Temporal convolutional networks for musical audio beat tracking

Matthew E. P. Davies
INESC TEC
Porto, Portugal
matthew.davies@inesctec.pt

Sebastian Böck
Austrian Research Institute for Artificial Intelligence (OFAI)
Vienna, Austria
sebastian.boeck@ofai.at

***Abstract*—We propose the use of Temporal Convolutional Networks for audio-based beat tracking. By contrasting our convolutional approach with the current state-of-the-art recurrent approach using Bidirectional Long Short-Term Memory, we demonstrate three highly promising attributes of TCNs for music analysis, namely: i) they achieve state-of-the-art performance on a wide range of existing beat tracking datasets, ii) they are well suited to parallelisation and thus can be trained efficiently even on very large training data; and iii) they require a small number of weights.**

***Index Terms*—Beat Tracking, Music Signal Processing, Convolutional Neural Networks**

I. Introduction

The task of musical audio beat tracking has been well-established over the last twenty-five years [1]. While the goal of estimating a sequence of quasi-periodic time instants to reflect how a human listener would synchronise their taps to the beat of the music has remained largely unchanged, there has been a shift away from purely signal processing-based approaches to those incorporating machine learning, and most recently deep learning. One means of understanding this change is to consider how early approaches to beat tracking relied on the use of onset strength functions as the primary input representation. Given such an input containing peaks at onset locations, the aim was to identify and track a latent periodicity and subsequently (or simultaneously) identify the subset of these peaks most likely to correspond to the beat of the music. In this sense, the aim was to recover a hidden sequence of beats from an observed representation related to musical onsets.

In an effort to filter out peaks that were unlikely to correspond to beats, Davies *et al.* [2], derived a so-called beat emphasis function as the linear combination of sub-band onset strength functions weighted by their respective beat strength. While effective in enhancing periodic peaks in the onset strength functions it yielded only a moderate improvement over existing state of the art approaches (*e.g.*, [3]). A radically different approach was proposed by Böck and Schedl [4] who reformulated the beat tracking task using Recurrent Neural Networks (RNNs), specifically, a Bidirectional Long Short-Term Memory (BLSTM) model, to output a beat activation function (with peaks only at beat locations) given a log magnitude spectrogram input representation and a training dataset of manually annotated beat locations. In this way, a detected sequence of beat locations could be obtained simply by peak-picking the beat activation function.

Limitations in both the amount of training data and the variable quality of the annotations led to more sophisticated, and ultimately more successful, approaches for obtaining beat times from *imperfect* beat activation functions. These included the selection between multiple trained models adapted to different types of musical content and the use of a dynamic Bayesian network (DBN) for decoding the beat activation function [5], with further gains possible by the combined modelling of beat and downbeat information [6], a step that echoes the probabilistic approach of Klapuri *et al.* [3] which simultaneously estimated beat, downbeat, and tatum levels.

In spite of the inclusion of more sophisticated post-processing applied to the beat activation function and the considerable improvements obtained from access to more training data, the core BLSTM approach [4] has remained essentially unchanged, perhaps due to the inherent modelling power of recurrent models for sequential data. Nevertheless, recurrent models are very hard to train and possess certain limitations, including: the vanishing gradient problem, difficulty in interpreting the different internal layers of the model, and a learning approach which doesn't lend itself to efficient parallelisation using GPUs.

Convolutional Neural Networks (CNNs) on the other hand, and particularly for image processing tasks, are amenable to exposing the representations held in different layers [7]. Furthermore, CNNs are highly parallelisable and can thus be trained very efficiently on GPUs. For the task of beat tracking they have only been deployed to predict local features (*e.g.*, [8]) or model beat sequences with (large) filters which extend over a context of several hundred frames [9]. Recently, convolutional recurrent neural networks (CRNNs) have emerged in an attempt to leverage the modelling power of both CNNs and RNNs, where recurrent layers are attached to the output of convolutional models [10], [11].

In this paper, we explore the ability of Temporal Convolutional Networks (TCNs) [12] for a sequential learning task,

Matthew E.P. Davies is supported by Portuguese National Funds through the FCT-Foundation for Science and Technology, I.P., under the project IF/01566/2015. Sebastian Böck is supported by the Austrian Promotion Agency (FFG) under the "BASIS, Basisprogramm" umbrella program.

978-9-0827-9703-9/19/$31.00

namely musical audio beat tracking. Having first appeared in the well-known *WaveNet* model [13], TCNs perform dilated convolutions (*i.e.*, convolutions across sub-sampled input representations) for learning sequential/temporal structure. In this way, they retain the parallelisation property of standard CNNs, and have been shown to outperform recurrent approaches on a range of sequential learning problems [12]. While the *WaveNet* approach used the raw audio waveform as input, we reformulate the current state-of-the-art approach for beat tracking [6], by substituting the BLSTM with a TCN and applying it to an input representation derived from a log magnitude spectrogram. In doing so, we demonstrate the TCN approach is able to perform on par with the state of the art on existing annotated beat tracking datasets, that it can be trained far more efficiently from a computational perspective, and it also benefits from a very small number of weights $(21,809)$ which is in stark contrast to many existing deep learning approaches, which can have millions of trainable parameters, and roughly a third of the BLSTM method [4].

The remainder of this paper is structured as follows. In Section II we describe our TCN approach for beat tracking. This is followed in Section III by an objective evaluation against the current state of the art. In Section IV we discuss the impact of our approach and propose areas for future work.

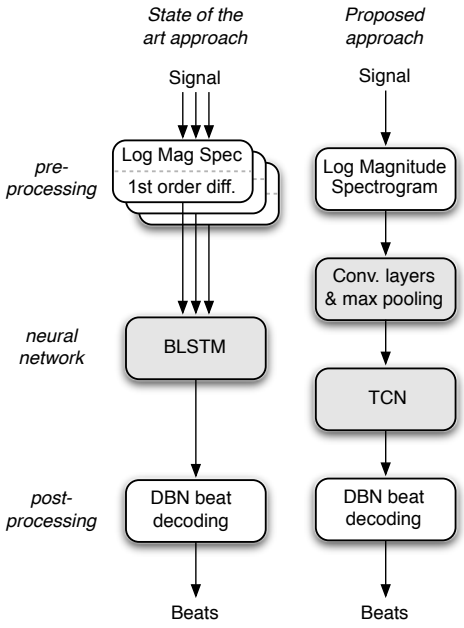

Fig. 1. Comparison between existing state of the art (left) with our proposed approach (right). The neural network blocks are shaded light grey.

II. Approach

A. Overview of existing state of the art

We base our approach around the model first presented by Böck and Schedl [4], and later extended in [5], [6] whose main processing pipeline is shown in the left hand side of Fig. 1. Given a mono audio input signal, sampled at $44.1\,\text{kHz}$, the input representation is derived from a set of log magnitude spectrograms which are grouped to have approximately logarithmic frequency spacing between adjacent bins. Three such spectrograms are calculated at a fixed hop size of $10\,\text{ms}$ with increasing window sizes of $23.2\,\text{ms}$, $46.4\,\text{ms}$ and $92.9\,\text{ms}$. From each, the per-bin first-order difference spectorgram is calculated, where only the positive differences are retained to capture the energy rise in individual frequency bands. All of these spectrogram representations are vertically stacked, and this multi-resolution input representation is then passed to a three layer BLSTM, with each layer having 25 recurrent units. In [4], the final beat locations were obtained by peak picking the beat activation function, however in [5], this processing step was replaced by a DBN approximated via a hidden Markov model (HMM). For more details see [4], [5].

B. Proposed reformulation using TCNs

An overview of our proposed approach is shown in the flow chart on the right of Fig. 1. By comparing the two processing pipelines, that of the existing state of the art (left), and our proposed method (right), we can observe: i) a simplification in the input representation, which replaces six spectrogram type inputs with just one and thus greatly reduces the input dimensionality; and ii) the inclusion of a set of convolution and max pooling layers prior to the main sequence learning model, the TCN, which replaces the BLSTM network. Our goal here is to minimise the need for explicit design choices in the front-end to our beat tracker, and thus place greater emphasis on what can be learned from a "simpler" input representation.

1) Input and Target Representations: As the initial input representation we use a single log magnitude spectrogram with a hop size of $10\,\text{ms}$ and a window size of $46.4\,\text{ms}$ (2048 samples). A logarithmic grouping of frequency bins with 12 bands per octave provides an input representation with a total of 81 frequency bands from $30\,\text{Hz}$ up to $17\,\text{kHz}$, as summarised in the *Signal Conditioning* section of Table I.

In the context of beat tracking, we are seeking to predict a beat activation function from this input representation which exhibits peaks at likely beat locations and can be used to recover an output sequence of beat times. To this end, we treat the beat tracking problem as a binary classification task, where annotated beat locations are first quantised to the temporal resolution of the input representation, and then represented as training targets. The goal is then to predict the likelihood of a beat occurring at any given time frame of the log magnitude spectrogram. Following the strategy of [14] for onset detection, we *widen* the temporal activation region around the annotations to include two adjacent temporal frames on either side of each quantised beat location and weight them with a value of 0.5 during training.

2) Convolutional Block: While the log magnitude spectrogram could be passed directly to the TCN, we first seek to learn some compact intermediate representation. To this end, we employ three convolutional layers: the first two layers with 16 filters of size 3×3 with subsequent max pooling over 3 bins in the frequency direction; and a third with 16 filters of

TABLE I
OVERVIEW OF SIGNAL PROCESSING AND LEARNING PARAMETERS

Signal Conditioning	
Audio sample rate	44.1 kHz
Window shape	*Hann*
Window & FFT size	2048 *samples*
Hop size	10 ms
Filterbank freq. range	30 . . . 17000 Hz
Sub-bands per octave	12
Total number of bands	81
Conv. Block	
Number of filters	16, 16, 16
Filter size	3×3, 3×3, 1×8
Max. pooling size	1×3, 1×3, —
Dropout rate	0.1
Activation function	*ELU*
TCN	
Number of stacks	1
Dilations	$2^{0,\ldots,10}$
Number of filters	16
Filter size	5
Spatial dropout rate	0.1
Activation function	*ELU*
Training	
Optimizer	*Adam*
Learning rate	0.001
Batch size	1
Output activation function	*sigmoid*
Loss function	*binary cross-entropy*

size 1×8 without pooling. In this way, small (overlapping) spectrogram snippets with a context of 5 frames get reduced to a single frame and 16 features. The exponential linear unit (*ELU*) [15] is used as activation function in the convolutional layers, and a dropout [16] rate of 0.1 applied afterwards. All parameters are summarised in the *Conv. Block* section of Table I. By learning these filters within the network we can derive an intermediate representation which is better adapted to the input data and much smaller than the hard-coded choice of the bin-wise temporal difference.

3) Temporal Convolution Network: The principal means by which the TCN is able to capture sequential structure is by learning filters via *dilated* convolutions. In our case, the input to the TCN is a 16-dimensional feature vector derived from the magnitude spectrogram by the convolutional block, which retains the same temporal resolution. The learning target for the TCN is to predict the beat locations from annotated training data as described in Section II-B1. By working on a highly sub-sampled feature representation compared to the raw audio, we can obtain a large temporal receptive field with far fewer layers and weights than the raw audio domain equivalent.

The TCN, as presented in [13], is highly parameterisable, with the principal degrees of freedom being: the number of TCN filters, the kernel size (*i.e.*, shape of the filters), the number of layers, their dilation rates, and the number of times the model can be stacked. While the number of filters and their shape are quite standard properties of CNNs, the number of dilations, their rate, and the number of stacks of the model, are what contribute to the width of the receptive field of the model. The TCN illustration shown in Fig. 2 contains four layers with an exponentially increasing dilation rate and demonstrates how the output depends on relations with time points which are potentially quite distant. In contrast to RNN approaches, they are not sequentially connected. Indeed, it is this lack of RNN-like long-term sequential connections which contribute to the highly parallelisable structure of the TCN, and thus drastically increase the computational efficiency when training on GPUs.

Two important distinctions between our TCN and the original formulation [13] are that we replace all activation functions within the TCN with *ELUs* [15], and modify it to operate non-causally, rather than in a purely causal way as defined in [12]. In practice, the latter modification means that for any given temporal frame of the input, the (dilated) convolutions extend both forwards and backwards in time. For purely causal operation the dilated convolutions are only performed using past data up to the current temporal frame with no access to future information in the signal. In the context of real-time beat tracking, such causal processing would be essential (as well as the need to adapt many other components of the beat tracking system), but for this paper where all other processing steps are performed offline, we allow full access to the input signal, and seek to benefit from the additional temporal context provided by the non-causal convolutions.

Concerning the specific parameterisation of our TCN, we have attempted to strike a balance between high beat tracking accuracy and the simplicity of the model (*i.e.*, minimising the number of weights to learn). To this end, we propose the parameters shown in the *TCN* section of Table I, and leave a more thorough optimisation of the parameters as a topic for future work. We train the model with the *Adam* optimiser [17], a batch size of 1 and a learn rate of $1e^{-3}$. We reduce the learn rate by a factor of 5 if the loss on the disjoint validation set reaches a plateau and stop training if no improvement in the validation loss is observed for 50 epochs.

4) Obtaining beat predictions: The output of the TCN is a one dimensional beat activation function, intended to exhibit peaks at likely beat locations and be close to zero at all other points in time. For musical examples where the beat activation function approximates this idealised structure, the output can be obtained by a simple peak-picking process. In practice, the beat activation function often has peaks at non-beat locations, or fails to produce peaks at annotated locations, and thus peak-picking alone is typically insufficient to provide a plausible beat tracking output. To this end, [5] proposed using a DBN to decode the beat activation function which yields better global alignment of beats. In this work we use the enhanced and more efficient state space proposed by Krebs *et al.* [18]. Given the beat activation function produced by our TCN has the same temporal resolution and target structure as in [5], we directly reuse this existing DBN together with the default parameters given in [18]: a tempo range of 55–215 beats per minute, and the transition-λ, which aims to control the ability of the model to react to tempo changes, at a value of 100.

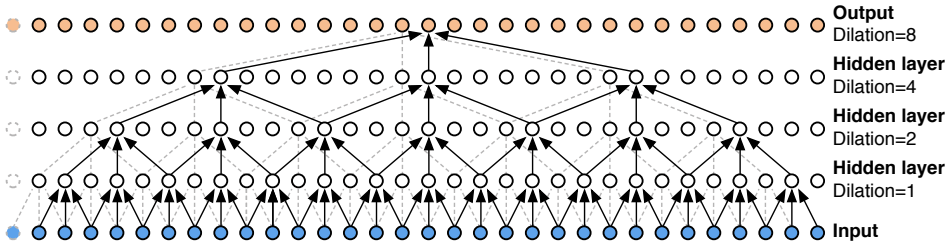

Fig. 2. Overview of the TCN structure (adapted from the original version [13]) to demonstrate non-causal operation. The grey dashed lines show the network connections shifted back one time step.

III. Experiments and Results

To determine the performance of our proposed TCN beat tracking system, we follow the *de facto* objective evaluation methodology by measuring beat tracking accuracy on annotated datasets and comparing it against state-of-the-art reference algorithms [19]. To permit the use of existing annotated datasets, both to train our model and measure its performance, we use 8-fold cross validation. We ensure the separation between testing and training data in each iteration of the cross validation, by using six folds for training, one for validation, and the remaining fold for testing, and rotate the folds eight times until each fold has uniquely been used for testing. The datasets used for cross validation are shown in the upper part of Table II. To provide some insight into the performance on totally unseen data, the *GTZAN* dataset [20] with the beat annotations from [21] is included as test data, but excluded from training, both for our TCN approach and the two reference state-of-the-art methods [5], [6].

TABLE II
Overview of the datasets used for training and evaluation.

Dataset	# files	length
Ballroom [22], [23] [1]	685	5 h 57 m
Beatles [19]	180	8 h 09 m
Hainsworth [24]	222	3 h 19 m
Simac [25]	595	3 h 18 m
SMC [26]	217	2 h 25 m
GTZAN [20], [21]	999	8 h 20 m

In Table III we list the beat tracking results using the widely adopted set of F-measure, continuity-based evaluation scores (CMLc, CMLt, AMLc, AMLt), and Information Gain (D), with the latter measured in bits. For further details, see [19]. Note that the results of [5] differ from the original publication since the DBN used for beat inference was updated to be the same as the one used in this paper which yields better performance and is computationally more efficient (cf. Section II-B4, [18]). Results on *GTZAN* were computed by averaging the predictions of all models trained with cross validation (*i.e.*, model bagging) before inferring the final beat locations with the DBN.

[1] We removed the 13 duplicates identified by Bob Sturm: http://media.aau.dk/null_space_pursuits/2014/01/ballroom-dataset.html

TABLE III
Overview of beat tracking performance.

	F-measure	CMLc	CMLt	AMLc	AMLt	D
Ballroom						
TCN	0.933	0.864	0.881	0.909	0.929	3.456
BLSTM [5]	0.917	0.832	0.849	0.905	0.926	**3.539**
BLSTM [6]	**0.938**	**0.872**	**0.892**	**0.932**	**0.953**	3.397
Hainsworth						
TCN	0.874	0.755	0.795	**0.882**	**0.930**	**3.518**
BLSTM [5]	**0.884**	**0.769**	**0.808**	0.873	0.916	3.507
BLSTM [6]	0.871	0.732	0.784	0.849	0.910	3.395
SMC						
TCN	**0.543**	**0.315**	**0.432**	**0.462**	**0.632**	**1.574**
BLSTM [5]	0.529	0.296	0.428	0.383	0.567	1.460
BLSTM [6]	0.516	0.307	0.406	0.429	0.575	1.514
GTZAN						
TCN	0.843	0.695	0.715	0.889	0.914	**3.096**
BLSTM [5]	**0.864**	**0.750**	**0.768**	**0.901**	**0.927**	3.071
BLSTM [6]	0.856	0.716	0.744	0.876	0.919	3.019

Inspection of Table III demonstrates that across datasets and evaluation methods, the performance of our proposed approach is highly comparable to the existing state of the art, with this pattern holding both for those datasets included in the cross validation and the withheld *GTZAN* dataset. For the *Ballroom*, *Hainsworth*, and *GTZAN* datasets performance is on a very high level, irrespective of evaluation method. Conversely, the *SMC* dataset reveals a significant and expected drop in performance across all methods due to the large proportion of highly challenging musical excerpts. However, performance is highest for our proposed method.

Perhaps what is most noteworthy about our TCN approach is that it can maintain competitive performance but with two distinct computational advantages over the state of the art. Putting aside the approach in [6] which also estimates downbeats and is thus far more complex, our TCN approach uses fewer than 35% of the weights of the BLSTM [5] (21,809 vs. 67,301). Furthermore, the learning of the TCN weights occurred at a rate of approximately 2 seconds/hour of audio on a recent GPU, compared to 2 minutes/hour of audio for the BLSTM, offering a 60x speed-up in training time. Put in context, this implies that our TCN can encode knowledge about the beat structure in music in a more compact representation than the BLSTM, and it can learn this information at a considerably

faster rate. This holds significant promise for learning on very large annotated datasets in practical computation time, *i.e.* in the order of hours rather than weeks, and therefore facilitating multiple training runs with different hyperparameter settings. One limitation of our proposed approach is that the inference is moderately slower than the BLSTM [5], but still much faster than real-time, processing 1 minute of audio in roughly 4–5 s on a recent laptop using only the CPU.

IV. Discussion and Conclusions

We have proposed a new approach for musical audio beat tracking using Temporal Convolutional Networks. Inspired by the well-known *WaveNet* generative model for raw audio signals [13], we re-purpose it to perform dilated convolutions along the temporal dimension of a jointly learned 16 dimensional feature vector in order to predict the locations of musical beats. Since the temporal resolution of the time-frequency representation is drastically lower than that of raw audio signals, this leads to a substantial decrease in the number of weights, which can be trained extremely efficiently, and is consequently less prone to overfitting.

In comparison with state-of-the-art recurrent beat trackers, we demonstrate that our TCN approach can achieve equivalent performance across a diverse range of annotated musical material, and improved performance on the dataset considered the most challenging for beat tracking. Furthermore, this high performance is embedded in a model which uses proportionally far fewer dimensions in its input representation by leveraging convolutional layers to implicitly learn a compact feature representation which is able to model the (local) temporal structure without explicitly encoding this information in the input representation, as done by existing recurrent approaches.

These promising initial results achieved with the TCN suggest significant potential for future work, including: learning the beats directly from the audio signal itself; simultaneously modelling beat and downbeats; developing a real-time approach using a causal TCN; and exploring TCNs for other time-based music analysis tasks such as chord recognition, note transcription, and structural segmentation.

References

[1] M. Goto and Y. Muraoka, "A beat tracking system for acoustic signals of music," in *Proc. of the 2nd ACM Intl. Conf. on Multimedia*, 1994, pp. 365–372.

[2] M. E. P. Davies, M. D. Plumbley, and D. Eck, "Towards a musical beat emphasis function," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2009, pp. 61–64.

[3] A. Klapuri, A. Eronen, and J. Astola, "Analysis of the meter of acoustic musical signals," *IEEE Transactions Speech and Audio Processing*, vol. 14, no. 1, pp. 342–355, 2006.

[4] S. Böck and M. Schedl, "Enhanced beat tracking with context-aware neural networks," in *Proc. of the 14th Intl. Conf. on Digital Audio Effects*, 2011, pp. 135–139.

[5] S. Böck, F. Krebs, and G. Widmer, "A multi-model approach to beat tracking considering heterogeneous music styles," in *Proc. of the 15th Intl. Society for Music Information Retrieval Conf.*, 2014, pp. 603–608.

[6] ——, "Joint beat and downbeat tracking with recurrent neural networks," in *Proc. of the 17th Intl. Society for Music Information Retrieval Conf.*, 2016, pp. 255–261.

[7] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. of the European Conf. on computer vision*. Springer, 2014, pp. 818–833.

[8] S. Durand, J. P. Bello, B. David, and G. Richard, "Feature Adapted Convolutional Neural Networks for Downbeat Tracking," in *Proc. of the 41st IEEE Intl. Conf. on Acoustics, Speech and Signal Processing*, 2016, pp. 296–300.

[9] A. Gkiokas and V. Katsouros, "Convolutional neural networks for real-time beat tracking: A dancing robot application," in *Proc. of the 18th Intl. Society for Music Information Retrieval Conf.*, 2017, pp. 286–293.

[10] R. Vogl, M. Dorfer, G. Widmer, and P. Knees, "Drum transcription via joint beat and drum modeling using convolutional recurrent neural networks," in *Proc. of the 18th Intl. Society for Music Information Retrieval Conf.*, 2017, pp. 150–157.

[11] M. Fuentes, B. McFee, H. C. Crayencour, S. Essid, and J. P. Bello, "Analysis of common design choices in deep learning systems for downbeat tracking," in *Proc. of the 19th Intl. Society for Music Information Retrieval Conf.*, 2018, pp. 106–112.

[12] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[13] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *CoRR abs/1609.03499*, 2016.

[14] J. Schlüter and S. Böck, "Improved musical onset detection with convolutional neural networks," in *Proc. of the 39th IEEE Intl. Conf. on Acoustics, Speech and Signal Processing*, 2014, pp. 6979–6983.

[15] D. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)," in *Proc. of the 4th Intl. Conf. on Learning Representations*, 2016.

[16] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[17] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. of the 3rd Intl. Conf. for Learning Representations*, 2015.

[18] F. Krebs, S. Böck, and G. Widmer, "An Efficient State Space Model for Joint Tempo and Meter Tracking," in *Proc. of the 16th Intl. Society for Music Information Retrieval Conf.*, 2015, pp. 72–78.

[19] M. E. P. Davies, N. Degara, and M. D. Plumbley, "Evaluation methods for musical audio beat tracking algorithms," Centre for Digital Music, Queen Mary University of London, Tech. Rep. C4DM-TR-09-06, 2009.

[20] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, 2002.

[21] U. Marchand and G. Peeters, "Swing ratio estimation," in *Proc. of the 18th Intl. Conf. on Digital Audio Effects*, 2015, pp. 423–428.

[22] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano, "An experimental comparison of audio tempo induction algorithms," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 5, p. 1832–1844, 2006.

[23] F. Krebs, S. Böck, and G. Widmer, "Rhythmic pattern modeling for beat and downbeat tracking in musical audio," in *Proc. of the 14th Intl. Society for Music Information Retrieval Conf.*, 2013, pp. 227–232.

[24] S. Hainsworth and M. Macleod, "Particle filtering applied to musical tempo tracking," *EURASIP Journal on Applied Signal Processing*, vol. 15, pp. 2385–2395, 2004.

[25] F. Gouyon, "A computational approach to rhythm description — audio features for the computation of rhythm periodicity functions and their use in tempo induction and music content processing," Ph.D. dissertation, Universitat Pompeu Fabra, 2005.

[26] A. Holzapfel, M. E. P. Davies, J. R. Zapata, J. L. Oliveira, and F. Gouyon, "Selective sampling for beat tracking evaluation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 9, pp. 2539–2548, 2012.