TOWARD POSTPROCESSING-FREE NEURAL NETWORKS FOR JOINT BEAT AND DOWNBEAT ESTIMATION

Tsung-Ping Chen **Li Su**
Institute of Information Science, Academia Sinica, Taiwan
{tearfulcanon, lisu}@iis.sinica.edu.tw

ABSTRACT

Recent deep learning-based models for estimating beats and downbeats are mainly composed of three successive stages—feature extraction, sequence modeling, and post processing. While such a framework is prevalent in the scenario of sequence labeling tasks and yields promising results in beat and downbeat estimations, it also indicates a shortage of the employed neural networks, given that the post-processing usually provides a notable performance gain over the previous stage. Moreover, the assumption often made for the post-processing is not suitable for many musical pieces. In this work, we attempt to improve the performance of joint beat and downbeat estimation without incorporating the post-processing stage. By inspecting a state-of-the-art approach, we propose reformulations regarding the network architecture and the loss function. We evaluate our model on various music data and show that the proposed methods are capable of improving the baseline approach without the aid of a post-processing stage.

1. INTRODUCTION

Beat and downbeat trackings have been of long-standing interests in the communities of signal processing and music information retrieval (MIR) [1–5]. The two tasks inherently possess the class imbalance issue [6], i.e., the highly imbalanced numbers of beat (downbeat) and non-beat (non-downbeat) frames in music data, and hence remain challenging in many cases even nowadays. To tackle the beat and downbeat tracking problems, early signal processing approaches have developed a three-stage framework which consists of *feature extraction*, *sequence modeling*, and *post-processing* [7–9]. Over the past few years, great progresses on the two tasks have been made through the combination of the three-stage pipeline and deep learning models such as recurrent neural networks (RNNs) [10–12], convolutional-recurrent neural networks (CRNNs) [13], and Transformer-based networks [14]. Among the thriving research, a convolutional approach achieves the state-of-the-art performance on joint estimation of tempo, beat, and downbeat [15], in which convolutional neural networks (CNNs) and temporal convolutional networks (TCNs) [16] are placed in charge of the feature extraction and of the sequence modeling, respectively, while dynamic Bayesian networks (DBNs) [17] are used for the post-processing.

In spite of the promising results obtained by this approach, the employed networks raise three concerns. First, the CNNs for feature extraction convolve an input spectrogram with small kernels at the very beginning, and a max-pooling layer is followed immediately. Therefore, the spectro-temporal patterns might not be well-captured [18]. Second, the dilated convolutions [19] of the TCNs involve a small kernel size and an exponentially increasing dilation rate. Despite the large receptive fields obtained at higher layers, this design leads to extremely sparse samplings which join distantly-separated frames, and consequently the contextual information could be irrelevant [20]. Lastly, the assumptions made for the DBNs that the meter is unchanged or the rhythmic patterns are known [11, 21] are not always applicable to various music genres, even to popular music. For example, the Hainsworth dataset [22], which is often employed for evaluation, includes several clips of pop or rock songs with changing meter. Moreover, the DBNs involve intricate design and assumptions for representing the state and the transition throughout a piece of music, and hence introduce non-trivial works in addition to the training of the preceding neural networks.

In this paper, we cope with the joint beat and downbeat estimation task by addressing the aforementioned issues. To get rid of the post-processing DBNs while maintaining the performance, we consider to improve the network architecture and the loss function. Precisely, we propose to use scaled depthwise separable convolutions [23–25] to aggregate rich contextual information at multiple time scales. To address the class imbalance issue, the focal loss [26] and the Dice loss [27] are employed in place of the commonly used cross entropy loss. Besides, we make our model aware of the periodic structure of beat or downbeat sequences by including a label embedding network [28] during training. We evaluate the proposed architecture on various music data, including both audio and MIDI files, and demonstrate a state-of-the-art performance on the Ballroom dataset. The main contribution of this work is that we thoroughly address the architectural issues for beat and downbeat estimations. Most of the problems are shared by sequence models in general, and therefore the proposed methods could be applied to many other MIR-related tasks.

2. OVERVIEW OF THE STATE-OF-THE-ART

The model of [15] without the DBNs is taken as our baseline upon which we build our new model to address the architectural issues mentioned beforehand. In the following, we briefly introduce the baseline model using the three-stage framework, and present the new model thereafter. We skip the post-processing stage for we aim to expel the DBNs from the system in this work. An overview of the baseline model is illustrated at the top of Figure 1.

2.1 Feature extraction

Given an audio signal represented as a log-magnitude spectrogram of size $T \times J$, where T denotes the number of time steps and J the number of frequency bins, the CNNs of the baseline model extract high-level features using three groups of 2-D convolution and max-pooling layers. The three convolutional layers (with a kernel size of 3×3, 1×12, and 3×3, respectively) [1] capture harmonic content at different frequency scales, while the max-pooling operations (with a kernel size of 1×3) reduce the frequency dimension in three steps. In other words, the CNNs gradually transcribe the harmonic information into high-level features, and output a sequence of size $T \times D$ with D indicating the dimension of high-level features.

2.2 Sequence modeling

The high-level feature sequence from the previous stage is then passed to the TCNs for learning temporal structure. As depicted at the top of Figure 1, each TCN layer performs two 1-D dilated convolutions in parallel with a kernel of size 5 and an exponentially increasing dilation rate 2^l (resp. 2^{l+1}), where l is the layer number. The outputs of the two dilated convolutions are concatenated and reprojected to keep the feature dimensionality constant. As a result, the TCNs efficiently enlarge the receptive field within a few layers with the amount of learnable parameters increasing linearly to the number of layers. Given that the TCNs has 11 layers, the two convolutions at the last layer will expand the kernels across over 4000 (resp. 8000) time frames with each element of the two kernels being spaced 1024 (resp. 2048) frames apart. Therefore, the temporal context modelled by the TCN is more than 80 seconds (assume a frame rate of 100 fps). Afterwards, the output of the TCNs is processed by the DBNs to obtain a sequence of beat or downbeat estimates.

2.3 Potential issues

In the community of computer vision, it has been shown that stacking small convolutional kernels followed by pooling layers is effectual for extracting high-level features of an image [29–31]. Such a architecture is often adopted by audio-based, or more specifically spectrogram-based research [32–34]. Considering the nature of images, it is reasonable to aggregate information hierarchically from spatial associations. However, it is questionable that a spectrogram could be well-represented with the same architecture. The characteristic spectral information of a musical piece does not necessarily reside in the adjacent frequency bins within a small kernel. Moreover, it is also suspicious that the *compressed* frequency dimension after pooling operations is as meaningful as a downsampled image.

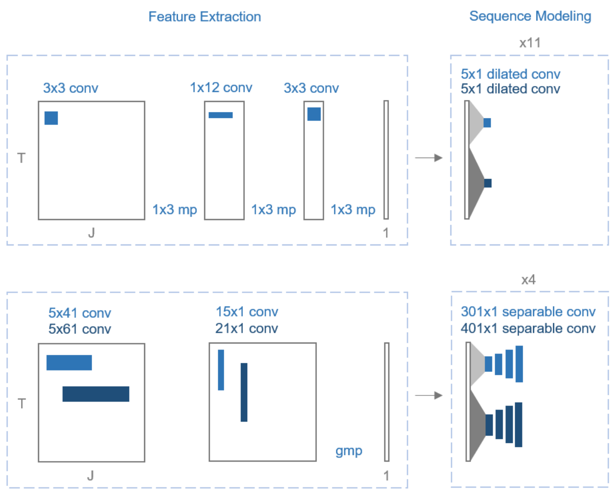

Figure 1: Architecture reformulation with respect to convolution (conv) and max pooling (mp) layers . Top: model of [15] without DBNs. Bottom: proposed architecture. The feature dimension is not shown in the figure.

On the other hand, it has been observed that dilated convolutions result in the so-called *gridding artifacts* [35, 36], due to that adjacent elements in the output are calculated from completely different sets of elements in the input. Additionally, small convolutional kernels with high dilation rates relate input elements which are highly sparsely distributed and hence might lack of correlations. In the scenario of beat or downbeat estimation, stacked 1-D dilated convolutions could be preferable for they are able to encode periodic structure of the beat (downbeat) through equidistant elements of dilated kernels. However, it is not guaranteed that an inquired musical piece has a steady tempo as well as a constant rhythm. Hence, a network equipped with dilated convolutions for modeling temporal information may have limitations on expressive music.

3. APPROACH

3.1 Task formulation

We treat the joint beat and downbeat estimation task as a sequence labeling problem [37, 38]. Given an input spectrogram $\mathbf{X} \in \mathbb{R}^{T \times J}$, the task involves the assignment of a categorical label $\in \{\texttt{downbeat}, \texttt{xbeat}, \texttt{neither}\}$ [2] to each time step $t \in$ T. We employ a model architecture which mainly consists of a feature extraction network (f_{ex}), a sequence modeling network (f_{seq}), and an estima-

[1] A kernel size $M \times N$ specifies a convolution window across M time frames and N frequency bins.

[2] The label `xbeat` refers to a beat which is not a downbeat.

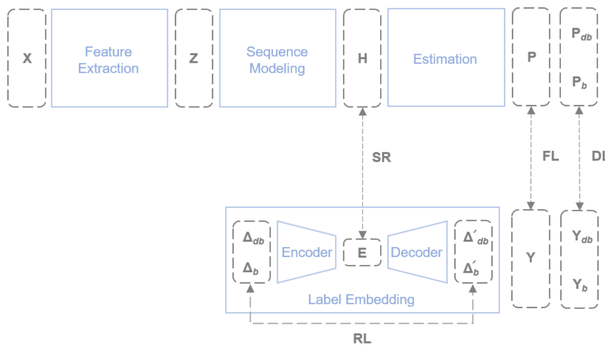

Figure 2: Proposed architecture. Dashed borders indicate data sequences, and solid ones denote neural networks. The losses includes focal loss (FL), Dice loss (DL), reconstruction loss (RL), and structural regularization (SR).

tion network (f_{est}) for the task:

$$\begin{aligned} \mathbf{P} &= f_{est}(\mathbf{H}), \\ \mathbf{H} &= f_{seq}(\mathbf{Z}), \\ \mathbf{Z} &= f_{ex}(\mathbf{X}), \end{aligned} \tag{1}$$

where $\mathbf{Z}, \mathbf{H} \in \mathbb{R}^{T \times D}$ are higher-level feature sequences, and $\mathbf{P} \in \mathbb{R}^{T \times 3}$ is a sequence of model estimates indicating the probabilities of the three categorical labels across all time steps. The overall model architecture is schematically shown in Figure 2.

3.2 Reformulation of feature extraction

We propose to use kernels of larger sizes for the 2-D convolution layers and remove the intermediate pooling layers in f_{ex}. As shown at the bottom of Figure 1, the feature extraction network consist of two layers, denoted as $f_{ex}^{(1)}$ and $f_{ex}^{(2)}$. The first layer captures harmonic information distributed across the frequency bins of an input spectrogram, while the second layer aggregates the harmonic information along the temporal dimension. Within each layer l, we employ two parallel convolutions with kernel sizes of $w_{ex,1}^{(l)}$ and $w_{ex,2}^{(l)}$. For these convolution operations, we keep the frequency dimension of the input spectrogram uncompressed, and thus the output shape will be $T \times J \times D$. Finally, we employ a global max pooling layer (GMP) [39] to reduce the frequency dimension (J) of the output, yielding the feature sequence $\mathbf{Z} \in \mathbb{R}^{T \times D}$. Let $f_c(\cdot, d, w)$ denote a standard convolution parameterized by the number of filters d and the kernel size w. The feature extraction network can be formulated as follows:

$$\begin{aligned} \mathbf{Z} = f_{ex}(\mathbf{X}) &= \underset{j \in J}{\mathrm{GMP}}(f_{ex}^{(2)}(f_{ex}^{(1)}(\mathbf{X}))), \\ f_{ex}^{(l)}(\cdot) &= [f_c(\cdot, D, w_{ex,1}^{(l)}), f_c(\cdot, D, w_{ex,2}^{(l)})]\mathbf{W}_{ex}^{(l)}, \end{aligned} \tag{2}$$

where $\mathbf{W}_{ex}^{(l)} \in \mathbb{R}^{2D \times D}$ is a learnable parameter matrix at the l-th layer, and $[\cdot, \cdot]$ indicates a concatenation operation.

3.3 Reformulation of sequence modeling

To maintain a large receptive field while get rid of the sparse sampling issue introduced by dilated convolutions, we use depthwise separable convolutions [23] (hereafter separable convolutions) instead. Separable convolution, denoted as $f_{sc}(\cdot, d, w)$, factorizes a standard convolution into a depthwise convolution and a pointwise convolution. This factorization allows a network to expand the receptive field using a large kernel size without drastically increasing the number of parameters. Similarly to the TCNs, we use two separable convolutions with kernel sizes of $w_{seq,1}$ and $w_{seq,2}$, as shown in Figure 1. Each sequence modeling layer, denoted as $f_{seq}(\cdot)$, is thus formulated as below:

$$f_{seq}(\cdot) = f_{sc}(\cdot, D, w_{seq,1}) + f_{sc}(\cdot, D, w_{seq,2}). \tag{3}$$

Considering that rhythmic patterns are tempo-dependent, that is, the inter-beat or the inter-downbeat interval will vary according to the current tempo, it is crucial for a network to learn tempo-invariant, or *scale-invariant* patterns [40]. To achieve scale invariance, we simply introduce a set of dilation rates, or *scale factors* $\mathbf{s} = \{r_s\}_{s=1}^{S}$ to each separable convolution. The *scaled* separable convolution, accordingly, will operate S times on the same input with the kernel being expanded by the given set of dilation rates:

$$\underset{\mathbf{s}}{f_{sc}}(\cdot, d, w) = (f_{sc}(\cdot, d, w, r_1), \ldots, f_{sc}(\cdot, d, w, r_S)), \tag{4}$$

where $(\cdot, \ldots, \cdot)$ denotes a stacking operation. Therefore, the scaled separable convolution is capable of capturing rhythmic patterns at different tempi simultaneously. Note that the dilation employed here differs from that of the original TCNs in two aspects. First, we employ the dilation for modeling tempo variance rather than for expanding the receptive field. Second, the parameters of a scaled separable convolution are shared among the S operations.

By introducing the scale factors, we add a new dimension to the output of the scaled separable convolution, whose shape thus becomes $T \times D \times S$. We summarize the scale information by applying a gating mechanism, termed *scale summarization* (SS), which is similar to the *Squeeze-and-Excitation* operation of the SENet [41]. Let $\mathbf{U} \in \mathbb{R}^{T \times D \times S}$ denote the output of a scaled separable convolution. We use global average pooling (GAP) along both the time and the feature dimensions to calculate scalewise statistics $\mathbf{z} \in \mathbb{R}^{S}$. Subsequently, a gating function $\mathbf{g} \in \mathbb{R}^{S}$ is generated by using a projection layer with learnable parameters $\mathbf{W} \in \mathbb{R}^{S \times S}$ and a softmax activation. Lastly, we apply the gate function to $\mathbf{U}$ for obtaining the summarized output $\mathbf{U}' \in \mathbb{R}^{T \times D}$:

$$\begin{gathered} \mathbf{U}' = \mathrm{SS}(\mathbf{U}) = \sum_{s=1}^{S} \mathbf{U}\mathbf{g}, \\ \mathbf{g} = \underset{s \in S}{\mathrm{softmax}}(\mathbf{W}\mathbf{z}), \;\; \mathbf{z} = \underset{t \in T, d \in D}{\mathrm{GAP}}(\mathbf{U}). \end{gathered} \tag{5}$$

In conclusion, the L-layer sequence modeling network can be formally expressed as follows:

$$\begin{aligned} \mathbf{H}^{(l)} &= f_{seq}^{(l)}(\mathbf{H}^{(l-1)}), \\ f_{seq}^{(l)}(\cdot) &= \mathrm{SS}(\underset{\mathbf{s}}{f_{sc}}(\cdot, D, w_{seq,1}) + \underset{\mathbf{s}}{f_{sc}}(\cdot, D, w_{seq,2})), \end{aligned} \tag{6}$$

with boundaries $\mathbf{H}^{(0)} = \mathbf{Z}$ and $\mathbf{H}^{(L)} = \mathbf{H}$.

3.4 Estimation

We employ a convolutional layer with a kernel size of w_{est} followed by a softmax activation function to estimate the probabilities of being `downbeat` (db), `xbeat` (xb), and `neither` (n) at each time step:

$$\begin{aligned}[\mathbf{P}_{db}, \mathbf{P}_{xb}, \mathbf{P}_n] &= \mathbf{P} = f_{est}(\mathbf{H}) \\ &= \underset{d}{\text{softmax}}(f_c(\mathbf{H}, d=3, w_{est})),\end{aligned} \tag{7}$$

where $\mathbf{P}_{db}, \mathbf{P}_{xb}, \mathbf{P}_n \in \mathbb{R}^T$ indicate the estimated probabilities of the three categorical labels across T time steps. During inference, we generate a downbeat sequence ($\mathbf{I}_{db}$) and a beat sequence ($\mathbf{I}_b$) from $\mathbf{P}_{db}$ and $\mathbf{P}_{xb}$ by finding the local maximums (LM):

$$\begin{aligned}\mathbf{I}_{db} &= \text{LM}(\mathbf{P}_{db}), \\ \mathbf{I}_b &= \text{LM}(\mathbf{P}_b) = \text{LM}(\mathbf{P}_{db} + \mathbf{P}_{xb}).\end{aligned} \tag{8}$$

3.5 Loss function

Cross entropy is often used as the loss function in the beat and the downbeat estimation tasks. Since the cross entropy loss treats all samples equally, the network optimization will be dominated by vast amount of easy samples, i.e., time frames not belonging to the beat and the downbeat. Similar cases can be found in computer vision-related tasks such as object detection, where exists an extreme foreground-background disproportion. The focal loss (FL) [26] is therefore proposed to weigh more on hard or easily mis-classified samples by introducing a modulating term to the cross entropy loss:

$$\text{FL} = -\frac{1}{N}\sum_{i=1}^{N} m_i \times y_i \times \log(p_i), \tag{9}$$

where p_i is the estimated probability of sample i being classified as the ground truth y_i, and $m_i = (1-p_i)^{\gamma_i}$ is the modulating term with γ_i being a controllable factor. It is noteworthy that the FL, as same as the cross entropy, is calculated on individual samples and hence unable to reflect spatial or temporal relations between samples. On the other hand, the Dice loss (DL) [27], which has been applied to semantic segmentation of images, measures the overlap between the estimations of N samples and the ground truths, and thus can delineate the regions of interest, e.g., beat positions in a given musical sequence:

$$\text{DL} = 1 - \frac{2\sum_{i=1}^{N} p_i \times y_i}{\sum_{i=1}^{N} p_i^2 + \sum_{i=1}^{N} y_i^2}. \tag{10}$$

We propose to use a hybrid of the FL and the DL in place of the cross entropy loss, for they address the class imbalance issue from two complementary aspects: the FL accentuates *individual* hard samples whereas the DL underlines *collective* similarity of the minority samples. In practice, we compute the FL on $\mathbf{P}$ while the DL on both $\mathbf{P}_{db}$ and $\mathbf{P}_b$. By combining the two losses, we urge our network to focus on the minority classes, i.e., beats and downbeats, both at the frame level and at the sequence level.

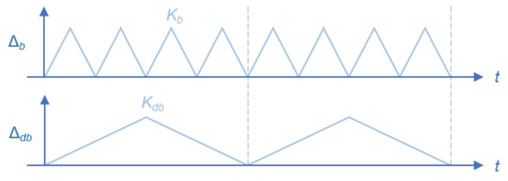

Figure 3: Representations of beat (top) and downbeat (bottom) for label embedding (assume a meter of `4/4`). The horizontal axes are the time dimensions while the vertical axes indicate the phase classes. K_b and K_{db} are the numbers of classes for beat and downbeat, respectively.

3.6 Label embedding

Another concern in regard to the loss function is that the periodic structure of a beat or downbeat sequence is not considered. A recent work deals with the periodicity by estimating beat *phase* instead of beat presence at each frame [42]. Accordingly, the beat estimation task is reformulated as a sequence labeling problem in which the beat phase is represented as a discrete sawtooth wave with period equal to the interbeat interval. This reformulation, however, enlarges the estimation space as the beat phase is categorized into K classes with K being the phase resolution.

Alternatively, we leverage the label embedding approach [28] for learning the periodic structure. As illustrated in Figure 3, a sequence of ground-truth annotations is represented as a discrete triangular wave $\mathbf{\Delta}_{\{db,b\}} \in \mathbb{R}^{T\times K_{\{db,b\}}}$ whose troughs locate at the beat or downbeat frames. We then employ a vanilla autoencoder to encode the represented annotations in an unsupervised manner. The encoder (f_{enc}) and the decoder (f_{dec}) used are both a simple 1-D convolution layer with a kernel size of w_e, i.e., $f_{enc}(\cdot) = f_{dec}(\cdot) = f_c(\cdot, D, w_e)$. The label embedding network is formulated as follows:

$$\begin{aligned}[\mathbf{V}'_{db}, \mathbf{V}'_b] &= f_{dec}(\mathbf{E}), \\ \mathbf{E} &= f_{enc}([\mathbf{V}_{db}, \mathbf{V}_b]), \\ \mathbf{V}_{db} &= \mathbf{\Delta}_{db}\mathbf{W}_{db}, \mathbf{V}_b = \mathbf{\Delta}_b\mathbf{W}_b,\end{aligned} \tag{11}$$

where $\mathbf{W}_{\{db,b\}} \in \mathbb{R}^{K_{\{db,b\}}\times D}$ are learnable phase embedding matrices, $\mathbf{V}_{\{db,b\}} \in \mathbb{R}^{T\times D}$ are sequences of embeddings, $\mathbf{E} \in \mathbb{R}^{T\times D}$ is a sequence of joint embeddings, and $\mathbf{V}'_{\{db,b\}} \in \mathbb{R}^{T\times D}$ are the reconstructions of $\mathbf{V}_{\{db,b\}}$. We compute the reconstruction loss (RL) as follows:

$$\begin{aligned}\text{RL} &= \text{FL}(\mathbf{\Delta}_{db}, \mathbf{\Delta}'_{db}) + \text{FL}(\mathbf{\Delta}_b, \mathbf{\Delta}'_b), \\ \mathbf{\Delta}'_{db} &= \underset{k\in K}{\text{softmax}}(\mathbf{V}'_{db}\mathbf{W}_{db}^{\top}), \\ \mathbf{\Delta}'_b &= \underset{k\in K}{\text{softmax}}(\mathbf{V}'_b\mathbf{W}_b^{\top}).\end{aligned} \tag{12}$$

As illustrated in Figure 2, the joint embedding $\mathbf{E}$ is taken as a structural regularization (SR) of $\mathbf{H}$ (the output of the sequence modeling network) by minimizing the mean squared error (MSE):

$$\text{SR} = \text{MSE}(\mathbf{E}, \mathbf{H}). \tag{13}$$

Hence, the total loss $\mathscr{L} = \text{FL} + \text{DL} + \text{RL} + \text{SR}$. We train the whole networks in an end-to-end fashion. As the contextual information of the annotations is encoded in $\mathbf{E}$, adding this *topology-aware* regularization term (SR) to the loss computation enables the network to explicitly learn the structural characteristics [43]. The source code of the proposed model is available online. [3]

4. EVALUATION

4.1 Data preparation

Four datasets are employed in the evaluation, including the Ballroom [44], the Hainsworth [22], the GTZAN [45], and the ASAP [46]. Each audio file (with a sampling rate of 44,100 Hz) in the datasets is transformed into a log-magnitude spectrogram with a total of 81 frequency bins from 30 Hz to 17,000 Hz, by using the Python package *librosa* [47]. Specifically, the short-time Fourier transform (STFT) with a window size of 92.9 ms (4,096 samples) and a hop size of 23.2 ms (1,024 samples) is applied, and the output spectrogram is mapped onto the Mel scale. The per-bin first-order difference is calculated for each spectrogram as an additional feature (only the positive differences are retained) [16]. In consequence, each audio file is represented as $\mathbf{X} \in \mathbb{R}^{T\times 81\times 2}$ with T being the time steps of the corresponding spectrogram. Besides, the MIDI data of the ASAP are also used for evaluation. Following [48], we represent each MIDI file by four features: the *pitch profile* $\in \mathbb{R}^{T\times 88}$ (i.e., pianoroll), the *onset profile* $\in \mathbb{R}^{T\times 88}$, the *spectral flux* $\in \mathbb{R}^{T}$, and the *inter-onset interval* $\in \mathbb{R}^{T}$. We modify our feature extraction network accordingly for the MIDI representation, and keep the other parts of the architecture unchanged. Specifically, the pitch profile and the onset profile are concatenated together and fed into the feature extraction network. The output of the feature extraction network is then concatenated with the other two features and taken as the input of the succeeding layer.

On the other hand, the beat and downbeat annotations are represented as binary sequences, $\mathbf{Y}_b, \mathbf{Y}_{db} \in \{0,1\}^T$, where a time step $t \in T$ has a value of 1 if it belongs to the beat or the downbeat, and 0 otherwise. To treat joint beat and downbeat estimation as a sequence labeling problem, we further generate $\mathbf{Y} \in \{0,1\}^{T\times 3}$ based on $\mathbf{Y}_{db}$ and $\mathbf{Y}_b$, which is a sequence of one-hot vectors indicating the categorical label of each time step. The $\mathbf{Y}$ is used to compute the FL (with $\mathbf{P}$) while the $\mathbf{Y}_b$ and the $\mathbf{Y}_{db}$ are for the DL (with $\mathbf{P}_b$ and $\mathbf{P}_{db}$, respectively), as depicted in Figure 2.

4.2 Experiment

We evaluate the proposed architecture on the four datasets and report the beat and the downbeat estimation performances using the standard F1 measure with a tolerance window of ±70 ms. The hyperparameters of the architecture are set as in Table 1. The model of [15] without DBNs (i.e., the top one in Figure 1) is employed as our baseline model. The numbers of learnable parameters are around 110K and 63K for the proposed model and the baseline, respectively. Evidently, we can employ separable convolutions to enlarge the receptive field without a catastrophic increase of model capacity. To investigate the effectiveness of the proposed methods, we further perform an ablation study by removing one of the reformulations: feature extraction (`abl_ex`), sequence modeling (`abl_seq`), loss function (`abl_loss`), and label embedding (`abl_lab`).

Input	
Number of frequency bins	$J = 81$ for Audio $J = 88$ for MIDI
Feature extraction	
Kernel size	$w^{(1)}_{ex,1} = 5 \times 41$
Kernel size	$w^{(1)}_{ex,2} = 5 \times 61$
Kernel size	$w^{(2)}_{ex,1} = 15 \times 1$
Kernel size	$w^{(2)}_{ex,2} = 21 \times 1$
Feature dimension	$D = 20$
Sequence modeling	
Kernel size	$w_{seq,1} = 301$
Kernel size	$w_{seq,2} = 401$
Scale factor	$\mathbf{s} = \{1, 2, 3, 4\}$
Number of scales	$S = 4$
Number of layers	$L = 4$
Estimation	
Kernel size	$w_{est} = 7$
Label embedding	
Number of beat phase classes	$K_b = 150$
Number of downbeat phase classes	$K_{db} = 500$
Kernel size	$w_e = 7$

Table 1: Hyperparameters of the proposed architecture.

In line with [15], the Ballroom and the Hainsworth are both split for 8-fold cross-validations. [4] Note that the two datasets are used separately rather than merged as a single cross-validation set. The GTZAN is divided into 10 parts according to the built-in genre labels, with which a 10-fold cross-validation is performed to inspect the performance variance with respect to genre. [5] For the ASAP, we first create a subset by selecting audio recordings which have a paired MIDI file (519 pairs in total). Afterwards, a test set is built from the subset with musical pieces by the four composers: Glinka, Mozart, Schubert, and Rachmaninoff; the remaining data of the subset are used for training. The paired MIDI files are used as the audio counterparts to examine the impact of input modality on the performance.

All the training data are augmented in two ways: 1) by shifting the pitch of an audio signal *before* the STFT is applied, and 2) by changing the hop size of the STFT window. [6] For the ASAP, we only apply the pitch augmentation as its total duration is significantly longer than the other three datasets. By the pitch augmentation, we can easily increase the data amount without extra annotation

[3] `https://github.com/Tsung-Ping/Joint-beat-and-downbeat-estimation`

[4] The details of the data splitting can be found at `https://github.com/superbock/ISMIR2020`.

[5] `jazz.00003`, `jazz.00009`, `jazz.00010`, `jazz.00014`, `jazz.00018`, `jazz.00020` are excluded for they have no downbeat annotations; `reggae.00086` is damaged and also excluded.

[6] We use pitch shifts $\in$ {-5 $\sim$ +6} (semitones) and hop sizes $\in$ {18.9, 20.3, 21.8, 23.2, 26.1, 29.0, 31.9} (ms) for the data augmentation.

Task	**Model**	**Ballroom**	**Hainsworth**	**GTZAN**										**ASAP**	
				blues	classical	country	disco	hiphop	jazz	metal	pop	reggae	rock	Audio	MIDI
Beat	baseline	96.60	84.18	83.20	51.62	**96.65**	**97.76**	94.61	68.40	82.46	93.84	92.76	89.90	**71.43**	72.45
	proposed	**96.81**	**86.28**	**86.63**	**63.23**	94.82	97.63	**95.58**	**79.80**	**86.48**	**94.81**	**93.05**	**93.01**	71.40	**75.70**
Downbeat	baseline	92.06	66.18	59.25	14.70	79.70	82.71	72.86	33.47	64.64	**84.67**	51.06	76.49	49.46	57.34
	proposed	**94.21**	**69.11**	**60.98**	**30.17**	**83.91**	**86.28**	**79.10**	**50.66**	**67.21**	83.86	**56.87**	**76.55**	**63.92**	**67.43**

Table 2: Evaluation results in terms of F1 score (%). For the Ballroom and the Hainsworth, the scores are the averaged performances over a 8-fold cross-validation. For the GTZAN, the score on each genre is obtained in a 10-fold cross-validation manner. For the ASAP, the performances on the audio data and on the MIDI data are separately shown.

Task	**Model**	**Ballroom**	**Hainsworth**
Beat	abl_ex	96.08 (-0.73)	85.49 (-0.79)
	abl_seq	96.98 (+0.17)	85.96 (-0.32)
	abl_loss	95.27 (-1.54)	86.52 (+0.24)
	abl_lab	96.54 (-0.27)	86.83 (+0.55)
Downbeat	abl_ex	92.16 (-2.05)	62.27 (-6.84)
	abl_seq	94.16 (-0.05)	69.73 (+0.62)
	abl_loss	91.99 (-2.22)	66.79 (-2.32)
	abl_lab	93.61 (-0.60)	69.58 (+0.47)

Table 3: Ablation study on the Ballroom and the Hainsworth. Performance in terms of F1 score (%) and relative improvement against the proposed model (in parentheses) are provided.

cost; moreover, the model can explicitly learn rhythmic features independent of absolute pitch. By the hop-size augmentation, we can obtain more training data of different tempi with a slight adjustment of the annotations.

4.3 Result

The evaluation results are summarized in Table 2, showing the performance on each dataset in terms of beat estimation and downbeat estimation. When evaluated on the Ballroom and the Hainsworth, the proposed model outperforms the baseline in both beat and downbeat estimations. Moreover, it is surprising that the evaluation results on the Ballroom are even better than [15] (beat: 96.20; downbeat: 91.6) and [14] (beat: 96.20; downbeat: 93.7) considering that we didn't use a combination of datasets for training and DBNs for post-processing; in addition, our model is smaller in capacity than the best model of [14] which has around 4.7M learnable parameters. On the other hand, the results on the Hainsworth indicate that more efforts should be put in dealing with music data of diverse styles. For instance, the choruses collected in the Hainsworth are extremely flexible in tempo at the beginning and particularly at the end of each phrase, and therefore a phrase segmentation technique might be incorporated in the tasks.

For the GTZAN, the proposed model demonstrates its superiority over the baseline in almost all the cases. The average beat estimation performances are 85.12% and 88.50% for the baseline and the proposed model, respectively, while the average downbeat estimation performances are 61.96% and 67.56%. According to the evaluation results, the rhythmic characteristics of `classical` and `jazz` are quite distinct from the other genres. Nevertheless, our model still surpasses the baseline by around 15.5 and 17.2 percentage points in estimating the downbeats of `classical` and `jazz`, respectively. These improvements are notable and indicate the capability of our model in both tasks, especially in downbeat estimation.

For the audio data of the ASAP, the proposed model performs comparably to the baseline on the beat estimation while remarkably outperforms the baseline on the downbeat estimation. For the MIDI data of the ASAP, our model achieves greater results both on estimating the beats and the downbeats. It is worth noting that in all cases, the estimation result is better on the MIDI data than on the audio counterpart. Given that automatic music transcription (AMT) [49–51] is an active research topic in the field of MIR, it could be promising to incorporate AMT techniques into the beat and the downbeat estimations.

Finally, as shown in Table 3, the ablation study indicates that our reformulations of the feature extraction and the loss function have notable positive effect especially on estimating downbeats, while the improvements by the other components are inconsistent over the tasks and datasets. We also observed that for the Hainsworth, the model without the label embedding (`abl_lab`) performs slightly better. This might result from the limited expressiveness of $\mathbf{H}$ due to the regularization on it. A more extensive study is required to justify the proposed reformulations.

5. CONCLUSION

We have addressed the joint beat and downbeat estimation task based on a state-of-the-art approach. By inspecting the potential issues in this approach, we proposed several reformulations to further the performance of deep neural networks. We experimentally showed that the proposed architecture is capable of outperforming the state-of-the-art approach without the aid of a post-processing network. In the scenario of deep learning-based beat and downbeat estimations, as well as in many sequence labeling frameworks, it is common to involve a post-processing stage in addition to the deep neural networks since the outputs by the networks are usually coarse when a simple thresholding method is applied. While involving a post-processing stage often leads to an improvement over the preceding deep learning models, it hinders the formulation of end-to-end training and indicates a necessity to reconsider the employed neural networks. Hopefully, we are able to build a model tailored to the task of interest with a deeper look at the network architecture from the perspective of data.

6. REFERENCES

[1] M. Goto and Y. Muraoka, “A beat tracking system for acoustic signals of music,” in *Proceedings of the 2nd ACM International Conference on Multimedia*, 1994, pp. 365–372.

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

[3] L. M. Smith, “Beat critic: Beat tracking octave error identification by metrical profile analysis,” in *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*, 2010, pp. 99–104.

[4] H. Grohganz, M. Clausen, and M. Müller, “Estimating musical time information from performed MIDI files,” in *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, 2014, pp. 35–40.

[5] S. Durand, J. P. Bello, B. David, and G. Richard, “Feature adapted convolutional neural networks for downbeat tracking,” in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 296–300.

[6] R. Anand, K. G. Mehrotra, C. K. Mohan, and S. Ranka, “An improved algorithm for neural network classification of imbalanced training sets,” *IEEE Transactions on Neural Networks*, vol. 4, no. 6, pp. 962–969, 1993.

[7] M. Goto and Y. Muraoka, “An audio-based real-time beat tracking system and its applications,” in *Proceedings of the International Computer Music Conference (ICMC)*, 1998.

[8] M. E. P. Davies and M. D. Plumbley, “A spectral difference approach to downbeat extraction in musical audio,” in *Proceedings of the 14th European Signal Processing Conference (EUSIPCO)*, 2006, pp. 1–4.

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

[10] F. Krebs, S. Böck, M. Dorfer, and G. Widmer, “Downbeat tracking using beat synchronous features with recurrent neural networks,” in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, 2016, pp. 129–135.

[11] S. Böck, F. Krebs, and G. Widmer, “Joint beat and downbeat tracking with recurrent neural networks,” in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, 2016, pp. 255–261.

[12] C. Chiu, A. W. Su, and Y. Yang, “Drum-aware ensemble architecture for improved joint musical beat and downbeat tracking,” *IEEE Signal Processing Letters*, vol. 28, pp. 1100–1104, 2021.

[13] M. Fuentes, B. McFee, H. C. Crayencour, S. Essid, and J. P. Bello, “A music structure informed downbeat tracking system using skip-chain conditional random fields and deep learning,” in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 481–485.

[14] Y.-N. Hung, J.-C. Wang, X. Song, W.-T. Lu, and M. Won, “Modeling beat and downbeat estimations with a time-frequency Transformer,” in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 401–405.

[15] S. Böck and M. E. P. Davies, “Deconstruct, analyse, reconstruct: How to improve tempo, beat, and downbeat estimation,” in *Proceedings of the 21th International Society for Music Information Retrieval Conference (ISMIR)*, 2020, pp. 574–582.

[16] M. E. P. Davies and S. Böck, “Temporal convolutional networks for musical audio beat tracking,” in *Proceedings of the 27th European Signal Processing Conference (EUSIPCO)*, 2019, pp. 1–5.

[17] F. Krebs, S. Böck, and G. Widmer, “An efficient state-space model for joint tempo and meter tracking,” in *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, 2015, pp. 72–78.

[18] J. Pons, O. Slizovskaia, R. Gong, E. Gómez, and X. Serra, “Timbre analysis of music audio signals with convolutional neural networks,” in *Proceedings of the 25th European Signal Processing Conference (EUSIPCO)*, 2017, pp. 2744–2748.

[19] F. Yu and V. Koltun, “Multi-scale context aggregation by dilated convolutions,” in *Proceedings of the 4th International Conference on Learning Representations (ICLR)*, 2016.

[20] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. W. Cottrell, “Understanding convolution for semantic segmentation,” in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018, pp. 1451–1460.

[21] F. Krebs, S. Böck, and G. Widmer, “Rhythmic pattern modeling for beat and downbeat tracking in musical audio,” in *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR)*, 2013, pp. 227–232.

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

[23] L. Sifre, “Rigid-motion scattering for image classification,” Ph.D. thesis, École Polytechnique, 2014.

[24] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1800–1807.

[25] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *ArXiv e-prints*, 2017. [Online]. Available: http://arxiv.org/abs/1704.04861

[26] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2999–3007.

[27] F. Milletari, N. Navab, and S. Ahmadi, "V-Net: Fully convolutional neural networks for volumetric medical image segmentation," in *Proceedings of the 4th International Conference on 3D Vision (3DV)*, 2016, pp. 565–571.

[28] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid, "Label-embedding for image classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 7, pp. 1425–1438, 2016.

[29] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[31] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2261–2269.

[32] F. Korzeniowski and G. Widmer, "A fully convolutional deep auditory model for musical chord recognition," in *Proceedings of the 26th IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2016, pp. 1–6.

[33] K. Choi, G. Fazekas, and M. B. Sandler, "Automatic tagging using deep convolutional neural networks," in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, 2016, pp. 805–811.

[34] S. Böck, M. E. P. Davies, and P. Knees, "Multi-task learning of tempo and beat: Learning one to improve the other," in *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*, 2019, pp. 486–493.

[35] F. Yu, V. Koltun, and T. A. Funkhouser, "Dilated residual networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 636–644.

[36] R. Hamaguchi, A. Fujita, K. Nemoto, T. Imaizumi, and S. Hikosaka, "Effective use of dilated convolutions for segmenting small object instances in remote sensing imagery," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018, pp. 1442–1450.

[37] X. Ma and E. H. Hovy, "End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

[38] A. Akbik, D. Blythe, and R. Vollgraf, "Contextual string embeddings for sequence labeling," in *Proceedings of the 27th International Conference on Computational Linguistics (COLING)*, 2018, pp. 1638–1649.

[39] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Is object localization for free? - weakly-supervised learning with convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 685–694.

[40] B. D. Giorgi, M. Mauch, and M. Levy, "Downbeat tracking with tempo invariant convolutional neural networks," in *Proceedings of the 21th International Society for Music Information Retrieval Conference (ISMIR)*, 2020, pp. 216–222.

[41] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 7132–7141.

[42] T. Oyama, R. Ishizuka, and K. Yoshii, "Phase-aware joint beat and downbeat estimation based on periodicity of metrical structure," in *Proceedings of the 22nd International Society for Music Information Retrieval Conference (ISMIR)*, 2021, pp. 493–499.

[43] A. Mosinska, P. Márquez-Neila, M. Kozinski, and P. Fua, "Beyond the pixel-wise loss for topology-aware delineation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 3136–3145.

[44] 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 (TASLP)*, vol. 14, no. 5, pp. 1832–1844, 2006.

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

[46] F. Foscarin, A. McLeod, P. Rigaux, F. Jacquemard, and M. Sakai, “ASAP: a dataset of aligned scores and performances for piano transcription,” in *Proceedings of the 21th International Society for Music Information Retrieval Conference (ISMIR)*, 2020, pp. 534–541.

[47] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, “librosa: Audio and music signal analysis in Python,” in *Proceedings of the 14th python in science conference (SCIPY)*, vol. 8, 2015.

[48] Y. Chuang and L. Su, “Beat and downbeat tracking of symbolic music data using deep recurrent neural networks,” in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, 2020, pp. 346–352.

[49] A. Ycart, A. McLeod, E. Benetos, and K. Yoshii, “Blending acoustic and language model predictions for automatic music transcription,” in *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*, 2019, pp. 454–461.

[50] Y. Wu, B. Chen, and L. Su, “Multi-instrument automatic music transcription with self-attention-based instance segmentation,” *IEEE ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, vol. 28, pp. 2796–2809, 2020.

[51] C. Hawthorne, I. Simon, R. Swavely, E. Manilow, and J. H. Engel, “Sequence-to-sequence piano transcription with transformers,” in *Proceedings of the 22nd International Society for Music Information Retrieval Conference (ISMIR)*, 2021, pp. 246–253.