FAST AND ACCURATE: IMPROVING A SIMPLE BEAT TRACKER WITH A SELECTIVELY-APPLIED DEEP BEAT IDENTIFICATION

Akira Maezawa
Yamaha Corporation

ABSTRACT

In music applications, audio beat tracking is a central component that requires both speed and accuracy, but a fast beat tracker typically has many beat phase errors, while an accurate one typically requires more computation. This paper achieves a fast tracking speed and a low beat phase error by applying a slow but accurate beat phase detector at only the most informative spots in a given song, and interpolating the rest by a fast tatum-level tracker. We present (1) a framework for selecting a small subset of the tatum indices that information-theoretically best describes the beat phases of the song, (2) a fast HMM-based beat tracker for tatum tracking, and (3) an accurate but slow beat detector using a deep neural network (DNN). The evaluations demonstrate that the proposed DNN beat phase detection halves the beat phase error of the HMM-based tracker and enables a 98% decrease in the required number of DNN invocations without dropping the accuracy.

1. INTRODUCTION

Offline audio beat tracking, the task of identifying beats in a music audio signal, is now a critical component in music applications, having uses in digital audio workstations, synthesizers, music recommendation and many others. In beat tracking, it is important to both estimate a reasonable tempo (inversely proportional to the period between two beats), as well as the timings of beat occurrence, or the beat phase. In these applications, a beat tracker must be fast and accurate for common types of musical pieces such as popular music and electronic dance music. The capability to analyze one song in a few seconds is often desirable in end-user products, while being satisfactorily accurate.

There is a trade-off, however, between the speed and the accuracy of a beat tracker. On the one hand, a fast beat tracker tends to make mistakes due to some simplifying assumptions. On the other hand, an accurate beat tracker that employs a more elaborate model like deep neural networks (DNN) tends to require more computation, which is proportional to the song duration.

We observe two points in these extrema. First, many of the errors in a fast tracker are attributed to incorrect beat *phase* estimation, especially for current popular music. For example, a simple beat tracker often mistakenly tracks the off-beat. This means that a simple method is already quite capable of tracking the tatum, *i.e.,* some integer subdivision of the beat, but poorly identifies *which* of them are the beats.

Second, using an accurate but slow beat tracker to sweep through an entire song is often wasteful. For many pieces where tatum tracking is possible with a simple method, the primary role of an elaborate method is in beat *phase* identification. In many musical pieces, however, the meter is mostly stationary, so the beat phase identification needs to be done only sparingly. If the beats are identified at the most informative spots in the music for beat phase identification, the rest may be interpolated by exploiting the stationarity of the meter.

In this paper, we combine the best of both worlds – a fast tracking of beats with a moderate amount of beat phase errors, and an accurate identification of the beats through the use of elaborate methods. Our key idea, as shown in Figure 1, is to efficiently fix the beat phase estimation errors of a simple but fast beat tracker, by *sparingly* applying an accurate but slow beat phase identifier, only at the most informative spots in the song. To elaborate, we (1) detect the tatum reliably with a simple beat tracker, (2) select a small disjoint subset of the tatums that best describes the beat phases of the entire song, (3) apply an elaborate beat

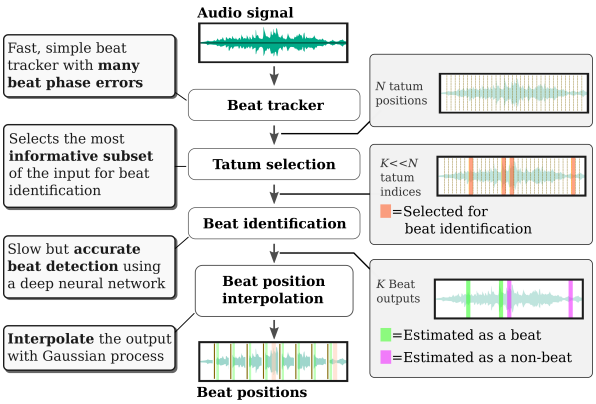

Figure 1. The overview of our method. Our system improves a simple beat tracker with little computational overhead by adding a slow but accurate beat phase estimator. Computation is reduced by only using a subset of the detected tatums for phase estimation, and interpolating the estimated phase output.

identifier only at the selected tatum subset, and (4) interpolate the beat identification for the remaining tatums. Such a framework is enabled by exploiting a strong tatum-level correlation of the beat phase: it allows us to select a small set of tatums that best describes, in the sense of mutual information, the rest of the beats, and to interpolate the rest.

Our contributions are (1) a low-overhead framework for improving an existing simple beat tracker by cascading a more elaborate beat phase detector, achieved by identifying the most informative tatums in a song for beat phase identification; (2) a beat phase identification method using a DNN that accurately identifies the beat phase of a tatum-sliced data; and (3) a simple and fast HMM-based beat detector that jointly decodes the BPM and beat phase.

2. RELATED WORK

2.1 Beat Tracking and Downbeat Estimation

Beat tracking is the task of identifying the beats in a music audio signal, a task that has been studied extensively. Earlier methods match hand-crafted onset features while assuming the evolution of the tempo, through soft rules [9] dynamic programming [8] or hidden Markov models [20], often with an explicit tempo induction step [5, 6]. One of the key design issues is the choice of the hand-crafted features, such as changes of harmony [9] or variants of spectral flux [8, 13, 18], and features indicating the salient beat interval [13].

Beat phase error is a common failure mode in beat trackers [4, 5, 18]. For example, it is often common for a beat tracker to track half a beat behind an acceptable beat position, or to track the off-beats (*e.g.*, tracking second and fourth beats in a $4/4$ time) when tracking at half the underlying tempo. The frequency of such a failure mode occurs suggests that tatum tracking is relatively easily done with a simple and fast method, but identifying the beat within the detected tatums is a more delicate problem.

To estimate the beat phase, and more generally downbeats, modeling of the rhythmic patterns [9, 16], or extracting the features indicating the spectral change at multiple temporal level [13] have been shown to be useful. More recently, significant improvements in downbeat estimation have been achieved through the use of DNN, which delegates the delicate task of designing the relevant features to machine learning. For example, convolutional neural networks [7] or recurrent neural networks [2, 15] have shown significant improvements, at a cost of more computation.

2.2 Sensor Placement and Submodular Optimization

The core idea of our paper is to find the "best" tatum positions to apply the computationally-heavy DNN output so that the most information may be extracted with each invocation of the DNN. In a related problem of acquiring data with costly sensors, the problem of determining the "best" way to place each sensor as to get the most information out is known as the *sensor placement problem* [14]. Sensor placement problem is often tackled by exploiting the spatial correlation. For example, if the spatial distribution of the temperature needs to be acquired, it is better to place the temperature sensors far apart with than near each other, since the sensor readings at two nearby points, as opposed to far-away points, are more strongly correlated and thus are less revealing.

The sensor placement problem can be formulated as to maximize the mutual information between the placed sensors and some other points of interest for which the sensors are not placed. While this problem is NP-hard, the *submodularity* of the mutual information may be exploited to arrive at a greedy near-optimal algorithm [19]. Submodularity amounts to concavity for sets, and means that a given function f over a set A satisfies $f(X \cup \{i\}) \geq f(Y \cup \{i\})$ for all $X \subseteq Y \subseteq A$ and $i \in A \setminus Y$.

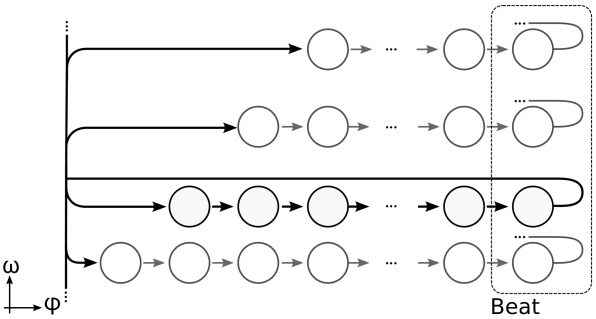

Figure 2. The state transition for the simple beat estimator. The beat phase counts down deterministically, and the next beat duration is chosen according to a beat period transition probability.

3. PROPOSED METHOD

Our method consists of (1) a simple tatum tracker that quickly tracks the tatum in an audio signal, (2) a slow but accurate beat identifier that identifies which of the tracked tatums are the beats, and (3) a tatum index selector, that selects a few tatum indices for applying the beat identifier, as to extract the most information regarding the presence of the beat.

3.1 Tracking Tatums with a Fast HMM Beat Tracker

We first track the tatum in a given audio signal. This is achieved by first using a simple beat tracker to extract the beat positions. Then, the tracked beat positions are subdivided equidistantly by a given factor Δn to obtain the tatums ($\Delta n = 4$ in this paper). Notice that while the beat detector may often track the wrong beat phase, it usually tracks the tatum properly.

To track the beat, we use an HMM-based beat detector, which uses onset and tempo features to jointly decode the beat position and the tempo. For the onset feature at frame t, we compute the first-order difference of the log-magnitude spectrum flux o_t, and for the BPM feature, we compute a comb filter-bank with the onset feature r_t, similar to [13].

We assume that the observed feature sequence is generated from an underlying sequence of discretized beat du-

ration (related to tempo) ω, and a "count-down" timer indicating the number of feature frames until the next beat. We assume that each normalized ϕ is associated to a unique onset feature observation likelihood $p(o|\phi)$, and each value of the beat duration ω is associated with a unique tempo feature observation likelihood $p(r|\omega)$. Based on these assumptions, the observation likelihood is given as follows:

$$p(o_t, r_t|\phi_t, \omega_t) = p(o_t|\phi_t)p(r_t|\omega_t). \quad (1)$$

We choose $p(o|\phi)$ to be a Normal distribution whose mean and the variance are selected based on the value of ϕ normalized by the beat duration. We choose three sets of the mean and the variance, based on whether the normalized beat phase is 0, 0.5 and others; the parameters are trained with maximum likelihood. Furthermore, $p(r|\omega)$ is chosen to be a von Mises-Fisher distribution [1] , whose parameters switches for each value of ω.

The time sequence of the beat duration ω_t and the count-down timer ϕ_t evolves such that (1) the ϕ_t decreases deterministically until reaching zero while ω_t remains constant and (2) when ϕ_t is zero, the beat duration ω_t switches to a new value according to a tempo transition matrix, and ϕ_{t+1} is set to ω_{t+1}. This amounts to the following generative process, also illustrated in Figure 2:

$$(\phi_t, \omega_t)|(\phi_{t-1}, \omega_{t-1}) \sim \begin{cases} \delta(\phi_t, \phi_{t-1} - 1)\delta(\omega_t, \omega_{t-1}) & \phi_{t-1} > 0 \\ R_{\omega_{t-1},\omega_t}\delta(\phi_t, \Phi_{\omega_t}) & \phi_{t-1} = 0 \end{cases}, \quad (2)$$

where Φ_ω is the number of audio frames corresponding to the beat duration for the beat duration ω, and R_{ω_1,ω_2} is a transition matrix that describes the probability of transitioning from beat duration ω_1 to ω_2. We reduce the search space by pruning negligible values of R and limiting the set of beat durations ω to consider, similar to [17]. The beat positions are decoded using the Viterbi algorithm to arrive at a set of N estimated *tatum* positions $\{\tau_n | n \in \mathcal{T} = \{1, 2 \dots N\}\}$.

This model is quite similar to the bar pointer model [24], except (1) we apply the bar pointer model only to decode the beats and not the underlying meter or rhythm and (2) we use both the tempo and the onset likelihoods to decode the beats and the beat durations. The inference is more efficient compared to the bar-pointer model because the search space is much smaller – *i.e.*, the state space is at the beat level instead of the bar level, the beat duration is discretized, and the allowed transition is pruned.

Despite the efficient processing, this method, like many beat trackers, suffers from beat phase estimation errors, occurring approximately once every ten songs, for example, for songs with strong syncopations. Thus, we consider using a more elaborate beat phase detection that is capable of directly modeling the kind of long-term characteristics required for beat phase identification.

[1] The likelihood is given by $p(x; \mu, \kappa) \propto \exp(\kappa \mu^T x)$ for some μ, x in $(D-1)$-sphere, D being the dimension of the comb-filter output, and κ is a scalar parameter.

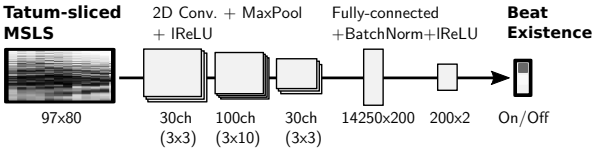

Figure 3. The architecture for predicting the Beat phase.

3.2 Identifying the Beats with Deep Neural Networks

To detect the presence of a beat at some tatum index, we use a DNN-based classifier of beats given tatum-sliced features. A DNN-based model is preferable because the notion of a beat depends on many factors like the rhythm and the harmony, and a manual feature design on such a problem is difficult.

To identify the beat position using a DNN, we use as the input the mel-scale log spectrogram (MSLS) that has been computed at each tatum. For each tatum, an 80-dimensional MSLS is extracted over a tatum window of 48 tatums before and after the current tatum, creating an input of $\mathbb{R}^{96\times80}$.

The network consists of three convolutional layers, each with a leaky ReLU activation followed by max-pooling. The number of channels and the kernel sizes, in increasing order of layers, are 30, 100, 30 and (3×3), (3×10) and (3×3), respectively. It is followed by dropout regularization [23] during learning and a fully-connected layer with 200-dimensional output with a batch-normalization layer [11] and a leaky ReLU activation. Finally, a fully connected layer with a softmax activation extracts the posterior probability of the beat presence. Thus, for some tatum index n, the DNN outputs $b_n \in \{0, 1\}$, which is 1 if tatum index n is a beat and 0 otherwise. Note that the model has no recurrent connections, allowing a random access to the tatum index.

Given a ground-truth annotation of the beat presence $\hat{b}_n$, we minimize the cross-entropy loss $L(\Theta)$:

$$L(\Theta) = \sum_n \hat{b}_n \log b_n(\Theta) + (1 - \hat{b}_n) \log(1 - b_n(\Theta)), \quad (3)$$

with n indexed over the training dataset. The optimization is done stochastically, using ADAM [12] with weight decay regularization. The mini-batch is shuffled randomly and we augment the data by pitch-shifting the input audio by -7 to +7 semitones, similar to [22].

This model is similar to the network in [7] in that we also use tatum-level features, but we (1) use the full MSLS instead of a band-passed input, allowing simultaneous extraction of both harmonic and rhythmic features that contribute to beats, and (2) use both convolutional and fully connected layers.

3.2.1 On the Choice of the Tatum Window Size

To justify the use of MSLS evaluated over a windows of 48 tatums before and after the current tatum, we have tested the accuracy of our beat identification method when changing the window radius. The accuracy on the validation data by changing the number of *beats* (assuming 4 tatums per

L [beats]	1	2	4	8	10	12
Accuracy[%]	83	87	90	93	93	93

Table 1. Validation accuracy of the beat estimator when changing the tatum window radius L, in beats.

beat) is shown in Table 1. Since the performance saturates at 10 beats, use of 48 tatums provides sufficient performance. Such a change in accuracy shows that a long-term analysis spanning a few measures is indeed required for properly identifying the beat.

3.3 Choosing Tatums for Beat Identification

Since the identification of the beats from a single frame is prone to errors, it is typical to identify the beat presence b_n for all N frames to arrive at the final beat position estimate. However, identifying the beat using all the detected tatums is computationally expensive. Therefore we consider using a small subset of the detected tatums with $K \ll N$ elements for identifying the beats at the selected tatums and interpolating the rest.

Let us formalize the problem. Let b_n be the beat phase estimation at some tatum index n. b_n can be evaluated by invoking an accurate but computationally expensive beat phase estimator as was discussed. Let $\mathcal{T} = \{1, 2, \cdots N\}$ be the set of tatum indices. The ultimate goal is to obtain an *estimate* of b_n for all n in $\mathcal{T}$ reliably while evaluating b_n only at a few spots. Thus, the goal is to find some small subset $\mathcal{D}_K \subset \mathcal{T}$ of K elements for which we do invoke the beat phase estimation algorithm. For the remaining tatums $\bar{\mathcal{D}}_K$, we interpolate the beat phase estimate by evaluating the expectation of $b_{\bar{\mathcal{D}}_K}$ given $b_{\mathcal{D}_K}$.

To both identify the small subset $\mathcal{D}_K$ and interpolate the beat phase output, we exploit the strong tatum-level correlation of the beat identification output. To illustrate, Figure 4 shows the auto-correlation r_i of the beat identification output b_n. Notice that a non-negligible correlation exists not only nearby but also far away, up to about 100 tatums. This means that the existence of a beat at some tatum index provides information about the beat existence of the neighboring tatums.

To exploit such a covariance we assume that b_n is a Gaussian process [21]. That is, for some disjoint ordered sets of indices $\mathcal{D}, \mathcal{U} \subseteq \mathcal{T}$, the joint pdf of $b_\mathcal{D} = \{b_i | i \in \mathcal{D}\}$ and $b_\mathcal{U} = \{b_i | i \in \mathcal{U}\}$ is expressed as follows:

$$\begin{pmatrix} b_\mathcal{D} \\ b_\mathcal{U} \end{pmatrix} \sim \mathcal{N}\left(\mu, \begin{pmatrix} \Sigma_{\mathcal{D}\mathcal{D}} & \Sigma_{\mathcal{D}\mathcal{U}} \\ \Sigma_{\mathcal{U}\mathcal{D}} & \Sigma_{\mathcal{U}\mathcal{U}} \end{pmatrix}\right). \tag{4}$$

Here, Σ_{AB} is the cross-covariance matrix $\in \mathbb{R}^{|A|\times|B|}$ between $\{b_i | i \in A\}$ and $\{b_i | i \in B\}$, such that element (i, j) of $\Sigma_{A,B}$ is the covariance between the ith element of A and the jth element of B. μ is the expectation of b_i that is computed from a training dataset.

3.3.1 Index Selection for Beat Phase Identification

Assuming the underlying Gaussian process, we seek to identify a set of tatum indices $\mathcal{D}_K \in \mathcal{T}$ subject to $|\mathcal{D}_K| = K$, as to maximize the mutual information between beat phase output $b_{\mathcal{D}_K}$ and the unobserved beat phase output $b_{\bar{\mathcal{D}}_K}$. This problem is NP-hard [14], but thanks to the submodularity of mutual information, a near-optimal greedy algorithm exists [19]. To solve the problem with a greedy near-optimal algorithm, we iteratively add a new index i to the set of indices $\mathcal{D}_k$ that maximizes the increase in mutual information, *i.e.*, $\mathcal{D}_k = \mathcal{D}_{k-1} \cup \{i\}$ where:

$$i = \arg \max_{i' \in \bar{\mathcal{D}}_{k-1}} \mathrm{MI}(\mathcal{D}_{k-1} \cup \{i'\}) - \mathrm{MI}(\mathcal{D}_{k-1}), \tag{5}$$

where $\mathrm{MI}(\mathcal{D})$ denotes the mutual information between $\mathcal{D}$ and $\bar{\mathcal{D}}$. This can be seen as a special case of the sensor placement problem based on mutual information maximization, where we ignore any points for which the sensor may not be placed.

Equation 5 for a Gaussian process amounts to setting $i = \arg\max_{i' \in \bar{\mathcal{D}}_k} \delta_{k-1,i'}$ at step k, with the following $\delta_{i,k}$:

$$\delta_{k,i} = \frac{\Sigma_{i,i} - \Sigma_{\{i\},\mathcal{D}_k} \Sigma^{-1}_{\mathcal{D}_k,\mathcal{D}_k} \Sigma_{\mathcal{D}_k,\{i\}}}{\Sigma_{i,i} - \Sigma_{\{i\},\mathcal{C}_{ki}} \Sigma^{-1}_{\mathcal{C}_{ki},\mathcal{C}_{ki}} \Sigma_{\mathcal{C}_{ki},\{i\}}}, \tag{6}$$

where $\mathcal{C}_{ki} = \overline{\mathcal{D}_k \cup \{i\}}$. Here, the numerator amounts to the conditional variance of b_i given $b_{\mathcal{D}_k}$ and the denominator amounts to the conditional variance of b_i given the remaining elements. Intuitively, therefore, this objective seeks to find an index i that is unpredictable based on the already-observed data $b_{\mathcal{D}_k}$, while being representative of the non-selected indices, *i.e.*, easily predictable from seeing the data of the non-selected indices.

It is important to notice that the computation of Equation 6 is independent of the actual values of b_n. Thus, $\mathcal{D}_K$ can be evaluated *without invoking the computationally expensive DNN beat identifier*. Furthermore, given the auto-correlation computed beforehand, the set $\mathcal{D}_K$ depends only on the number of tatums in a given song and not the actual observations. Thus, $\mathcal{D}_K$ may be pre-computed for all practical values of the number of tatums, incurring zero runtime overhead.

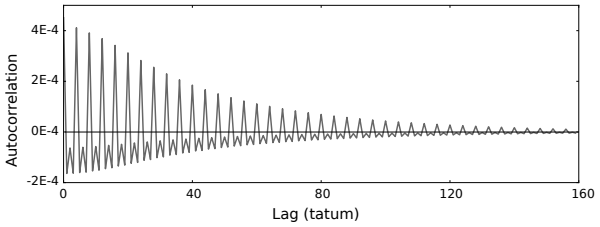

Figure 4. The auto-correlation of the estimated beat phase.

3.3.2 Analysis of the Selected Indices

To see how the indices are chosen with our method, Figure 5 shows the index chosen as the index selection algorithm proceeds. Notice how in the initial stage ($K = 2$ and 4; red and yellow boxes), the algorithm selects the middle and the edges of the song while selecting pairs of indices that are 2 tatums apart, congruent mod 4. This shows that

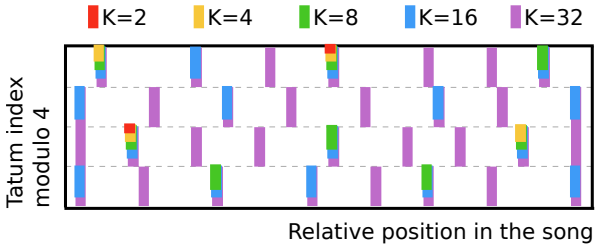

Figure 5. The indices chosen by the the mutual information maximization criterion as the algorithm progresses with $K = 2, 4$, up to 32 (best viewed in color).

the method tries to disambiguate the beat versus the off-beat, while staking out the entire song. As the algorithm progresses it selects tatum indices so that it (1) is more-or-less uniformly sampled throughout the song and (2) samples indices two tatums apart due to the weak correlation with lag 2 as was shown in Figure 4.

3.3.3 Interpolation of the Beat Phase Outputs

For interpolation, we evaluate the conditional expectation of $b_{\mathcal{T}}$ given $b_{\mathcal{D}_K}$:

$$\mathrm{E}[b_i|b_{\mathcal{D}_K}] = \Sigma_{i,\mathcal{D}_K}\Sigma^{-1}_{\mathcal{D}_K,\mathcal{D}_K}b_{\mathcal{D}_K}. \quad (7)$$

The evaluation of this function over all tatum indices requires a total of K invocations to the beat phase estimator.

Finally, given the interpolated beat phase estimates computed from Equation 7, we use in this paper a simple heuristic to decide the beat positions. For the given level of beat subdivision Δn used to compute the tatums, we find the beat phase given a beat subdivision. For each beat phase hypothesis ρ, we compute the following quantity:

$$R_\rho = \sum_{k=0}^{N/\Delta n} E[b_{k\Delta n+\rho}|b_{\mathcal{D}_K}] \quad (8)$$

Then, the beats are estimated as all tatum positions with indices $\hat{\rho} + k\Delta n$ with $k \in N$ and $\hat{\rho} = \arg\max_\rho R_\rho$. This heuristic is valid if the tatum tracking is successful and the number of tatums per beat remains fixed at Δn.

4. EXPERIMENTAL EVALUATION

4.1 Dataset

For the training and the validation dataset of the DNN beat identification and for estimating the auto-correlation of the beat identifier output at the tatum level, we used 100 popular songs from the RWC Popular Music Database [10]. For the test dataset, we prepared an in-house dataset consisting of 410 popular music in the United States and Japan. The median duration of the songs was four minutes. The tatum was extracted with the proposed method and the beat phase was hand-annotated based on a music score data created by professional musicians.

Method	Beat phase error	Real-time factor
BOCK15	14.0%	0.149
Baseline	12.2%	0.012
Proposed (Full)	6.1%	0.328
Proposed ($K = 4$)	16.2%	0.016
Proposed ($K = 8$)	10.1%	0.017
Proposed ($K = 16$)	7.6%	0.020
Proposed ($K = 32$)	6.1%	0.026

Table 2. The beat phase estimation error for songs that succeeded at tatum tracking, and the mean real-time factor.

4.2 Experiment 1: Beat Detection Improvement

First, we evaluated the beat phase estimation error of the proposed DNN beat identification method.

4.2.1 Experimental Condition

We extracted the beats using four methods: (1) an implementation [2] of the DNN-based beat detector in [3] with the tempo estimation method of [1], denoted "BOCK15," (2) the tatum detector used as a beat tracker (denoted "Baseline"), (3) the tatum detector with the DNN beat identifier evaluated over the entire data (denoted "Proposed (full)"), (4) the tatum detector with the DNN beat identifier evaluated over a subset of the data that has been selected with the proposed method (denoted "Proposed ($K = n$)" when using n tatum indices to evaluate the DNN). Since our focus is on fixing beat detection that succeeds at tatum tracking but fails at beat phase identification, we compared the methods for songs for which tatum extraction was successful (393 songs out of 410).

In addition, we computed the real-time factor, measured on a machine with Intel Core i5 processor running at 2.6 GHz with 4 GB of RAM with no GPU, utilizing one CPU core. Notice that the condition "Proposed (full)" is the baseline, in terms of computational speed, for most previous DNN-based downbeat detectors such as [7], as the previous method applies DNN to the entire audio data. The baseline method was implemented in C++ using SSE2 for SIMD instructions. The DNN was implemented in Python using the Chainer [3] library (an SSE2-optimized implementation of the DNN in C++ yielded in a similar benchmark and thus is omitted).

4.2.2 Results and Discussion

The results are shown in Table 2. It can be seen that by choosing $K = 32$, the method performs identically to when using the entire data for beat estimation, while being twelve times faster (38x faster than real-time). The modest increase in computation time over the Baseline suggests that there is only a marginal overhead, especially when the beat phase identification is executed in tandem with the beat extraction.

[2] Obtained from `https://github.com/CPJKU/madmom`, commit `de906fb`

[3] `https://github.com/chainer/chainer`

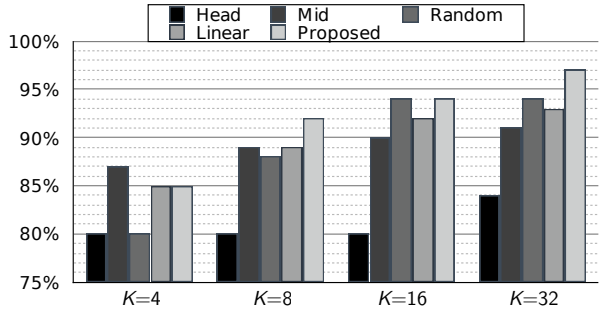

Figure 6. The match to the DNN beat detector when sampling K tatums in a song using different methods. Greater value indicates that the estimated beat phase better matches that estimated by the DNN.

To get a better idea of the computational costs, the beat detector takes 740 msec to process a minute of audio using a single core. The selection of indices incurred virtually no overhead, since we pre-computed the indices for all possible number of tatums. Note that this kind of index precomputation is viable for $K = 32$; even when evaluating the indices for songs with 65000 tatums, the total memory for storing the indices is at most only 4 MB.

Each call to the DNN used about 100 msec. Of this, the convolutional layer comprised about 60% of the total time, and the fully-connected layer comprised about 40%. In our implementation, the convolution layer is performed fully for each call to the DNN, but this is redundant because portions of MSLS spectrograms may be shared across different audio frames. For songs with fewer than $K \times 96$ tatums (*e.g.,* song of six and a half minutes with a BPM of 120 and $K = 32$), there are redundant outputs of the convolutive layers. Such a redundancy potentially enables us to further speed up the convolutive layers.

4.3 Experiment 2: Tatum Selection Strategy

Second, we evaluated the capability for the proposed tatum index selection method to approximate the DNN evaluated over all tatums, by comparing the index selection method to other possible ways of selecting the tatum index subset.

4.3.1 Experimental Condition

For each song in the test dataset, including those for which tatum tracking had failed, we selected K indices with $K = [4, 8, 16, 32]$, with five different strategies: (1) select the first K tatums (denoted "Head"), (2) select the middle K tatums (denoted "Mid"), (3) select K uniformly-sampled tatums (denoted "Random"), (4) select K linearly-spaced tatums (denoted "Linear"), and (5) select K tatums with the proposed method. Then we evaluated, for each strategy and K, the agreement rate of the beat detection output, evaluated with Equation 8, between that obtained using (1) the indices obtained by each strategy and (2) all index. Notice that a high agreement for a given index selection scheme suggests its capability to approximate the beat identification using all indices.

4.3.2 Results and Discussion

Figure 6 shows the agreement rate between each tatum selection strategy and the full DNN.

When trying to identify the beat with only more than four tatums, the proposed method consistently outperformed the baselines. When choosing only four tatums ($K = 4$), the strategy of choosing the middle performs the best, perhaps because it is better to focus on one region to estimate the beat phase instead of dispersing the selection throughout the piece. The result nonetheless demonstrates the capability of our method to select a "good" set of indices for beat identifications compared to other intuitively-arrived methods.

Comparing the result with the previous experiment, with $K = 32$ the agreement of the proposed method does not reach 100% even though the beat detection accuracy for $K = 32$ is identical to those using the entire DNN output. This means that interpolated beats disagree for songs for which tatum tracking has failed. Such a disagreement occurs because (1) if the tatum tracking fails, the assumed covariance of the DNN output Σ poorly describes the underlying DNN output and (2) our beat position decoding method relies on a proper tracking of tatums with no change of meter.

5. CONCLUSION

This paper presented a method to improve a fast and simple beat tracker with little computatinal overhead by using an elaborate DNN-based beat identifier to fix the error in the simple beat tracker at carefully-selected tatums.

We addressed the critical issue of achieving both the accuracy enjoyed by a DNN-based beat identification of slow and elaborate methods, and the fast speed enjoyed by a simple but erroneous beat tracking methods. This was tackled by applying a DNN beat identification *sparingly*, only at the most informative tatum indices given by a simple beat-tracker. The selection was done as to maximize the mutual information between the selected and the non-selected indices for invoking the DNN.

Evaluation demonstrated that the DNN halved the beat phase error, and the tatum selection strategy provided the same performance as when sweeping through the entire audio signal, resulting in a twelve-fold speed improvement for a typical song. Furthermore, the subset selection method was also shown to be consistently efficient at approximating the DNN output compared to other index selection methods.

Future work includes (1) application of the index selection framework to other tasks in MIR such as downbeat estimation, (2) relaxing the assumptions made for the index selection, such as assuming a known and a fixed covariance of the output, (3) allowing the parameter K to be determined automatically, (4) improving the heuristics for deciding the beat positions.

6. REFERENCES

[1] Sebastian Böck, Florian Krebs, and Gerhard Widmer. Accurate tempo estimation based on recurrent neural networks and resonating comb filters. In *Proc. International Conference on Music Information Retrieval*, pages 625–631, 2015.

[2] Sebastian Böck, Florian Krebs, and Gerhard Widmer. Joint beat and downbeat tracking with recurrent neural networks. In *Proc. International Conference on Music Information Retrieval*, 2016.

[3] Sebastian Böck and Markus Schedl. Enhanced beat tracking with context-aware neural networks. In *Proc. International Conference on Digital Audio Effects*, pages 135–139, 2011.

[4] Matthew E.P. Davies, Norberto Degara, and Mark 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.

[5] Matthew E.P. Davies and Mark D. Plumbley. Context-dependent beat tracking of musical audio. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(3):1009–1020, 2007.

[6] Simon Dixon. Evaluation of the audio beat tracking system Beatroot. *Journal of New Music Research*, 36(1):39–50, 2007.

[7] Simon Durand, Juan P. Bello, Bertrand David, and Gaël Richard. Feature adapted convolutional neural networks for downbeat tracking. In *Proc. International Conference on Acoustics, Speech and Signal Processing*, pages 296–300, 2016.

[8] Daniel P.W. Ellis. Beat tracking by dynamic programming. *Journal of New Music Research*, 36(1):51–60, 2007.

[9] Masataka Goto. An audio-based real-time beat tracking system for music with or without drum-sounds. *Journal of New Music Research*, 30(2):159–171, 2001.

[10] Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka. RWC music database: Popular, classical, and jazz music databases. In *Proc. International Conference on Music Information Retrieval*, pages 287–288, 2002.

[11] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[12] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

[14] Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9:235–284, 2008.

[15] Florian Krebs, Sebastian Böck, Matthias Dorfer, and Gerhard Widmer. Downbeat tracking using beat-synchronous features and recurrent neural networks. In *Proc. International Conference on Music Information Retrieval*, pages 129–135, 2016.

[16] Florian Krebs, Sebastian Böck, and Gerhard Widmer. Rhythmic pattern modeling for beat and downbeat tracking in musical audio. In *Proc. International Conference on Music Information Retrieval*, pages 227–232, 2013.

[17] Florian Krebs, Sebastian Böck, and Gerhard Widmer. An efficient state-space model for joint tempo and meter tracking. In *Proc. International Conference on Music Information Retrieval*, pages 72–78, 2015.

[18] Jean Laroche. Efficient tempo and beat tracking in audio recordings. *Journal of the Audio Engineering Society*, 51(4):226–233, 2003.

[19] George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functionsi. *Mathematical Programming*, 14(1):265–294, 1978.

[20] Geoffroy Peeters and Helene 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.

[21] Carl E. Rasmussen. Gaussian processes for machine learning. 2006.

[22] Jan Schlüter and Thomas Grill. Exploring data augmentation for improved singing voice detection with neural networks. In *Proc. International Conference on Music Information Retrieval*, pages 121–126, 2015.

[23] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[24] Nick Whiteley, Ali Taylan Cemgil, and Simon J Godsill. Bayesian modelling of temporal structure in musical audio. In *Proc. International Conference on Music Information Retrieval*, pages 29–34, 2006.