU for research purposes.

8. REFERENCES

[1] Christian Dittmar and Daniel Gärtner, "Real-time transcription and separation of drum recordings based on NMF decomposition," in *Proc 17th International Conference on Digital Audio Effects*, Erlangen, Germany, Sept. 2014.

[2] Richard Vogl, Matthias Dorfer, and Peter Knees, "Recurrent neural networks for drum transcription," in *Proc 17th International Society for Music Information Retrieval Conference*, New York, NY, USA, Aug. 2016.

[3] Jouni Paulus and Anssi Klapuri, "Drum sound detection in polyphonic music with hidden markov models," *EURASIP Journal on Audio, Speech, and Music Processing*, 2009.

[4] Andrio Spich, Massimiliano Zanoni, Augusto Sarti, and Stefano Tubaro, "Drum music transcription using prior subspace analysis and pattern recognition," in *Proc 13th International Conference of Digital Audio Effects*, Graz, Austria, Sept. 2010.

[5] Chih-Wei Wu and Alexander Lerch, "Drum transcription using partially fixed non-negative matrix factorization with template adaptation," in *Proc 16th International Society for Music Information Retrieval Conference*, Málaga, Spain, Oct. 2015.

[6] Derry FitzGerald, Robert Lawlor, and Eugene Coyle, "Prior subspace analysis for drum transcription," in *Proc 114th Audio Engineering Society Conference*, Amsterdam, Netherlands, Mar. 2003.

[7] Olivier Gillet and Gaël Richard, "Enst-drums: an extensive audio-visual database for drum signals processing," in *Proc 7th International Conference on Music Information Retrieval*, Victoria, BC, Canada, Oct. 2006.

[8] Marius Miron, Matthew EP Davies, and Fabien Gouyon, "Improving the real-time performance of a causal audio drum transcription system," in *Proc Sound and Music Computing Conference*, Stockholm, Sweden, July 2013.

[9] Olivier Gillet and Gaël Richard, "Transcription and separation of drum signals from polyphonic music," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 3, 2008.

[10] Marius Miron, Matthew EP Davies, and Fabien Gouyon, "An open-source drum transcription system for pure data and max msp," in *Proc IEEE International Conference on Acoustics, Speech and Signal Processing*, Vancouver, BC, Canada, May 2013.

[11] Kazuyoshi Yoshii, Masataka Goto, and Hiroshi G Okuno, "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, 2007.

[12] Haşim Sak, Andrew W. Senior, and Françoise Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proc 15th Annual Conference of the International Speech Communication Association*, Singapore, Sept. 2014.

[13] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber, "A novel connectionist system for improved unconstrained handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, 2009.

[14] Sebastian Böck and Markus Schedl, "Polyphonic piano note transcription with recurrent neural networks," in *Proc IEEE International Conference on Acoustics, Speech and Signal Processing*, Kyoto, Japan, Mar. 2012.

[15] Carl Southall, Ryan Stables, and Jason Hockman, "Automatic drum transcription using bidirectional recurrent neural networks," in *Proc 17th International Society for Music Information Retrieval Conference*, New York, NY, USA, Aug. 2016.

[16] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," `http://arxiv.org/abs/1409.1259`, 2014.

[17] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, June 2014.

[18] Tijmen Tieleman and Geoffrey Hinton, "Lecture 6.5rmsprop: Divide the gradient by a running average of its recent magnitude," in *COURSERA: Neural Networks for Machine Learning*, Oct. 2012.

[19] Jan Schlüter and Thomas Grill, "Exploring data augmentation for improved singing voice detection with neural networks," in *Proc 16th International Society for Music Information Retrieval Conference*, Málaga, Spain, Oct. 2015.