Addressing Tempo Estimation Octave Errors in Electronic Music by Incorporating Style Information Extracted from Wikipedia

Florian Hörschläger, Richard Vogl, Sebastian Böck, Peter Knees
Dept. of Computational Perception, Johannes Kepler University Linz, Austria
florian.hoerschlaeger@jku.at, richard.vogl@jku.at,
sebastian.boeck@jku.at, peter.knees@jku.at

ABSTRACT

A frequently occurring problem of state-of-the-art tempo estimation algorithms is that the predicted tempo for a piece of music is a whole-number multiple or fraction of the tempo as perceived by humans (tempo octave errors). While often this is simply caused by shortcomings of the used algorithms, in certain cases, this problem can be attributed to the fact that the actual number of beats per minute (BPM) within a piece is not a listener's only criterion to consider it being "fast" or "slow". Indeed, it can be argued that the perceived style of music sets an expectation of tempo and therefore influences its perception.

In this paper, we address the issue of tempo octave errors in the context of electronic music styles. We propose to incorporate stylistic information by means of probability density functions that represent tempo expectations for the individual music styles. In combination with a style classifier those probability density functions are used to choose the most probable BPM estimate for a sample. Our evaluation shows a considerable improvement of tempo estimation accuracy on the test dataset.

1. INTRODUCTION

A well-known problem of tempo estimation algorithms is the so called *tempo octave error*, i.e., the tempo as predicted by the algorithm is a whole-number multiple or fraction of the actual tempo as perceived by humans. Since these errors on the metrical level are not always clearly agreed on by humans, evaluations performed in the literature discount octave tempo errors by introducing *secondary accuracy* values (i.e. accuracy2) which also consider double, triple, half, and third of the ground truth tempo as a correct prediction. In average, these values exceed the primary accuracy values based only on exact matches by about 20 percentage points, cf. [1, 2].

While for tasks such as automatic tempo alignment for DJ mixes this can be a tolerable mistake, for making accurate predictions of the semantic category of musical "speed," i.e., whether a piece of music is considered "fast" or "slow," this discrepancy shows that there is still a need for improvement. To this end, several approaches have directly addressed the octave error problem, either by incorporating a-priori knowledge of tempo distributions [3], spectral and rhythmic similarity [1, 2], source separation [4, 5], or classification into speed categories based on audio [6, 7] and user-generated meta-data [8, 9].

The importance of stylistic context for the task of beat tracking has been stated before [10]. Similarly, in this work, we argue that there is a connection between the style of the music and its perceived tempo (which is strongly related to the perception of the beat). More precisely, we assume that human listeners take not only rhythmic information (onsets, percussive elements) but also stylistic cues (such as instrumentation or loudness) into account when estimating tempo. [1] Therefore, when including knowledge on the style of the music, tempo estimation accuracy should improve. Consider this simple example: If we knew that an audio sample is a *drum and bass* track, it would be unreasonable to estimate a tempo below 160 BPM. Yet our findings show that uninformed estimators can produce such an output. Therefore we propose to incorporate stylistic information into the tempo estimation process by means of a music style classifier trained on audio data. In addition to predicting multiple hypotheses on the tempo of a piece of music using a state-of-the-art tempo estimation algorithm, we determine its style using the classifier and choose the tempo hypothesis being most likely in the context of the determined style. For this, we utilize probability density functions (PDF) constructed from data extracted from Wikipedia articles (i.e., BPM ranges or values as well as tempo relationships).

The remainder of this paper is organized as follows. Section 2 covers a representative selection of related work. In section 3 the proposed system is presented. This includes our strategy to extract style information from Wikipedia, in particular information on style-specific tempo ranges. In section 4 we evaluate our approach using a new data set for tempo estimation in electronic music. The paper concludes with a short discussion and ideas for future work in section 5.

[1] This assumption is supported by psychological evidence that identification of pieces as well as recognition of styles and emotions can be performed by humans within 400 msecs [11]. This information can therefore prime the assessment of rhythm and tempo which requires more context.

2. RELATED WORK

Gouyon et al. compare and discuss 11 tempo estimation algorithms submitted to the ISMIR'04 tempo induction contest [12]. Their paper shows that all submitted algorithms perform much better if tempo octave errors are considered as correctly estimated tempos. By ignoring this kind of error it was already possible to reach accuracies beyond 80%. A more recent comparison of state-of-the art tempo estimation algorithms is given by Zapata and Gómez [13]. Again the 11 algorithms compared in [12] are discussed along with 12 new approaches. In this comparison, again, the algorithm presented by Klapuri et al. [3] performs best, if tempo octave errors are ignored.

The tempo estimation algorithm described in [3] uses a bank of comb filters similarly to the approach by Scheirer [14]. One important difference is that while Scheirer uses only five frequency subbands to calculate the input signal for the comb filters, Klapuri et al. use 36 frequency subbands which are combined into 4 so-called "accent bands". This was done with the goal that changes in narrower frequency bands are also detected while maintaining the ability to detect more global changes which was already the case in [14].

The two approaches presented by Seyerlehner et al. [1] are based on two periodicity sensitive features (the autocorrelation function and fluctuation patterns) which are each used to train a k-Nearest-Neighbour classifier. The results obtained with this algorithm is at least comparable to the best results found in [12].

Peeters [2] uses a frequency domain analysis of an onset-energy function to extract so called spectral templates. During training, reference spectral patterns are created used in two different approaches. First in an unsupervised approach where clustering of similar spectral templates is done via a fuzzy k-means algorithm and second in an supervised variant where the 8 genres of the training dataset (ballroom) are used. Viterbi decoding is then used to determine the two hidden variables (tempo and rhythmical pattern) of the spectra templates to estimate the tempo of an audio track.

Gkiokas et al. [7] and Eronen and Klapuri [6] use machine learning approaches to further improve tempo estimation results. While [7] uses support vector machines to classify additional tempo cues in combination with the periodicity vectors, [6] uses k-nearest-neighbour regression in combination with the autocorrelation function of the accent signal. In [5], Elowsson et al. try to improve the tempo estimation results by separating percussive and harmonic sound sources and extracting different features on the resulting signals. Gärtner [15] proposes tempo detection based on non-negative matrix factorization and reports good results on a dataset comprised of urban club music.

Other approaches that aim at classifying music speed use external meta-data. Hockman and Fujinaga learn to classify music pieces into the categories "fast" and "slow" based on user tags found on YouTube [8]. Using simple frame-level features, they could reach a classification accuracy of 96%. Following a similar argumentation, Levy claims that the tempo octave error rate can be reduced by utilizing user tags of "fast" and "slow" [9]. Independent of a specific method for tempo estimation, Moelants and McKinney investigate the factors of a piece being perceived as fast, slow, or temporally ambiguous [16]. In this work, we focus on predicting the correct *beats per minute (bpm)* for a music piece rather than directly classifying music into speed categories.

3. METHOD

Our approach consists of a two stage tempo estimation process (visualized in figure 1). First, the *Tempo Estimator* generates $n = 10$ tempo estimates using a state-of-the-art tempo estimation approach. Second, the *Style Estimator* classifies the audio file into a style. Finally, the *Tempo Ranker* chooses the most probable tempo in the context of the classified style. In the following, we describe the used tempo induction approach (that also serves as a reference baseline for our evaluations), the construction of the style classifier and the strategy for picking the most probable tempo estimate. In the context of this work we also present an approach for extraction of music style specific tempo information from Wikipedia articles and how it is used within the described scheme.

3.1 Baseline Tempo Estimator

In the tempo estimation stage, any state-of-the-art tempo induction algorithm that can provide more than one tempo hypothesis can be used. In this work, we make use of the beat detection method introduced by Böck in [17]. The algorithm is based on *bidirectional long short-term memory (BLSTM)* recurrent neural networks. As network input, six variations of *short time fourier transform (STFT)* spectrograms transformed to the Mel-scale (20 bands) are used. The six variations consist of three spectrograms which are calculated using window sizes of 1024, 2048, and 4096 samples. For this process, audio data with a sampling rate of 44.1kHz is used which results in windows lengths of 23.2ms, 46.4ms and 92.8ms, respectively. In addition to these three spectrograms, the positive first order difference to the median of the last 0.41s of every spectrogram is used as input. The neural networks are randomly initialized and trained using manually annotated ground truth for beats from the *ballroom dataset*.[2] The trained neural network produces beat activation functions which are then used to calculate an autocorrelation function. The peaks in the smoothed autocorrelation function represent the tempo candidates expressed in *beats per minute (BPM)*. The height of the peaks corresponds to the probability of being the dominant tempo of the track.

3.2 Style Classification

The construction of the style classifier is based on the approach proposed by Seyerlehner et al. [18] that consistently yielded top-ranked results in the recent MIREX tasks on similarity estimation and genre and tag classification. As described in [19], this genre classification algorithm uses

[2] http://mtg.upf.edu/ismir2004/contest/tempoContest/node5.html

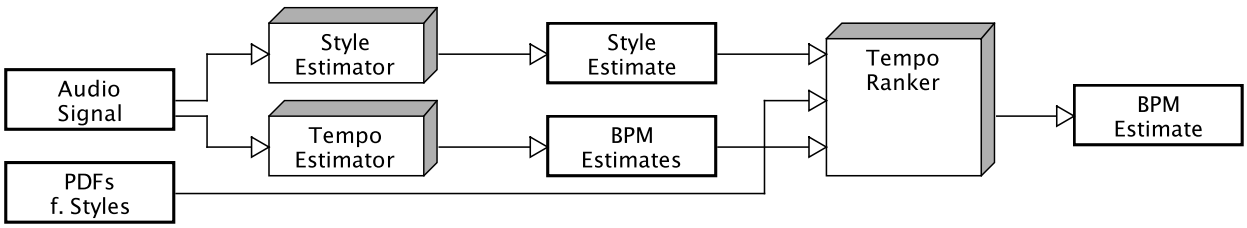

Figure 1. Schematic overview of the proposed system. The audio signal is used to derive multiple tempo estimations as well as a style estimation. The Tempo Ranker utilizes the style estimate, the tempo estimations and a set of predefined probability density functions (PDFs) to chose the most probable tempo in the context of the estimated style.

six block-level feature types, namely *spectral pattern*, *delta spectral pattern*, *variance delta spectral pattern*, *logarithmic fluctuation pattern*, *correlation pattern*, and *spectral contrast pattern*. Furthermore we add another vector, containing ten preferred BPM estimates produced by the tempo induction algorithm described in subsection 3.1. Directly concatenating the six block-level feature vectors would result in a combined feature vector with a length of 9,448 dimensions. To speed up training and reduce redundancy we first normalize those individual feature vectors and perform a separate *principal component analysis (PCA)* on each of the six feature types over the whole training set. For each feature type, we then take the first l dimensions of the PCA-transformed feature vectors which have a cumulative sum of their latent values (eigenvalues of covariance matrix) of more than 80%. The final feature vector is then obtained by concatenation of the reduced feature vectors as well as the vector containing the BPM estimates. Note that we do not transform the vector containing the tempo estimates. On the used dataset, this results in vectors with a length of 113 dimensions. Using this feature vectors we train a *random forest classifier* with 500 trees on a dataset containing 23,000 random samples downloaded from the *Beatport*® website. Those samples are almost equally distributed among the 23 different styles.

Samples are classified by first computing the block-level features and tempo estimates individually for each file. The six block-level feature vectors are individually (i) normalized by reusing the averages obtained from normalizing the training data (ii) transformed by reusing the coefficient matrices obtained by the PCA of the training data (iii) reduced by only taking the first l dimensions, where l are the same numbers as for the training vectors. The final vector is then composed of the six resulting vectors as well as the vector containing the BPM estimates and classified by the random forest.

Although not our primary interest in this work, we test the quality of the music style classification component by conducting 8-fold cross-validation on the training dataset. The average accuracy on the individual folds is 52.3 percent while the standard deviation is 1.0 percent. The overall accuracy on the *GiantSteps* tempo data set, that is used for the tempo estimation experiments, is 56.3 percent.

Style	from	to	slower than
chill-out	80	160	house
funk-r-and-b	80	160	
house	115	130	trance*
minimal	125	130	
electro-house	128	130	
glitch-hop	128	130	
hip-hop	124	135	
breaks	110	150	
indie-dance-nu-disco	120	140	
progressive-house	110	150	
pop-rock	130	140	
techno	120	150	psy-trance*
dubstep	130	142	
reggae-dub	130	142	
deep-house	110	170	
trance	120	160	
hard-dance	140	150	
psy-trance	140	150	
electronica	119	180	
drum-and-bass	130	180	
hardcore-hard-techno	160	200	
tech-house	180	220	

Table 1. Tempo ranges and tempo relationships for the *GiantSteps* dataset styles extracted from Wikipedia articles. Relationships marked with an asterisk where converted from *faster than* to *slower than* relationships.

3.3 Incorporating Style Information

To make use of the classified style in terms of tempo estimation, we need a suitable way to add tempo restrictions to each style. We do this by linking the different styles in our dataset to probability density functions, which are used to rank the different estimates. This section describes how the ranking works, the tempo information was extracted from Wikipedia and the probability density functions were modeled given the tempo information.

3.3.1 Deriving tempo information from Wikipedia articles

In this work we focused on deriving two different kinds of tempo information from Wikipedia articles:

- tempo annotations, which can either be BPM ranges or BPM values
- tempo relationships, which model whether one style is faster than another one (or vice verca)

This is achieved by a combination of heuristics and regular expression patterns, which were hand-crafted after reviewing a considerable amount of examples. Experience has shown that this task offers some major challenges: (i) multiple tempo annotations for styles, (ii) tempo annotations that need to be associated with other styles and (iii) the usage of synonyms and the complexity of the natural language.

For the purpose of this experiment we crawled a Wikipedia dump using JWPL [20] and Sweble [21] and used a hybrid strategy to decide if an article is about a music genre/style or not. If an article contains an instance of the *infobox music genre* [3] we assume it is indeed about a genre (infobox data is due to it's high quality utilized in many projects e.g. [22]). If this is not the case a WEKA [23] classifier is used. This classifier was trained using tf-idf weights, as well as some features based on infobox availability and the Wikipedia category and article graph [24] (e.g. number of referenced artists or the minimal category-graph distance to the root category of music genres). The training dataset was constructed by utilizing instances of the infobox music genre: articles containing the infobox as well as those referred as subgenres were added as positive training examples, articles referred as instruments and cultural origins were added as negative training examples. Given the resulting genres sub- and supergenre relationships were extracted by using data available in music genre infoboxes. Using this approach we were able to extract 775 genres (with a precision of 96.6 %) as well as 2.217 sub- and supergenre relationships from the Wikipedia snapshot created on 2nd May 2014.

In order to extract tempo information the derived genre-graph is traversed and all article texts are processed. The article texts are scanned for relevant sentences (e.g., containing a notation for BPM). Those sentences are matched with the hand-crafted patterns. Due to the fact that there are some genre articles that contain tempo annotations relevant to other genres it was necessary to perform a sanity check: For each match, the current section's title is cross-checked with a blacklist made up of the names and synonyms associated with the related genres (either sub- or supergenres of the current genre) as well as some words indicating that the content is about another genre (e.g. subgenres, related, influences). Whenever the section title matches an entry of the blacklist the system tries to associate the tempo annotation with the correct genre. This is done by matching names or synonyms of related genres within the current sentence or section title. All annotations that cannot be associated with a unique genre are dropped. To give an impression of what a sentence and a pattern might look like we provide the following example [4] (containing booth a range and a single value):

- "The average tempo of a minimal techno track is between 125 and 130 beats per minute. Richie Hawtin suggests 128 BPM as the perfect tempo."
  - between([^\d]{0,5}|)(**d**{**2,4**})([^\d]{0,5}|)(and)([^\d]{0,5}|)(**d**{**2,4**})
  - (**d**{**2,4**})[^\d]{0,5}BPM

Depending on the case, the bold face printed regions are then used as upper or lower boundary of a BPM range or a single BPM value. Tempo relationships are derived in a similar manner. Again all irrelevant sentences are dropped. Identified genre's names (and synonyms) are masked in order to make matching easier. Whenever a regular expression matches, a relationship is instantiated between the identified genres. This approach works reasonably well for extracting tempo annotations (precision = 90.2%) while the tempo relationship extraction (precision = 81.7%) would probably benefit from more sophisticated natural language processing techniques. Using this simple technique we were able to extract 94 tempo annotations - most of them associated with electronic music genres. Furthermore we were able to extract 38 tempo relationships.

3.3.2 Ranking of BPM estimates

In our approach the baseline estimator computes *ten* BPM estimates, those estimates are ranked by the Tempo Ranker. The Tempo Ranker uses predefined probability density functions (PDFs) to choose the most likely BPM estimate given by tempo estimation algorithm. To further formalize the behavior of the ranker, let S be the set of music styles, PDF_s be the probability density function (a function assigning a probability to BPM value) for the individual style estimate s and $E = \{e_0, e_1, .., e_9\}$ be the set of computed tempo estimates. Then the ranker chooses the tempo estimate $e_{max} \in E$ that maximizes the result of PDF_s (see equation 1).

$$e_{max} = \arg\max_{e \in E} PDF_s(e) \tag{1}$$

In the concrete test setup a PDF_s was derived for each style of the *GiantSteps* tempo dataset $s \in S$ except *dj-tools*. [5] In order to provide PDFs for the different styles of the *GiantSteps* dataset the first step was to create a mapping from Wikipedia genres to *GiantSteps* dataset styles. Since this step strongly depends on the dataset, it needs to be carried out manually. Given these mappings a BPM range $r_s = (min_s, max_s)$ is extracted for each style (the corresponding ranges are given in table 1). Also tempo relationships between the styles are considered. For the cases where the style s is not perceived to be slower than one of the other styles, PDF_s is defined analogous to the PDF of a normal distribution (see equation 2). With $\mu_s =$

[3] `http://en.wikipedia.org/wiki/Template:Infobox_music_genre`

[4] The syntax of regular expressions is conform to the Java Pattern class, for details see `https://docs.oracle.com/javase/7/docs/api/java/util/regex/Pattern.html`

[5] No corresponding Wikipedia article could be found. Since there is no PDF for the style *dj-tools*, the Tempo Ranker skips ranking and chooses the first estimate in such cases.

$\frac{min_s+max_s}{2}$, $\sigma_s = \frac{\mu_s - \min_s}{3}$ and $\sigma_s \geq 3$ the PDF is basically a normal distribution with its center and standard deviation defined by the values of the range.

$$PDF_s(x) = \frac{1}{\sigma_s\sqrt{2\pi}} \cdot e^{-\frac{1}{2}(\frac{x-\mu_s}{\sigma_s})^2} \quad (2)$$

For styles k that are (according to the Wikipedia tempo relationships) slower than another style $s \in S$, the PDF is modeled analogous to the PDF of a gamma distribution (see equation 3). With $\gamma = 3$ (the shape parameter), $beta_k = 0.4 - \lfloor \frac{(max_k - min_k)-15}{15} \rfloor 0.05$ (the decay parameter - this parameterization of β_k defines the PDF's decay wrt. the difference of min_k and max_k, i.e., the larger the difference, the lower the decay and therefore the PDF is smoother fading towards max_k) and $\mu_k = min_k - \frac{\gamma}{\beta_k} - \frac{max_k - min_k}{4}$ (the position parameter trying to position the PDF in a way that the PDF reflects the *slower* relationship). The resulting PDFs are visualized in figure 2.

$$PDF_k(x) = \frac{(\frac{x-\mu_k}{\beta_k})^{\gamma-1} exp(-\frac{x-\mu_k}{\beta_k})}{\beta_k \int_0^\infty t^{\gamma-1}e^{-t}dt} \quad (3)$$

4. EVALUATION

In this section we describe the conducted experiments, introduce the used dataset and discuss the results. All experiments were conducted using the *GiantSteps* tempo dataset. For every algorithm we provide accuracy1 and accuracy2 within a $\pm 4\%$ tolerance window. Accuracy1 considers an estimate to be correct if it is within $\pm 4\%$ of the true tempo. Accuracy2 also considers an estimate to be correct if it is within $\pm 4\%$ of either a third, half, double or triple of the true tempo.

4.1 Experiments

For the evaluation of our approach we conducted two experiments. In the first experiment we test the composition of style classification and tempo ranking, as visualized in figure 1 (for details see section 3). First the style estimation is carried out and ten tempo estimates are computed. Given those tempo and style estimates the Tempo Ranker chooses the most probable tempo based on the probability density function of the estimated style. This experiment therefore tests the impact of the overall composition of style estimation and ranking based on the data obtained from Wikipedia on tempo estimation accuracy. For simplicity this experiment is referred to as *wikidata-1*.
In the second experiment we test tempo ranking with respect to a known style, hence this shows how well the ranking itself performs given correct style assumptions. The style estimation step is skipped and the ranking algorithm is provided with the correct style. The experiment therefore tests the actual impact of the ranking procedure on the tempo estimates. For simplicity this experiment is referred to as *wikidata-2*.

We compare our results with tempo estimators, that are shipped with popular DJ tools. Namely *Cross DJ Free* [6], *Deckadance v2 (trail)* [7] and *Traktor 2 PRO*. [8] We argue that those estimators are tailored for electronic music and therefore should be able to perform well on the dataset.
Each of the products enables the user to choose some parameters for BPM prediction. *Deckadance* offers to choose among a predefined set of lower bounds, based on the ranges extracted from Wikipedia we decided to use 80 BPM. In the *Traktor* option pane the user can choose between a predefined set of tempo ranges, we decided to evaluate two ranges: 88-175 BPM (*TraktorA*) and 60-200 BPM (*TraktorB*). *CrossDJ* also provides a predefined set of tempo ranges, we chose 75-150 BPM for evaluation. In order to perform the evaluation we imported the audio files in the individual tools and analyzed them, the predicted values were later obtained from XML files (Deckadance, CrossDJ) or via ID3 tags that were instantiated during the analysis (Traktor).

4.2 The *GiantSteps* tempo dataset

The *GiantSteps* tempo dataset created in the course of the *GiantSteps* project [9] was obtained using the *Beatport*® website. [10] It contains tempo and stylistic ground truth for 664 samples. *Beatport*® is an online music store targeting producers and DJs of electronic music. For each track available on the website a preview sample can be downloaded. Furthermore, a variety of annotations for the tracks are provided – among them are *music style* presumably assigned by the composer out of the 23 maintained styles and the *tempo in BPM*. Since the BPM annotations might be calculated by an undisclosed algorithm they cannot be used as tempo ground truth data. However customers were encouraged to report false BPM annotations within a discussion forum. Typically users posted a reference to the track plus the correct BPM annotation. This discussion was crawled for comments containing a link to a track, the term BPM and a two or three digits long number. The association of a BPM annotation and a track was then conducted by putting this three pieces of information together. In cases where unambiguous information could be derived, the corresponding samples were downloaded. It can therefore be argued that this approach derives a human annotated dataset appropriate for evaluating tempo estimations. Unfortunately *Beatport*® recently abandoned this discussion forum.

As can be seen in table 3 this dataset has a strong bias towards the style *drum-and-bass*, 20% of the samples are within this style. This bias is most likely triggered by the fact that tempo estimation of drum-and-bass tracks is frequently affected by the tempo octave error, which implies many reports of incorrect tempi for this style.

4.3 Results and Discussion

Table 2 gives an overview of the obtained accuracy1 and accuracy2 values on the *GiantSteps* tempo dataset. In table

[6] http://www.mixvibes.com/products/cross

[7] http://www.image-line.com/deckadance/

[8] http://www.native-instruments.com/en/products/traktor/dj-software/traktor-pro-2/

[9] http://www.giantsteps-project.eu

[10] http://www.beatport.com

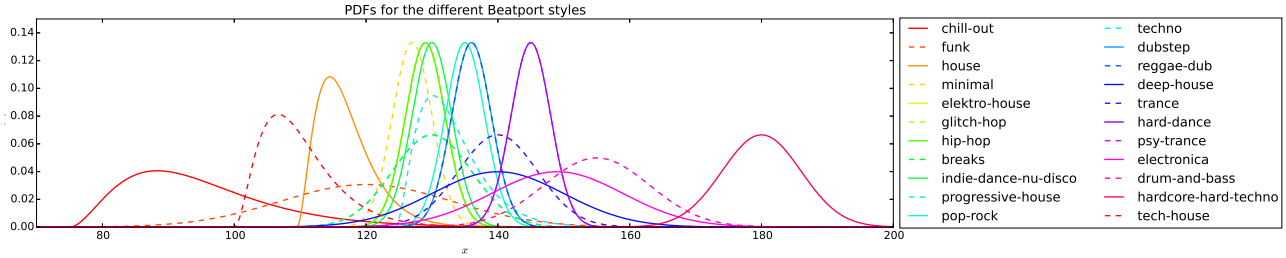

Figure 2. Probability density functions for the *GiantSteps* tempo dataset styles based on data extracted from Wikipedia articles.

	baseline	wikidata-1	wikidata-2	TraktorA	TraktorB	Deckadance	Cross DJ
accuracy1	45.48%	75.00%	72.89%	76.96%	64.61%	57.68%	63.40%
accuracy2	73.04%	82.83%	80.87%	88.71%	88.71%	81.63%	90.21%

Table 2. Tempo estimation accuracies for the different algorithms on the *GiantSteps* tempo dataset within a $\pm 4\%$ tolerance window. Apart from TraktorA (tempo range 88-175 BPM) the proposed approach clearly outperforms others and considerably improves the baseline performance.

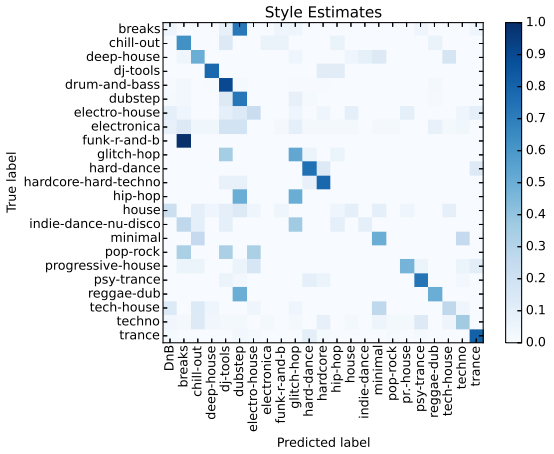

Figure 3. Confusion matrix for the style estimation task on the *GiantSteps* tempo dataset.

3 we provide detailed, per-style accuracies for the baseline, wikidata-1 and wikidata-2. Considering the accuracy1 values, which punish octave errors, it is apparent that the proper boundaries for the different styles help to increase tempo estimation accuracy. Compared to the performance of the baseline, we were able to increase the accuracy1 by 29 percentage points in scenario wikidata-1 and 27 percentage points in scenario wikidata-2. After having a detailed look on the per-style accuracy values in table 3 we noticed that especially for drum-and-bass, which makes up 20% of the dataset, we were able to considerably increase estimation accuracy (i.e. decrease the influence of the octave error). The baseline estimator only got 7.19 % right while the ranking boosts this value to 78.42 %. Overall we can report an increase of tempo estimation accuracy for most of the styles. A particularly interesting finding is, that despite the bad style classification performance wikidata-1 slightly outperforms wikidata-2, the tempo estimation accuracy still improves. Having a look at the confusion matrix in figure 3 reveals that styles which are hard to distinguish have similar tempo ranges (see table 1 and figure 2). This applies for instance to breaks and dubstep or reggae-dub and dubstep. Therefore is not too surprising that the ranking approach is able to chose a proper tempo estimate. For those styles (e.g. chill-out) for which the ranking decreases performance we assume that the extracted ranges do not properly represent the style. Except for the Traktor algorithm (TraktorA) our approach outperforms others in terms of accuracy1, we argue that despite reaching a higher accuracy the Traktor algorithm very much depends on the selected BMP range in order to reduce octave errors. Apart from that the algorithm is is highly tailored for tempo estimations of electronic music styles. In contrast to that our approach can (given proper input ranges for the styles of interest) be used for a wide range of music styles and does not enforce estimations within certain, predefined ranges. Note that we were not able to apply our approach on top of the audio-based tempo estimations given by Traktor, as our proposed method builds upon a tempo estimator that outputs multiple hypotheses. Also, it is not possible to set arbitrary tempo output ranges in Traktor, which would be another possibility of using external stylistic information.

In terms of accuracy2 other algorithms outperform our approach. Nevertheless it is apparent that wikidata-1 and wikidata-2 do only benefit by a small magnitude from the simpler task. While the other algorithms are able to increase their accuracy by between 12 and 27 percentage points our approaches only increase by about 8 percentage points. This means that there is only a small fraction of octave errors produced by the baseline that could not be corrected by the ranking procedure.

5. CONCLUSION

In this paper we have presented and evaluated a novel approach to further improve tempo estimation results of state-

style	#	baseline	wikidata-1	wikidata-2
drum-and-bass	139	7.19	78.42	83.45
dubstep	76	42.11	76.32	73.68
trance	74	75.68	97.30	98.65
techno	61	44.26	65.57	60.66
electronica	54	42.59	53.70	53.70
psy-trance	34	76.47	85.29	85.29
breaks	25	72.00	96.00	84.00
deep-house	24	75.00	75.00	83.33
house	23	47.83	65.22	73.91
tech-house	22	54.55	72.73	9.09
electro-house	22	63.64	77.27	68.18
progressive-house	19	57.89	89.47	94.74
glitch-hop	17	47.06	41.18	47.06
chill-out	16	62.50	43.75	37.50
hardcore-hard-techno	14	14.29	85.71	92.86
indie-dance-nu-disco	11	63.64	63.64	45.45
dj-tools	9	44.44	55.56	44.44
minimal	8	75.00	75.00	75.00
hard-dance	8	37.50	62.50	62.50
pop-rock	3	33.33	66.67	33.33
reggae-dub	2	0.00	0.00	0.00
hip-hop	2	50.00	50.00	50.00
funk-r-and-b	1	100.00	100.00	100.00
Weighted Average	664	45.33	74.85	72.74

Table 3. Primary tempo estimation accuracy (within 4% tolerance) of baseline estimator, wikidata-1 and wikidata-2 for the individual styles of the *GiantSteps* dataset.

of-the-art tempo induction algorithms. From the presented experiments we can see that using additional stylistic information of music can be beneficial for the task of tempo estimation. We were able to considerably reduce the amount of octave errors made by our baseline estimator and boost it's performance to be comparable with the performance of tempo estimators shipped with popular DJ tools. The facts that these (i) heavily depend on the predefined BPM output ranges / boundaries and (ii) do strictly enforce this parameters can be considered as drawbacks. Due to the use of probability density functions our approach does not come with this drawbacks. However, finding and assigning the right information in order to describe the styles is not trivial. To this end we proposed a strategy to extract tempo information from Wikipedia. The majority of tempo annotations derived from Wikipedia belong to electronic styles – this implies a limitation to the domain of electronic music. In our experiments, using information extracted from Wikipedia gives advantages over the uninformed baseline approach. The results we have obtained show the potential of tapping external and contextual information – unfortunately they are still rather inconsistent. The fact that some styles do not benefit from the extra knowledge suggests that we need to invest some more effort in the tempo range extraction strategy. Apart from that one could utilize more data sources or different tempo estimators. As mentioned before, the majority of the tempo annotations extracted from Wikipedia belong to electronic music styles, hence we were forced to carry out the experiments on an appropriate dataset. It is hard to predict how well the introduced method would perform on a dataset containing music from genres with wide tempo ranges like "classical," "metal," or "pop", cf. [25]. In these examples, tempo estimation would benefit only little if the chosen styles can not be linked to specific tempo ranges – which might be the case for e.g. "classical". On the other hand, if there are diverse subgenres with specific tempo ranges (consider "speed metal" vs. "rock ballad") the challenge is finding an appropriate set of styles and tempo ranges. For electronic music styles, where tempo can be one of the major factors that determine the style, we could show that the introduced tempo estimation approach improves results of state-of-the-art algorithms – mainly by preventing tempo octave errors.

6. ACKNOWLEDGEMENTS

The research leading to these results was performed in the *GiantSteps* project, which has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement no. 610591.

7. REFERENCES

[1] K. Seyerlehner, G. Widmer, and D. Schnitzer, "From rhythm patterns to perceived tempo," in *Proceedings of the 8th International Society for Music Information Retrieval Conference (ISMIR 2007)*, Vienna, Austria, Sept 2007.

[2] G. Peeters, "Template-based estimation of tempo: Using unsupervised or supervised learning to create better spectral templates," in *Proceedings of the 13th International Conference on Digital Audio Effects (DAFx-10)*, Graz, Austria, Sept 2010.

[3] A. Klapuri, A. Eronen, and J. Astola, "Analysis of the meter of acoustic musical signals," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 1, pp. 342–355, Jan 2006.

[4] A. Gkiokas, V. Katsouros, G. Carayannis, and T. Stafylakis, "Music tempo estimation and beat tracking by applying source separation and metrical relations," in *Proceedings of the 37th International Conference on Acoustics, Speech and Signal Processing (ICASSP 2012)*, Kyoto, Japan, Mar 2012, pp. 421–424.

[5] A. Elowsson, A. Friberg, G. Madison, and J. Paulin, "Modelling the speed of music using features from harmonic/percussive separated audio," in *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR 2013)*, Curitiba, Brazil, Nov 2013.

[6] A. Eronen and A. Klapuri, "Music tempo estimation with k-nn regression," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 1, pp. 50–57, Jan 2010.

[7] A. Gkiokas, V. Katsouros, and G. Carayannis, "Reducing tempo octave errors by periodicity vector coding and svm learning," in *Proceedings of the 13th International Society for Music Information Retrieval Conference (ISMIR 2012)*, Porto, Portugal, Oct 2012, pp. 301–306.

[8] J. Hockman and I. Fujinaga, "Fast vs slow: Learning tempo octaves from user data," in *Proceedings of the*

11th International Society for Music Information Retrieval Conference (ISMIR 2010), Utrecht, the Netherlands, 2010, pp. 231–236.

[9] M. Levy, "Improving perceptual tempo estimation with crowd-sourced annotations." in *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011)*, Miami, FL, USA, 2011, pp. 317–322.

[10] N. Collins, "Towards a style-specific basis for computational beat tracking," in *Proceedings of the 9th International Conference on Music Perception and Cognition (ICMPC9) and 6th Triennial Conference of the European Society for the Cognitive Sciences of Music (ESCOM)*, Bologna, Italy, 2006.

[11] C. Krumhansl, "Plink: "Thin Slices" of Music," *Music Perception: An Interdisciplinary Journal*, vol. 27, no. 5, pp. 337–354, June 2010.

[12] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano, "An experimental comparison of audio tempo induction algorithms," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 5, pp. 1832–1844, Sept 2006.

[13] J. Zapata and E. Gómez, "Comparative evaluation and combination of audio tempo estimation approaches," in *AES 42nd International Conference*, Ilmenau, Germany, July 2011.

[14] E. Scheirer, "Tempo and beat analysis of acoustic musical signals," *The Journal of the Acoustical Society of America*, vol. 103, no. 1, pp. 588–601, 1998.

[15] D. Gärtner, "Tempo detection of urban music using tatum grid non negative matrix factorization," in *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR 2013)*, Curitiba, Brazil, Nov 2013.

[16] D. Moelants and M. F. McKinney, "Tempo perception and musical content: What makes a piece fast, slow or temporally ambiguous?" in *In Proceedings of the 8th International Conference on Music Perception and Cognition (ICMPC8)*, Evanston, USA, August 2004.

[17] 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, pp. 135–139.

[18] K. Seyerlehner, M. Schedl, T. Pohle, and P. Knees, "Using block-level features for genre classification, tag classification and music similarity estimation," in *online Proceedings of the 6th Annual Music Information Retrieval Evaluation eXchange (MIREX-2010)*, Utrecht, the Netherlands, Aug 2010.

[19] K. Seyerlehner, G. Widmer, and T. Pohle, "Fusing block-level features for music similarity estimation," in *Proceedings of the 13th International Conference on Digital Audio Effects (DAFx-10)*, Graz, Austria, Sept 2010.

[20] T. Zesch, C. Müller, and I. Gurevych, "Extracting Lexical Semantic Knowledge from Wikipedia and Wiktionary," in *Proceedings of the Conference on Language Resources and Evaluation (LREC), electronic proceedings*. Ubiquitious Knowledge Processing, Universität Darmstadt, Mai 2008.

[21] H. Dohrn and D. Riehle, "Design and implementation of the Sweble Wikitext parser: unlocking the structured data of Wikipedia." in *Int. Sym. Wikis*, F. Ortega and A. Forte, Eds. ACM, 2011, pp. 72–81.

[22] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer, "DBpedia - A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia," *Semantic Web Journal*, 2014.

[23] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, 2009.

[24] T. Zesch and I. Gurevych, "Analysis of the Wikipedia Category Graph for NLP Applications," in *Proceedings of the Second Workshop on TextGraphs: Graph-Based Algorithms for Natural Language Processing*. Rochester, NY, USA: Association for Computational Linguistics, 2007, pp. 1–8.

[25] F. Pachet and D. Cazaly, "A Taxonomy of Musical Genre," in *Proceedings of Content-Based Multimedia Information Access (RIAO) Conference*, Paris, France, Apr 2000.