TRANSFORMER-BASED BEAT TRACKING WITH LOW-RESOLUTION ENCODER AND HIGH-RESOLUTION DECODER

Tian Cheng **Masataka Goto**
National Institute of Advanced Industrial Science and Technology (AIST), Japan
{tian.cheng, m.goto}@aist.go.jp

ABSTRACT

In this paper, we address the beat tracking task which is to predict beat times corresponding to the input audio. Due to the long sequential inputs, it is still challenging to model the global structure efficiently and to deal with the data imbalance between beats and no beats. In order to meet the above challenges, we propose a novel Transformer-based model consisting of a low-resolution encoder and a high-resolution decoder. The encoder with low temporal resolution is suited to capture global features with more balanced data. The decoder with high temporal resolution is designed to predict beat times at a desired resolution. In the decoder, the global structure is considered by the cross attention between the global features and high-dimensional features. There are two key modifications in the proposed model: (1) adding 1D convolutional layers in the encoder and (2) replacing positional embedding by the upsampled encoder features in the decoder. In the experiment, we achieved the state-of-the-art performance and showed that the decoder produced more precise and stable results.

1. INTRODUCTION

Beat tracking is an important task in Music Information Retrieval (MIR) area with a long history. The task is to predict beat times, a periodic sequence of time instants which people can tap along with, from musical pieces. The first attempt of beat tracking for polyphonic musical audio signals can date back to around 30 years ago [1]. In the past three decades, we see the techniques shifting from signal processing to machine learning. In the most recent deep-learning-based methods, sequence models have been used to produce beat probabilities for each input frame, with the final beat times detected by the HMM on the beat probabilities in the post-processing step. In these models, various sequence models have been used, including Recurrent Neural Network (RNN) [2–4], Temporal Convolutional Network (TCN) [5–7], and Transformers [8–10]. Convolutional Neural Networks (CNNs) are also commonly combined in the models for front-end feature embedding [9, 11, 12].

To produce good beat tracking results, the model needs to consider both local timing and global consistency. This brings a contradiction on choosing the temporal resolution. The problem of using low temporal resolution (i.e., low frame rate) is that we cannot predict beats with precise times. On the other hand, using high temporal resolution (i.e., high frame rate) results in long sequential inputs and imbalanced output labels. The current commonly-used 10ms temporal resolution enables an easy comparison on the results. With such high temporal resolution, the sequential inputs are already relatively long for the RNN, causing the gradient vanishing problem. Using TCN and Transformer helps to solve the gradient vanishing problem, while modeling long sequences can still be challenging. To model long sequences more efficiently, more compact models have been proposed, such as dilated self-attention [9] and linear Transformer [10]. Another problem caused by the high temporal resolution is the data imbalance issue between beats and no beats. Given the same tempo, the higher the temporal resolution is, the more the no-beat labels exist between the beat labels. In order to solve this problem, smoothed labels [7, 9, 13] and weighted loss functions designed for the data imbalance problem [14–16] are applied to achieve more efficient training. The above long sequence modelling issue and data imbalance issue can be more challenging if a higher temporal resolution than 10ms is needed. In fact, there are some commercial music applications that potentially require more temporally precise beat tracking for sample-wise audio editing/mixing/mashups based on beat timings and highly rigid music synchronization.

In order to tackle the contradiction between high and low temporal resolutions, we propose a novel beat tracking model based on the Transformer with low-resolution encoder and high-resolution decoder. With the low temporal resolution, the sequential inputs become shorter and the training data become more balanced, which makes the global structure easier to model by the encoder. At the same time, the beat time precision in the output can still be preserved by the decoder with the high temporal resolution. The Transformer is a good architecture for joining the two parts because the encoder and decoder are not required to be the same length, and features of different dimensions can be jointly learned by the cross attention in the decoder. We modify the original Transformer in several ways to make it work for beat tracking with the proposed combination of the encoder and decoder. First, we

stack 2D convolutional layers for feature learning from the spectral inputs and 1D convolutional layers inside the encoder layers for feature smoothing and dimension adjustment. Second, we use the upsampled encoder feature to replace the position encoding in the decoder. In the experiments, we produced results comparable to the state-of-the-art performance. The analysis of experimental results showed that the decoder not only produced more precise results, but also helped to recover the missing beats and to filter out unwanted peaks between beats, making the beat tracking more stable.

The rest of paper is organised as follows. Section 2 summarises the related work on Transformer-based beat tracking models and multi-scale models. In Section 3, we give a detailed description of the proposed model, especially focusing on the proposed modifications. Section 4 presents the experiments with ablation study, results, and attention visualisation. In the last section, we conclude the paper and show aspects for future improvements.

2. RELATED WORKS

2.1 Transformer in beat tracking

Recently, Transformers have been used for many MIR tasks with promising performance, such as music transcription [17–19], music tagging [20], and beat tracking [8–10]. In the SpectTNT model, Transformer encoders are used for modeling both the spectral and temporal features [8]. The model also combines the Temporal Convolutional Network (TCN) model for better beat tracking results. Since the Transformer is computationally expensive for long sequences, the inputs are divided in 6-second chunks to process. For modelling the long sequences efficiently, more compact Transformers have been applied, including the dilated Transformer in the Beat Transformer model [9] and the linear Transformers for singing beat tracking [10].

These existing methods are based on Transformer encoders, while in our model, we use both the encoder and decoder, which is an important contribution of this paper. By adding the decoder layers, we can set a more reasonable temporal resolution for the encoder input, more specifically, low-temporal-resolution inputs. In other words, in the proposed model, the temporal resolution of the encoder can be independent of the high temporal resolution of the beat tracking output. With the low-resolution encoder, we are able to model the sequences more efficiently and obtain more balanced training data.

2.2 Multi-scale structure

In the proposed model, we leverage features at different scales: low-dimensional features for modeling the global structure and high-dimensional features for predicting precise beat times. Such multi-scale structure has also been used in related domains. In our previous work, we proposed a multi-scale beat tracking model based on the Wave-U-Net, which learned features at different scales from waveform and spectral inputs with downsampling and upsampling blocks [21]. Schreiber et al. achieved tempo estimation by concatenating multi-scale features learned from a series of convolutional layers with different filter size from 32 to 256 [22]. Sun et al. [23] propose a multi-scale structure for tempo estimation by downsampling/upsampling the feature to different scales and combining multi-scale features repeatedly.

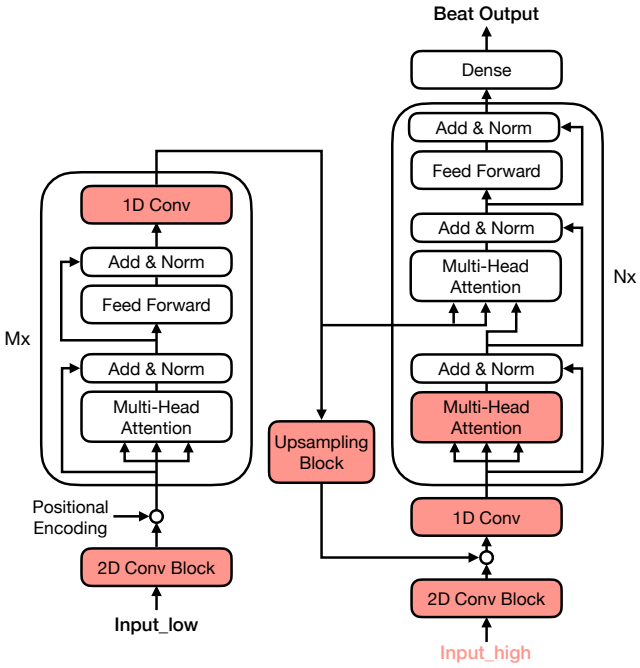


Figure 1: The model architecture of the beat tracking model. The coloured parts indicate the modifications from the original Transformer.

3. MODEL ARCHITECTURE

The proposed model architecture is shown in Figure 1. Our model is based on the Transformer, consisting of both encoder and decoder layers. As we have already written, the key idea is to use a low-resolution encoder for modelling the global structure, depicted on the left side of the figure (starting from "Input_low" denoting the low-resolution input), and a high-resolution decoder for predicting beats more precisely, depicted on the right side of the figure (starting from "Input_high" denoting the high-resolution input).

To make it work for beat tracking, we make some modifications on the original Transformer. The modified parts are coloured in Figure 1, including adding 1D convolutional layers ("1D Conv") in the encoder, replacing the positional embedding by upsampled encoder features (from "Upsampling Block") in the decoder, and stacking 2D convolutional layers for feature learning from the spectral inputs ("2D Conv Block") in both the encoder and decoder.

In the following subsections, we illustrate the proposed model in detail.

Network parameter	Setting
filter size	3×3, 3×3, 3×3, 1×3
maxpooling size	1×3, 1×3, 1×3
activation function	ReLU

Table 1: Parameters used in the convolution block.

Encoder	
Conv Block filter number	48, **64**, 72
encoder layer number	3, 4, **5**, 6
head number	4, 8, 12, **16**
key dim.	8, 16, 24, **32**
inner-layer dim. in feed-forward	32, 48, **64**, 72, 128
Conv 1D filter number	32, **64**, 96
Conv 1D filter size	5, **15**

Decoder	
Conv Block filter number	**32**
decoder layer number	1, **2**
head number	**4**

Table 2: Hyperparameters in the proposed beat tracking model.

3.1 Input features and 2D convolutional layers

We use the Mel-spectrogram as the input features. For the low-resolution encoder, we computed 80-dimensional Mel-spectrogram with a 22050 sample rate and a hop size of 1024, roughly corresponding to a 46 ms temporal resolution (more precisely, 46.44 ms). The high-resolution Mel-spectrogram is computed the same way but in a hop size of 256, roughly corresponding to 12 ms temporal resolution (more precisely, 11.61ms). The dimensions of the inputs are (T, 80) and (4T, 80), respectively, where T is the frame length of the low-resolution input. We choose such resolutions so that the low-resolution can still distinguish beat and no beat frames for fast-tempo pieces, and the high-resolution outputs can be easily compared to those of other methods. In the 2D convolutional block, we stack four 2D convolutional layers and three maxpooling layers for feature embedding, with details show in Table 1.

3.2 Encoder with 1D convolutional layers

The encoder consists of identical encoder layers which process the low-dimensional features. As shown on the left side in Figure 1, each encoder layer includes a multi-head attention sub-layer and a fully connected feed-forward network with residual connections. Before the encoder, we concatenate the features with the positional encoding. Since in the Transformer, the input dimension is not changeable within the encoder layers, we stack a 1D convolutional layer after the feed-forward network for feature smoothing and channel number adjustment.

3.3 Upsampling Block

Another important change of the proposed model is that we replace the original positional encoding by upsampled encoder features for the decoder. In the upsampling block, there are two upsampling layers with linear interpolation. The upsampled features are then concatenated with the high-dimensional features. We also stack a 1D convolutional layer to re-dimension the concatenated features. In the preliminary experiment, we confirmed that the original positional encoding does not work well and the upsampled features worked for indicating rough beat positions. The ablation study for this replacement is presented in Section 4.3.

3.4 Decoder

The decoder processes the high-dimensional features for predicting more precise beats. The decoder layer consists of three components. As shown on the right side in Figure 1, between the multi-head attention sub-layer and a fully connected feed-forward network, there is another multi-head attention sub-layer which computes the cross-attention between the low- and high-dimensional features. We use the decoder as a discriminative model for predicting the output based on the input, rather than a generative model as the original Transformer decoder. Hence we do not need to use the causal mask in the first multi-head attention sub-layer.

3.5 Output Layer and Post-Processing

As shown in Figure 1, we stack a dense layer with the sigmoid activation at the end of the decoder for producing beat outputs. Then, we apply the DBN from Madmom [2] for post-processing. We first take the nearest integer of frame per second (fps) for the post-processing, and then map the results to the original fps.

3.6 Complete Architecture

We apply random search to find the best hyperparameters for the model. We set up a grid of hyperparameter values according to Table 2, and randomly select a subset to compare. The decoder uses the same parameters as the encoder if not present. The finally chosen parameters are shown in bold.

4. EXPERIMENTS

4.1 Data

We train, validate, and test the proposed model by using the standard music datasets with beat annotations as shown in Table 3. For training and validation sets, all musical pieces are segmented into 30-second clips with 50% overlap. Segments from the same musical piece appear only in either the training or validation set to ensure that there is no overlap between the training and validation sets. Test results are obtained on the whole pieces without segmentation.

We do data augmentation for better tempo balance. We follow the strategy in [7] to generate input features for less

Usage	Datasets
training only	Beatles [24], Harmonix [25], 5 RWC datasets [26, 27], tapcorrect [28]
8-fold cross-validation	Ballroom [29, 30], Hainsworth [31], SMC [32]
testing only	GTZAN [33, 34]

Table 3: The usage of the standard datasets for beat tracking evaluation.

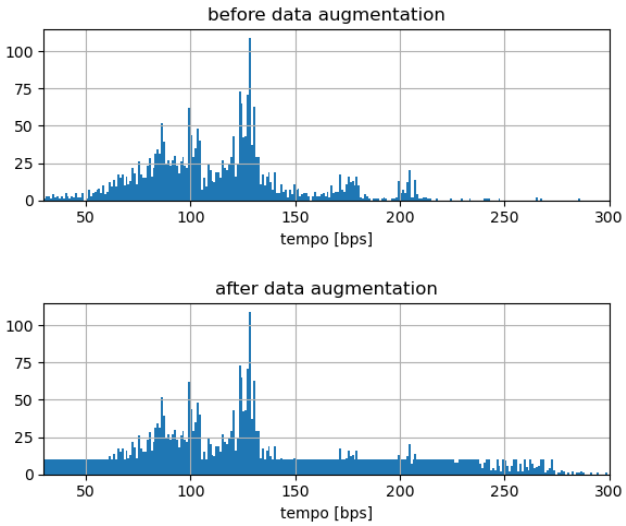


Figure 2: The tempo distribution before and after the data augmentation for the model training.

representative tempos by changing the hop size when computing the mel-spectrogram. The tempo distribution before and after the data augmentation is shown in Figure 2.

Inspired by [9], we also process the input mel-spectral features by Harmonic Percussive Source Separation (HPSS) and obtain the original mel-spectrogram S, the harmonic part H, and the percussive part P. In the preliminary experiment, we compare two way of using HPSS: one way is as data augmentation which triples the training data (i.e., we can use all of S, H, and P with the same beat annotations); the other way is to concatenate three parts, S, H, and P, as the more informative input features. Since the results showed that using HPSS as the data augmentation works better, we decided to take that way.

4.2 Training

In order to train the model effectively, we compare three training methods as shown in Table 4. The first method is training the model (i.e., both the encoder and decoder) from scratch.

For the other two methods, we first temporarily stack a dense layer at the end of the encoder and pre-train the encoder only with the low-resolution labels. Then we initialize the encoder with this pre-trained model and start training the decoder. The second method trains the parameters of the decoder only, by freezing the parameters of this pre-traind encoder. The third method trains the parameters of both the encoder and decoder after the above initialization of the encoder.

Method	Initialization	Encoder parameters
1	None	Trained with decoder
2	Pre-trained encoder	Not trainable in training decoder
3	Pre-trained encoder	Trainable in training decoder

Table 4: Three training methods (the third method was the best).

We choose the third method for training the model because it worked best in our preliminary experiments. The model is trained with binary cross-entropy by using the RMSprop optimiser [35] with a learning rate of 0.0002. The batch size is set to 16 for pre-training the encoder, and 4 for training the whole model.

4.3 Ablation Study

To illustrate the influence of replacing the positional encoding by the upsampled encoder features in the decoder, we show the differences on the training with and without the proposed modification (replacement) in Figure 3. We see that using the upsampled encoder features decreased the validation loss slightly and decreased the training loss in a large degree in comparison to using the positional encoding. This shows that the modified model can learn better from the training data and generalise well in the validation set, resulting in better beat tracking results.

4.4 Evaluation

We evaluate the proposed method with three standard metrics: F-measure with a tolerance window of 70ms, continuity-based metrics CMLt (tracking accuracy on the correct metrical level), and AMLt (tracking accuracy with alternate metrical levels allowed) [24].

4.4.1 The proposed method

In order to validate our model design, besides results on the decoder outputs, we also show results on the pre-trained encoder outputs. In Table 5, "Encoder (Th)" indicates the results obtained by applying a threshold of 0.1 without using the DBN. "Encoder" and "Decoder (Proposed)" results are processed by the DBN in the post-processing step, with the encoder outputs linear interpolated. If we compare the results of "Encoder (Th)" and "Encoder", we observe that the DBN post-processing step increased the performance in all the four datasets, especially for the continuity-based results (CMLt and AMLt). Furthermore, if we compare them with the "Decoder (Proposed)" results, which corresponds to the proposed model, we see performance further increased on the Ballroom, SMC, and GTZAN datasets.

Figure 3: The training and validation losses for training the decoder with the original positional embedding or with upsampled encoder features.

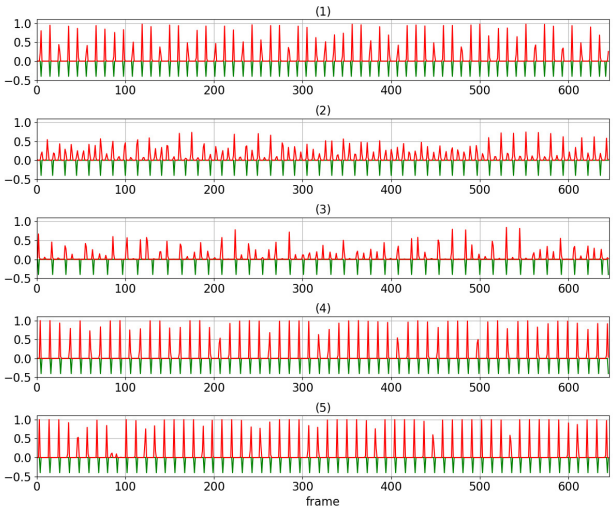
Method	F-measure	CMLt	AMLt
Dataset: Ballroom			
Encoder (Th)	90.7	80.1	85.7
Encoder	93	87.4	96.1
Decoder (Proposed)	95	91.1	96.4
Beat trans [9]	96.8	95.4	96.6
TF trans [8]	96.2	93.9	96.7
TCN [7]	96.2	94.7	96.1
Dataset: Hainsworth			
Encoder (Th)	84.4	66.7	81.8
Encoder	88.2	81	93.4
Decoder (Proposed)	87	76.2	93.6
Beat trans [9]	90.2	84.2	91.8
TF trans [8]	87.7	86.2	91.5
TCN [7]	90.4	85.1	93.7
Dataset: SMC			
Encoder (Th)	53.9	32.9	45.6
Encoder	55	45.8	64.1
Decoder (Proposed)	55.4	45.1	65.6
Beat trans [9]	59.6	45.6	63.5
TF trans [8]	60.5	51.4	66.3
TCN [7]	55.2	46.5	64.3
Dataset: GTZAN			
Encoder (Th)	87.1	72.8	85.5
Encoder	87.8	78.5	93.7
Decoder (Proposed)	88.4	80.8	94
Beat trans [9]	88.5	80	92.2
TF trans [8]	88.7	81.2	92
TCN [7]	88.5	81.3	93.1

Table 5: Testing results for comparing the proposed method with three state-of-the-art beat tracking models [7–9]. The GTZAN dataset is held out for testing only; other datasets are used in the 8-fold cross-validation. (Th) means results obtained with a threshold of 0.1 without using the DBN post-processing step.

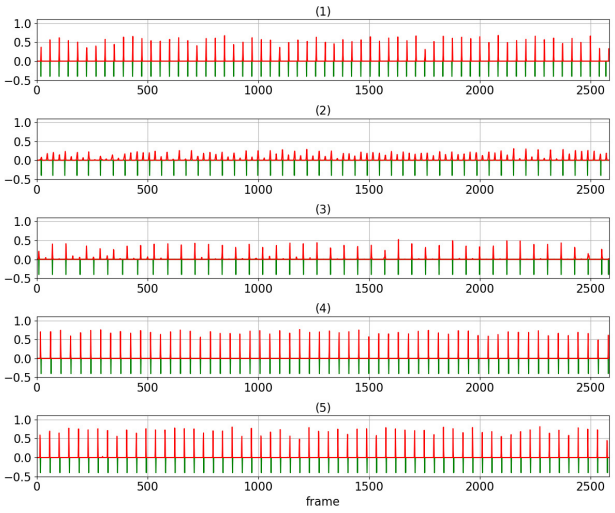
In order to understand the effect of the proposed decoder better, we show the outputs examples from the pre-trained encoder and the decoder in Figure 4. We can see that the encoder outputs are large at beat times, which is benefit from the more balanced training data in shorter sequences. On the other hand, the decoder outputs are small and even (i.e., more stable), as we expected. As we design, the decoder basically predicts the beat times at a higher resolution in comparison to the encoder. The decoder also helped to recover missing beats as shown in the 5th example, where some beats are missing in the encoder output but they are recovered in the decoder output. Moreover, the decoder helped to filter out peaks between beats as shown in the 3rd example, where peaks are more regularly placed in the decoder output. With the above effects, using the proposed decoder generally improved results except for the Hainsworth dataset. For the Hainsworth dataset, we see the CMLt decreased, but the AMLt remained the same level, which means that the decrease on the performance is caused by phase and octave errors. Since the peaks from the decoder are evener, in some cases it would be more difficult for the DBN to exclude the peaks between beats.

4.4.2 Comparison to state-of-the-art beat tracking models

As shown in Table 5, results of the proposed model ("Decoder (Proposed)") were comparable to the state-of-the-art results obtained by three beat tracking models [7–9], despite not the best. Since our goal is not to achieve better performances than all the state-of-the-art models, these results are satisfactory since we can show the high performances of the proposed model with different temporal resolutions. We see noticeable gap on the CMLt in comparison to other methods, which means the proposed model encountered more phase and octave errors. We hope this could be improved by including related topics in the multi-task learning as in [7, 9]. In addition, for the testing-only dataset GTZAN, the F-measures achieved by thresholding the encoder outputs ("Encoder (Th)") are better than what we expected, given the fact that it did not use the DBN post-processing step.



(a) Encoder outputs



(b) Decoder outputs

Figure 4: Output examples from the pre-trained encoder and the decoder for five different pieces. The ground-truth beat annotations are indicated by lines pointing down.

4.5 Attention Visualisation

In order to understand how low- and high-dimensional features are jointly learned, we show the cross attention matrix between low- and high-dimensional features in the second decoder layer in Figure 5. We found that for high-dimension features at beat times (frames), it got attention at each beat on the low-dimensional features. For no-beat times (frames), all attentions were drawn to the frames after the corresponding beat times, which formed horizontal lines in this figure. With such attentions, the final high-dimensional beat outputs were predicted with the captured global beat structure considered.

5. CONCLUSIONS AND FUTURE WORK

We present a novel Transformer-based model for beat tracking. The proposed model consists of both encoder layers and decoder layers which work on low- and high-dimensional features, respectively. We obtained beat tracking performances which are comparable to the state-of-the-art beat tracking results. The experimental results showed that the proposed model worked well as designed: with the low-dimensional (low-temporal-resolution) encoder for capturing the global beat structure and high-dimensional (high-temporal-resolution) decoder for predicting more precise beats. Thus, the proposed Transformer-based encoder and decoder structure succeeds in providing a new framework for handling multi-scale features for beat tracking. Beyond beat tracking, the advantage of this framework can be summarized as follows.

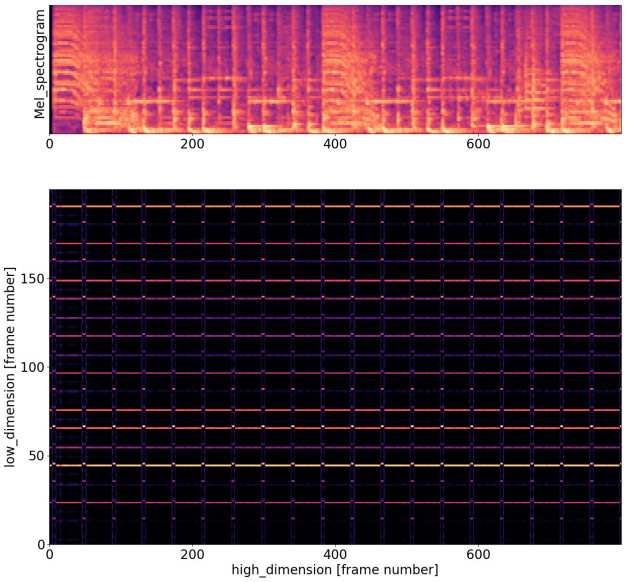


Figure 5: Cross attention matrix between low dimensional features and high dimensional features in the decoder layer.

- The encoder and decoder do not require inputs to be the same length (same temporal resolution), they can be used to handling features at different scales, which enables us to sample the features with more reasonable time resolutions.

- We can make use of features at different scales jointly learned by the cross attention in the decoder.

We therefore believe that this framework is also adaptable for other MIR tasks, such as musical structure boundary detection.

As the analysis of our experimental results showed, phase and octave errors are relatively high in our results. As future work, we would like to tackle the problems by combining downbeat tracking and tempo estimation in the proposed model by using multi-task learning. In addition, we also plan to use our model to produce beat outputs in a higher temporal resolution, which is demanded by some practical music applications as we discussed in Section 1. Yet another advantage of our model is that such precise beats can be achieved by using transfer learning with a higher-temporal-resolution input for the decoder.

6. ACKNOWLEDGMENTS

This work was supported in part by JST CREST Grant Number JPMJCR20D4 and JSPS KAKENHI Grant Number JP21H04917, Japan.

7. REFERENCES

[1] M. Goto and Y. Muraoka, "A Beat Tracking System for Acoustic Signals of Music," in *Proc. ACM Multimedia*, 1994, pp. 365–372.

[2] S. Böck, F. Krebs, and G. Widmer, "A Multi-Model Approach to Beat Tracking Considering Heterogeneous Music Styles," in *Proc. ISMIR*, 2014.

[3] ——, "Joint Beat and Downbeat Tracking with Recurrent Neural Networks," in *Proc. ISMIR*, 2016.

[4] F. Krebs, S. Böck, and G. Widmer, "Downbeat Tracking Using Beat-Synchronous Features and Recurrent Networks," in *Proc. ISMIR*, 2016.

[5] M. E. P. Davies and S. Böck, "Temporal Convolutional Networks for Musical Audio Beat Tracking," in *Proc. EUSIPCO*, 2019.

[6] S. Böck, M. E. P. Davies, and P. Knees, "Multi-Task Learning of Tempo and Beat: Learning One to Improve the Other," in *Proc. ISMIR*, 2019, pp. 486–493.

[7] S. Böck and M. E. Davies, "Deconstruct, Analyse, Reconstruct: How to Improve Tempo, Beat, and Downbeat Estimation," in *Proc. ISMIR*, 2020.

[8] Y. Hung, J. Wang, X. Song, W. T. Lu, and M. Won, "Modeling beats and downbeats with a time-frequency transformer," in *Proc. ICASSP*, 2022, pp. 401–405.

[9] J. Zhao, G. Xia, and Y. Wang, "Beat Transformer: Demixed Beat and Downbeat Tracking with Dilated Self-Attention," in *Proc. ISMIR*, 2022, pp. 169–177.

[10] M. Heydari and Z. Duan, "Singing Beat Tracking With Self-supervised Front-end and Linear Transformers," in *Proc. ISMIR*, 2022, pp. 617–624.

[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. ISMIR*, 2018.

[12] T. Cheng, S. Fukayama, and M. Goto, "Joint Beat and Downbeat Tracking Based on CRNN Models and a Comparison of Using Different Context Ranges in Convolutional Layers," in *Proc. ICMC*, 2020.

[13] ——, "Convolving Gaussian Kernels for RNN-based Beat Tracking," in *Proc. EUSIPCO*, 2018, pp. 1919–1923.

[14] F. Pedersoli and M. Goto, "Dance Beat Tracking from Visual Information Alone," in *Proc. ISMIR*, 2020, pp. 400–408.

[15] C. J. Steinmetz and J. D. Reiss, "WaveBeat: End-to-end beat and downbeat tracking in the time domain," in *151st AES Convention*, 2021.

[16] T.-P. Chen and L. Su, "Toward postprocessing-free neural networks for joint beat and downbeat estimation," in *Proc. ISMIR*, 2022, pp. 27–35.

[17] C. Hawthorne, I. Simon, R. Swavely, E. Manilow, and J. Engel, "Sequence-to-sequence piano transcription with transformers," in *Proc. ISMIR*, 2021, pp. 246–253.

[18] L. Oua, Z. Guo, E. Benetos, J. Han, and Y. Wang, "Exploring transformer's potential on automatic piano transcription," in *Proc. ICASSP*, 2022, pp. 776–780.

[19] J. Gardner, I. Simon, E. Manilow, C. Hawthorne, and J. H. Engel, "Mt3: multi-task multitrack music transcription," in *Proc. of the Tenth International Conference on Learning Representations (ICLR)*, 2022.

[20] M. Won, K. Choi, and X. Serra, "Semi-supervised music tagging transformer," in *Proc. ISMIR*, 2021, pp. 769–776.

[21] T. Cheng and M. Goto, "U-Beat: A multi-scale beat tracking model based on Wave-U-Net," in *Proc. ICASSP*, 2023.

[22] H. Schreiber and M. Meinard, "A single-step approach to musical tempo estimation using a convolutional neural network," in *Proc. ISMIR*, 2018, pp. 98–105.

[23] X. Sun, Q. He, Y. Gao, and W. Li, "Musical Tempo Estimation Using a Multi-scale Network," in *Proc. ISMIR*, 2021.

[24] M. E. P. Davies, N. Degara, and M. D. Plumbley, "Evaluation Methods for Musical Audio Beat Tracking Algorithms," Queen Mary University of London, London, United Kingdom, Tech. Rep. C4DM-TR-09-06, 2009.

[25] O. Nieto, M. McCallum, M. Davies, A. Robertson, A. Stark, and E. Egozy, "The harmonix set: Beats, downbeats, and functional segment annotations of western popular music," in *Proc. ISMIR*, 2019.

[26] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC Music Database: Popular, Classical, and Jazz Music Databases," in *Proc. ISMIR*, 2002, pp. 287–288.

[27] ——, "RWC Music Database: Music Genre Database and Musical Instrument Sound Database," in *Proc. ISMIR*, 2003, pp. 229–230.

[28] J. Driedger, H. Schreiber, W. B. de Haas, and M. Müller, "Towards automatically correcting tapped beat annotations for music recordings," in *Proc. ISMIR*, 2019.

[29] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano, "An Experimental Comparison of Audio Tempo Induction Algorithms," *IEEE Trans. Audio, Speech, Language Process.*, vol. 14, no. 5, pp. 1832–1844, 2006.

[30] F. Krebs, S. Böck, and G. Widmer, "Rhythmic Pattern Modeling for Beat and Downbeat Tracking in Musical Audio," in *Proc. ISMIR*, 2013, pp. 227–232.

[31] S. W. Hainsworth and M. D. Macleod, "Particle Filtering Applied to Musical Tempo Tracking," *EURASIP Journal on Applied Signal Process.*, vol. 15, 2004.

[32] A. Holzapfel, M. E. P. Davies, J. R. Zapata, J. a. L. Oliveira, and F. Gouyon, "Selective Sampling for Beat Tracking Evaluation," *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 9, pp. 2539–2548, 2012.

[33] G. Tzanetakis and P. Cook, "Musical Genre Classification of Audio Signals," *IEEE Speech Audio Process.*, vol. 10, no. 5, 2002.

[34] U. Marchand and G. Peeters, "Swing Ratio Estimation," in *Proc. DAFx*, 2015.

[35] T. Tieleman and G. Hinton, "Lecture 6.5—RMSProp: Divide the Gradient by a Running Average of its Recent Magnitude," COURSERA: Neural Networks for Machine Learning, 2012.