Beat and Downbeat Tracking of Symbolic Music Data Using Deep Recurrent Neural Networks

Yi-Chin Chuang and Li Su
Dept. of Computer Science and Engineering, National Chung Hsing University, Taiwan
Institute of Information Science, Academia Sinica, Taiwan
Email: ychin.chuang@gmail.com, lisu@iis.sinica.edu.tw

***Abstract*—Musical beat tracking is one of the most investigated tasks in music information retrieval (MIR). Research endeavors on this task have mostly been focused on the modeling of audio data representations. In contrast, beat tracking of symbolic music contents (e.g., MIDI, score sheets) has been relatively overlooked in the past years. In this paper, we revisit the task of symbolic music beat tracking and present to solve this task with modern deep learning approaches. To the extent of our knowledge, it is the first time that utilizing deep learning approaches to track beats and downbeats of symbolic music data. The proposed symbolic beat tracking framework performs joint beat and downbeat tracking in a multi-task learning (MTL) manner, and we investigate various types of networks which are based on the recurrent neural networks (RNN), such as bidirectional long short-term memory (BLSTM) network, hierarchical multi-scale (HM) LSTM, and BLSTM with the attention mechanism. In the experiments, a systematic comparison of these networks and state-of-art audio beat tracking methods are performed on the MusicNet dataset. Experiment results show that the BLSTM model trained specifically on symbolic data outperforms the state-of-the-art beat tracking methods utilized on synthesized audio. Such a comparison of performance also indicates the technical challenges in symbolic music beat tracking.**

I. Introduction

Human's perception of beats creates the fundamental temporal structure of music. Beat tracking is the task of modeling the periodicity and metrical accents within a naturally performed rhythm. Automatic beat tracking of music has therefore been one of the most widely investigated tasks in music information retrieval (MIR) research [1]. Early discussions of this task, most of which were made in the last century, were on *symbolic beat tracking*, the beat tracking task on MIDI- or score-like music data containing the onset, duration, and pitch values of each note [2]. In the past two decades, tremendous research efforts have been shifted to *audio beat tracking*, where real-world performed audio rather than symbolic data are taken as input [3]. Being a rapidly developed field, the approaches to audio beat tracking have also undergone huge paradigm shifts, as they range from audio features [4], dynamic programming [5], state-space models [6], to today's deep learning [7], [8]. Currently, most of the state-of-the-art beat tracking methods and tools are designed for audio signals only. In contrast, symbolic beat tracking is rarely discussed, with solutions outdated, though it is never proven a solved task. Since symbolic beat tracking is important as it represents a key portion of musical language modeling (i.e. inferring the temporal structure from a note sequence), and has various practical application such as score parsing [9], automatic accompaniment [10], [11], and music generation [12], it would therefore be of high advantage to revisit the task of symbolic beat tracking under the viewpoint of modern deep learning techniques.

This paper represents a revisit of symbolic beat and downbeat tracking, as well as a first attempt to solve the symbolic beat tracking problem with modern deep learning methods. We argue that symbolic beat tracking is still a challenging task in some aspects, but is highly overlooked in contrast to audio beat tracking. Specifically, symbolic beat tracking is a task to infer the beats without any information on human-performed music accents and dynamics. Since state-of-the-art audio beat tracking models are mostly based on the recurrent neural networks (RNN), we manage to solve the symbolic beat and downbeat tracking problem with three models based on RNN, namely the bidirectional long-short-term memory (BLSTM) RNN [13], BLSTM RNN with the attention mechanism [14], and the hierarchical multi-scale RNN (HM-RNN) [15]. To the best of our knowledge, this is the first time that track symbolic music beats through deep neural networks and compare the tracking results systematically. The proposed models are constructed with a multi-task learning (MTL) manner where beat and downbeat are learned jointly. Results indicate that symbolic beat tracking is far from being solved, while competitive performance can be achieved by applying data-driven learning approaches with recurrent neural networks.

II. Related work

Automatic beat tracking on the symbolic music data (such as scores, MIDI note lists, and note onset sequences) has been investigated for a much longer time than audio beat tracking. Pioneer works in this direction are mostly based on rule-based algorithms. Dixon proposed ruled-based tempo estimation and beat tracking algorithms based on the clustering of inter onset intervals (IOI) and a multi-agent beat testing framework [2]. This algorithm is used in pretty_midi, which is one of the few open-source packages for the MIR of symbolic music. Besides this rule-based approach, Large used nonlinear oscillators to perform beat tracking [16], and Desain and Honing proposed a connectionist approach for the quantization of musical time [17]. Temperley described the metrical, rhythm, and beat structure of music jointly with the stream, harmony, and note patterns in a Bayesian probability model [18]. Similar approaches can also be seen in [19]. Tempo and beat inference

of MIDI are highly related to the application of rendering readable score sheets from human-performed MIDI. Grohganz et al. dealt with this problem by analyzing the inconsistency between the estimated tempo and the performance [9].

In contrast to symbolic music beat tracking, audio beat tracking has caught wide attention in the past decade, where various novel techniques were developed with efficient tools published. Examples include Ellis' dynamic programming approach in the librosa library [5] and Boeck's recurrent neural network (RNN) for beat and downbeat tracking in the madmom library [7], to name but a few. Audio beat tracking has been the most evaluated task in the MIREX campaign. Particularly, deep learning approaches have been proven to be the state of the art in audio beat tracking and have achieved over 95% of F1-scores on some standard evaluation datasets. Utilizing neural networks on the beat tracking of symbolic music data is, however, a rather unexplored task.

III. Proposed Method

This section describes the proposed beat and downbeat tracking methods on symbolic music data. In our system, first, symbolic data representations and labels are extracted from the MIDI and score data. Second, neural network models are trained on the extracted data representations and labels. The individual blocks of this system will be described in more detail in the following sections.

A. Data representation

The input data representation used in this work is directly extracted from MIDI data. MIDI data are a list of notes, each of which contains at least three attributes, namely onset time, offset time (or note duration), and pitch value. Onset and offset are recorded in seconds, and pitch values are represented in terms of MIDI numbers. A 2-D piano roll $P := \{p_t\}_{t=1}^{T}$ is extracted from a music clip, where T is the number of frames. As shown in Figures 2 to 4, the ith frame of P, $p_t \in \{0,1\}^{88}$, is a multi-hot 88-D vector which represents pitch activation indexed from A0 to C8: value 1 in p_t represents activated pitch and value 0 represents silence. The frame rate of the piano roll is 100 Hz, that means, there are 100 frames per second.

The data representation contains four parts: pitch profile, onset profile, spectral flux, and inter-onset interval (IOI). The pitch profile is simply the frame-level piano roll p_t. Similarly, the onset profile $o_t \in \{0,1\}^{88}$ represents the onset events of the activated pitches: value 1 represents an onset event of a certain pitch index occurs, and value 0 represents that no onset event is activated. Spectral flux[1] s_t is the sum of the temporal increments (i.e. positive) in P at the tth frame, and is therefore defined as:

$$s_t := \|\mathrm{ReLU}(p_t - p_{t-1})\|_1 \,, \tag{1}$$

where $\mathrm{ReLU}(\cdot)$ is the rectified linear unit, defined as $\mathrm{ReLU}(x) = x$ if $x > 0$, and 0 otherwise. $\|x\|_1$ represents the l_1 norm of x, which is the sum over all the elements of x in absolute value. The IOI i_t calculates the duration between the onset time of the current note and the onset time of its previous note. More specifically, if there is an onset event at t, and the nearest onset event before t is at u, then $i_t = t - u$. If there is no onset events at t, then $i_t = 0$. Both the spectral flux and IOI are utilized to further enhance the onset information. The above-mentioned four data representations are concatenated into one vector $[p_t, o_t, s_t, i_t] \in \mathbb{R}^{178}$, and are then fed into the neural network.

As the input of the neural network, each music clip is segmented into an overlapped packed sequence with a length of 12 seconds (i.e. 1,200 frames) and an overlap of 6 seconds. It should be noted that in this study, the output labels are represented without label extension. The beat and downbeat annotations are labeled with linear interpolation over the given note value data, as will be described in Section IV-B.

B. Recurrent Neural Network

The models applied to symbolic music beat tracking are on the basis of the Recurrent Neural Network (RNN) [13], which is a classical neural network that is used to analyze sequential data prediction. Current state-of-art models on audio beat tracking are also mostly based on RNN. In this paper, variants of RNN with BLSTM cells are considered in the modeling stage. The architecture of the basic BLSTM RNN is shown in Figure 1(a). Given the entire input data representation $X := \{p_t, o_t, s_t, i_t\}_{t=1}^{T}$, the RNN-based model aims at learning the relationship between the input units, and the model predicts two desired outputs, $b_t \in [0,1]^2$ and $d_t \in [0,1]^2$, which represent the probability of beat and downbeat occurrence at time step t, respectively.

We consider three types of neural networks: the first is the conventional bidirectional LSTM (BLSTM) network, the second is the Hierarchical Multiscale RNN, and the third is the BLSTM with attention mechanism. Since BLSTM is relatively well-known in the field, in the following only the Hierarchical Multiscale RNN and the BLSTM with an attention mechanism are described in detail.

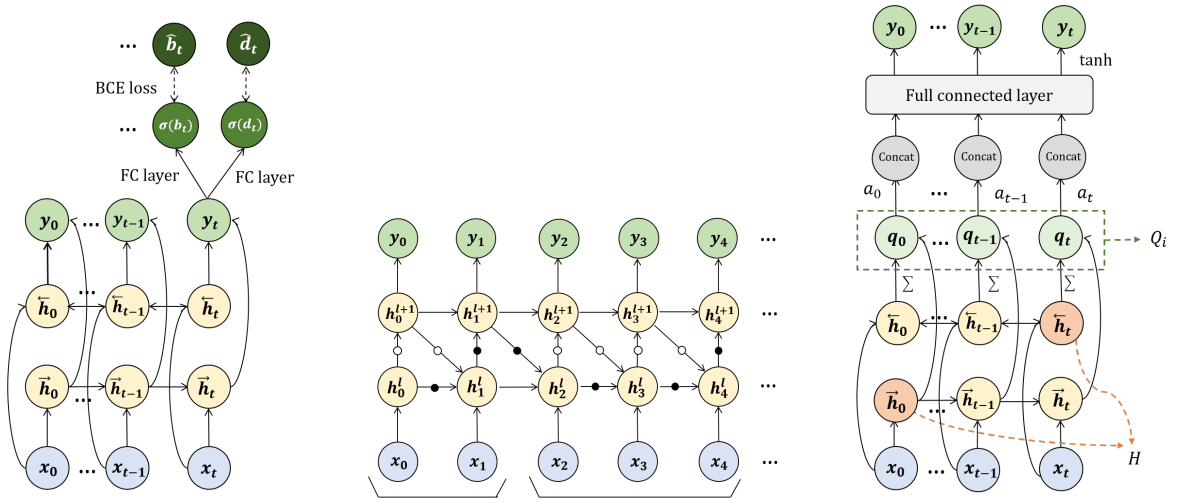
C. Hierarchical Multiscale RNN

The main idea of Hierarchical Multiscale RNN (HM-RNN) is to capture the hierarchical structure with different time-scale in sequential data. This model was originally proposed to cope with natural language processing tasks, we consider this model here since symbolic music data possess hierarchical structures as well. The architecture of the HM-LSTM is shown in Figure 1(b). Each layer l in the Hierarchical Multiscale LSTM (HM-LSTM) architecture computes:

$$h_t^l, c_t^l, z_t^l = F^l(c_{t-1}^l, h_{t-1}^l, h_t^{l-1}, h_t^{l+1}, z_{t-1}^l, z_t^{l-1}). \tag{2}$$

where updated function F is served for HM-LSTM cell. Three states above are h, c, and z represent hidden state, cell state, and boundary state respectively, where the latest one is introduced to divide a sequence into sub-sequences and to create the hierarchical relationship among them. The

[1]Though the data representation here is the flux of piano roll rather than spectrum, we follow the literature and keep using the term "spectral flux."



(a) BLSTM network and outputs (b) HM-RNN (c) BLSTM with attention mechanism

Fig. 1: The network structures considered in this work.

major difference between the standard LSTM and HM-LSTM is the latter provided three operations: UPDATE, COPY, and FLUSH to for each cell state and hidden state at time step t. A boundary will be conducted with the UPDATE operation, while the COPY operation simply passes the hidden state and cell state from the previous time step. If the end of a certain sub-sequence is detected and the initialization of the next sub-sequence is needed, the FLUSH operation will be executed. An in-depth description of these operations can be found in [15]. The selection of which operations should be performed at each time step is controlled by the binary value of the boundary state $z = \{0, 1\}$. In other words, with this boundary checking mechanism, a hierarchical multi-scale architecture can be formed automatically, which meets the hierarchical character of the beats in music theory. In this paper, two hidden layers ($l = 2$) are used. Both sizes of h state and c state are set as 25.

D. Attention Mechanism

The architecture of the basic BLSTM with attention mechanism is shown in Figure 1(c). The idea of an attention mechanism is to focus at each time step on certain elements of the sequence data. It has been adopted to current neural networks, and has shown improvement in learning long-term dependencies. Accordingly, the attention mechanism leads to better performances on various sequence-to-sequence learning tasks. Since the beat tracking is a sequence-to-sequence learning task as well, we therefore consider an additional attention layer to the BLSTM model in this paper. With the attention mechanism, the modified BLSTM output Q_o is computed as:

$$A = \text{softmax}(Q_i H^\top H), \tag{3}$$

$$Q_o = \tanh(W(A; Q_i) + b). \tag{4}$$

Here, the outputs of the forward and backward parts from the last BLSTM layer are summed up, and the result is denoted as Q_i, whose dimension is (M, T, N), where M is the batch size, T is the number of time steps, and N is the number of hidden units. Since BLSTM contains two directions, the concatenation of each time steps for l-layer BLSTM is denoted as H in the shape of $(2l, M, N)$. In this paper, we set $l = 2$, $M = 8$, and $N = 25$. In order to guide the model to focus on information relevant to the labels, we multiply the matrix Q_i by H, where Q_i contains features at each time step, and H is the last time step that had passed through the entire music clip. The resulting multiplication therefore is an alignment score of Q_i with respect to H. We then take softmax on the alignment score, and multiply it by H to obtain the attended feature A, which implies the influence of Q_i on H. The operation $(A; Q_i)$ in (4) represents the concatenation of two given features: attended feature A and raw feature from BLSTM Q_i along the last dimension. The matrix $W \in \mathbb{R}^{x \times 2x}$ is a learnable matrix in a fully connected layer.

In the following of this paper, the three models are denoted as **BLSTM**, **HM-LSTM**, and **BLSTM-Attn**, respectively.

E. Network architecture

To stable the hidden state dynamics in recurrent neural networks, we apply the layer normalization technique on the training data. After normalization, variants of a two-layer (i.e. $l = 2$) RNN-based network followed by fully-connected (FC) layers are used. The variants of RNN can be either BLSTM or HM-LSTM. Each RNN hidden layer has 25 units. In BLSTM-Attn, the attention mechanism is embedded after RNN layers. The outputs of RNN are fed into two separated FC layers for beat and downbeat outputs, where both outputs have a dimension of 2, one indicates the probability of beat/ downbeat occurrence, and the other indicates the probability of non-beat/ non-downbeat occurrence. The sigmoid function σ is then

used as the activation function for the outputs. The network is trained with labeled ground truth $\hat{b}_t \in \{0,1\}$ and $\hat{d}_t \in \{0,1\}$ for every time step t. The multi-task objective function $\mathcal{L}$ is to minimize the binary cross entropy (BCE) between $\hat{b}_t$ and b_t, and between $\hat{d}_t$ and d_t:

$$\mathcal{L} := \sum_t \left(\mathrm{BCE}(b_t, \hat{b}_t) + \lambda \cdot \mathrm{BCE}(d_t, \hat{d}_t) \right) , \qquad (5)$$

where the value λ controls the weights of both BCE losses. The implementation details can be found in our Github repository.[2]

IV. Experiment

A. Data

We evaluate our networks on the MusicNet dataset [20], [21], which is a collection of 330 freely-licensed classical music recordings, with over one million annotated labels for each note in every recording. We adopted this dataset because it provides not only audio recording and aligned MIDI events, but also annotated values of *start beat* of each note. Here, start beat refers to the note onset time in terms of beats. The ground truths of beat and downbeat can be extracted from such annotation. For example, a note starting from beat number 4 in a music clip with meter 4/4 is on beat (and also on downbeat), while another note starting from beat number 4.25 is not. The term "meter" is a notational convention in music to specify how many beats are contained in each bar. Beat/ downbeat time can therefore be obtained from a linear interpolation process if there is no note onset at the beat/ downbeat position. For instance, given the start beat and onset time of two notes by (B_0, T_0) and (B_1, T_1), the time $\bar{t}$ at beat B ($B \in \mathbb{N}$, $B_0 \leq B < B_1$) can be calculated from:

$$\bar{t} = T_0 + \frac{T_1 - T_0}{B_1 - B_0}(B - B_0) . \qquad (6)$$

We remove the recordings with too many *staccato* or *tremolo* notes from experiments since we observed that the annotations of both kinds of notes in the dataset are not precise enough. After this data cleaning process, we come up with a subset containing 155 recordings for our experiments. In this subset, the training set consists of 111 songs, with the total length being 12h 23m; the validation set consists of 12 songs, with the total length being 1h 27m; and the testing set consists of 31 songs, with the total length being 3h 16m.

B. Baseline

Since audio beat tracking has been a well-developed field, the importance of symbolic beat tracking may be questioned: symbolic music data could be synthesized to audio first, then the symbolic beat tracking could be simply regarded as a subproblem of the audio beat tracking. To take a response to this question and justify the importance of symbolic beat tracking, we adopt previous works on both audio and symbolic beat tracking as the baseline methods.

[2]https://github.com/chuang76/symbolic-beat-tracking

We compare the proposed methods with three state-of-the-art open-source packages on either audio or symbolic beat tracking, namely `madmom` library, `librosa` library, and `pretty_midi` library. The `madmom` library predicts the beat positions with RNNs and a dynamic Bayesian network approximated by a Hidden Markov Model (HMM). That is, we acquire the beat activation function via the `RNNBeatProcessor` function first, then we call the `DBNBeatTrackingProcessor` function to obtain the locations of the beats in seconds. The `librosa` library computes onset envelope first, then uses dynamic programming algorithm to perform beat tracking. It should be noted that the `librosa` library only provides beat tracking. The `pretty_midi` library predicts the beat positions according to MIDI tempo changes. For compound meters, it returns every third denominator note as beat positions. For all other meters, it returns every denominator note as beat positions.

Since the `madmom` and `librosa` libraries only allow audio inputs, we consider two scenarios for comparison: 1) evaluation of the original audio recordings (i.e. human performance) in the MusicNet dataset, and 2) evaluate of the audio data which are synthesized from the note event annotation. In the second scenario, we first obtain MIDI tracks rendered from note event annotation in the MusicNet dataset with `pretty_midi` library, and set the constant velocity (volume) for each note. Subsequently, with the given MIDI tracks, the audio signals are synthesized through software synthesizer `fluidsynth`. The synthesized audio signals are mono-channel sampled at 44.1 kHz. In the following, we refer to the first scenario as **real** and the second scenario as **syn**.

C. Experiment Setup

1) Training procedure: The weights and biases of the neural network are initialized with a uniform random distribution in a range [-0.1, 0.1]. We train the network with stochastic gradient descent (SGD) where the learning rate is 10^{-2} and momentum is 0.9 to minimize binary cross-entropy loss for beat and downbeat tracking tasks. The learning rate is reduced by a factor of 10 every 20 epochs. We select the model whose validation loss is the minimum value before 50 epochs as the best model. With empirical tuning on the validation set, we found that setting $\lambda = 5$ on the objective function (5) gives optimal results on beat and downbeat tracking.

2) Thresholding: To obtain the predicted beat and downbeat position, we apply thresholds on the activated outputs of the neural network. The threshold values are obtained by fine-tuning on the validation set. The tuning range is in $[0, 1]$, and 0.1 is as a step. For BLSTM and BLSTM-attn, the optimal threshold values of beat and downbeat tasks are 0.3 and 0.2, respectively. For HM-LSTM, the optimal threshold values of beat and downbeat tasks are 0.3 and 0.1, respectively.

3) Evaluation Measure: In line with most of the studies on beat and downbeat tracking [7], we report the precision (P), recall (R), and F1-score (F) with the tolerance window of $\pm$70 ms. The F1-score is defined as $2PR/(P + R)$. In

TABLE I: Beat tracking results on the testing set.

Method		Precision	Recall	F1-score
Proposed	BLSTM	0.520	**0.724**	**0.605**
	BLSTM-Attn	**0.522**	0.715	0.603
	HM-LSTM	0.513	0.675	0.583
Baseline	madmom (syn)	0.497	0.641	0.560
	madmom (real)	0.427	0.547	0.480
	librosa (syn)	0.388	0.600	0.471
	librosa (real)	0.277	0.394	0.325
	pretty_midi	0.207	0.303	0.246

our experiments, the whole evaluation is conducted with the `mir_eval` library.

TABLE II: Downbeat tracking results on the testing set.

Method		Precision	Recall	F1-score
Proposed	BLSTM	0.262	0.448	0.331
	BLSTM-Attn	0.264	0.466	**0.337**
	HM-LSTM	0.198	**0.643**	0.303
Baseline	madmom (syn)	**0.319**	0.135	0.190
	madmom (real)	0.286	0.138	0.186
	pretty midi	0.067	0.078	0.072

D. Results

Table I lists the beat tracking results on the test set using various models including BLSTM, BLSTM-Attn, HM-LSTM, madmom (syn/ real), librosa (syn/ real), and pretty_midi. Comparing the F1-scores among the three proposed networks, both BLSTM and BLSTM-Attn outperform HM-LSTM by around 2%. The results of BLSTM and BLSTM-Attn are almost the same. This implies that taking the hidden states from all the time steps into account is of quite limited contribution, probably because the data representation in the symbolic domain is not as rich as in the audio domain. Interestingly, the F1-score of HM-LSTM is lower than BLSTM's. One possible reason is that HM-LSTM predicts beats in a parsimonious manner: while BLSTM tends to predict both on-beat and off-beat positions and gets a high recall, HM-LSTM tends to predict only either on-beat or off-beat positions, where the latter case could result in low precision and low recall.

We then compare the proposed system with baseline models. We observe that, first, in madmom and librosa, using synthesized audio performs better than using real audio. The improvement of synthesized audio over real audio is 8% for madmom and 14.6% for librosa. This is to verify that when performing on synthesized audio, these audio beat tracking methods still work, at least better than on real audio, and can be taken as reliable references to be compared with symbolic beat tracking. Second, we observe that each of the proposed model outperforms the baseline models. More specifically, BLSTM, BLSTM-Attn, and HM-LSTM outperform madmom (syn) by 4.5%, 4.3%, and 2.3%, respectively. This indicates the importance of taking symbolic beat tracking as a specific task independent from audio beat tracking, as state-of-the-art audio beat tracking cannot replace the symbolic beat tracking method in basic neural network models. Finally, the proposed system also outperforms the symbolic beat tracking library pretty_midi, as pretty_midi is rule-based and has less generalizability on large-scale data.

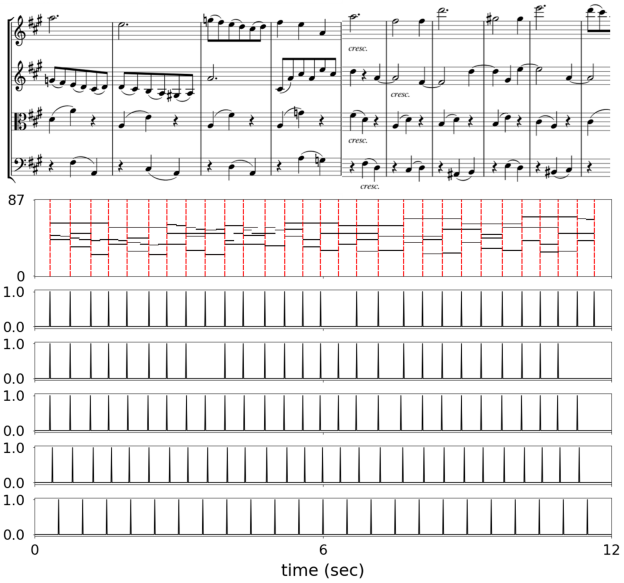


Fig. 2: Beat tracking results on mm. 63-72 in Beethoven's *String Quartet in A major No. 5, Op. 18 , II. Menuetto*. The time signature is $\mathbf{3}/\mathbf{4}$, which means each bar contains three quarter-note beats. Annotations are derived from `2481.csv` in the MusicNet dataset. From top to bottom: sheet music, piano roll representation, and outputs of BLSTM, HM-LSTM, madmom (syn), librosa (syn), and pretty_midi. Ground truth beats are marked in vertical red dashed lines.

Table II lists the results of downbeat tracking. It should be noted that librosa does not support the downbeat tracking task, therefore the downbeat tracking result of librosa is not presented in the Table. A trend similar to Table I is observed; BLSTM and BLSTM-Attn outperform HM-LSTM, and HM-LSTM outperforms other baseline methods. BLSTM outperforms madmom (syn) by 14.1%, and outperforms pretty_midi by 25.9%. These results again show the challenges of symbolic beat tracking, and the importance of data-driven approaches of symbolic beat tracking.

E. Illustration

Figures 2 to 4 illustrate the beat tracking results of various methods on three music samples. Figure 2 illustrates a string quartet sample with regular speed and metrical structures. In this case, the three baseline methods appear to successfully predict all the beat positions, while BLSTM and HM-LSTM miss a few beat positions. The advantage of the proposed approach can be shown in Figure 3. It can be seen that only the neural network models trained on MIDI data (i.e. BLSTM and HM-LSTM) capture meter information correctly. madmom predicts beat position in a stable manner but failed to

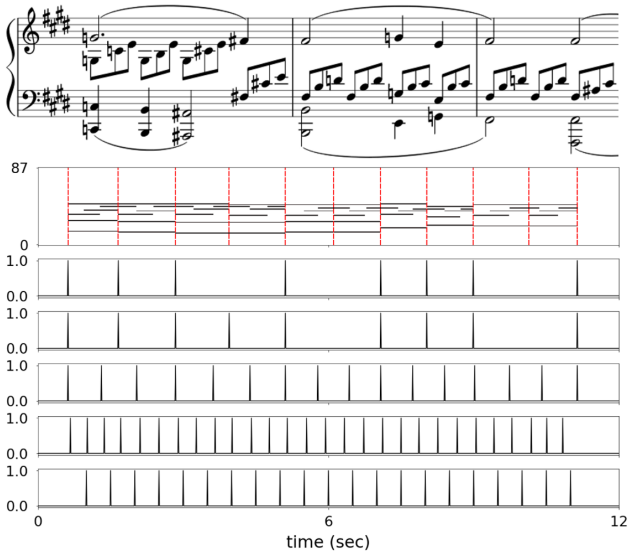


Fig. 3: Beat tracking results on mm. 12-14 in Beethoven's *Piano Sonata No. 14 (Moonlight) in C-sharp minor, Op. 27, I. Adagio sostenuto.* The time signature is $\mathbf{4/4}$, which means each bar contains four quarter-note beats. Annotations are derived from `2350.csv` in the MusicNet dataset. From top to bottom: sheet music, piano roll representation, and outputs of BLSTM, HM-LSTM, madmom (syn), librosa (syn), and pretty_midi. Ground truth beats are marked in vertical red dashed lines.

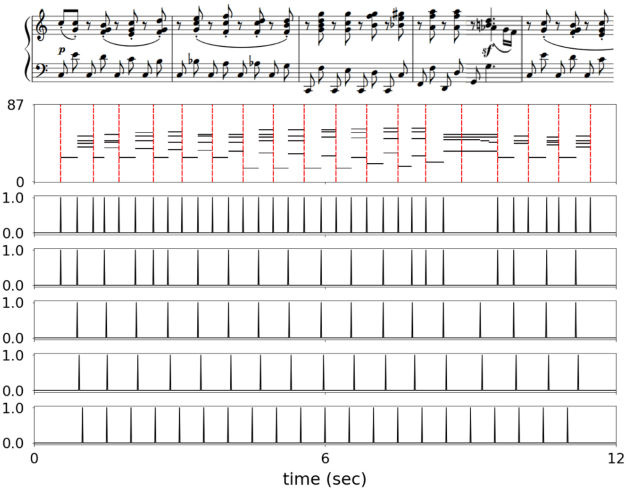


Fig. 4: Beat tracking results on mm. 41-45 in Beethoven's *Piano Sonata No. 10 in G Major, Op. 14, II. Andante.* The time signature is $\mathbf{4/4}$, which means each bar contains four quarter-note beats. Annotations are derived from `2632.csv` in the MusicNet dataset. From top to bottom: sheet music, piano roll representation, and outputs of BLSTM, HM-LSTM, madmom (syn), librosa (syn), and pretty_midi. Ground truth beats are marked in vertical red dashed lines.

specify the boundary of the triplets by predicting a sequence of doublet. In addition, the dynamic-programming-based librosa and the rule-based pretty_midi give rather conservative predictions by taking every eighth note as a beat or every 0.5 second as a beat. Figure 4 presents a challenging case in which most of the notes in the music are at off-beat positions. Here, all the three baseline methods output only the off-beat positions. In contrast, BLSTM outputs both on-beat and off-beat positions, and HM-LSTM outputs off-beat positions and some on-beat positions. As a result, BLSTM and HM-LSTM outperform the baseline methods in this case. Finally, we can also observe the different behaviors between BLSTM and HM-LSTM: HM-LSTM mispredicts most of the beat position at off-beat, only a few predicted beat positions are on-beat. Under the same situation, BLSTM predicts both on-beat and off-beat as beat positions, which leads to a higher F1-score. The tracking results in Table I also demonstrate the performance of BLSTM is better than HM-LSTM.

V. Discussion

As the first attempt to tackle the symbolic beat tracking problem with deep learning, the scope of this paper is limited in technical aspects. The input data representations used in this paper are just a preliminary event set. There is only one dataset for model training; in comparison, state-of-the-art audio beat tracking adopted a much richer set of data for training [7]. Under such limited resources, the classic BLSTM outperforms a more advanced model design, a counter-intuitive result.

Our experiment results however indicate a few open issues for further study. First, symbolic beat tracking is demonstrated far from a trivial problem. With the lack of timbre, energy, and musical accent information in symbolic data, the features to discriminate onset and beat, and the features to discriminate beat and downbeat would be much more difficult to learn, in comparison to the case of audio beat tracking. This might be the reason why the models cannot distinguish between beat and downbeat precisely in some cases. Second, the models tend to ignore the beat predictions when there is a note whose duration is much longer than its neighboring notes, such as a full note at the end of a musical phrase. To address this problem, the Dynamic Bayesian Network (DBN) [22] adopted in `madmom` can be utilized as post-processing to infer to missed beat positions. Third, we found that beat and downbeat annotation of human-performed MIDI data is urgently needed, but such annotation is also not easy to obtain. Finally, it is still unknown whether the recent advance in sequential models (e.g., autoregressive models [23], Transformers [24], etc.) can further improve the performance. This direction is worth investigating based on more delicate data representation and richer sets of data.

VI. Conclusions

In this study, we present the joint beat and downbeat tracking task on symbolic music and cope with this task

with modern deep learning approaches. Our major conclusions are two-fold. First, taking symbolic beat tracking as a task independently from audio beat tracking and building a specific model upon the symbolic data is necessary for the related application. Second, experiment results demonstrate the performances of BLSTM-based methods (BLSTM and BLSTM-Attn) outperform HM-LSTM and other baseline methods. This suggests that BLSTM-based methods can be adopted in the application at the current stage. Future work includes the design of input data representation, and more advance sequence-to-sequence models such as autoregressive models and Transformers for beat and downbeat tracking.

REFERENCES

[1] Stephen Hainsworth, "Beat tracking and musical metre analysis," in *Signal processing methods for music transcription*, pp. 101–129. Springer, 2006.

[2] Simon Dixon, "Automatic extraction of tempo and beat from expressive performances," *Journal of New Music Research*, vol. 30, no. 1, pp. 39–58, 2001.

[3] Masataka Goto, "An audio-based real-time beat tracking system for music with or without drum-sounds," *Journal of New Music Research*, vol. 30, no. 2, pp. 159–171, 2001.

[4] Matthew EP Davies and Mark D Plumbley, "Beat tracking with a two state model," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*. IEEE, 2005, vol. 3, pp. iii–241.

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

[6] Yu Shiu and C-C Jay Kuo, "A hidden markov model approach to musical beat tracking," in *Proc. ICASSP*, 2008.

[7] Sebastian Böck, Florian Krebs, and Gerhard Widmer, "Joint beat and downbeat tracking with recurrent neural networks.," in *ISMIR*. New York City, 2016, pp. 255–261.

[8] EP MatthewDavies and Sebastian Böck, "Temporal convolutional networks for musical audio beat tracking," in *2019 27th European Signal Processing Conference (EUSIPCO)*. IEEE, 2019, pp. 1–5.

[9] Harald Grohganz, Michael Clausen, and Meinard Müller, "Estimating musical time information from performed midi files.," in *ISMIR*, 2014, pp. 35–40.

[10] Matthew EP Davies, Paul M Brossier, and Mark D Plumbley, "Beat tracking towards automatic musical accompaniment," in *Audio Engineering Society Convention 118*. Audio Engineering Society, 2005.

[11] Grigore Burloiu, "An online tempo tracker for automatic accompaniment based on audio-to-audio alignment and beat tracking," in *Sound and Music Computing conference (SMC)*, 2016.

[12] Jean-Pierre Briot, Gaëtan Hadjeres, and François-David Pachet, "Deep learning techniques for music generation–a survey," *arXiv preprint arXiv:1709.01620*, 2017.

[13] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[15] Junyoung Chung, Sungjin Ahn, and Yoshua Bengio, "Hierarchical multiscale recurrent neural networks," in *5th International Conference on Learning Representations (ICLR)*, 2017.

[16] Edward W Large, "Modeling beat perception with a nonlinear oscillator," in *Proceedings of the eighteenth annual conference of the cognitive science society*, 1996, pp. 420–425.

[17] Peter Desain and Henkjan Honing, "The quantization of musical time: A connectionist approach," *Computer Music Journal*, vol. 13, no. 3, pp. 56–66, 1989.

[18] David Temperley, "A unified probabilistic model for polyphonic music analysis," *Journal of New Music Research*, vol. 38, no. 1, pp. 3–18, 2009.

[19] Andrew McLeod and Mark Steedman, "Meter detection and alignment of midi performance.," in *ISMIR*, 2018, pp. 113–119.

[20] John Thickstun, Zaid Harchaoui, Dean P. Foster, and Sham M. Kakade, "Invariances and data augmentation for supervised music transcription," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2018.

[21] John Thickstun, Zaid Harchaoui, and Sham M. Kakade, "Learning features of music from scratch," in *International Conference on Learning Representations (ICLR)*, 2017.

[22] Florian Krebs, Sebastian Böck, Matthias Dorfer, and Gerhard Widmer, "Downbeat tracking using beat synchronous features with recurrent neural networks.," in *ISMIR*, 2016.

[23] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.