Music Tempo Estimation with \(k\)-NN Regression

*Antti Eronen and Anssi Klapuri

Abstract—An approach for tempo estimation from musical pieces with near-constant tempo is proposed. The method consists of three main steps: measuring the degree of musical accent as a function of time, periodicity analysis, and tempo estimation. Novel accent features based on the chroma representation are proposed. The periodicity of the accent signal is measured using the generalized autocorrelation function, followed by tempo estimation using \(k\)-Nearest Neighbor regression. We propose a resampling step applied to an unknown periodicity vector before finding the nearest neighbors. This step improves the performance of the method significantly. The tempo estimate is computed as a distance-weighted median of the nearest neighbor tempi. Experimental results show that the proposed method provides significantly better tempo estimation accuracies than three reference methods.

Index Terms—Music tempo estimation, chroma features, \(k\)-Nearest Neighbor regression.

I. INTRODUCTION

Musical meter is a hierarchical structure, which consists of pulse sensations at different time scales. The most prominent level is the tactus, often referred as the foot tapping rate or beat. The tempo of a piece is defined as the rate of the tactus pulse. It is typically represented in units of beats per minute (BPM), with a typical tempo being of the order of 100 BPM.

Human perception of musical meter involves inferring a regular pattern of pulses from moments of musical stress, a.k.a. accents [1], p.17. Accents are caused by various events in the musical surface, including the beginnings of all discrete sound events, especially the onsets of long pitched sounds, sudden changes in loudness or timbre, and harmonic changes. Many automatic tempo estimators try to imitate this process to some extent: measuring musical accentuation, estimating the periods and phases of the underlying pulses, and choosing the level corresponding to the tempo or some other metrical level of interest [2].

Tempo estimation has many applications, such as making seamless "beatmixes" of consecutive music tracks with the help of beat alignment and time stretching. In disc jockey applications metrical information can be used to automatically locate suitable looping points. Visual appeal can be added to music players with beat synchronous visual effects such as virtual dancing characters. Other applications include finding music with certain tempo from digital music libraries in order to match the mood of the listener or to provide suitable motivation for the different phases of a sports exercise. In addition, automatically extracted beats can be used to enable musically-synchronized feature extraction for the purposes of structure analysis [3] or cover song identification [4], for example.

A. Previous work

Tempo estimation methods can be divided into two main categories according to the type of input they process. The earliest ones processed symbolic (MIDI) input or lists of onset times and durations, whereas others take acoustic signals as input. Examples of systems processing symbolic input include the ones by Rosenthal [5] and Dixon [6].

One approach to analyze acoustic signals is to perform discrete onset detection and then use e.g. inter onset interval (IOI) histogramming to find the most frequent periods, see e.g. [7], [8]. However, it has been found better to measure musical accentuation in a continuous manner instead of performing discrete onset detection [9]. A time-frequency representation such as energies at logarithmically distributed subbands is usually used to compute features that relate to the accents [2], [10]. This typically involves differentiation over time within the bands. Alonso et al. use a subspace analysis method to perform harmonic+noise decomposition before accent feature analysis [11]. Peeters proposes the use of a reassigned spectral energy flux [12], and Davies and Plumbley use the complex spectral difference [3].

Accent feature extraction is typically followed by periodicity analysis using e.g. the autocorrelation function (ACF) or a bank of comb-filter resonators. The actual tempo estimation is then done by picking one or more peaks from the periodicity vector, possibly weighted with the prior distribution of beat periods [2], [13], [10]. However, peak picking steps are error prone and one of the potential performance bottlenecks in rhythm analysis systems.

An interesting alternative to peak picking from periodicity vectors was proposed by Seyerlehner et al., who used the \(k\)-Nearest Neighbor algorithm for tempo estimation [14]. Using the \(k\)-Nearest Neighbor algorithm was motivated based on the observation that songs with close tempi have similar periodicity functions. The authors searched the nearest neighbors of a periodicity vector and predicted the tempo according to the value that appeared most often within the \(k\) songs but did not report significant performance improvement over reference methods.

It should be noted that in the tempo estimation task, the temporal positions of the beats are irrelevant. In this sense, the present task differs from full meter analysis systems, where the positions of the beats need to be produced for example with dynamic programming [2], [10], [12], [15], [11] or Kalman filtering [16]. A full review of meter analysis systems is outside the scope of this article due to space restrictions. See [17] and [18] for more complete reviews.

B. Proposed method

In this paper, we study the use of the \(k\)-Nearest Neighbor algorithm for tempo estimation further. This is referred as \(k\)-NN regression as the tempo to be predicted is continuous-valued. Several improvements are proposed that significantly improve the tempo estimation accuracy using \(k\)-NN regression compared to the approach presented in [14]. First, if the training data does not have instances with very close tempi to the test instance, the tempo estimation is likely to fail. This is a quite common situation in tempo estimation because the periodicity vectors tend to be sharply peaked at the beat period and its multiples and because the tempo value to be predicted is continuous valued. With distance measures such as the Euclidean distance even small
A. Musical accent analysis

1) Chroma feature extraction: The purpose of musical accent analysis is to extract features that effectively describe song onset information and discard information irrelevant for tempo estimation. In our earlier work [2], we proposed an accent feature extractor which utilizes 36 logarithmically distributed subbands for accent measurement and then folds the results down to four bands before periodicity analysis.

In this work, a novel accent analysis front end is described which further emphasizes the onsets of pitch events and harmonic changes in music and is based on the chroma representation used earlier for music structure analysis in [21]. Figure 2 depicts an overview of the proposed accent analysis. The chroma features are calculated using a multiple fundamental frequency (F0) estimator [22]. The input signal sampled at 44.1 kHz sampling rate and 16-bit resolution is first divided into 93 ms frames with 50% overlap. In each frame, the salience, or strength, of each F0 candidate is calculated as a weighted sum of the amplitudes of its harmonic partials in a spectrally whitened signal frame. The range of fundamental frequencies used here is 80 – 640 Hz. Next, a transform is made into a musical frequency scale having a resolution of 1/3rd-semitone (36 bins per octave). This transform is done by retaining only the maximum-salience fundamental frequency component for each 1/3rd of a semitone range.

Finally, the octave equivalence classes are summed over the whole pitch range using a resolution of three bins per semitone to produce a 36 dimensional chroma vector $\bar{x}_k$, where $k$ is the frame index and $b = 1, 2, \ldots, b_0$ is the pitch class index, with $b_0 = 36$. The matrix $x_k$ is normalized by removing the mean and normalizing the standard deviation of each chroma coefficient over time, leading to a normalized matrix $\hat{x}_k$.

2) Musical accent calculation: Next, musical accent is estimated based on the normalized chroma matrix $\hat{x}_k$, $k = 1, \ldots, K$, $b = 1, 2, \ldots, b_0$, much in a similar manner as proposed in [2], the main difference being that frequency bands are replaced with pitch classes. First, to improve the time resolution, the chroma coefficient envelopes are interpolated by a factor eight by adding zeros between the samples. This leads to the sampling rate $f_s = 172$ Hz. The interpolated envelopes are then smoothed by applying a sixth-order Butterworth low-pass filter (LPF) with $f_{LP} = 10$ Hz cutoff. The resulting smoothed signal is denoted by $z_k(n)$. This is followed by half-wave rectification and weighted differentiation steps. A half-wave rectified (HWR) differential of $z_k(n)$ is first calculated as

$$z'_k(n) = \text{HWR}(z_k(n) - z_k(n - 1)),$$

where the function $\text{HWR}(x) = \max(x, 0)$ sets negative values to zero and is essential to make the differentiation useful. Next we form a weighted average of $z_k(n)$ and its differential $z'_k(n)$:

$$u_k(n) = (1 - \lambda)z_k(n) + \lambda \frac{f_s}{f_{LP}} z'_k(n),$$

where $0 \leq \lambda \leq 1$ determines the balance between $z_k(n)$ and $z'_k(n)$, and the factor $f_s/f_{LP}$ compensates for the small amplitude of the differential of a low-pass-filtered signal [2].

Finally, bands are linearly averaged to get a single accent signal $a(n)$ to be used for periodicity estimation. It represents the degree of musical accent as a function of time.

B. Periodicity analysis

Periodicity analysis is carried out on the accent signal. Several periodicity estimators have been proposed in the literature, such as the inter-onset interval histogramming [7], autocorrelation function (ACF) [23], or comb filter banks [24]. In this paper, we use the generalized autocorrelation function (GACF) which is computationally efficient and has proven to be a robust technique in multipitch analysis [20]. The GACF is calculated without windowing in successive frames of length $W$ and 16% overlap. The input vector $a_m$ at the $m$th frame has the length of $2W$ after zero padding to twice its length:

$$a_m = [a((m - 1)W), \ldots, a(mW - 1), 0, \ldots, 0]^T,$$

where $T$ denotes transpose. The GACF is defined as (19):

$$\rho_m(\tau) = \text{IDFT}(|\text{DFT}(a_m)|^p),$$

where DFT stands for Discrete Fourier Transform and IDFT its inverse. The coefficient $p$ controls the frequency domain compression. $\rho_m(\tau)$ gives the strength of periodicity at period (lag) $\tau$. The GACF
was selected because it is straightforward to implement as usually the fast Fourier transform routines are available, and it suffices to optimize the single parameter \( p \) to make the transform optimal for different accent features. The conventional ACF is obtained with \( p = 2 \). We optimized the value of \( p \) for different accent features by testing a range of different values and performing the tempo estimation on a subset of the data. The value that led to the best performance was selected for each feature. For the proposed chroma accent features, the value used was \( p = 0.65 \).

At this step we have a sequence of periodicity vectors computed in adjacent frames. If the goal is to perform beat tracking where the tempo can vary in time, we would consider each periodicity vector separately and estimate the tempo as a function of time from each vector separately. In this paper, we are interested in getting a single representative tempo value for each musical excerpt. Therefore, we obtain a single representative periodicity vector \( \rho_{med}(\tau) \) for each musical excerpt by calculating point-wise median of the periodicity vectors over time. This assumes that the excerpt has nearly constant tempo and is sufficient in applications where a single representative tempo value is desired. The median periodicity vector is further normalized to remove the trend due to the shrinking window for larger lags

\[
\hat{\rho}_{med}(\tau) = \frac{1}{W} \rho_{med}(\tau). \tag{5}
\]

The final periodicity vector is obtained by selecting the range of bins corresponding to periods from 0.06 s to 2.2 s, and removing the mean and normalizing the standard deviation to unity for each periodicity vector.

The resulting vector is denoted by \( s(\tau) \). Figure 3 presents the periodicity vectors for the songs in our evaluation database, ordered in ascending tempo order. Indeed, the shape of the periodicity vectors is similar across music pieces, with the position of the peaks changing with tempo.

C. Tempo estimation by k-NN regression

The tempo estimation is formulated here as a regression problem: given the periodicity observation \( s(\tau) \), we estimate the continuous valued tempo \( T \). In this paper, we propose to use locally weighted learning ([25]) to solve the problem. More specifically, we use \( k \)-Nearest Neighbors regression and compute the tempo as a weighted median of the nearest neighbor tempi. In conventional \( k \)-NN regression, the property value of an object is assigned to be the average of the values of its \( k \) nearest neighbors. The distance to the nearest neighbors is typically calculated using the Euclidean distance.

In this paper, several problem-specific modifications are proposed to improve the performance of tempo estimation using \( k \)-NN regression. First, a resampling step is proposed to alleviate problems caused by mismatches of the exact tempo values in the testing and training data. Distance measures such as the Euclidean distance or correlation distance are sensitive to whether the peaks in the unknown periodicity vector and the training vectors match exactly. With the resampling step it is more likely that similarly shaped periodicity vector(s) with a close tempo are found from the training set. Resampling is applied to “stretch” and “shrink” the unknown test vectors to increase the likelihood of a matching training vector to be found from the training set. Since the tempo values are continuous, the resampling ensures that we do not need to have a training instance with exactly the same tempo as the test instance in order to find a good match.

Thus, given a periodicity vector \( s(\tau) \) with unknown tempo \( T \), we generate a set of resampled test vectors \( s_i(\tau) \), where subscript \( r \) indicates the resampling ratio. A resampled test vector will correspond to a tempo of \( T/r \). We tested various possible ranges for the resampling ratio, and 15 linearly spaced ratios between 0.87 and 1.15 were taken into use. Thus, for a piece having a tempo of 120 BPM the resampled vectors correspond to a range of tempi from 104 to 138 BPM.

When receiving an unknown periodicity vector, we first create the resampled test vectors \( s_i(\tau) \). The Euclidean distance between each training vector \( t_m(\tau) \) and the resampled test vectors is calculated as

\[
d(m, r) = \sqrt{\sum_\tau (t_m(\tau) - s_i(\tau))^2} \tag{6}
\]

where \( m = 1, ..., M \) is the index of the training vector. The minimum distance \( d(m) = \min_r d(m, r) \) is stored for each training instance \( m \), along with the resampling ratio that leads to the minimum distance \( r(m) = \arg\min_r d(m, r) \). The \( k \) nearest neighbors that lead to the \( k \) lowest values of \( d(m) \) are then used to estimate the unknown tempo.

The annotated tempo \( T_{ann}(i) \) of the nearest neighbor \( i \) is now an estimate of the resampled test vector tempo. Multiplying the nearest neighbor tempo with the ratio gives us an estimate of the original test vector tempo: \( T(i) = T_{ann}(i) r(i) \).

The final tempo estimate is obtained as a weighted median of the nearest neighbor tempo estimates \( T(i), i = 1, ..., k \). Due to the weighting, training instances close to the test point have a larger effect on the final tempo estimate. The weights \( w_i \) for the \( k \) nearest neighbors are calculated as

\[
w_i = \frac{\exp(-\gamma d(i))}{\sum_{i=1}^k \exp(-\gamma d(i))}, \tag{7}
\]

where the parameter \( \gamma \) controls how steeply the weighting decreases with increasing distance \( d \), and \( i = 1, ..., k \). The value \( \gamma = 40 \) was found by monitoring the performance of the system with a subset of the data. The exponential function fulfills the requirements for a weighting function in locally weighted learning: the maximum value is at zero distance, and the function decays smoothly as the distance increases [25]. The tempo estimate is then calculated as a weighted median of the tempo estimates \( T(i) \) using the weights \( w_i \) with the procedure described in [26]. The weighted median gives significantly better results than a weighted mean. The difference between weighted median and unweighted median is small but consistent in favor of the weighted median when the parameter \( \gamma \) is properly set.

In addition, the use of an outlier removal step is evaluated to
improve the quality of the training data. We implemented leave-one-out outlier removal as described in [27]. It works within the training data by removing each sample in turn from the training data, and classifying it by all the rest. Those training samples that are misclassified are removed from the training data.

III. RESULTS

This section looks at the performance of the proposed method in simulations and compares the results to three reference systems and three accent feature extractors.

A. Experimental setup

A database of 355 musical pieces with CD quality audio was used to evaluate the system and the three reference methods. The musical pieces were a subset\(^1\) of the material used in [2]. The database contains examples of various musical genres whose distribution is the following: 82 classical pieces, 28 electronic/dance, 12 hip hop/rap, 60 jazz/blues, 118 rock/pop, 42 soul/RnB/funk, and 13 world/folk. Full listing of the database is available at www.cs.tut.fi/~cronen/taalp08-tempo-dataset.html. The beat was annotated from approximately one-minute long representative excerpts by a musician who tapped along with the pieces. The ground truth tempo for each excerpt is calculated based on the median inter-beat-interval of the tapped beats. The distribution of tempi is depicted in figure 4.

We follow here the evaluation presented in [14]. Evaluation is done using leave-one-out cross validation: the tempo of the unknown song is estimated using all the other songs in the database. The tempo estimate is defined to be correct if the predicted tempo estimate is within 4% of the annotated tempo.

Along with the tempo estimation accuracy, we also report a tempo category classification accuracy. Three tempo categories were defined: from 0 to 90 BPM, 90 to 130 BPM, and above 130 BPM. Classification of the tempo category is considered successful if the predicted tempo falls within the same category as the annotated tempo. This kind of "rough" tempo estimate is useful in applications that would only require e.g. classifying songs to slow, medium, and fast categories.

The decision whether the differences in error rates is statistically significant is done using McNemar’s test [28]. The test assumes that the trials are independent, an assumption that holds in our case since the tempo estimation trials are performed on different music tracks. The null hypothesis \(H_0\) is as follows: given that only one of the two algorithms makes an error, it is equally likely to be either one. Thus, this test considers those trials where two systems make different predictions, since no information on their relative difference is available from trials in which they report the same outcome. The test is calculated as described in [28, Section 3], and \(H_0\) is rejected if the \(P\)-value is less than a selected significance level \(\alpha\). We report the results using the following significance levels and wordings: \(P \geq 0.05\), not significant (NS); \(0.01 \leq P < 0.05\), significant (S); \(0.0001 \leq P < 0.01\), very significant (VS); and \(P < 0.0001\), highly significant (HS).

B. Reference methods

To put the results in perspective, the results are presented in comparison to three reference methods. The first was described by Ellis [10], and is based on an accent feature extractor using the mel-frequency filterbank, autocorrelation periodicity estimation, and dynamic programming to find the beat times. The implementation is also provided by Ellis [29]. The second reference method was proposed by ourselves in [2] and was the best performing method in the Music Information Retrieval Evaluation eXchange (MIREX 2006) evaluations [9]. The third has been described in [13] and is based on a computationally efficient accent feature extraction based on multirate analysis, discrete cosine transform periodicity analysis, and period determination utilizing simplified musicological weight functions. The comparison against the Ellis method may not be completely fair as it has not received any parameter optimization on any subset of the data used. However, the two other methods have been developed on the same data and are thus good references.

In addition to comparing the performance of the proposed method to the complete reference systems, we also evaluate the proposed musical accent measurement method against four other features. This is done by using the proposed k-NN regression tempo estimation with accent features proposed elsewhere. Comparisons are presented to two auditory spectrogram based accent features: first using a critical band scale as presented in [2] (KLAP) and the second using the Mel-frequency scale (MEL). Another two accent features are based on the quadrature mirror filter bank of [13] (QMF), and a straightforward chroma feature analysis (SIMPLE). The main difference between the various methods is how the frequency decomposition is done, and how many accent bands are used for periodicity analysis. In the case of the MEL features, the chroma vector \(x(k)\) is replaced with the output band powers of the corresponding auditory filterbank. In addition, logarithmic compression is applied to the band envelopes before the interpolation step, and each nine adjacent accent bands are combined into one resulting into four accent bands. Periodicity analysis is done separately for four bands, and final periodicity vector is obtained by summing across bands. See the details in [2]. In the case of the QMF and KLAP front ends, the accent feature calculation is as described in the original publications [13] and [2]. The method SIMPLE differs from the method proposed in this paper in how the chroma features are obtained: whereas the proposed method uses saliences of F0 estimates mapped on a musical scale, the method SIMPLE simply accumulates the energy of FFT bins to 12 semitone bins. The accent feature parameters such as \(\lambda\) were optimized for both the chroma accent features and the MEL accent features using a subset of the data. The parameters for the KLAP and QMF methods are as presented in the original publications [13] and [2]. The frame size and frame hop for the methods MEL and SIMPLE is fixed at 92.9 ms and 46.4 ms, respectively. The KLAP feature extractor utilizes a frame size of 23 ms with 50% overlap.

C. Experimental results

1) Comparison to reference methods: Table I shows the results of the proposed method in comparison with the reference systems. The statistical significance is reported under each accuracy percentage in comparison to the proposed method. All the reference systems output both the period and timing of the beat time instants and the output tempo is calculated based on the median inter beat interval. We

\(^1\) The subset consisted of all music tracks to which the first author had access.
observe a highly significant or very significant performance difference in comparison to all the reference methods in both tasks.

2) Importance of different elements of the proposed method: The following experiments study the importance of different elements of the proposed method in detail. Table II presents the results obtained using different accent feature extractors. The performance of a certain accent feature extractor depends on the parameters used, such as the parameter λ controlling the weighted differentiation described in section II-A2. There is also some level of dependency between the accent features and periodicity estimation parameters, i.e. the length of the GACF window, and the exponent used in computing the GACF. These parameters were optimized for all accent features using a subset of the database, and the results are reported for the best parameter setting.

The proposed chroma accent features based on F0 salience estimation perform best, although the difference is not statistically significant in comparison to the accent features proposed earlier in [2]. The difference in comparison to the three other front ends in tempo estimation is statistically significant. The accent features based on the QMF-decomposition are computationally very attractive and may be a good choice if the application only requires classification into rough tempo categories, or if the music consists mainly of fast pieces as slow. Very fast tempi might deserve special treatment by a factor of two: the proposed method would still classify 28% of fast pieces as slow. Very fast tempi might deserve special treatment in future work.

3) Performance across tempo categories: Examining the performance across in classifying within different tempo categories is illustrative of the performance of the method, showing how evenly the method performs with slow, medium, and fast tempi. Tables IV and V depict the confusion matrices in tempo category classification for the proposed method and the best performing reference method, respectively. Rows correspond to presented tempo, columns to the estimated tempo category. Errors with slow and fast tempi cause the accuracy of tempo category classification to be generally smaller than that of tempo estimation. Both methods perform very well in classifying the tempo category within the medium range of 90 to 130 BPM. However, especially fast tempi are often underestimated by a factor of two: the proposed method would still classify 28% of fast pieces as slow. Very fast tempi might deserve special treatment in future work.

4) Effect of training data size: The quality and size of the training data has an effect on the performance of the method. To test the effect of the training data size, we ran the proposed method while varying the size of the training data. The outlier removal step is omitted. Figure 5 shows the result of this experiment. Uniform random samples with a fraction of the size of the complete training data were used to perform classification. A graceful degradation in performance is observed. The drop in performance becomes statistically significant at training data size of 248 vectors, however, over 70% accuracy is obtained using only 71 reference periodicity vectors. Thus, good performance can be obtained with small training data sizes if the reference vectors span the range of possible tempi in a uniform manner.
methods were Matlab implementations. The accent feature extraction was implemented completely in C++. The Seppänen paper generally believed to highlight information on harmonic changes ([31]), during a measure, and the beat or measure phase is then predicted as a feature vector is constructed by taking values of the accent signal. In this case, the estimate is wrong using both accent features. Nevertheless, there are many cases where the proposed method makes tempo doubling or halving errors. But at the same range there are also many cases where the proposed method correctly estimates tempo halving or doubling. Out of the 355 test instances, 255 instances were incorrectly estimated using both accent features. At indices between 310 and 350 the method KLAP correctly estimates tempo, which might be utilized in the future.

6) Computational complexity: To get a rough idea of the computational complexity of the method, a set of 50 musical excerpts were processed with each of the methods and the total run time was measured. From fastest to slowest, the total run times are 130 seconds for Seppänen et al. [13], 144 seconds for the proposed method, 187 seconds for Ellis [10], and 271 seconds for Klapuri et al. [2]. The Klapuri et al. method was the only one that was implemented completely in C++. The Seppänen et al. and Ellis methods were Matlab implementations. The accent feature extraction of the proposed method was implemented in C++, the rest in Matlab.

IV. DISCUSSION AND FUTURE WORK

Several potential topics exist for future research. There is some potential for further improving the accuracy by combining different types of features as suggested by one of the reviewers. Figure 6 presents a pairwise comparison of the two best performing accent front ends: the F0-salience based chroma accent proposed in this paper and the method KLAP. The songs have been ordered with respect to increasing error made by the proposed method. The error is computed as follows ([9]):

\[
E = \log_2(\frac{\text{computed tempo}}{\text{correct tempo}}).
\]  

The value 0 corresponds to correct tempo estimates, and the value 1 to tempo halving or doubling. Out of the 355 test instances, 255 instances were correctly estimated using both accent features. 60 instances were incorrectly estimated using both accent features. At indices between 310 and 350 the method KLAP correctly estimates some cases where the proposed method makes tempo doubling or halving errors. But at the same range there are also many cases where the estimate is wrong using both accent features. Nevertheless, there is some complementary information in these accent feature extractors which might be utilized in the future.

Second direction is to study whether a regression approach can be implemented for beat phase and barline estimation. In this case, a feature vector is constructed by taking values of the accent signal during a measure, and the beat or measure phase is then predicted using regression with the collected feature vectors. Chroma is generally believed to highlight information on harmonic changes ([31]), thus the proposed chroma accent features would be worth testing in barline estimation.

V. CONCLUSION

A robust method for music tempo estimation was presented. The method estimates the tempo using locally weighted k-NN regression and periodicity vector resampling. Good performance was obtained by combining the proposed estimator with different accent feature extractors. The proposed regression approach was found to be clearly superior compared to peak picking techniques applied on the periodicity vectors. We conclude that most of the improvement is attributed to the regression based tempo estimator with a smaller contribution to the proposed F0-salience chroma accent features and GACF periodicity estimation, as there is no statistically significant difference in error rate when the accent features used in [2] are combined with the proposed tempo estimator.

In addition, the proposed regression approach is straightforward to implement and requires no explicit prior distribution for the tempo as the prior is implicitly included in the distribution of the k-NN training data vectors. The accuracy degrades gracefully when the size of the training data is reduced.

REFERENCES


