Beat Tracking: Is 44.1 kHz Really Needed?

Matěj Ištvánek, Štěpán Miklánek
Department of Telecommunications, Faculty of Electrical Engineering and Communication
Brno University of Technology, Technicka 12, 616 00 Brno, Czech Republic
matej.istvanek@vut.cz, stepan.miklanek@vut.cz

***Abstract*—Beat tracking is essential in music information retrieval, with applications ranging from music analysis and automatic playlist generation to beat-synchronized effects. In recent years, deep learning methods, usually inspired by well-known architectures, outperformed other beat tracking algorithms. The current state-of-the-art offline beat tracking systems utilize temporal convolutional and recurrent networks. Most systems use an input sampling rate of 44.1 kHz. In this paper, we retrain multiple versions of state-of-the-art temporal convolutional networks with different input sampling rates while keeping the time resolution by changing the frame size parameter. Furthermore, we evaluate all models using standard metrics. As the main contribution, we show that decreasing the input audio recording sampling frequency up to 5 kHz preserves most of the accuracy, and in some cases, even slightly outperforms the standard approach.**

***Index Terms*—Beat tracking, music information retrieval, temporal convolutional networks, machine learning.**

I. Introduction

In Music Information Retrieval (MIR), one of the core tasks is beat tracking or beat detection. It aims at detecting "tactus" positions in an audio signal — described as "the most comfortable foot-tapping rate when unconsciously tapping to a piece of music" [1]. Early conventional approaches to beat tracking usually utilized a two-stage strategy. First, an onset detection function was computed from time-frequency representations, such as spectrograms or mel-spectrograms. Then a post-processing phase with prior musical knowledge was implemented to determine which onsets might correspond to beats. Well-known examples of non-machine learning approaches are BeatRoot [2] or beat tracking based on dynamic programming by D. Ellis [3]. Furthermore, a method called Predominant Local Pulse [4] that captures local tempo deviations was implemented, for example, in [5].

Over the years, numerous beat tracking methods have been proposed, ranging from rule-based systems and probability methods to machine learning models. With the rise of deep learning techniques, beat tracking has significantly shifted toward data-driven deep neural networks. The increasing availability of data and their annotation[1] helped to outperform every conventional non-machine learning beat tracker (see MIREX results[2]). However, expressive music and genres with more complex metric and rhythmic structures are still considered highly challenging. The Recurrent Neural Networks (RNN) proved that data-driven approaches provide better results than conventional systems [6], [7]. Furthermore, the multi-model approach [8] was implemented to reflect different rhythms depending on the music style and genre. Selecting the best-performing model as the state-of-the-art is challenging, due to the absence of a beat tracking competition (the last MIREX beat tracking competition ended in 2019, and further evaluation is based on the beat tracking community). However, many studies use n-fold cross-validation and similar datasets, achieving more than 90 % F-score (explained in Section III-C) for non-classical and less expressive music.

The Temporal Convolutional Networks (TCNs), based on the original implementation of WaveNet [9], are one of the newest methods for beat tracking and usually achieve the highest F-score. The well-known examples are [10] and [1]. All mentioned models use time-frequency representation (modified spectrograms). Authors in [11] show an end-to-end approach using time-domain representation, feeding raw audio samples into the TCN, achieving similar results as state-of-the-art systems. It is also possible to train beat, downbeat, and tempo activation functions jointly [1], [12], solving more tasks with just one model. For more details about deep learning beat and downbeat tracking TCN models, we refer to [13].

In this paper, we implement multiple TCN beat tracking systems and evaluate their abilities to detect beats on standard datasets. We add skip connections to the networks as one of the possible network modifications and treat them as separate models. As the main contribution, we train five models on 44.1, 22.05, 11.025, and 5.5 kHz input sampling rates to demonstrate how much higher-frequency information is needed for the beat tracking task. We show that lower sampling rates slightly outperform the standard approach with 44.1 kHz input signal in most models. This may be useful for applying beat tracking systems in other MIR-related tasks, such as improving the synchronization accuracy when used jointly with Dynamic Time Warping methods [14] without the need for resampling. The results indicate that even a system trained on the 5.5 kHz input audio signal is comparable to the standard 44.1 kHz model. The higher frequency content seems redundant for the universal beat tracking task.

The rest of the paper is organized as follows. Section II describes the architecture, pre-processing, and models of TCN beat tracking systems. Section III explains the training and evaluation process. Finally, Section IV shows the results followed by discussion and conclusions in Section V.

[1]Beat annotation consists of discrete time points of beat occurrence in a given audio recording.

[2]https://www.music-ir.org/mirex/wiki/2019:Audio_Beat_Tracking (accessed on 27 March 2023)

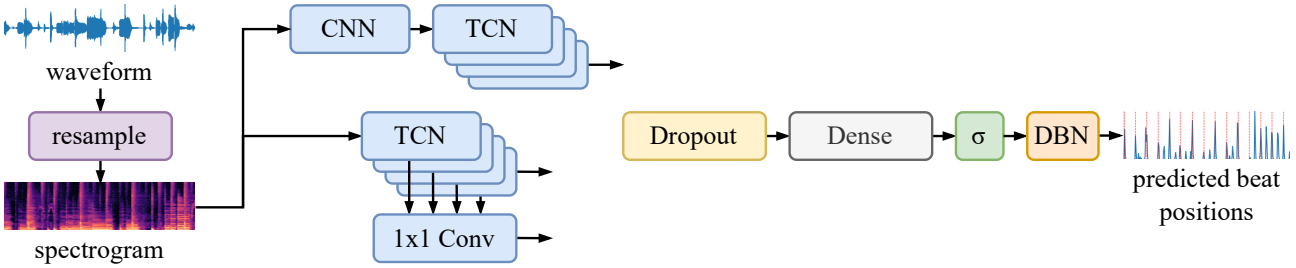

Fig. 1. High-level overview of different approaches to TCN beat tracking.

II. Methods

A. Architecture

Temporal convolutional networks are a class of deep neural networks that have gained popularity in various time-series applications. TCNs consist of multiple layers of temporal convolutions and non-linear activations, allowing them to capture long-term dependencies in sequential data. The ability to model long-term dependencies makes TCNs an attractive option for beat tracking, as the tempo of a musical piece is inherently a temporal pattern, with the exception of expressive performances. For beat tracking, the input to the TCN is a sequence of audio features, usually modified spectrograms. The output of the TCN is a sequence of beat activations, representing the likelihood of a beat occurring at each time step. The beat activations are then post-processed to estimate the exact time positions of beats. We apply a post-processing method called Dynamic Bayesian Network (DBN) [15] as a standard approach to obtain a sequence of beats.

To adapt TCNs to beat tracking, researchers have proposed various modifications to the standard TCN architecture. One of the modifications is to use skip connections, allowing the network to bypass certain layers and directly propagate information from earlier to later ones. Skip connections have been shown to improve the training stability and convergence of TCNs. In our paper, we experiment with three slightly different versions of the TCN beat tracker and modify two of them with additional skip connections.

Figure 1 shows different variants of beat tracking neural networks. The first approach is to use a two-dimensional Convolutional Neural Network (CNN) to extract musically motivated features from the spectrograms and then use a sequence of TCN blocks to capture temporal information. In this work, we experiment with discarding the CNN and using only the TCN blocks to reduce the number of trainable parameters. We also utilize skip connections and combine the intermediate outputs of the TCN blocks using a 1×1 convolutional layer instead of taking only the output from the last TCN block. We evaluate all models using 44.1, 22.05, 11.025, and 5.5 kHz as input sampling rates.

B. Pre-processing

We use frame sizes 2048, 1024, 512, and 256 samples to maintain the same temporal context for 44.1, 22.05, 11.025, and 5.5 kHz sampling rates, respectively. However, the 5.5 kHz sampling rate is a rounded number (the correct rate would be 5 512.5, which is impractical). Therefore, this model does not exactly follow the sampling rate/frame size compromise. We apply the Short-Time Fourier Transform to the audio frames, followed by a filter bank to obtain magnitude spectrograms with logarithmically spaced frequency bins. We refer to [13] for a detailed description of the pre-processing step. The number of frequency bins per octave was set to 12. We ensured that the fps = 100 stayed the same for each scenario. The time resolution also corresponds to the output beat activation function.

C. Models

We use the model `bock_2020` from [1] as a baseline for our experiments. This model includes a CNN to extract relevant spectral features from the spectrograms, which are then fed as input to the first TCN block. The inner structure of the TCN block is shown in Figure 2. The input gets first processed by two parallel dilated convolutional layers with different dilation rates. The output of the layers is then concatenated by the channel dimension, followed by an Exponential Linear Unit (ELU) activation function. The next block is a spatial dropout layer used only during training to prevent overfitting. We set the value of spatial dropout to 0.1 in all experiments. Then, a 1×1 convolutional layer is used to reduce the number of convolution channels in half. The TCN block also contains a residual connection with an additional 1×1 convolutional layer, which helps to retain information from previous TCN blocks.

We also use a simplified TCN block described in [16] and implemented in `simple_tcn` and `tcn_dp`. The structure is depicted in Figure 3. The models are listed below; each row represents different architecture:

- `bock_2020_`*x*
- `simple_tcn_`*x*
- `simple_tcn_skip_`*x*
- `tcn_dp_`*x*
- `tcn_dp_skip_`*x*

To differentiate between various inputs of each model, we added a postfix: *x* stands for 44, 22, 11, or 5, which is equal to 44.1, 22.05, 11.025, and 5.5 kHz sampling rates, respectively.

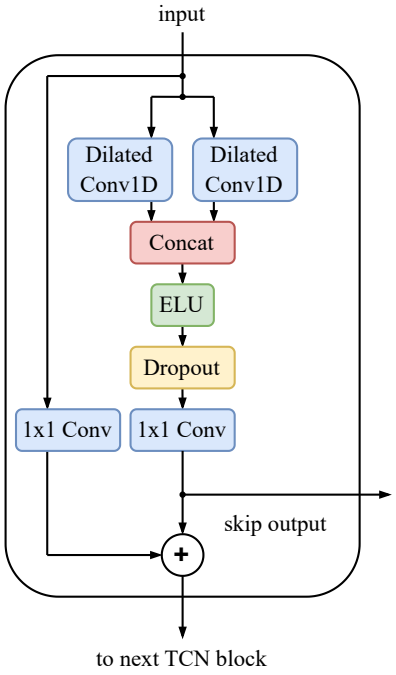

Fig. 2. Diagram of a TCN block used in [1].

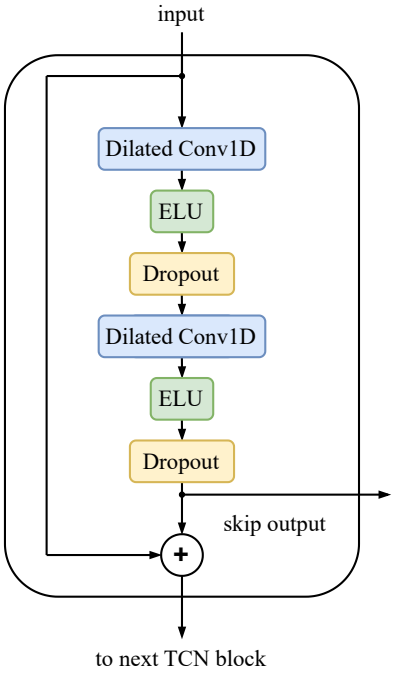

Fig. 3. Diagram of a TCN block presented in [16].

III. Experiments

A. Dataset

In our experiments, we use well-known datasets that have been used for beat tracking tasks for many years. We used the corrected annotations from S. Böck[3] with the exception of the Beatles dataset, in which all annotations were manually corrected to the corresponding ground-truth beat positions based on [16]. All datasets combined consist of 2 263 recordings with a total duration of around 26 hours and 175 127 ground-truth beat annotations. The list of datasets is described below:

- Ballroom [17] – excerpts around 30 s in length, dance music genres such as cha-cha, jive, quickstep, rumba, waltz, or tango,
- Hainsworth [18] – excerpts around 60 s in length, organized into six categories: rock/pop, dance, jazz, folk, classical, and choral music,
- GTZAN [19] – a large dataset containing 30 s excerpts and 10 different genres,
- SMC [20] – excerpts around 40 s in length, specifically selected to be challenging for the state-of-the-art beat tracking systems (for example, expressive performances, local tempo deviations, or complex music compositions),
- Beatles [21] – a collection of songs from Beatles with corrected annotations based on [1] and [16].

B. Training

First, we merge all datasets from Section III-A into one dataset and split it to train, test, and validation sets using the 80/10/10 strategy, respectively. Train and validation sets are shuffled for training, but the test set always contains the same recordings and annotation data.

We train each model on the training data while monitoring the performance on the validation set. We use the following settings: Adam optimizer, binary cross-entropy loss, and reduction of learning rate by a factor of 0.2 if the training does not improve for 10 epochs with the lower bound of learning rate set to 1×10^{-7}. Furthermore, early stopping is called if the change of validation loss is less than 1×10^{-4} for more than 20 epochs. The best checkpoint is then saved as the final model. Contrary to the original implementations, we use an augmentation inspired by [11]. During training, we shift the beat positions forward or back by a random amount between ±70 ms. Table I shows the number of trainable parameters and training time for all models. The average training time of all proposed models combined was 50 minutes. The trainable parameters of the networks ranged from 48 481 to 71 521.

The model `bock_2020` derived from [1] was trained only on 44.1 and 22.05 kHz sampling rates, and without skipping modifications. Changing the network's input size was impossible without changing the inner structure — for example, convolution channels or the dilation factor. However, we decided to keep it in our experiments and show the difference between the original 44.1 and 22.05 kHz models.

The training and validation losses are shown in Figures 4 and Figure 5, respectively. We only display one of the models (`tcn_dp_skip`) with all sampling rates for brevity.

[3]https://github.com/superbock/ISMIR2020 (accessed on 27 March 2023)

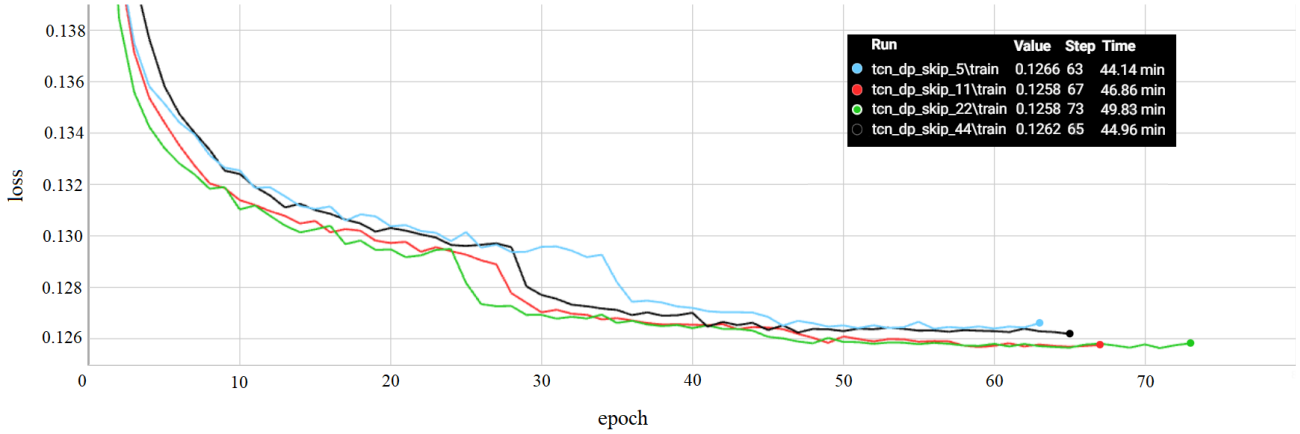

Fig. 4. Loss of the `tcn_dp_skip` model on the training data for each epoch.

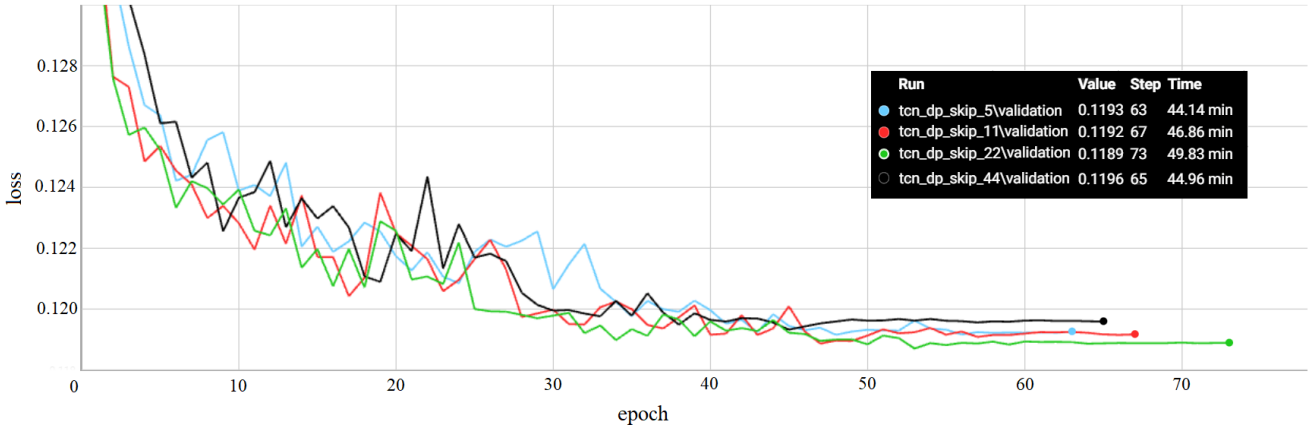

Fig. 5. Loss of the `tcn_dp_skip` model on the validation data for each epoch.

C. Evaluation

We use standard F-score metrics on the test set to evaluate proposed models in terms of prediction accuracy. The F-score is a harmonic mean of precision and recall based on true positives, false positives, and false negatives [22]. To decide if the target beat is within the range of ground-truth beat position, we use a window of length 70 ms, which is a standard value in the beat tracking community [23]. We compute additional metrics (Cemgil, P-score, Goto, and CMLc metrics) and refer to [21] for more details about their implementation. Contrary to other studies, we do not use n-fold cross-validation due to the nature of our experiments.

IV. Results

We evaluated all models on the test set described in Section III-A and III-B. Table II shows the F-score, Cemgil, P-score, Goto, and CMLc metrics for each architecture and sampling rate. Bold numbers indicate the best result for given metrics and architecture. The `bock_2020_22` model achieves the highest scores overall (F-score = 0.928, Cemgil = 0.829, P-score = 0.912, Goto = 0.828, and CMLc = 0.840), surpassing the 44.1 kHz version. Lower sampling rates slightly increase the models' accuracy. An exception is the `tcn_dp_44` model with F-score = 0.912 compared to the `tcn_dp_22` model with F-score = 0.900. The differences, however, are not significant. Furthermore, `tcn_dp_skip_22` is comparable to the state-of-the-art beat tracking model `bock_2020_44` with worse Cemgil and slightly better P-score and CMLc metrics. The difference between the training and validation process of the same architecture but varied input sampling rates is shown in Figures 4 and 5. There is no connection between training time and the input sampling frequency due to the early stopping mechanism.

V. Discussion and Conclusions

In this paper, we trained multiple beat tracking systems with slightly modified architectures on standard datasets and evaluated their performance. Using additional skip connections increased the metrics in most cases, except for `tcn_dp_44` and `simple_tcn_22` models. The well-known `bock_2020` system achieved the highest detection accuracy when trained on a 22.05 kHz audio input sampling rate, although its authors used 44.1 kHz. All networks except `tcn_dp` provided better results when trained on lower sampling rates. This may be thanks

TABLE I

THE NUMBER OF PARAMETERS, TRAIN TIME IN SECONDS FOR EACH MODEL, AND THE MEAN TRAIN TIME FOR EACH ARCHITECTURE.

model	params	train time [s]	mean [s]
bock_2020_22	65 941	3 286	3 099
bock_2020_44	65 941	2 911	
simple_tcn_5	64 701	3 308	3 249
simple_tcn_11	67 341	2 782	
simple_tcn_22	69 981	3 635	
simple_tcn_44	71 521	3 272	
simple_tcn_skip_5	64 483	3 355	3 340
simple_tcn_skip_11	67 123	3 422	
simple_tcn_skip_22	69 763	3 474	
simple_tcn_skip_44	71 303	3 110	
tcn_dp_5	48 481	2 374	2443
tcn_dp_11	49 921	2 180	
tcn_dp_22	51 361	2 570	
tcn_dp_44	52 201	2 648	
tcn_dp_skip_5	48 683	2 777	2 918
tcn_dp_skip_11	50 123	2 951	
tcn_dp_skip_22	51 563	3 117	
tcn_dp_skip_44	52 403	2 826	

TABLE II

BEAT TRACKING EVALUATION OF ALL MODELS ON THE TEST SET USING STANDARD METRICS. BOLD NUMBERS INDICATE THE BEST RESULT FOR GIVEN METRICS AND ARCHITECTURE.

model	F-score	Cemgil	P-score	Goto	CMLc
bock_2020_22	**0.928**	**0.829**	**0.912**	**0.828**	**0.840**
bock_2020_44	0.925	0.815	0.904	0.806	0.821
simple_tcn_5	0.907	**0.799**	0.889	0.771	0.799
simple_tcn_11	0.900	0.773	0.878	0.744	0.778
simple_tcn_22	**0.917**	**0.799**	**0.903**	**0.789**	**0.823**
simple_tcn_44	0.907	0.765	0.887	0.767	0.798
simple_tcn_skip_5	0.907	0.772	0.890	**0.784**	0.810
simple_tcn_skip_11	**0.915**	**0.778**	**0.894**	0.771	0.801
simple_tcn_skip_22	0.907	0.767	0.890	0.775	0.807
simple_tcn_skip_44	0.909	0.773	0.893	**0.784**	**0.811**
tcn_dp_5	0.899	0.790	0.879	0.767	0.783
tcn_dp_11	0.885	0.804	0.863	0.740	0.763
tcn_dp_22	0.900	**0.805**	0.884	0.758	0.796
tcn_dp_44	**0.912**	**0.805**	**0.896**	**0.806**	**0.813**
tcn_dp_skip_5	0.905	0.751	0.890	0.771	0.799
tcn_dp_skip_11	0.918	0.770	0.900	0.784	0.809
tcn_dp_skip_22	**0.925**	**0.776**	**0.912**	**0.806**	**0.838**
tcn_dp_skip_44	0.909	0.764	0.895	0.775	0.806

to, for example, redundant information in higher frequencies. In most music genres, the beat structure is defined by lower frequencies and specific pulsations. Even 5.5 kHz models show comparable performance, considering many instruments contain overtones and timbre above 2.5 kHz. It seems that 44.1 kHz might not be needed for the beat tracking task. For some applications, the lower input sampling rates may be beneficial, as most of the common music processing pipelines and extraction tools work with 22.05 kHz audio signals. In the future, we want to build on these experiments and release open-source models with different input sampling rates to provide more options for subsequent applications.

REFERENCES

[1] S. Böck, J. S. Cardoso, and M. E. P. Davies, "Deconstruct, analyse, reconstruct: How to improve tempo, beat, and downbeat estimation," in *21st International Society for Music Information Retrieval Conference (ISMIR)*, 2020.

[2] S. Dixon, "An interactive beat tracking and visualisation system," in *Proceedings of the 2001 International Computer Music Conference (ICMC'2001)*, 2001.

[3] D. P. Ellis, "Beat tracking by dynamic programming," *Journal of New Music Research*, vol. 36, no. 1, pp. 51–60, 2007.

[4] P. Grosche and M. Müller, "Extracting predominant local pulse information from music recordings," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 6, pp. 1688–1701, 2011.

[5] ——, "Tempogram toolbox: Matlab implementations for tempo and pulse analysis of music recordings," in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, 2011.

[6] S. Böck and M. Schedl, "Enhanced beat tracking with context-aware neural networks," in *Proceedings of the 14th International Conference on Digital Audio Effects (DAFx)*, 2011.

[7] S. Böck, F. Krebs, and G. Widmer, "Accurate tempo estimation based on recurrent neural networks and resonating comb filters," in *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, 2015.

[8] ——, "A multi-model approach to beat tracking considering heterogeneous music styles," in *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR 2014)*, 2014.

[9] A. van den Oord *et al.*, "WaveNet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499v2*, 2016.

[10] M. E. P. Davies and S. Böck, "Temporal convolutional networks for musical audio beat tracking," in *2019 27th European Signal Processing Conference (EUSIPCO)*, 2019.

[11] C. J. Steinmetz and J. D. Reiss, "Wavebeat: End-to-end beat and downbeat tracking in the time domain," in *151st Audio Engineering Society Convention*, 2021.

[12] S. Böck, F. Krebs, and G. Widmer, "Joint beat and downbeat tracking with recurrent neural networks," in *17th International Society for Music Information Retrieval Conference (ISMIR)*, 2016.

[13] M. E. P. Davies, S. Böck, and M. Fuentes, *Tempo, Beat and Downbeat Estimation*. https://tempobeatdownbeat.github.io/tutorial/intro.html, November 2021, (accessed on 27 March 2023).

[14] Y. Özer, M. Ištvánek, V. Arifi-Müller, and M. Müller, "Using activation functions for improving measure-level audio synchronization," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2022.

[15] F. Krebs, S. Böck, and G. Widmer, "An efficient state-space model for joint tempo and meter tracking," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2015.

[16] T. Suchánek, "Detektor tempa hudebních nahrávek na bázi neuronové sítě," Master Thesis, Brno University of Technology, Department of Telecommunications, 2021.

[17] F. Krebs, S. Böck, and G. Widmer, "Rhythmic pattern modeling for beat and downbeat tracking in musical audio," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2013.

[18] S. W. Hainsworth and M. D. Macleod, "Particle filtering applied to musical tempo tracking," *EURASIP Journal on Advances in Signal Processing*, vol. 15, 2004.

[19] 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.

[20] 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.

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

[22] M. F. McKinney, D. Moelants, M. E. P. Davies, and A. Klapuri, "Evaluation of audio beat tracking and music tempo extraction algorithms," *Journal of New Music Research*, vol. 36, no. 1, pp. 1–16, 2007.

[23] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, D. P. Ellis, and C. C. Raffel, "Mir_eval: A transparent implementation of common mir metrics." in *15th International Society for Music Information Retrieval Conference (ISMIR)*, 2014.