Drum-Aware Ensemble Architecture for Improved Joint Musical Beat and Downbeat Tracking

Ching-Yu Chiu, Student Member, IEEE, Alvin Wen-Yu Su, and Yi-Hsuan Yang, Senior Member, IEEE

Abstract—This paper presents a novel system architecture that integrates blind source separation with joint beat and downbeat tracking in musical audio signals. The source separation module segregates the percussive and non-percussive components of the input signal, over which beat and downbeat tracking are performed separately and then the results are aggregated with a learnable fusion mechanism. This way, the system can adaptively determine how much the tracking result for an input signal should depend on the input’s percussive or non-percussive components. Evaluation on four testing sets that feature different levels of presence of drum sounds shows that the new architecture consistently outperforms the widely-adopted baseline architecture that does not employ source separation.

Index Terms—Beat/downbeat tracking, source separation.

I. INTRODUCTION

BEATS and downbeats, usually referred to as the sequence of times to tap to when listening to a piece of music, are fundamental information for analyzing and understanding music [1]–[5]. Besides being important in its own right, automatic beat and downbeat tracking holds downstream applications in tasks such as music transcription [6], structural segmentation, automatic accompaniment [7], and music generation [8]. As such, beat/downbeat tracking has been an important topic in signal processing and music information retrieval. And, along with the development of deep learning based techniques, the performance of beat/downbeat tracking has been improved greatly over the recent years [9]–[15].

Research on blind monaural source separation, which concerns with segregating the sound sources involved in a monaural audio recording, has also seen remarkable progress in recent years thanks to deep learning [16]–[19]. Although not yet widely noted in recent work, we conjecture that source separation is mature enough to be integrated to a beat/downbeat tracking model, with the parameters of both optimized jointly, to improve the performance of the latter. We are in particular interested in the case of using a source separation module that separates an input audio to merely two streams—a “drum” component and a “non-drum” component—and then learning to track beats and downbeats also over the resulting two streams, which are supposed to have the same metrical structure as the original input. The rationale behind is that, with or without the drum sounds, human can steadily tap to a song by switching their attention between the percussive and non-percussive parts [20]. Presenting not only the original input but also the separated components to the machine allows it to develop specialized trackers for three types of inputs (i.e., original mixture, drum and non-drum), whose predictions can be fused later on adaptively depending on the acoustic characteristics of the input (see Fig. 1 for an illustration and Section III for details). Doing so endows the machine with the ability to shift its attention to non-conventional parts of the music, and may in turn improve its tracking accuracy across signals that feature different usages of the drum sounds.

It is a recurring observation (e.g., [11]) that the performance of beat/downbeat tracking can vary a lot across different test sets. We conjecture that this is partly because the drum sounds can play fairly different roles in different songs. For example, while acting as a timekeeper in many cases (e.g., in Pop and Rock), drum sounds can be more creative in Jazz and Funk, while be absent in classical or choral music [21]. Using a single network to fit it all, as done in many recent work (e.g.,...
may lead to a model that relies on features that do not necessarily present in all types of audio signals. While there are many other factors that can limit the performance of beat/downbeat tracking (e.g., the presence of local tempo changes [12], [23]), the impact of the versatile role of the drum sounds has not been much studied in the literature, to our best knowledge. It is therefore our goal to quantify this factor. Accordingly, the evaluation of the proposed model architecture is carried out using four test sets that are large enough, and that feature different levels of presence of drum sounds, including a new dataset of classical piano music proposed lately [24], as listed in Table I. Three of the four test sets are kept completely unseen to our training scheme.

We implement our source separation and beat/downbeat tracking models based on the bidirectional long-short term memory (BLSTM) architectures of the respective SOTA, the Open-Unmix model [16] and the RNNDownBeatProcessor of the Madmom library [9], [10]. We open source our code at https://github.com/SunnyCYC/drum-aware4beat.

II. RELATED WORK

Early work on beat/downbeat tracking tends to target on a one specific genre or style at a time with in-depth analysis of the signal properties [4], [23], [25]. For example, the work from Goto and Muraoka [26]–[28] presented a series of signal analysis methods for beat tracking in music with or without drum sounds. Harmonic-percussive source separation methods were also adopted to enhance the performance of dynamic programming-based beat tracking models [29], [30].

In more recent years, researchers began to focus on building systems capable of dealing with different kinds of music, via either feature or model design [31]. The multi-model approach used to be mainstream. For example, both Durand et al. [32], [33] and Krebs et al. [34] tried to improve the performance and robustness of drumtracking with multiple complementary musical features that are fed to independent neural network modules, using the average of these modules to yield the activation functions for the final postprocessing stage. For beat tracking, Böck et al. [2] also proposed a system that combines multiple recurrent neural networks specialised on certain musical styles with a model switcher.

While beat and downbeat tracking used to be treated as separate tasks, multi-task models that jointly estimate beat, downbeat, and even also tempo, have become popular due to the work of Böck et al. [9], [12], [22]. Joint beat/downbeat tracking models proposed since then tend to be a single model that aims to fit all the musical genres. Under the single-model framework, researchers employed data augmentation to deal with unseen or unpopular type of data [12], [35].

The proposed model can be viewed as a multi-task multi-model system that uses learnable drum separation to divide the joint estimation of beats and downbeats in a signal into subproblems, conquer them in parallel, and then fuse the result. This divide-and-conquer idea, which has not been explored before in deep learning based beat/downbeat tracking to our best knowledge, can be easily integrated with any existing models, such as the BLSTM-based model [9] adopted here.

III. PROPOSED METHODS

Table I depicts the proposed model architecture for joint beat and downbeat tracking. As shown on the right hand side of the figure, the model comprises four modules: feature separation, beat/downbeat tracking, fusion, and post processing. From the result of feature separation, which performs source separation in a feature domain, the model uses an ensemble of three beat trackers to get the activation likelihoods of beats, downbeats, and non-beats for the input audio mixture, the drum, and non-drum parts respectively. The fuser aggregates the result from the three trackers (optionally with a two-stage fusion method; see later), and then a postprocessing module, implemented as a hidden Markov model (HMM) [10], makes the final beat and downbeat prediction.

While we follow the BLSTM architectures of Open-Unmix [16] and RNNDownBeatProcessor [9] for the first two modules of our model, the parameters of these two modules, as well as those of the third fuser module, are all to be learned from the training set. Specifically, the loss of all these three modules are detached so that the parameters of each module are optimized using its own training loss function. All the three beat trackers (i.e. mixture, drum, non-drum) are independently supervised by the same set of beat/downbeat annotations for each song in the training set, in the light that the corresponding audio signals share the same metrical structure.

For the last module, we use the HMM available in Madmom [10], with the hyperparameters tuned using our validation set.

A. Feature Separation

The input audio is firstly represented by the combination of three magnitude spectrograms and their first-order derivatives with different window sizes following [9]. This representation, dubbed the mixture feature in Fig. I, is then fed to a drum/non-drum source separation module with a three-layer BLSTM architecture akin to the Open-Unmix (OU) model [16] to generate the drum and non-drum features by masking. The feature...
separation module is supervised by the mean squared error between the features it produced and the features calculated from the reference drum and non-drum audio computed in advance by another source separation library named Spleeter [17]. The officially pretrained Spleeter was trained on a large private dataset and was claimed to outperform the officially pretrained OU. We use the officially pretrained Spleeter for preparing the supervisory drum and non-drum audio signals, instead of adopting its fully-convolutional model architecture for our own feature separation module, as the recurrent neural network based architecture adopted by OU can more easily deal with variable-length audio signals at the inference time.

B. Parallel Beat/Downbeat Trackers

It happens that the RNNDownBeatProcessor of Madmom also has a three-layer BLSTM architecture [10]. We implement such an architecture on our own following the referenced publication [9]. We use the same BLSTM architecture for our three trackers, which take as input the mixture feature, drum feature, and non-drum feature respectively. They each produces a $3 \times T$ matrix indicating the activation likelihood of beat, downbeat, and “non-beat” of each of the $T$ time frames, shown as separate rows in each $3 \times 8$ matrix in Fig. 1 ($T = 8$ here). Namely, each tracker tracks the presence of beats and downbeats jointly (and with separate activation functions), regarding the rest as the non-beat. Cross-entropy loss is applied with an empirically determined weight (67:200:1) to deal with the class imbalance among beat, downbeat, and non-beat.

C. Fusion Mechanism

At the fusion stage, the activation functions from the parallel trackers are combined and used as input to the fuser tracker, also a three-layer BLSTM but with smaller input feature size and less hidden units (10 units in our implementation). The output of the fuser is also a $3 \times T$ matrix. We implement two variants of the fuser. The first variant, DA1 (drum-aware ensemble), simply concatenates the output of the three individual trackers as its input. The second variant, DA2 (two-stage DA), uses additionally the output of the last BLSTM layer of the preceding trackers as input, with an enlarged number of 25 hidden units in the fuser tracker. We assume DA2 may work better as it has access to not only the final output but also the intermediate states of the individual trackers.

IV. EXPERIMENTAL SETUP

A. Datasets

Table I shows the datasets adopted in our experiment. We use the first ten datasets to train our models. Specifically, each of these datasets is randomly split into 80%, 10%, 10% to become part of the training, validation, and the first test set called “Merged.” The last three datasets are kept as completely unseen test sets, including ASAP—a drum-less collection of classical piano music [24], Rock—200 of the Rolling Stone magazine’s list of the “500 Greatest Songs of All Time” [45], and HJDB—a drum-heavy dataset comprising Hardcore, Jungle and Drum&Bass music excerpts [46].

To have an idea of the difference in drum usage in these datasets, we also present in Table I the “drum presence” rate (in %) calculated by the following ad-hoc method. For each song, we employ the officially pretrained Spleeter [17] to derive its drum stem. Then, we compute the mean value of the drum stem’s absolute magnitude (ABSM) in the time domain, and consider the song as a drum-less song if the ABSM is less than 0.01, an empirically set threshold. The drum presence rate is defined as the percentage of songs in a dataset that are not drum-less. We can see that the datasets differ a lot in this rate.

B. Baselines

We implement the following baselines or ablated variants to validate the effectiveness of the proposed ensemble architecture, which is denoted as fuser in Table II.

• baseline: The widely-adopted architecture comprising a single BLSTM-based beat/downbeat tracker and an HMM postprocessing module, implemented following Madmom [9]. Namely, neither source separation nor fusion is used. Similar to [34], we find increasing the layers of BLSTM not helpful and retain the three-layer BLSTM setting.

• mix, drum, nodrum: The single-headed cases where we train the proposed model as usual yet use the output of only one of the individual trackers (i.e., mixture tracker, drum tracker, or non-drum tracker, respectively) as input to the HMM to get the final estimate at evaluation time. This is to validate the advantage of fusion.

• bagging: We implement a simple fuser that takes the averaged activation of the three individual trackers, instead of using the sophisticated BLSTM-based fusion.

C. Model Training & Evaluation Metrics

Our models are trained with the Lookahead Adam optimizer [47] with $10^{-2}$ learning rate. If no improvement on validation loss can be observed for 20 epochs, we reduce the learning rate by a factor of five. To reduce the influence of random initialization, we repeat the training of all models for five times and report the averaged evaluation results. Following the convention of beat/downbeat tracking (e.g., [12]), we report the F-measure (F1) with a tolerance window of $\pm 70$ ms [48].

V. RESULTS AND DISCUSSION

Table II shows the evaluation result. Looking at the first row shows that the Madmom-like baseline already performs quite well across the test sets, especially for beat tracking. The F1 scores degrade and exhibit larger variation across the datasets for downbeat tracking, suggesting downbeats are in general more difficult to track than beats. Both tasks are challenging on ASAP, which is expected as our training set does not contain many classical music pieces.

The next block of rows in Table II, namely the result of the DA1 based models, shows that the proposed ensemble model DA1_fuser consistently outperforms its ablated versions across different test sets in both beat & downbeat tracking, validating the effectiveness of the proposed design. Compared to the result of the baseline, we see salient performance gain in beat
TABLE II

| Model                  | Beat                | Downbeat            |
|------------------------|---------------------|---------------------|
|                        | HDJB                | Merged Rock | ASAP | Mean     | HDJB | Merged Rock | Rock | ASAP | Mean     |
| Baseline (our implementation of [9]) | 0.886 | 0.884 | 0.908 | 0.585 | 0.762 | 0.699 | 0.715 | 0.842 | 0.399 | 0.599 |
| Madmom API [9]         | 0.994 | 0.904 | 0.957 | 0.468 | 0.746 | 0.970 | 0.737 | 0.927 | 0.275 | 0.616 |
| DA1 mix                | 0.875 | 0.876 | 0.901 | 0.584 | 0.756 | 0.656 | 0.713 | 0.817 | 0.396 | 0.585 |
| DA1 drum               | 0.897 | 0.847 | 0.891 | 0.512 | 0.722 | 0.720 | 0.669 | 0.768 | 0.273 | 0.528 |
| DA1 nodrum             | 0.759 | 0.854 | 0.878 | 0.578 | 0.724 | 0.487 | 0.671 | 0.814 | 0.393 | 0.542 |
| DA1 bagging            | 0.875 | 0.875 | 0.907 | 0.574 | 0.753 | 0.754 | 0.736 | 0.844 | 0.383 | 0.608 |
| DA1 fuser              | 0.914 | 0.894 | 0.907 | 0.587 | 0.770 | 0.769 | 0.743 | 0.842 | 0.402 | 0.620 |
| DA2 mix                | 0.879 | 0.877 | 0.901 | 0.587 | 0.759 | 0.651 | 0.701 | 0.831 | 0.403 | 0.587 |
| DA2 drum               | 0.906 | 0.847 | 0.886 | 0.522 | 0.727 | 0.750 | 0.666 | 0.775 | 0.273 | 0.534 |
| DA2 nodrum             | 0.775 | 0.863 | 0.885 | 0.584 | 0.732 | 0.521 | 0.699 | 0.813 | 0.402 | 0.558 |
| DA2 bagging            | 0.904 | 0.880 | 0.910 | 0.582 | 0.764 | 0.802 | 0.739 | 0.853 | 0.387 | 0.621 |
| DA2 fuser              | 0.914 | 0.891 | 0.911 | 0.596 | 0.774 | 0.778 | 0.743 | 0.853 | 0.415 | 0.628 |

and downbeat tracking for HJDB (+7.0% relative improvement in downbeat), which comprises songs from musical genres that are seldom seen in the training set. Looking at the performance of the individual trackers DA1_mix, DA1_drum, DA1_nodrum for HJDB shows that the performance of DA1_drum is the strongest among the three, which is again expected since HJDB features fairly high drum presence rate. This might also explain why the fused result outperforms the baseline. We take this as an evidence of the benefit of building tailored trackers for percussive and non-percussive sounds to take care of the versatile role of the drums in different music signals.

Table II also shows that DA1_drum performs the worst among the three individual trackers for beat & downbeat tracking for ASAP. While the simple fusion method DA1_bagging may suffer from the inconsistent performance of the individual trackers across datasets, DA1_fuser can nicely aggregate the result of the three trackers. Instead of simply taking the average, DA1_fuser might be aware of the absence of the drum sounds from the output of the drum tracker and accordingly decide to rely more on the other two individual trackers.

The last block of rows in Table II, the result of the DA2 based models, shows that DA2 is in general slightly better than DA1. DA2_fuser outperforms DA1_fuser in most cases and leads to the best mean F1 scores in both beat tracking (0.774) and downbeat tracking (0.628). This indicates that we can achieve more effective fusion by taking advantage of the intermediate result of the individual trackers.

In mean F1 scores, DA2_fuser outperforms the baseline by 0.012 in tracking beats and by 0.029 in downbeats. The larger improvement in the latter might be partly due to the available room for improvement, but might also suggest the prediction of downbeats benefits more from separating the effect of the percussive and non-percussive instruments.

One-tailed t-test shows that the per-song performance difference in F1 (averaged over the five runs) between either the pair baseline vs. DA1_fuser, or baseline vs. DA2_fuser, is significant (p-value<0.05) in many cases (bolded in Table II). We also see significant performance difference between DA1_fuser, DA2_fuser for both beats & downbeats on ASAP.

Lastly, we evaluate the models using another metric, called CMLt, defined as the ratio of the total number of correct beats (or downbeats) at correct metrical level to the total number of annotations. While F1 treats a beat/downbeat estimate as correct by whether or not it falls within a fixed-length tolerance window (±70 ms), CMLt adopts a variable-length tolerance window (i.e., ±17.5% of the current inter-annotation interval), and additionally demands consistency between the inter-annotation interval and the inter-beat (or -downbeat) interval. This downplays the effect of tempo drift where occasional beats will be in phase [43]. Table III shows that both DA1_fuser and DA2_fuser still outperform the baseline in CMLt. In many cases (highlighted in Table III), the per-song performance difference in CMLt between either baseline vs. DA1_fuser, or baseline vs. DA2_fuser, is also significant (p-value<0.05) under the one-tailed t-test.

VI. CONCLUSION

Aiming to improve the performance of joint beat/downbeat tracking across different types of musical audio signals, we have presented in this paper a drum-aware ensemble architecture that employs and fuses the result of multiple parallel beat/downbeat trackers for different sound sources in an input signal. Experiment results demonstrated the advantage of such a new multi-model approach over a single-model, Madmom’s RNNDownBeatProcessor-like baseline [9]. Ablation experiments also confirmed the effectiveness of the proposed BLSTM-based fusion mechanism. For future work, we are interested in experimenting with different architectures for the tracking and fusion modules, such as temporal convolutional networks [49] or Transformer [50], and in further improving the performance of beat/downbeat tracking for classical music.
Our system would require extra feature processing to convert the output of the source separation model to become the input to the beat and downbeat tracking system.

To empirically examine the effect of this, we implement such a “Spleeter DA” (or SDA for short) where we replace the trainable feature separation module with a fixed, officially pretrained Spleeter module. Table IV shows the result, averaged over five runs for each model. We can see that, except for the beat F-score on ASAP, the SDA_fusers benefit from the DA idea just like the proposed DA models reported in Table II. Similar trends between SDA and DA models can also be observed. For example, fusers gain more improvements on HJDB and Merged test sets and bagging performs well on the Rock test set. The major differences between SDA and DA are in the results on ASAP. First, unlike the DA_fusers, the SDA_fusers do not gain improvement on ASAP beat performance. Second, the SDA_drum results significantly worse performance on ASAP. We conjecture that this is due to the high-quality SS results of Spleeter. As Spleeter could perform very well on SS, the drum stems it isolates out from pieces in ASAP would be nearly silent, providing SDA_drum nearly no information to track the beats and downbeats. In similar lights, we can also see that the SDA_nodrum perform much worse than the DA_nodrum on HJDB. We take this as an empirical support of the design proposed in Section III-A.

C. Inside the Activation Functions

To gain insights into the behavior of the DA idea, we plot the “averaged” activation function of different trackers for the regions from 10 frames before beat/downbeat position to 10 frames after beat/downbeat position, over all the pieces in a specific test set. This approach enables us to see how each head (i.e., fuser, mix, drum, nondrum) reacts to a beat/downbeat on average. Figures 2 and 3 show the results for ASAP and HJDB, respectively. We can see that for both test sets, each head reacts differently (and sometimes complementarily) to a beat/downbeat. And the fusers of both SDA and DA generally integrate information from other heads and produce the highest peak near beat/downbeat position (i.e., position 10.0 at x-axis). More specifically, we can observe that the drum heads of DA and SDA are clearly more confident than the mix heads on HJDB dataset. And on ASAP dataset, the nodrum heads still react differently than the mix heads. These observations echo

| Model       | Beat | Downbeat |
|-------------|------|----------|
|             | HJDB | Merged   | Rock | ASAP | Mean | Mean |      | HJDB | Merged | Rock | ASAP | Mean | Mean |
| SDA1 mix    | 0.885| 0.885    | 0.917| 0.587| 0.764| 0.661| 0.707| 0.838| 0.404| 0.592|
| SDA1 drum   | 0.897| 0.781    | 0.871| 0.130| 0.543| 0.763| 0.596| 0.724| 0.043| 0.415|
| SDA1 nodrum | 0.701| 0.835    | 0.875| 0.572| 0.705| 0.421| 0.666| 0.801| 0.395| 0.527|
| SDA1 bagging | 0.846| 0.845    | 0.919| 0.452| 0.691| 0.758| 0.708| 0.850| 0.291| 0.564|
| SDA1 fuser  | 0.927| 0.894    | 0.916| 0.585| 0.773| 0.800| 0.735| 0.844| 0.414| 0.630|
| SDA2 mix    | 0.882| 0.871    | 0.888| 0.596| 0.760| 0.701| 0.707| 0.825| 0.428| 0.607|
| SDA2 drum   | 0.886| 0.793    | 0.858| 0.173| 0.559| 0.745| 0.606| 0.698| 0.057| 0.416|
| SDA2 nodrum | 0.679| 0.846    | 0.871| 0.577| 0.705| 0.391| 0.673| 0.795| 0.402| 0.525|
| SDA2 bagging | 0.855| 0.855    | 0.896| 0.457| 0.694| 0.775| 0.727| 0.835| 0.313| 0.579|
| SDA2 fuser  | 0.908| 0.890    | 0.895| 0.587| 0.767| 0.808| 0.744| 0.817| 0.429| 0.635|

APPENDIX

We add to this ArXiv preprint some materials that cannot fit in the SPL paper due to the space limit. First, we comment on the difference between the present work and our earlier work adopting source separation for data augmentation (Aug4Beat) [51]. Second, we present the result of an ablated version of our system which uses the officially pretrained Spleeter model instead of a trainable feature separation module. Lastly, we discuss the averaged activation functions of different trackers on two test sets.

A. On the Difference between Aug4Beat and the Present Work

Aug4Beat [51], a precursor of the present paper, represents our first attempt in using source separation as a way to create additional data for model training. We consider similar experimental setup there and here, using exactly the same training sets and almost the same test sets, making it valid to compare the results in these two papers. But, we remark that we actually use a different optimizer in this paper, and this may play some role. Specifically, in Aug4Beat, we adopt stochastic gradient descent with momentum (SGDM) with a learning rate of $10^{-3}$, following [9]. However, in this paper we adopt the Lookahead Adam optimizer [47] with $10^{-2}$ learning rate, following [12]. We find the new optimizer greatly reduces training time and achieves better overall performance for the DA architecture. Comparing the result of the ‘baseline’ systems in Table III of [51] and Table II of this paper, we see that SGDM performs better than Lookahead Adam on ASAP, but much worse on the Rock dataset. Future work can be done to more closely integrate information from other heads and produce the highest peak near beat/downbeat position (i.e., position 10.0 at x-axis).
On the other hand, since the drum is sensitive than a non-drum feature. Therefore, it could be less sensitive than the mixture input feature, it learns to rely on patterns related to the composition of training data. As a hypothesis that the models are driven by the sound source information/pattern and generate more confident results.

And, the fuser head therefore has higher chance to access to more or better preserved beat/downbeat related information/pattern and generate more confident results.

**REFERENCES**

[1] F. Krebs, S. Böck, and G. Widmer, “Rhythmic pattern modeling for beat and downbeat tracking in musical audio,” in *Proc. Int. Soc. Music Inf. Retr. Conf.*, 2013.

[2] S. Böck, F. Krebs, and G. Widmer, “A multi-model approach to beat tracking considering heterogeneous music styles,” *Proc. Int. Soc. Music Inf. Retr. Conf.*, pp. 603–608, 2014.

[3] M. E. P. Davies, N. Degara, and M. D. Plumbley, “Measuring the performance of beat tracking algorithms using a beat error histogram,” *IEEE Signal Processing Letters*, vol. 18, no. 3, pp. 157–160, 2011.

[4] A. Holzapfel, M. E. P. Davies, J. R. Zapata, J. L. Oliveira, and F. Gouyon, “Selective sampling for beat tracking evaluation,” *IEEE Trans. Audio, Speech Lang. Process.*, vol. 20, no. 9, pp. 2539–2548, 2012.

[5] S. Durand, J. P. Bello, B. David, and G. Richard, “Robust downbeat tracking using an ensemble of convolutional networks,” *IEEE/ACM Trans. Audio, Speech, and Language Processing*, vol. 25, no. 1, pp. 76–89, 2017.

[6] E. Benetos et al., “Automatic music transcription: An overview,” *IEEE Signal Processing Magazine*, vol. 36, no. 1, pp. 20–30, 2019.

[7] G. Burlou, “An online tempo tracker for automatic accompaniment based on audio-to-audio alignment and beat tracking,” in *Proc. Sound and Music Computing Conf.*, 2016, pp. 93–98.

[8] Y.-S. Huang and Y.-H. Yang, “Pop Music Transformer: Beat-based modeling and generation of expressive pop piano compositions,” in *Proc. ACM Int. Conf. Multimedia*, 2020, p. 1180–1188.

[9] S. Böck, F. Krebs, and G. Widmer, “Joint beat and downbeat tracking with recurrent neural networks,” *Proc. Int. Soc. Music Inf. Retr. Conf.*, pp. 255–261, 2016.

[10] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and G. Widmer, “Madmom: A new Python audio and music signal processing library,” *Proc. ACM Multimedia Conf.*, pp. 1174–1178, 2016.

[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. Int. Soc. Music Inf. Retr. Conf.*, 2018, pp. 106–112.

[12] S. Böck and M. E. P. Davies, “Deconstruct, analyse, reconstruct: How to improve tempo, beat, and downbeat estimation,” in *Proc. Int. Soc. Music Inf. Retr. Conf.*, 2020, p. 574–582.

[13] B. D. Giorgi, M. Mauch, and M. Levy, “Downbeat tracking with tempo invariant convolutional neural networks,” in *Proc. Int. Soc. Music Inf. Retr. Conf.*, 2020, p. 216–222.

[14] E. Cano, F. M. Angel, G. A. L. Gil, J. R. Zapata, A. Escamilla, J. F. A. Londoño, and M. B. Pelaez, “Sesquialtera in the Colombian Bambuco: Perception and estimation of beat and meter,” in *Proc. Int. Soc. Music Inf. Retr. Conf.*, 2020, p. 409–415.

[15] F. Pedersoli and M. Goto, “Dance beat tracking from visual information alone,” in *Proc. Int. Soc. Music Inf. Retr. Conf.*, 2020, p. 400–408.

[16] F.-R. Stöter, S. Uhlich, A. Liutkus, and Y. Mitsufuji, “Open-Unmix - A reference implementation for music source separation,” *J. Open Source Softw.*, vol. 4, no. 41, p. 1667, 2019.

[17] R. Hennequin, F. V. A. Khlf, and M. Moussallam, “Spleeter: A fast and state-of-the art music source separation tool with pre-trained models,” *J. Open Source Softw.*, vol. 5, no. 50, p. 2154, 2020.

[18] J.-Y. Liu and Y.-H. Yang, “Dilated convolution with dilated GRU for music source separation,” in *Proc. Int. Joint Conf. Artificial Intelligence*, 2019, pp. 4718–4724.

[19] C.-Y. Chiu, W.-Y. Hsiao, Y.-C. Yeh, Y.-H. Yang, and A. W. Y. Su, “Mixing-specific data augmentation techniques for improved blind violin/piano source separation,” in *Proc. IEEE Int. Workshop on Multimedia Signal Processing*, 2020.

[20] F. Gouyon, G. Widmer, X. Serra, and A. Flexer, “Acoustic cues to beat induction: A machine learning perspective,” *Music Perception*, vol. 24, no. 2, pp. 177–188, 2006.

[21] B. Jia, J. Lv, and D. Liu, “Deep learning-based automatic downbeat
tracking: a brief review,” Multimedia Systems, vol. 25, no. 6, pp. 617–638, 2019.

22. S. Böck, M. E. P. Davies, and P. Knees, “Multi-task learning of tempo and beat: Learning one to improve the other,” Proc. Int. Soc. Music Inf. Retr. Conf., pp. 486–493, 2019.

23. P. Grosche, M. Müller, and C. S. Sapp, “What makes beat tracking difficult? a case study on Chopin Mazurkas,” Proc. Int. Soc. Music Inf. Retr. Conf., pp. 649–654, 2010.

24. F. Foscarin, A. McLeod, P. Rigaux, F. Jacquemard, and M. Sakai, “ASAP: a dataset of aligned scores and performances for piano transcription,” in Proc. Int. Soc. Music Inf. Retr. Conf., 2020, pp. 534–541.

25. J. R. Zapata and E. Gomez, “Using voice suppression algorithms to improve beat tracking in the presence of highly predominant vocals,” Proc. IEEE Int. Conf. Acoust. Speech Signal Process., pp. 51–55, 2014.

26. M. Goto and Y. Muraoka, “Music understanding at the beat level real-time beat tracking for audio signals,” Proc. Int. Joint Conf. Artificial Intelligence Workshop on Computational Auditory Scene Analysis, 1995.

27. ——, “Real-time beat tracking for drumless audio signals: Chord change detection for musical decisions,” Speech Communication, vol. 27, no. 3, pp. 311–335, 1999.

28. M. 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.

29. B. McFee and D. P. Ellis, “Better beat tracking through robust onset aggregation,” in Proc. IEEE Int. Conf. Acoust. Speech Signal Process., 2014, pp. 2154–2158.

30. A. Robertson, A. Stark, and M. E. P. Davies, “Percussive beat tracking using real-time median filtering,” Int. Workshop on Machine Learning and Music, 2013.

31. S. Durand, B. David, and G. Richard, “Enhancing downbeat detection when facing different music styles,” in Proc. IEEE Int. Conf. Acoust. Speech Signal Process., 2014, pp. 3132–3136.

32. S. Durand, J. P. Bello, B. David, and G. Richard, “Downbeat tracking with multiple features and deep neural networks,” Proc. IEEE Int. Conf. Acoust. Speech Signal Process., pp. 409–413, 2015.

33. ——, “Robust downbeat tracking using an ensemble of convolutional networks,” IEEE/ACM Trans. Audio Speech Lang. Process., vol. 25, no. 1, pp. 72–85, 2017.

34. F. Krebs, S. Böck, M. Dorfer, and G. Widmer, “Downbeat tracking using beat-synchronous features and recurrent neural networks,” Proc. Int. Soc. Music Inf. Retr. Conf., pp. 120–135, 2016.

35. H. Schreiber and M. Muller, “A single-step approach to musical tempo estimation using a convolutional neural network,” Proc. Int. Soc. Music Inf. Retr. Conf., pp. 98–105, 2018.

36. M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, “RWC Music Database: Popular, Classical, and Jazz music databases,” in Proc. Int. Soc. Music Inf. Retr. Conf., 2002, pp. 287–288.

37. ——, “RWC Music Database: Music genre database and musical instrument sound database,” in Proc. Int. Soc. Music Inf. Retr. Conf., 2003, pp. 229–230.

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

39. M. Macleod and S. Hainsworth, “Particle filtering applied to musical tempo tracking,” EURASIP Journal on Advances in Signal Processing, vol. 2004, no. 5, pp. 2385–2395, 2004.

40. U. Marchand and G. Peeters, “Swing ratio estimation,” in Proc. Digital Audio Effects, Trondheim, Norway, 2015, pp. 423–428.

41. G. Tzanetakis and P. Cook, “Musical genre classification of audio signals,” IEEE Trans. Speech and Audio Processing, vol. 10, no. 5, pp. 293–302, 2002.

42. A. Srinivasasurumathy and X. Serra, “A supervised approach to hierarchical metrical cycle tracking from audio music recordings,” in Proc. IEEE Int. Conf. Acoust. Speech Signal Process., 2014, pp. 5217–5221.

43. M. E. P. Davies, N. D. Quintela, and M. Plumley, “Evaluation methods for musical audio beat tracking algorithms,” in Queen Mary University of London, Centre for Digital Music, Tech. Rep. C4DM-TR-09-06, 2009.

44. B. D. Giorgi, M. Zanoni, A. Sarti, and S. Tubaro, “Automatic chord recognition based on the probabilistic modeling of diatonic modal harmony,” in Proc. Int. Workshop on Multidimensional Systems, 2013.

45. T. de Clercq and D. Temperley, “A corpus analysis of rock harmony,” Journal of New Music Research, vol. 30, no. 1, pp. 47–70, 2011.

46. J. A. Hockman, M. E. P. Davies, and I. Fujinaga, “One in the jungle: Downbeat detection in hardcore, jungle, and drum and bass,” Proc. Int. Soc. Music Inf. Retr. Conf., pp. 169–174, 2012.