SOURCE SEPARATION-BASED DATA AUGMENTATION TECHNIQUES FOR IMPROVED JOINT BEAT AND DOWNBEAT TRACKING IN CLASSICAL MUSIC

Ching-Yu Chiu[1,2]*, Joann Ching*[2]*, Wen-Yi Hsiao*[3]*, Yu-Hua Chen*[2,3]*, Alvin Wen-Yu Su*[1]*, and Yi-Hsuan Yang*

[1] Graduate Program of Multimedia Systems and Intelligent Computing,
National Cheng Kung University and Academia Sinica, Taiwan
[2] Research Center for IT Innovation, Academia Sinica, Taiwan
[3] Yating Music Team, Taiwan AI Labs, Taiwan
{sunnycyc,joann8512,yang}@citi.sinica.edu.tw

ABSTRACT

Beat/downbeat tracking have been fundamental and important topics in music information retrieval research. Though deep learning-based models have achieved great success for music with stable and clear beat positions, it remains quite challenging for classical music in the absence of a steady drum sound. In this paper, we propose a novel source separation-based data augmentation technique that is tailored for beat/downbeat tracking in classical music. This involves a model that separates drum and non-drum sounds, and mechanisms to perform drum stem selection. We report comprehensive experiments validating the usefulness of the proposed methods.

***Index Terms*—** Beat/downbeat tracking, source separation, data augmentation, classical music

1. INTRODUCTION

Beats/downbeats are usually referred to as a sequence of time instants that human would tap their feet to [1, 2]. Such concepts define the metrical structure of a music piece, and enable numerous higher level tasks such as structure segmentation, automated DJ mixing, and score alignment [2–4]. Moreover, they are essential foundations for machines to understand music. As humans are able to track musical beats without difficulty in various musical genres regardless of changes in local tempo, people expect machines to have similar skills. Thanks to cumulative efforts in the research community, recent years witness great success of deep learning-based supervised models for beat/downbeat tracking for music with steady and strong beats, such as pop, rock, and dance [5]. Unfortunately, their performances in classical music are seldom reported or are at a level that is far from satisfaction [5–8].

Existing deep learning-based methods for beat/downbeat tracking are mainly supervised models that require paired training data (i.e., audio with annotated labels of beat and downbeat timing). These approaches usually consist of a neural network module and a post-processing dynamic model. The neural network module takes as input a low-level feature (e.g., chroma or spectral flux [2, 9–11]) and outputs an activation function that indicates the most likely beats/downbeats time candidates. The post-processing model, be it a dynamic Bayesian network (DBN), hidden Markov model (HMM) or conditional random field (CRF), makes binary predictions from the activation function [2, 11, 12].

Beat/downbeat tracking for classical music is challenging mainly due to the inherent signal characteristics of classical music, such as the presence of local tempo changes and rhythmic variation for purposes of expressive performance, and the rather blurred note onsets and transitions caused by non-percussive instruments [6, 13]. From a machine learning perspective, the difficulty may also be related to limitations of model capacity, insufficient paired data for training, or potential imbalance of training data. For example, we note that most available training datasets for beat/downbeat tracking are pop/rock songs [5].

Although not often mentioned together with beat tracking, music source separation has been another important topic in music information research. Given a monaural input mixture (i.e., an audio recording of multiple instruments sounding together), a source separation model generates separated stems of different sound sources that composed the mixture [14–18]. Noticing that the total duration and metrical structure of the stems are basically the same, we conjecture that it may be useful to apply drum/non-drum separation as a means of data augmentation, to increase the amount of non-drum data in our training set. It is therefore our goal in this paper to investigate variants of such augmentation data and their usefulness in improving the performance of beat/downbeat tracking in classical music. To the best of our knowledge, such a study has never been presented before.

As our focus is on the data augmentation methods, we simply adopt as our model architecture the pipeline of cascading a long short-term memory (LSTM) network with an HMM-based dynamic network proposed by `Madmom` [2, 19], which represents the state-of-the-art for beat and downbeat

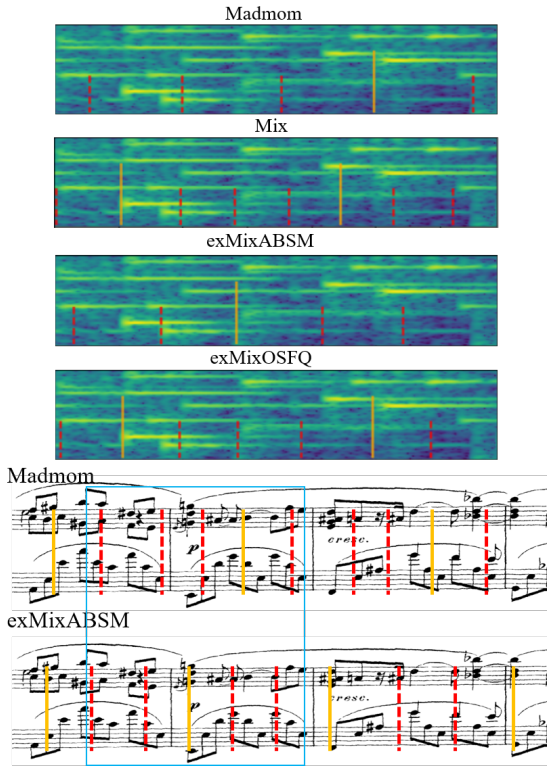

Fig. 1. Estimation of beat (red dash lines) and downbeat (yellow lines) on a piano performance of Scriabin Fantasy Op. 28 in B Minor. The score corresponding to the spectrograms is enclosed by the blue square. We show two copies of the score, each marked with the beat/downbeat estimation of a model. This is a rather complicated piece of music so all the models do not perform well. Yet, the proposed model *exMixABSM* seems to produce relatively more reasonable result, presumably because it is trained with more non-percussive data.

tracking across various genres. Our experiment shows that, with the proposed data augmentation, we are already able to improve beat/downbeat tracking for classical music, as illustrated in Figure 1 (see Section 5.3 for details). We provide audio demos at `https://sunnycyc.github.io/aug4beat_demo/`.

2. BACKGROUND

As available paired data for training supervised deep learning models for beat tracking is limited, research has been done to enhance the generalizability of models, or to compensate for the relatively insufficient properties in the training set. Giorgi *et al.* [20], for example, proposed a deterministic time-warping operation to help their model learn rhythmic patterns independently of tempo. Bock & Davies [5] devised a novel multi-task approach to leverage shared connections in musical structure, and to simultaneously estimate tempo, beat, and downbeat. They also proposed an augmentation method based on changing parameters of the short-time Fourier transform to expose their model to a wider range of tempi. Zapata & Gomez [3] proposed an audio voice suppression technique and an simple low-pass filter to improve beat tracking.

Efforts have been made to tackle beat tracking by treating percussive/non-percussive features separately. Goto [21] developed a method to judge if the input audio signal contains drum sounds, and then used different ways to track the beats for music with or without drum sounds. Gkiokas *et al.* [22] presented an tempo estimation and beat tracking algorithm which utilized source separation to extract features of percussive/harmonic components separately. Our work is different from these prior arts in that we use source separation as a means to create augmented data for training a supervised neural network model.

We implement a beat/downbeat tracking model on our own following the LSTM+HMM architecture described in the `Madmom` library. The network consists of three fully-connected bidirectional recurrent layers with 25 LSTM units each. After the LSTM layers, a softmax classification layer with three units produces three activation functions (i.e., curves) corresponding to the probability of a frame being 'a beat but no a downbeat,' 'a downbeat' or 'a non-beat' position. The output activation functions are then processed by an HMM to produce the final binary beat/downbeat predictions. In other words, the model predicts beat and downbeat jointly. In this paper, the bar length setting required by HMM is set as three or four beats, following the default setting of `Madmom`.

3. PROPOSED METHODS

3.1. Source Separation-based Data Augmentation

We propose to employ a source separation model to isolate out the drum sounds and non-drum sounds for the music pieces in the training data for beat/downbeat tracking. In doing so, we employ Spleeter [16], the current state-of-the-art model for musical source separation that has publicly available model checkpoints. Spleeter can generate five separated stems, each stem corresponding to the sounds of the piano, vocal, drum, bass, and others, respectively. We simply sum all but the drum stem as the *non-drum* stem. If we call the original pieces in the training set as *mixes*, we would get the same number of drum stems and non-drum stems after source separation. For those pieces that correspond to the same song, we assume that they share the same metrical structure. Accordingly, we can assume that the labels of beat and downbeat timing apply equally well to that corresponding drum stem and non-drum stem.

3.2. Drum Stem Selection

Since some of our training data may not contain any drum sounds originally, the drum stems derived from them would

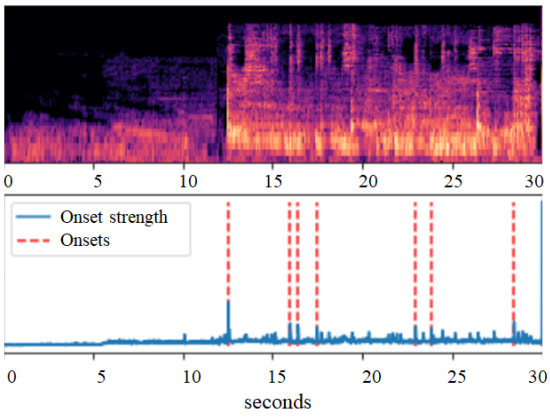

Fig. 2. An example of a separated drum stem that would pass the ABSM filter as it has recognizable drum sounds, yet would not pass the OSFQ filter, for it lacks regular beat structure as can be seen from the result of onset detection.

be nearly silent and could be harmful if adopted as training data. We devise two drum selection criteria to tackle this. **ABSM**: Exclude a drum stem if the mean value of its absolute magnitude in the time domain is less than 0.01, an empirically set threshold. **OSFQ**: Exclude a drum stem if the mean value of its absolute magnitude is less than 0.001 (i.e., looser than ABSM), *or* if it has less than one prominent onset per second, as detected by `librosa.onset.onset_detect` [23]; i.e., when its onset patterns is sparse or irregular. See Figure 2 for an example of bass drum excluded by OSFQ.

After drum selection, we have in total four types of data: *Mix*, *non-drum*, *only-drum (ABSM)* and *only-drum (OSFQ)*, for all the pieces in our training set, not necessarily classical pieces only. To train our models, we can use any data type alone, or different combinations of them. For example, if we use only *non-drum* to train our trackers, we are essentially enforcing the tracker to recognize the downbeats and beats for all those pop/rock/dance songs in the training set as if there is no drums in the original songs. This is likely to be more challenging for the trackers, but this may improve the result when testing on classical music. Please see Table 1 for the datasets employed in training our models.

4. EXPERIMENTS

We implement models with many data type combinations that are enabled by source separation, as listed in Table 2. The goal is to see any of these can outperform the baseline case of using the *Mix* alone for model training, as it represents the standard approach as of now. For these models, we randomly split the datasets listed in Table 1 into 80%, 10%, 10% for training, validation and test. During validation, the parameters of the post-processing HMM model are optimized. The proposed data augmentation applies to the training sets only.

Dataset	#	duration	ABSM	OSFQ
Ballroom	685	5h 57m	2h 46m	3h 55m
Beatles	180	8h 09m	5h	6h 05m
Hainsworth	222	3h 19m	1h 02m	1h 18m
RWC Popular	100	6h 47m	4h 40m	4h 45m
Robbie Williams	65	4h 31m	3h 19m	3h 33m
Carnatic	176	16h 38m	9h 56m	11h 26m
GTZAN	999	8h 20m	4h 47m	5h 18m
RWC Classical	54	5h 19m	0	0
RWC Jazz	50	3h 42m	1h	1h 14m
RWC Music genre	100	7h 20m	2h 10m	3h 38m

Table 1. The datasets employed in our experiments; '#' denotes the number of pieces. The last two columns indicate the total duration of the selected drum stems for training. Most of the 10 datasets listed here have the non-drum and drum-only versions after source separation, except for RWC Classical [24], which has no presence of drum sounds at all.

Model	mix	non-drum	onlyDrum	
			ABSM	OSFQ
Mix	√			
noDrum		√		
onlyDrumABSM			√	
onlyDrumOSFQ				√
mix+noDrum	√	√		
exMix_ABSM		√	√	
exMix_OSFQ		√		√
combine3_ABSM	√	√	√	
combine3_OSFQ	√	√		√

Table 2. The data type combinations tested in our experiments, along with the short names of the corresponding models. Each model is trained from scratch using the ticked data types of the training split of the 10 datasets listed in Table 1.

We also include the result of `Madmom` API in our experiments as a reference. However, we note that it is likely that the model `Madmom` has been trained with many other datasets that are unavailable to us (e.g., the HJDB, Rock, Cretan, Turkish, Klapuri datasets [2, 5]). Hence, we consider the *Mix* model as the main baseline.

We are mainly interested in the result on the test split of RWC Classical [24] in the experiments. To make a contrast, we also report the result on RWC Popular [24].

5. RESULTS AND DISCUSSION

5.1. Result on the RWC Classical test set

Table 3 shows the results of the proposed methods. The following observations can be made. Models trained with non-

Model	F_1 beat	F_1 downbeat
exMix_OSFQ	**0.815**	**0.570**
exMix_ABSM	0.794	0.568
noDrum	0.804	0.553
Mix	**0.779**	**0.557**
mix+noDrum	0.742	0.576
Madmom	**0.800**	**0.517**
combine3_ABSM	0.794	0.510
onlyDrumOSFQ	0.750	0.552
combine3_OSFQ	0.740	0.524
onlyDrumABSM	0.707	0.275

Table 3. Beat and downbeat tracking F-measure on the RWC Classical test set. Except for Madmom, all the models are trained with the combinations of data types (cf. Table 2) of the training split of all the datasets listed in Table1. We highlight the result of Madmom, our baseline, and our best model.

Model	F_1 beat	F_1 downbeat
Madmom	**0.986**	**0.980**
mix+noDrum	**0.949**	**0.909**
combine3_OSFQ	0.907	0.884
combine3_ABSM	0.849	0.824
Mix	**0.885**	**0.764**
onlyDrumOSFQ	0.848	0.708
exMix_ABSM	0.818	0.702
noDrum	0.796	0.626
exMix_OSFQ	0.675	0.637
onlyDrumABSM	0.818	0.323

Table 4. Evaluation result on the RWC Popular test set. We note that the significant performacne difference between Madmom [2, 19] and our reproduced *Mix* model could be caused by differences in the adopted training datasets.

drum data and without mixture seem to perform better here. As *noDrum* model achieves overall performance better than baseline model *Mix*, we may say that simply non-drum feature could be enough for models to have a basic idea about beats for classical music. And if selected drum stems are included in training, the performance could even be better. However, a significant performance drop is also observed when using all data (i.e., *combine3*), or using only drum stems for training. This may look counter-intuitive as using more data can even deteriorate a model's performance. We conjecture that this is due to the inherent acoustic differences of mixture, non-drum, and drum. The high level features that the models could extract to track beats should be different in the three kinds of input features. Therefore, manipulating the composition of training data could significantly change the way models process the input features. As our classical test set contains no drum (see Table 1), including some drum feature may be seen as augmentation (e.g., the cases of *exMix*), yet including too much data heterogeneous to the target data would not be helpful (e.g., the cases of *combine3* and *onlyDrum*).

5.2. Result on the RWC Popular test set

Table 4 shows the results on the RWC Popular test set. The following observations can be made. We see a fairly reverse performance ranking order of the implemented models in the result for pop music and classical music. Since pop music usually has percussive sounds throughout the song, models never seen mixture during training all falls behind the baseline *Mix*, and suffers from significant performance drop. In contrast, models trained using mixture with some extra data as augmentations all gain improvement and perform better than the baseline *Mix*. Again, including more data (e.g., the cases of *combine3*) is not always better; *mix+noDrum* gains the most improvement in this experiment. Another interesting observation is that our reproduced baseline *Mix* is not on par with Madmom, presumably because we are not able to collect all the training sets mentioned in the literature.

5.3. Case Study: Expressive Classical Piano Performance

To evaluate the capability of these models (*Mix*, *exMix_OSFQ*, *exMix_ABSM* and Madmom) on real classical performances that are much more complicated, we apply them to music pieces of MAESTRO [25]. We find that the augmented models have better adaptability to difficult cases and are more likely to produce a reasonable result than the baseline *Mix* model. Figure 1 presents the result for a representative example. Despite of minor mistakes, *exMix_ABSM* is the only model that correctly recognizes that the piece is in 3 beats per measure. In a piece that groups three notes together, mistaking the 3-beat meter to 4-beat can be unacceptable.

6. CONCLUSIONS

In this paper, we have proposed and studied variants of source separation-enabled data augmentation methods tailored for training deep learning-based supervised beat/downbeat tracking models. In-depth analysis of the proposed methods shows that applying drum/non-drum separation to the training set of beat/downbeat tracking models, which are not necessarily classical music pieces, improves the performance of these models on unseen classical music pieces. For future work, we are interested in building a beat/downbeat tracking model that incorporates source separation as part of the model, and in conducting experiments on more classical music pieces.

7. REFERENCES

[1] S. Böck, F. Krebs, and G. Widmer, "A multi-model approach to beat tracking considering heterogeneous mu-

sic styles," *Proc. Int. Soc. Music Inf. Retr. Conf.*, pp. 603–608, 2014.

[2] 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.

[3] 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, 2013.

[4] H. Grohganz, M. Clausen, and M. Müller, "Estimating musical time information from performed MIDI files," *Proc. Int. Soc. Music Inf. Retr. Conf.*, pp. 35–40, 2014.

[5] S. Böck and M. E. P. Davies, "Deconstruct, analyse, reconstruct: How to improve tempo, beat, and downbeat estimation," *Proc. Int. Soc. Music Inf. Retr. Conf.*, 2020.

[6] 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.

[7] 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.

[8] S. Durand and S. Essid, "Downbeat detection with conditional random fields and deep learned features," *Proc. Int. Soc. Music Inf. Retr. Conf.*, pp. 386–392, 2016.

[9] 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.

[10] 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. 129–135, 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," *Proc. Int. Soc. Music Inf. Retr. Conf.*, pp. 106–112, 2018.

[12] S. Durand, J. P. Bello, B. David, and G. Richard, "Robust downbeat tracking using an ensemble of convolutional networks," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 25, no. 1, pp. 72–85, 2017.

[13] D. P. W. Ellis, "Better beat tracking through robust onset aggregation," *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, pp. 2154–2158, 2014.

[14] S. Uhlich et al., "Improving music source separation based on deep neural networks through data augmentation and network blending," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2017, pp. 261–265.

[15] Z. Rafii, A. Liutkus, F.-R. Stöter, S. I. Mimilakis, D. FitzGerald, and B. Pardo, "An overview of lead and accompaniment separation in music," *IEEE/ACM Trans. Audio, Speech, and Language Processing*, vol. 26, no. 8, pp. 1307–1335.

[16] F. Voituret R. Hennequin, A. Khlif and M. Moussallam, "Spleeter: A fast and state-of-the art music source separation tool with pre-trained models," *J. Open Source Softw.*, 2020.

[17] 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, pp. 1667, 2019.

[18] 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," *Proc. IEEE Int. Workshop on Multimedia Signal Processing*, 2020.

[19] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and G. Widmer, "Madmom: A new python audio and music signal processing library," *Proc. ACM Multimed. Conf.*, pp. 1174–1178, 2016.

[20] B. D. Giorgi, M. Mauch, and M. Levy, "Downbeat tracking with tempo-invariant convolutional neural networks," *Proc. Int. Soc. Music Inf. Retr. Conf.*, pp. 216–222, 2020.

[21] M. Goto, "An audio-based real-time beat tracking system for music with or without drum-sounds," *Int. J. Phytoremediation*, vol. 21, no. 1, pp. 159–171, 2001.

[22] A. Gkiokas, V. Katsouros, G. Carayannis, and T. Stajylakis, "Music tempo estimation and beat tracking by applying source separation and metrical relations," *Proc. IEEE Int. Conf. Acoust. Speech Signal Process*, pp. 421–424, 2012.

[23] B. McFee, C. Raffel, D. Liang, D. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "Librosa: Audio and music signal analysis in Python," *Proc. Python in Science Conf.*, pp. 18–24, 2015.

[24] M. Goto et al., "RWC Music Database: Popular, Classical, and Jazz Music Databases," *Proc. Int. Soc. Music Inf. Retr. Conf.*, 2002.

[25] C. Hawthorne et al., "Enabling factorized piano music modeling and generation with the Maestro dataset," *Proc. Int. Conf. Learning Representations*, 2019.