PROBABILISTIC EXTRACTION OF BEAT POSITIONS FROM A BEAT ACTIVATION FUNCTION

Filip Korzeniowski, Sebastian Böck, and Gerhard Widmer
Department of Computational Perception
Johannes Kepler University, Linz, Austria
filip.korzeniowski@jku.at

ABSTRACT

We present a probabilistic way to extract beat positions from the output (activations) of the neural network that is at the heart of an existing beat tracker. The method can serve as a replacement for the greedy search the beat tracker currently uses for this purpose. Our experiments show improvement upon the current method for a variety of data sets and quality measures, as well as better results compared to other state-of-the-art algorithms.

1. INTRODUCTION

Rhythm and pulse lay the foundation of the vast majority of musical works. Percussive instruments like rattles, stampers and slit drums have been used for thousands of years to accompany and enhance rhythmic movements or dances. Maybe this deep connection between movement and sound enables humans to easily tap to the pulse of a musical piece, accenting its beats. The computer, however, has difficulties determining the position of the beats in an audio stream, lacking the intuition humans developed over thousands of years.

Beat tracking is the task of locating beats within an audio stream of music. Literature on beat tracking suggests many possible applications: practical ones such as automatic time-stretching or correction of recorded audio, but also as a support for further music analysis like segmentation or pattern discovery [4]. Several musical aspects hinder tracking beats reliably: syncopation, triplets and offbeat rhythms create rhythmical ambiguousness that is difficult to resolve; varying tempo increases musical expressivity, but impedes finding the correct beat times. The multitude of existing beat tracking algorithms work reasonably well for a subset of musical works, but often fail for pieces that are difficult to handle, as [11] showed.

In this paper, we further improve upon the beat tracker presented in [2]. The existing algorithm uses a neural network to detect beats in the audio. The output of this neural network, called activations, indicates the likelihood of a beat at each audio position. A post-processing step selects from these activations positions to be reported as beats. However, this method struggles to find the correct beats when confronted with ambiguous activations.

We contribute a new, probabilistic method for this purpose. Although we designed the method for audio with a steady pulse, we show that using the proposed method the beat tracker achieves better results even for datasets containing music with varying tempo.

The remainder of the paper is organised as follows: Section 2 reviews the beat tracker our method is based on. In Section 3 we present our approach, describe the structure of our model and show how we infer beat positions. Section 4 describes the setup of our experiments, while we show their results in Section 5. Finally, we conclude our work in Section 6.

© Filip Korzeniowski, Sebastian Böck, Gerhard Widmer. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Filip Korzeniowski, Sebastian Böck, Gerhard Widmer. "Probabilistic Extraction of Beat Positions from a Beat Activation Function", 15th International Society for Music Information Retrieval Conference, 2014.

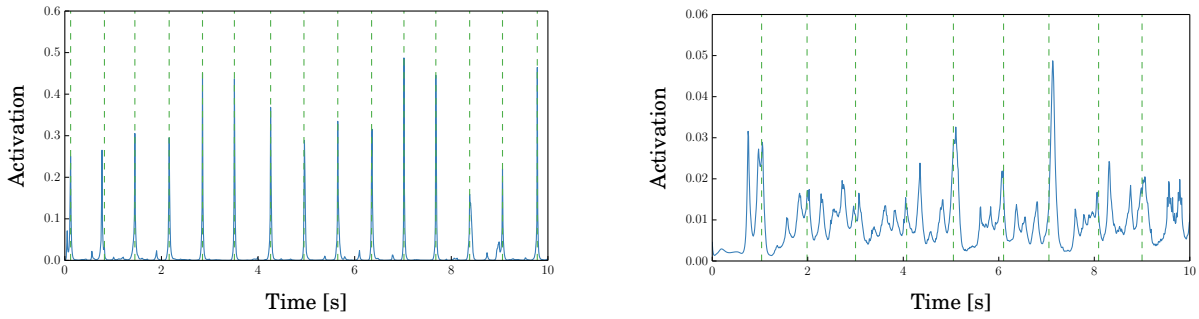
2. BASE METHOD

In this section, we will briefly review the approach presented in [2]. For a detailed discourse we refer the reader to the respective publication. First, we will outline how the algorithm processes the signal to emphasise onsets. We will then focus on the neural network used in the beat tracker and its output in Section 2.2. After this, Section 3 will introduce the probabilistic method we propose to find beats in the output activations of the neural network.

2.1 Signal Processing

The algorithm derives from the signal three logarithmically filtered power spectrograms with window sizes W of 1024, 2048 and 4096 samples each. The windows are placed 441 samples apart, which results in a frame rate of $f_r = 100$ frames per second for audio sampled at 44.1kHz. We transform the spectra using a logarithmic function to better match the human perception of loudness, and filter them using 3 overlapping triangular filters per octave.

Additionally, we compute the first order difference for each of the spectra in order to emphasise onsets. Since longer frame windows tend to smear spectral magnitude values in time, we compute the difference to the last, second to last, and third to last frame, depending on the window size W. Finally, we discard all negative values.



(a) Activations of a piece from the Ballroom dataset

(b) Activations of a piece from the SMC dataset

Figure 1. Activations of pieces from two different datasets. The activations are shown in blue, with green, dotted lines showing the ground truth beat annotations. On the left, distinct peaks indicate the presence of beats. The prominent rhythmical structure of ballroom music enables the neural network to easily discern frames that contain beats from those that do not. On the right, many peaks in the activations do not correspond to beats, while some beats lack distinguished peaks in the activations. In this piece, a single woodwind instrument is playing a solo melody. Its soft onsets and lack of percussive instruments make detecting beats difficult.

2.2 Neural Network

Our classifier consists of a bidirectional recurrent neural network of Long Short-Term Memory (LSTM) units, called *bidirectional Long Short-Term Memory* (BLSTM) recurrent neural network [10]. The input units are fed with the log-filtered power spectra and their corresponding positive first order differences. We use three fully connected hidden layers of 25 LSTM units each. The output layer consists of a single sigmoid neuron. Its value remains within $[0, 1]$, with higher values indicating the presence of a beat at the given frame.

After we initialise the network weights randomly, the training process adapts them using standard gradient descent with back propagation and early stopping. We obtain training data using 8-fold cross validation, and randomly choose 15% of the training data to create a validation set. If the learning process does not improve classification on this validation set for 20 training epochs, we stop it and choose the best performing neural network as final model. For more details on the network and the learning process, we refer the reader to [2].

The neural network's output layer yields activations for every feature frame of an audio signal. We will formally represent this computation as mathematical function. Let N be the number of feature frames for a piece, and $\mathbb{N}_{\leq N} = \{1, 2, \ldots, N\}$ the set of all frame indices. Furthermore, let υ_n be the feature vector (the log-filtered power spectra and corresponding differences) of the n^{th} audio frame, and $\Upsilon = (\upsilon_1, \upsilon_2, \ldots, \upsilon_N)$ denote all feature vectors computed for a piece. We represent the neural network as a function

$$\Psi : \mathbb{N}_{\leq N} \to [0, 1]\,, \tag{1}$$

such that $\Psi(n; \Upsilon)$ is the activation value for the n^{th} frame when the network processes the feature vectors Υ. We will call this function *"activations"* in the following.

Depending on the type of music the audio contains, the activations show clear (or, less clear) peaks at beat positions. Figure 1 depicts the first 10 seconds of activations for two different songs, together with ground truth beat annotations. In Fig. 1a, the peaks in the activations clearly correspond to beats. For such simple cases, thresholding should suffice to extract beat positions. However, we often have to deal with activations as those in Fig. 1b, with many spurious and/or missing peaks. In the following section, we will propose a new method for extracting beat positions from such activations.

3. PROBABILISTIC EXTRACTION OF BEAT POSITIONS

Figure 1b shows the difficulty in deriving the position of beats from the output of the neural network. A greedy local search, as used in the original system, runs into problems when facing ambiguous activations. It struggles to correct previous beat position estimates even if the ambiguity resolves later in the piece. We therefore tackle this problem using a probabilistic model that allows us to globally optimise the beat sequence.

Probabilistic models are a frequently used to process time-series data, and are therefore popular in beat tracking (e.g. [3, 9, 12, 13, 14]). Most systems favour generative time-series models like hidden Markov models (HMMs), Kalman filters, or particle filters as natural choices for this problem. For a more complete overview of available beat trackers using various methodologies and their results on a challenging dataset we refer the reader to [11].

In this paper, we use a different approach: our model represents each beat with its own random variable. We model time as dimension in the sample space of our random variables as opposed to a concept of time driving a random process in discrete steps. Therefore, all activations are available at any time, instead of one at a time when thinking of time-series data.

For each musical piece we create a model that differs from those of other pieces. Different pieces have different lengths, so the random variables are defined over different sample spaces. Each piece contains a different number of beats, which is why each model consists of a different

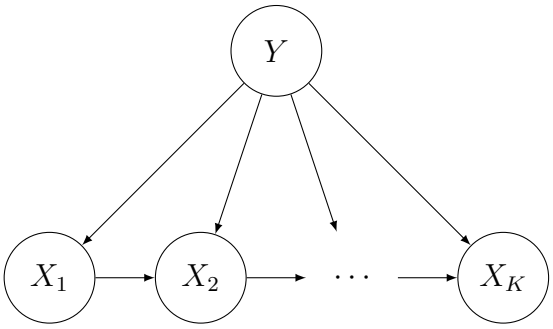


Figure 2. The model depicted as Bayesian network. Each X_k corresponds to a beat and models its position. Y represents the feature vectors of a signal.

number of random variables.

The idea to model beat positions directly as random variables is similar to the HMM-based method presented in [14]. However, we formulate our model as a Bayesian network with the observations as topmost node. This allows us to directly utilise the whole observation sequence for each beat variable, without potentially violating assumptions that need to hold for HMMs (especially those regarding the observation sequence). Also, our model uses only a single factor to determine potential beat positions in the audio – the output of a neural network – whereas [14] utilises multiple features on different levels to detect beats and downbeats.

3.1 Model Structure

As mentioned earlier, we create individual models for each piece, following the common structure described in this section. Figure 2 gives an overview of our system, depicted as Bayesian network.

Each X_k is a random variable modelling the position of the k^{th} beat. Its domain are all positions within the length of a piece. By position we mean the frame index of the activation function – since we extract features with a frame rate of $f_r = 100$ frames per second, we discretise the continuous time space to 100 positions per second.

Formally, the number of possible positions per piece is determined by N, the number of frames. Each X_k is then defined as random variable with domain $\mathbb{N}_{\leq N}$, the natural numbers smaller or equal to N:

$$X_k \in \mathbb{N}_{\leq N} \quad \text{with } 1 \leq k \leq K, \tag{2}$$

where K is the number of beats in the piece. We estimate this quantity by detecting the dominant interval τ of a piece using an autocorrelation-based method on the smoothed activation function of the neural network (see [2] for details). Here, we restrict the possible intervals to a range $[\tau_l..\tau_u]$, with both bounds learned from data. Assuming a steady tempo and a continuous beat throughout the piece, we simply compute $K = {}^{N}/_{\tau}$.

Y models the features extracted from the input audio. If we divide the signal into N frames, Y is a sequence of vectors:

$$Y \in \{(y_1, \ldots, y_N)\}, \tag{3}$$

where each y_n is in the domain defined by the input features. Although Y is formally a random variable with a distribution $P(Y)$, its value is always given by the concrete features extracted from the audio.

The model's structure requires us to define dependencies between the variables as conditional probabilities. Assuming these dependencies are the same for each beat but the first, we need to define

$$\begin{aligned} &\mathrm{P}\left(X_1 \mid Y\right) \quad \text{and} \\ &\mathrm{P}\left(X_k \mid X_{k-1}, Y\right). \end{aligned}$$

If we wanted to compute the joint probability of the model, we would also need to define $P(Y)$ – an impossible task. Since, as we will elaborate later, we are only interested in $P(X_{1:K} \mid Y)$ [1], and Y is always given, we can leave this aside.

3.2 Probability Functions

Except for X_1, two random variables influence each X_k: the previous beat X_{k-1} and the features Y. Intuitively, the former specifies the spacing between beats and thus the rough position of the beat compared to the previous one. The latter indicates to what extent the features confirm the presence of a beat at this position. We will define both as individual factors that together determine the conditional probabilities.

3.2.1 Beat Spacing

The pulse of a musical piece spaces its beats evenly in time. Here, we assume a steady pulse throughout the piece and model the relationship between beats as factor favouring their regular placement according to this pulse. Future work will relax this assumption and allow for varying pulses.

Even when governed by a steady pulse, the position of beats is far from rigid: slight modulations in tempo add musical expressivity and are mostly artistic elements intended by performers. We therefore allow a certain deviation from the pulse. As [3] suggests, tempo changes are perceived relatively rather than absolutely, i.e. halving the tempo should be equally probable as doubling it. Hence, we use the logarithm to base 2 to define the intermediate factor $\tilde{\Phi}$ and factor Φ, our beat spacing model. Let x and x' be consecutive beat positions and $x > x'$, we define

$$\tilde{\Phi}\left(x, x'\right) = \phi\left(\log_2\left(x - x'\right); \log_2\left(\tau\right), \sigma_\tau^2\right), \tag{4}$$

$$\Phi\left(x, x'\right) = \begin{cases} \tilde{\Phi}\left(x, x'\right) & \text{if } 0 < x - x' < 2\tau \\ 0 & \text{else} \end{cases}, \tag{5}$$

where $\phi\left(x; \mu, \sigma^2\right)$ is the probability density function of a Gaussian distribution with mean μ and variance σ^2, τ is the dominant inter-beat interval of the piece, and σ_τ^2 represents the allowed tempo variance. Note how we restrict the non-zero range of Φ: on one hand, to prevent computing the logarithm of negative values, and on the other hand, to reduce the number of computations.

[1] We use $X_{m:n}$ to denote all X_k with indices m to n

The factor yields high values when x and x' are spaced approximately τ apart. It thus favours beat positions that correspond to the detected dominant interval, allowing for minor variations.

Having defined the beat spacing factor, we will now elaborate on the activation vector that connects the model to the audio signal.

3.2.2 Beat Activations

The neural network's activations Ψ indicate how likely [2] each frame $n \in N_{\leq N}$ is a beat position. We directly use this factor in the definition of the conditional probability distributions.

With both factors in place we can continue to define the conditional probability distributions that complete our probabilistic model.

3.2.3 Conditional Probabilities

The conditional probability distribution $P(X_k \mid X_{k-1}, Y)$ combines both factors presented in the previous sections. It follows the intuition we outlined at the beginning of Section 3.2 and molds it into the formal framework as

$$\mathrm{P}(X_k \mid X_{k-1}, Y) = \frac{\Psi(X_k; Y) \cdot \Phi(X_k, X_{k-1})}{\sum_{X_k} \Psi(X_k; Y) \cdot \Phi(X_k, X_{k-1})}. \tag{6}$$

The case of X_1, the first beat, is slightly different. There is no previous beat to determine its rough position using the beat spacing factor. But, since we assume that there is a steady and continuous pulse throughout the audio, we can conclude that its position lies within the first interval from the beginning of the audio. This corresponds to a uniform distribution in the range $[0, \tau]$, which we define as beat position factor for the first beat as

$$\Phi_1(x) = \begin{cases} 1/\tau & \text{if } 0 \leq x < \tau, \\ 0 & \text{else} \end{cases}. \tag{7}$$

The conditional probability for X_1 is then

$$\mathrm{P}(X_1 \mid Y) = \frac{\Psi(X_1; Y) \cdot \Phi_1(X_1)}{\sum_{X_1} \Psi(X_1; Y) \cdot \Phi_1(X_1)}. \tag{8}$$

The conditional probability functions fully define our probabilistic model. In the following section, we show how we can use this model to infer the position of beats present in a piece of music.

3.3 Inference

We want to infer values $x^*_{1:K}$ for $X_{1:K}$ that maximise the probability of the beat sequence given $Y = \Upsilon$, that is

$$x^*_{1:K} = \underset{x_{1:K}}{\operatorname{argmax}} P(X_{1:K} \mid \Upsilon). \tag{9}$$

Each x^*_k corresponds to the position of the k^{th} beat. Υ are the feature vectors computed for a specific piece. We use a dynamic programming method similar to the well known Viterbi algorithm [15] to obtain the values of interest.

We adapt the standard Viterbi algorithm to fit the structure of model by changing the definition of the "Viterbi variables" δ to

$$\delta_1(x) = \mathrm{P}(X_1 = x \mid \Upsilon) \quad \text{and}$$
$$\delta_k(x) = \max_{x'} \mathrm{P}(X_k = x \mid X_{k-1} = x', \Upsilon) \cdot \delta_{k-1}(x'),$$

where $x, x' \in \mathbb{N}_{\leq N}$. The backtracking pointers are set accordingly.

$P(x^*_{1:K} \mid \Upsilon)$ gives us the probability of the beat sequence given the data. We use this to determine how well the deducted beat structure fits the features and in consequence the activations. However, we cannot directly compare the probabilities of beat sequences with different numbers of beats: the more random variables a model has, the smaller the probability of a *particular* value configuration, since there are more *possible* configurations. We thus normalise the probability by dividing by K, the number of beats.

With this in mind, we try different values for the dominant interval τ to obtain multiple beat sequences, and choose the one with the highest normalised probability. Specifically, we run our method with multiples of τ ($1/2$, $2/3$, 1, $3/2$, 2) to compensate for errors when detecting the dominant interval.

4. EXPERIMENTS

In this section we will describe the setup of our experiments: which data we trained and tested the system on, and which evaluation metrics we chose to quantify how well our beat tracker performs.

4.1 Data

We ensure the comparability of our method by using three freely available data sets for beat tracking: the *Ballroom* dataset [8,13]; the *Hainsworth* dataset [9]; the *SMC* dataset [11]. The order of this listing indicates the difficulty associated with each of the datasets. The Ballroom dataset consists of dance music with strong and steady rhythmic patterns. The Hainsworth dataset includes of a variety of musical genres, some considered easier to track (like pop/rock, dance), others more difficult (classical, jazz). The pieces in the SMC dataset were specifically selected to challenge existing beat tracking algorithms.

We evaluate our beat tracker using 8-fold cross validation, and balance the splits according to dataset. This means that each split consists of roughly the same relative number of pieces from each dataset. This way we ensure that all training and test splits represent the same distribution of data.

All training and testing phases use the same splits. The same training sets are used to learn the neural network and to set parameters of the probabilistic model (lower and upper bounds τ_l and τ_u for dominant interval estimation and σ_τ). The test phase feeds the resulting tracker with data from the corresponding test split. After detecting the beats

[2] technically, it is not a likelihood in the probabilistic sense – it just yields higher values if the network thinks that the frame contains a beat than if not

for all pieces, we group the results according to the original datasets in order to present comparable results.

4.2 Evaluation Metrics

A multitude of evaluation metrics exist for beat tracking algorithms. Some accent different aspects of a beat tracker's performance, some capture similar properties. For a comprehensive review and a detailed elaboration on each of the metrics, we refer the reader to [5]. Here, we restrict ourselves to the following four quantities, but will publish further results on our website [3].

F-measure The standard measure often used in information retrieval tasks. Beats count as correct if detected within ±70ms of the annotation.

Cemgil Measure that uses a Gaussian error window with $\sigma = 40$ms instead of a binary decision based on a tolerance window. It also incorporates false positives and false negatives.

CMLt The percentage of correctly detected beats at the correct metrical level. The tolerance window is set to 17.5% of the current inter-beat interval.

AMLt Similar to CMLt, but allows for different metrical levels like double tempo, half tempo, and off-beat.

In contrast to common practice [4], we do not skip the first 5 seconds of each audio signal for evaluation. Although skipping might make sense for on-line algorithms, it does not for off-line beat trackers.

5. RESULTS

Table 1 shows the results of our experiments. We obtained the raw beat detections on the Ballroom dataset for [6, 12, 13] from the authors of [13] and evaluated them using our framework. The results are thus directly comparable to those of our method. For the Hainsworth dataset, we collected results for [6, 7, 12] from [7], who does skip the first 5 seconds of each piece in the evaluation. In our experience, this increases the numbers obtained for each metric by about 0.01.

The approaches of [6, 7] do not require any training. In [12], some parameters are set up based on a separate dataset consisting of pieces from a variety of genres. [13] is a system that is specialised for and thus only trained on the Ballroom dataset.

We did not include results of other algorithms for the SMC dataset, although available in [11]. This dataset did not exist at the time most beat trackers were crafted, so the authors could not train or adapt their algorithms in order to cope with such difficult data.

Our method improves upon the original algorithm [1, 2] for each of the datasets and for all evaluation metrics. While F-Measure and Cemgil metric rises only marginally (except for the SMC dataset), CMLt and AMLt improves considerably. Our beat tracker also performs better than the other algorithms, where metrics were available.

SMC	**F**	**Cg**	**CMLt**	**AMLt**
Proposed	0.545	0.436	0.442	0.580
Böck [1, 2]	0.497	0.402	0.360	0.431

Hainsworth	**F**	**Cg**	**CMLt**	**AMLt**
Proposed	0.840	0.718	0.784	0.875
Böck [1, 2]	0.837	0.717	0.763	0.811
Degara* [7]	-	-	0.629	0.815
Klapuri* [12]	-	-	0.620	0.793
Davies* [6]	-	-	0.609	0.763

Ballroom	**F**	**Cg**	**CMLt**	**AMLt**
Proposed	0.903	0.864	0.833	0.910
Böck [1, 2]	0.889	0.857	0.796	0.831
Krebs [13]	0.855	0.772	0.786	0.865
Klapuri [12]	0.728	0.651	0.539	0.817
Davies [6]	0.764	0.696	0.574	0.864

Table 1. Beat tracking results for the three datasets. **F** stands for F-measure and **Cg** for the Cemgil metric. Results marked with a star skip the first five seconds of each piece and are thus better by about 0.01 for each metric, in our experience.

The proposed model assumes a stable tempo throughout a piece. This assumption holds for certain kinds of music (like most of pop, rock and dance), but does not for others (like jazz or classical). We estimated the variability of the tempo of a piece using the standard deviation of the local beat tempo. We computed the local beat tempo based on the inter-beat interval derived from the ground truth annotations. The results indicate that most pieces have a steady pulse: 90% show a standard deviation lower than 8.61 bpm. This, of course, depends on the dataset, with 97% of the ballroom pieces having a deviation below 8.61 bpm, 89% of the Hainsworth dataset but only 67.7% of the SMC data.

We expect our approach to yield inferior results for pieces with higher tempo variability than for those with a more constant pulse. To test this, we computed Pearson's correlation coefficient between tempo variability and AMLt value. The obtained value of ρ = -0.46 indicates that our expectation holds, although the relationship is not linear, as a detailed examination showed. Obviously, multiple other factors also influence the results. Note, however, that although the tempo of pieces from the SMC dataset varies most, it is this dataset where we observed the strongest improvement compared to the original approach.

Figure 3 compares the beat detections obtained with the proposed method to those computed by the original approach. It exemplifies the advantage of a globally optimised beat sequence compared to a greedy local search.

[3] http://www.cp.jku.at/people/korzeniowski/ismir2014

[4] As implemented in the MatLab toolbox for the evaluation of beat trackers presented in [5]

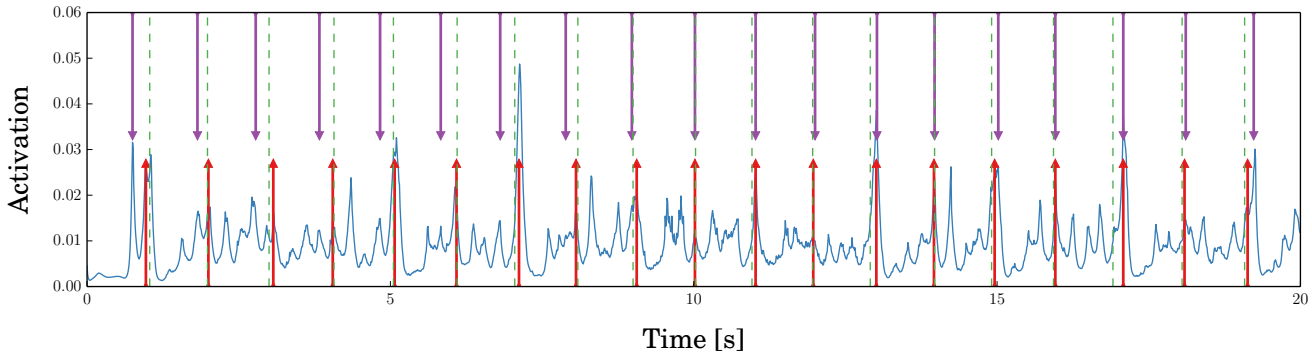


Figure 3. Beat detections for the same piece as shown in Fig. 1b obtained using the proposed method (red, up arrows) compared to those computed by the original approach (purple, down arrows). The activation function is plotted solid blue, ground truth annotations are represented by vertical dashed green lines. Note how the original method is not able to correctly align the first 10 seconds, although it does so for the remaining piece. Globally optimising the beat sequence via back-tracking allows us to infer the correct beat times, even if the peaks in the activation function are ambiguous at the beginning.

6. CONCLUSION AND FUTURE WORK

We proposed a probabilistic method to extract beat positions from the activations of a neural network trained for beat tracking. Our method improves upon the simple approach used in the original algorithm for this purpose, as our experiments showed.

In this work we assumed close to constant tempo throughout a piece of music. This assumption holds for most of the available data. Our method also performs reasonably well on difficult datasets containing tempo changes, such as the SMC dataset. Nevertheless we believe that extending the presented method in a way that enables tracking pieces with varying tempo will further improve the system's performance.

ACKNOWLEDGEMENTS

This work is supported by the European Union Seventh Framework Programme FP7 / 2007-2013 through the GiantSteps project (grant agreement no. 610591).

7. REFERENCES

[1] MIREX 2013 beat tracking results. `http://nema.lis.illinois.edu/nema_out/mirex2013/results/abt/`, 2013.

[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-11)*, Paris, France, Sept. 2011.

[3] A. T. Cemgil, B. Kappen, P. Desain, and H. Honing. On tempo tracking: Tempogram Representation and Kalman filtering. *Journal of New Music Research*, 28:4:259–273, 2001.

[4] T. Collins, S. Böck, F. Krebs, and G. Widmer. Bridging the Audio-Symbolic Gap: The Discovery of Repeated Note Content Directly from Polyphonic Music Audio. In *Proceedings of the Audio Engineering Society's 53rd Conference on Semantic Audio*, London, 2014.

[5] M. E. P. Davies, N. Degara, and M. D. Plumbley. Evaluation methods for musical audio beat tracking algorithms. *Queen Mary University of London, Centre for Digital Music, Tech. Rep. C4DM-TR-09-06*, 2009.

[6] M. E. P. Davies and M. D. Plumbley. Context-Dependent Beat Tracking of Musical Audio. *IEEE Transactions on Audio, Speech and Language Processing*, 15(3):1009–1020, Mar. 2007.

[7] N. Degara, E. A. Rua, A. Pena, S. Torres-Guijarro, M. E. P. Davies, and M. D. Plumbley. Reliability-Informed Beat Tracking of Musical Signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):290–301, Jan. 2012.

[8] 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*, 14(5):1832–1844, 2006.

[9] S. W. Hainsworth and M. D. Macleod. Particle Filtering Applied to Musical Tempo Tracking. *EURASIP Journal on Advances in Signal Processing*, 2004(15):2385–2395, Nov. 2004.

[10] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computing*, 9(8):1735–1780, Nov. 1997.

[11] A. Holzapfel, M. E. P. Davies, J. R. Zapata, J. a. L. Oliveira, and F. Gouyon. Selective Sampling for Beat Tracking Evaluation. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(9):2539–2548, Nov. 2012.

[12] A. P. Klapuri, A. J. Eronen, and J. T. Astola. Analysis of the meter of acoustic musical signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1):342–355, 2006.

[13] F. Krebs, S. Böck, and G. Widmer. Rhythmic Pattern Modeling for Beat and Downbeat Tracking in Musical Audio. In *Proc. of the 14th International Conference on Music Information Retrieval (ISMIR)*, 2013.

[14] G. Peeters and H. Papadopoulos. Simultaneous beat and downbeat-tracking using a probabilistic framework: theory and large-scale evaluation. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(6):1754–1769, 2011.

[15] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.