Convolving Gaussian Kernels for RNN-Based Beat Tracking

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

***Abstract*—Because of an ability of modelling context information, Recurrent Neural Networks (RNNs) or bi-directional RNNs (BRNNs) have been used for beat tracking with good performance. However, there are two problems associated with RNN-based beat tracking. The first problem is the imbalanced data: usually only around 2% frames are labelled as 'beat'. The second one is the disagreement on the precise positions of beats in human annotations or the delay of annotations caused by human tapping. In order to tackle these problems, we propose to convolve the original ground truth with a Gaussian kernel as the target output of the network for a more robust training. We conduct a comparison experiment using five different Gaussian kernels on five individual datasets. The results on the validation sets show that we can train a better or at least competitive model in a shorter time by using the convolved ground truth with a proper Gaussian kernel.**

I. Introduction

Beat tracking [1] is defined as the task to derive a sequence of beat instants from a music audio signal. Due to temporal continuity, contextual information of previous beats and following beats is valuable for beat tracking. Since 2010, it has become popular to apply deep neural networks for beat tracking and downbeat (the first beat of a measure in music) tracking. Because Recurrent Neural Networks (RNNs) are good at modeling context information (with bi-directional RNNs (BRNNs) for modeling both the past and future contexts), they have been used for beat tracking [2], [3], [4], downbeat tracking [5] and joint beat and downbeat tracking [6]. Other related work includes an ensemble of Convolutional Neural Networks (CNNs) which are trained on harmonic and rhythm features for downbeat tracking [7], [8]. Metrical continuity is tackled in a post-processing stage with the learned features from CNNs.

In this paper we are interested in beat tracking based on the bi-directional RNN, because such models show good performance in beat tracking [2], [3], [4]. A drawback of these models is the biased labels which may lead to an inefficient learning. The proportions of two labels ('beat' and 'no beat') are very imbalanced. With a tempo of 120 bpm and a frame of 10ms, only 2% data will be labelled as beat. With a slower tempo, the occurrence of beats is even sparser. The other problem is that human annotations may slightly deviate from the precise timing of beats, because of the subjective perception or the delay caused by manually tapping [4].

In order to tackle these two problems, we propose to convolve the ground truth labels with a Gaussian kernel for training BRNNs. Similar techniques with smooth or fuzzy labels are used for different reasons. The target output of an RNN is transformed into a conditional probability distribution over label sequences in order to classify unsegmented sequence data [9]. An RNN is trained with fuzzy ground truth (all possible annotations) because of imperfect manual annotations [10]. A CNN-based music boundary detection method [11] proposes to use frames around annotated boundaries as positive examples (weighted by a Gaussian kernel) because of the disagreement between human annotaions. [12] convolves the ground truth labels with a Gaussian kernel in a CNN-based music boundary detection system, because of the sparse occurrence of the segment boundaries and the perceptual variance in the ground truth. In this paper, we apply the convolved labels to train BRNNs for beat tracking. In addition, we study the impact of using different labels on the training. The results show that a better or competitive performance can be achieved in a shorter time by using the convolved labels with a proper Gaussian kernel.

In the rest of the paper, we introduce existing RNN-based beat tracking models in Section II. Section III conducts a comparison experiment with labels convolved with different Gaussian kernels on five datasets. In Section IV, we illustrate the comparison results with examples. Discussions and conclusions are drawn in Section V and Section VI, respectively.

II. Existing RNN-based models

We study existing RNN-based beat tracking models [2], [3], [4] in this section.

A. Pre-processing

All three models use input features based on Mel spectrograms [2], [3], [4]. There are three Mel spectrograms computed with various window lengths (23.2ms, 46.4ms and 92.8ms) and the same hop-size of 10ms. For the spectral differences, [3], [4] use the positive first order differences of the Mel spectrograms; in [2], instead of the previous spectrum, a median spectrum of previous spectra is subtracted to obtain the positive first order median difference. Then the three Mel spectrogram and their differences are concatenated along the frequency axis as the input of the networks.

B. Neural Network

All three models use a bi-directional RNN with Long Short-Term Memory (LSTM) units [2], [3], [4]. The LSTM unit is usually used with the RNN to overcome the vanishing gradient

ISBN 978-90-827970-1-5

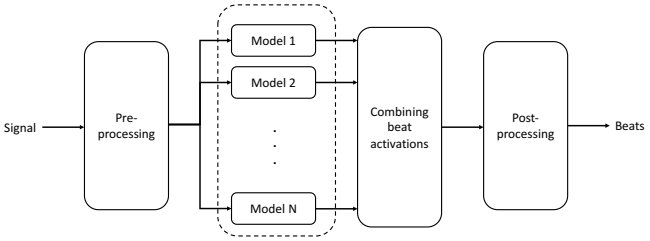


Fig. 1: Overview of the beat tracking systems.

TABLE I: Optimisation of the RNN trainings.

Ref.	optimiser	learning rate
[2]	standard gradient descent	not available
[3]	standard gradient descent	10^{-4} (fine-tuned with 10^{-5})
[4]	stochastic gradient descent	10^{-4} (a momentum of 0.9)

problem [13]. The fully connected network has 3 hidden layers in each direction, with 25 LSTM units in each layer. The output layers (with the softmax activation function) has two units for two classes of 'beat' and 'no beat'. The network is trained as a classifier with the cross entropy error function. The output nodes represent the probabilities of the two classes.

C. Network training

Different models are trained and combined for beat tracking in [2], [3], as shown in Figure 1. In [2] five models are trained on different sets of the training datasets, then the outputs of different models are averaged as the input of the post-processing stage. In [3] models are trained on individual training datasets (assuming different music styles) for a better beat detection in heterogeneous music styles. The output mostly close to the output of the reference model (trained on all the data) is chosen as the input of the post-processing stage. In [4] the network is trained in an 8-fold cross validation setting based on a random splitting of the training datasets.

Weights of the network are randomly initialised using a Gaussian distribution with mean 0 and standard deviation 0.1 in [2], [3], and with a uniform random distribution with range $[-0.1, 0.1]$ in [4]. Standard gradient descent and stochastic gradient decent are used for optimisation, with settings shown in Table I. To prevent overfitting, the training is stopped if no improvement of the cross entropy error of the validation set can be observed for 20 epochs.[1]

D. Post-processing

Beats are tracked on beat activations obtained by the BRNN models, by using autocorrelation function [2], dynamic Bayesian network [3].

In [4], the output of the BRNN (the beat activation function) is convolved with a Hamming window of length 140ms to obtain a smooth activation function. Then beats are tracked on the convolved beat activations by using comb filters. The convolution in this post-processing stage [4] is different from the convolution in our method, because it works on the output of the network and makes no change to the training stage. Our method manipulates the target labels, which changes the training stage.

[1]One epoch consists of one full training cycle on the training set.

III. A comparison experiment

In this section, we study the impact of using different labels on training BRNNs for beat tracking. The labels are generated by convolving the ground truth labels with different Gaussian kernels. We compare the training of the BRNNs using 5 different labels on 5 individual datasets.

A. The basic network architecture

In this experiment we adopt the pre-processing, post-processing steps and the network architecture from [2], [3], [4], [14].

1) Pre-processing and post-processing: We adopt the signal pre-processing step in [3], [4] to compute the input features for the neural network. For the post-processing stage, we apply the comb filter periodicity estimation method [4] to track beat positions from the network outputs, with the implemetations of [14]. See Section II-A and II-D for details.

2) Neural network: We apply a bi-directional RNN with LSTM units as in [2]. There are 3 hidden layers in both directions, with 25 LSTM units each layer (Section II-B). Weights are initialised with a Gaussian distribution of mean 0 and standard deviation 0.1.

B. Comparison parameters

The annotation data are processed in 10ms frames, with beat frames labelled 1 and other frames labelled 0. We convolve the ground truth beat labels with a Gaussian kernel function $g(x)$ as the target outputs for training the BRNNs:

$$g(x) = e^{-\frac{1}{2}(\frac{x}{\sigma})^2}, x \in [-2\sigma, 2\sigma]. \quad (1)$$

The Gaussian kernels with different standard deviations are shown in Figure 2. The s_i means labels obtained by convolving a Gaussian kernel with a standard deviation of i. The s_0 denotes the original labels.

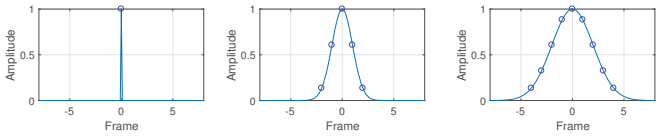
By doing so, the labels of the BRNN model are no longer binary values, but indicate the probabilities of 'beats' and 'no beat'. The sum of the values of the 'beat' and 'no beat' labels is 1 for each frame. Because there is a post-processing step to track beats on the BRNN outputs, we do not need to train the BRNN model as a classifier, but train the BRNN model with beat probabilities.

In the comparison experiment, we compare different Gaussian kernels with standard deviations (σ) ranging from 0 to 4 frames to find a proper kernel.

C. Network training

We train BRNN models on RWC pop, jazz, classic [15], SMC [16] and ballroom datasets [17], [18]. The usage of datasets is shown in Table II. For example, we use clips from 10 seconds to 50 seconds of 100 pieces in the RWC pop dataset, with data of a total length of 4000 seconds used in

ISBN 978-90-827970-1-5 © EURASIP 2018



(a) s_0 ($\sigma = 0$) (b) s_1 ($\sigma = 1$) (c) s_2 ($\sigma = 2$)

Fig. 2: Gaussian kernels with different standard deviations (σ).

TABLE II: The usage of the datasets.

No.	dataset	files	range	length	batch size
1	RWC pop [15]	100	10-50s	4000s	128
2	RWC jazz [15]	50	10-50s	2000s	64
3	RWC classic [15]	50	10-50s	2000s	64
4	SMC [16]	200	0-40s	8000s	256
5	Ballroom [17], [18]	250	0-30s	7500s	256

the experiment. The batch size is set in proportion to the total length of data in each dataset, so that weights are updated for similar times in each epoch for all datasets. For example, after arranging the input data, there are 640 and 320 training samples for the RWC pop and jazz datasets, respectively. With the batch sizes of 128 and 64, weights are updated for 5 times in an epoch for both datasets. We train models on individual datasets with 80% data for training and 20% for validation, based on a random splitting of the data.

We apply a simple truncation strategy to feed data into the BRNNs. The method reshapes the input data into short sequences (key sequences), then attaches some previous frames and following frames to the key sequences (zero-padding if no frames before or after). We show the beat interval histogram of pieces in the datasets in Figure 3. The largest interval is less than 2 seconds. We assume that each sample should include several beats in order to learn the context relations in the BRNN. So we set the length of the key sequence to 500 frames (5 seconds including at least 2 beats), and the numbers of previous frames and following frames to 250. Then each sample is a sequence of 1000 frames.

We minimise the cross entropy loss using an optimisation method of Adam [19] with a learning rate of 10^{-4} (other parameters with default values in tensorflow). We stop training when the validation loss does not decrease for 20 epochs.

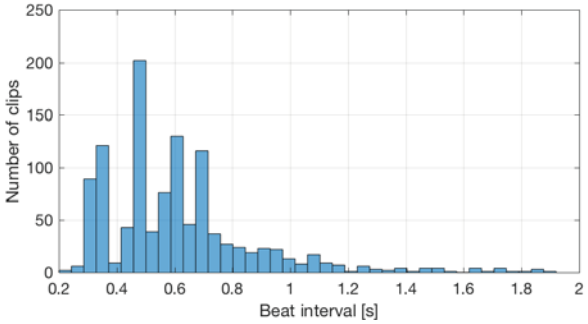


Fig. 3: The histogram of beat intervals in the datasets.

TABLE III: Comparison results on validation sets. '*Epochs*' denotes the number of epochs when the validation loss is smallest. '*F*' is the average F-measure obtained by the trained models on the validation sets.

	RWC pop		RWC jazz		RWC classic		SMC		Ballroom	
	Epochs	*F*	*Epochs*	*F*	*Epochs*	*F*	*Epochs*	*F*	*Epochs*	*F*
s0	538	81	526	**73**	382	54	470	45	808	**91**
s1	242	**82**	466	**73**	182	**55**	368	**47**	410	89
s2	200	79	222	63	156	51	204	45	180	**91**
s3	126	80	148	65	130	51	170	43	176	89
s4	124	78	118	63	140	50	174	43	178	89

D. Network testing

When producing beat activations on the testing data, we unroll the BRNN model for the whole length of the input sequence (30s or 40s), and feed in the whole sequence at a time. No truncation of the data is needed for this step.

IV. Comparison results

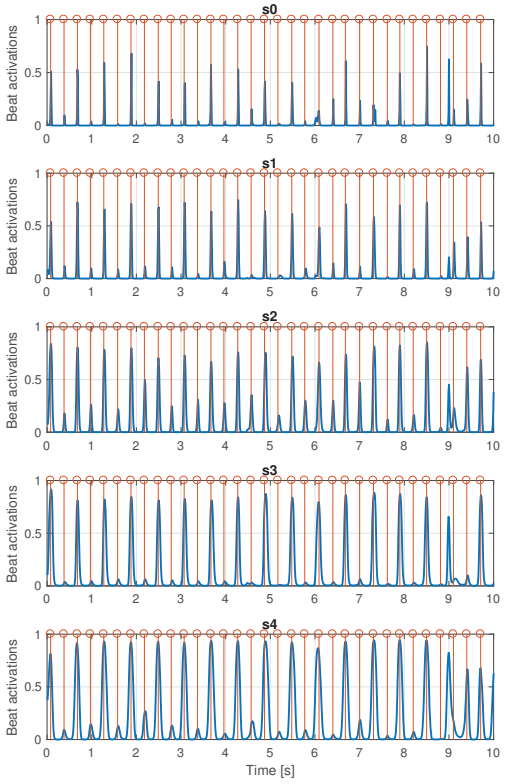
For each dataset, we compare beat tracking performance (in F-measures) of 5 models trained with 5 different labels on the validation set. The detailed results are shown in Table III. We find that using convolved labels (s_1) provides better results than using the original ground truth labels (s_0) on three datasets: 1 percentage point improvement for the RWC pop and classic datasets and 2 percentage point improvement for the SMC dataset. For the other two datasets, the results are the same for using the original labels and using the proper convolved labels (with the s_1 model for RWC jazz and the s_2 model for ballroom). With convolved labels, the best performance is achieved by the s_1 model on all datasets except the ballroom dataset. For the ballroom dataset, the s_2 model produces the best beat tracking result. There is no improvement in performance by further increasing the standard deviation of the Gaussian kernel to 3 and 4 frames.

The results on the validation sets can be used as an indication of the difficulty of the datasets. We show two examples of the outputs of the BRNN models of the easiest (ballroom) dataset and of the most difficult (SMC) dataset in Figure 4. We find that with a larger standard deviation, the output beat activations are larger, and the peaks are wider and more similar to the Gaussian kernel used for generating labels for training. The beat activations of the ballroom dataset are relatively larger than those of the SMC dataset. As shown in Figure 4a, we find the beat activations of the s_2 model are more related to the ground truth; beat activations of other models are more related to half-tempo beats. As shown in Figure 4b, from 5 to 10 second, the beat activations of the s_0, s_1 and s_5 models indicate the ground truth beats. However, from 0 to 5 second, the beat activations of the s_0 model are small and those of the other four models are noisy.

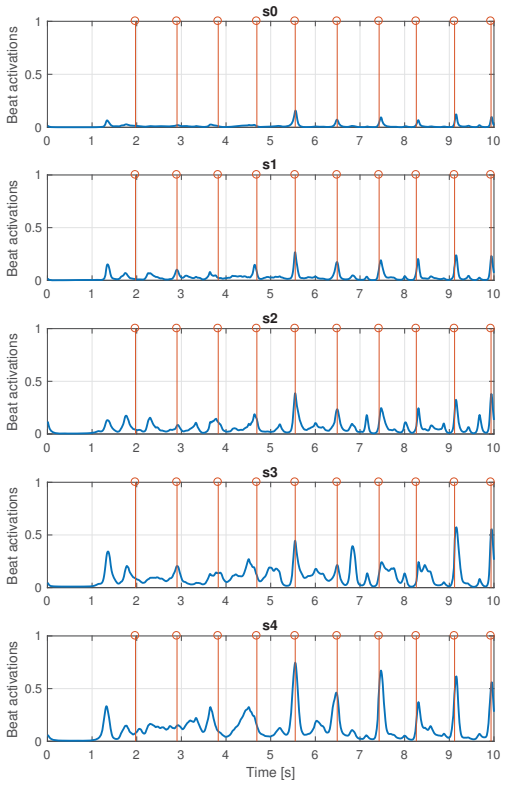
For standard deviations from 0 to 3 frames, the training time is shorter with a larger standard deviation, as indicated by the numbers of epochs in Table III.[2] We find that the decreasing

[2]An epoch takes the same amount of time on the same dataset.

ISBN 978-90-827970-1-5 © EURASIP 2018



(a) Beat activations of a clip from the ballroom dataset



(b) Beat activations of a clip from the SMC dataset

Fig. 4: Beat activations of two 10s clips—(a) and (b)—from the validation sets obtained by models trained with different labels ($s_i, i \in [0, 4]$). Stems indicate the ground truth beats.

rate of the training time with the increasing standard deviation is generally related to the average tempo of the dataset. For example, with the s_2 model, the number of epochs for training decreases from 808 to 180 for the ballroom dataset (with faster tempi), and decreases from 470 to 204 for the SMC dataset (with slower tempi). One possible reason is when convolving with a Gaussian kernel, the ratio of non-zero frames increases more quickly for pieces with faster tempi, hence a sharper decrease on the training time.

The comparison experiment shows that we can learn a better or competitive model in a shorter time with the convolved labels. Because there is a post-processing step to track the beats, the difference of the average performance of different models are small. The individual examples in Figure 4 indicate that better beat activations can be obtained by using a proper Gaussian kernel.

V. Discussions

Due to the overlapped frames used for computing the input spectrogram, the spectra of frames around beats are similar to each other. With the original labels, only beat frames are labelled 1, while the adjacent frames are labelled 0. Then some similar inputs will have different labels, which makes it less efficient to train the model. Then with the proper convolved labels, similar inputs have similar labels, making the model training more efficient (with better or competitive performance in a shorter period of time). There could be arguments that deep learning should be able to learn the inherent structure of the data of the above situation. We think the experiment shows that with the same network architecture, it is more efficient to train a network with more reasonable labels. In addition, as mentioned in the introduction the original labels of beat tracking are imbalanced. When using the convolved labels, there are more non-zero frames, resulting in less biased data, which also contributes to a more efficient training.

The results show that the convolved labels can help a more efficient training in a training-validation setting on five datasets. We need to fine-tune the BRNN models for testing unknown music pieces. We train the models with sequences of 10 seconds. A preliminary experiment shows that feeding the BRNN with shorter sequences can obtain better results in some occasions. To find a proper sequence length for beat tracking is worth exploring in the future.

VI. Conclusions

In this paper, we propose to use smooth labels for training BRNN-based beat tracking models by convolving the ground truth labels with a Gaussian kernel. We conduct a comparison experiment with different Gaussian kernels on five individual datasets. The results show that a more efficient training can be obtained by using the labels convolved with a proper Gaussian kernel.

We make use of existing techniques for building a BRNN-based beat tracking model and also provide details for the training and testing processes. We think that these details can help conduct a re-production of the work.

ISBN 978-90-827970-1-5 © EURASIP 2018

ACKNOWLEDGEMENT

This work was supported in part by JST ACCEL Grant Number JPMJAC1602, Japan.

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 (ACM Multimedia)*, 1994, pp. 365–372.

[2] S. Böck and M. Schedl, "Enhanced Beat Tracking with Context-Aware Neural Networks," in *Proceedings of the 14th International Conference on Digital Audio Effects (DAFx)*, 2011.

[3] S. Böck, F. Krebs, and G. Widmer, "A Multi-Model Approach to Beat Tracking Considering Heterogeneous Music Styles," in *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, 2014.

[4] ——, "Accurate Tempo Estimation Based on Recurrent Neural Networks and Resonating Comb Filters," in *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, 2015, pp. 625–631.

[5] F. Krebs, S. Böck, M. Dorfer, and G. Widmer, "Downbeat Tracking Using Beat-Synchronous Features and Recurrent Neural Networks," in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, 2016.

[6] 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.

[7] S. Durand and S. Essid, "Downbeat Detection with Conditional Random Fields and Deep Learned Features," in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, 2016.

[8] S. Durand, J. Pablo Bello, B. David, and G. Richard, "Robust Downbeat Tracking Using an Ensemble of Convolutional Networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 1, pp. 76–89, 2017.

[9] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks," in *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, 2006, pp. 369–376.

[10] M. Jenckel, S. S. Parkala, S. S. Bukhari, and A. Dengel, "Impact of Training LSTM-RNN with Fuzzy Ground Truth," in *Proceedings of the 9th International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, 2018.

[11] T. Grill, J. Schlüter, and K. Ullrich, "Boundary Detection in Music Structure Analysis Using Convolutional Neural Networks," in *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, 2014, pp. 417–422.

[12] T. O'Brien, "Musical Structure Segmentation with Convolitional Neural Networks," in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, 2016.

[13] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and G. Widmer, "madmom: a New Python Audio and Music Signal Processing Library," in *Proceedings of the 24th ACM International Conference on Multimedia (ICM)*, 2016, pp. 1174–1178.

[15] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC Music Database: Popular, Classical, and Jazz Music Databases," in *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR)*, 2002, pp. 287–288.

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

[17] 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.

[18] 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.

[19] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *Proceedings of the 3rd International Conference for Learning Representations (ICLR)*, 2015.

ISBN 978-90-827970-1-5 © EURASIP 2018