TEMPO ESTIMATION FOR MUSIC LOOPS AND A SIMPLE CONFIDENCE MEASURE

Frederic Font and Xavier Serra
Music Technology Group, Universitat Pompeu Fabra
frederic.font@upf.edu, xavier.serra@upf.edu

ABSTRACT

Tempo estimation is a common task within the music information retrieval community, but existing works are rarely evaluated with datasets of music loops and the algorithms are not tailored to this particular type of content. In addition to this, existing works on tempo estimation do not put an emphasis on providing a confidence value that indicates how reliable their tempo estimations are. In current music creation contexts, it is common for users to search for and use loops shared in online repositories. These loops are typically not produced by professionals and lack annotations. Hence, the existence of reliable tempo estimation algorithms becomes necessary to enhance the reusability of loops shared in such repositories. In this paper, we test six existing tempo estimation algorithms against four music loop datasets containing more than 35k loops. We also propose a simple and computationally cheap confidence measure that can be applied to any existing algorithm to estimate the reliability of their tempo predictions when applied to music loops. We analyse the accuracy of the algorithms in combination with our proposed confidence measure, and see that we can significantly improve the algorithms' performance when only considering music loops with high estimated confidence.

1. INTRODUCTION

Tempo estimation is a topic that has received considerable attention within the music information retrieval (MIR) community and has had a dedicated task in the Music Information Retrieval Evaluation eXchange (MIREX) since its first edition in 2005. Tempo estimation consists in the automatic determination of the "rate of musical beats in time" [10], that is to say, in the identification of the rate at which periodicities occur in the audio signal that convey a rhythmic sensation. Tempo is typically expressed in beats per minute (BPM), and is a fundamental property to characterise rhythm in music [13]. Applications of tempo estimation include, just to name a few, music recommendation, music remixing, music browsing, and beat-aware audio analysis and effects.

Our particular research is aimed at automatically annotating user provided music loops hosted in online sound sharing sites to enhance their potential reusability in music creation contexts. We can define music loops as short music fragments which can be repeated seamlessly to produce an "endless" stream of music. In this context, BPM is an important music property to annotate. The kind of music loops we are targeting can include noisy and low quality content, typically not created by professionals. This may increase the difficulty of the tempo estimation task. Taking that into consideration, it is particularly relevant for us to not only estimate the tempo of music loops, but to also quantify how reliable an estimation is (i.e., to provide a confidence measure). Except for the works described in [10, 14] (see below), tempo estimation has been rarely evaluated with datasets of music loops, and we are not aware of specific works describing algorithms that are specifically tailored to this particular case.

In this paper we evaluate the accuracy of six state of the art tempo estimation algorithms when used to annotate four different music loop datasets, and propose a simple and computationally cheap confidence measure that can be used in combination with any of the existing methods. The confidence measure we propose makes the assumption that the audio signal has a steady tempo thorough its whole duration. While this assumption can be safely made in the case of music loops, it does not necessarily hold for other types of music content such as music pieces. Hence, the applicability of the confidence measure we propose is restricted to music loops. Using our confidence measure in combination with existing tempo estimation algorithms, we can automatically annotate big datasets of music loops and reach accuracies above 90% when only considering content with high BPM estimation confidence. Such reliable annotations can allow music production systems to, for example, present relevant loops to users according to the BPM of a music composition, not only by showing loops with the same BPM but also by automatically transforming loops to match a target BPM. This effectively increases the reusability of user provided music loops in real-world music creation contexts.

The rest of the paper is organised as follows. In Sec. 2 we give a quick overview of related work about tempo estimation. In Sec. 3 we describe the confidence measure that we propose. Sections 4 and 5 describe the evaluation methodology and show the results of our work, respectively. We end this paper with some conclusions in Sec. 6.

In the interest of research reproducibility, the source code and one of the datasets used in this paper have been made available online in a public source code repository [1] .

2. RELATED WORK

A significant number of works within the MIR research field have been focused on the task of tempo estimation. In general, tempo estimation algorithms are based on detecting onsets in an audio signal, either as a continuous function [3, 14, 15] or as discrete events in time [5]. Then, a dominant period is extracted from the onsets either by analysing inter-onset intervals, using autocorrelation [11] or resonating filters [12]. Some approaches perform more complex operations such as analysing periodicities in different frequency bands [8, 19], performing source separation [6, 9], or using neural networks to learn features to use instead of usual onset information [1].

While comparative studies of tempo estimation algorithms have been carried out in the past [10, 21], we are not aware of any study solely devoted to the evaluation of tempo estimation algorithms for music loops. One of the typical datasets that some of the existing tempo estimation works use for evaluation is the ISMIR 2004 dataset released for the tempo induction contest of that year [2] . This dataset is divided into three subsets, one of them composed of 2k audio loops. Gouyon et. al. [10] published the evaluation results for the contest considering the different subsets of the dataset, but no significant differences are reported regarding the accuracies of the tempo estimation algorithms with the loops subset compared to the other subsets. To the best of our knowledge, the only other work that uses the loops subset of the ISMIR 2004 dataset and reports its accuracy separated form other datasets is by Oliveira et. al. [14]. The authors report lower estimation accuracies when evaluating with the loops dataset and attribute this to the fact that loops are typically shorter than the other audio signals (in many cases shorter than 5 seconds).

Surprisingly enough, there has not been much research on confidence measures for tempo estimation algorithms. Except for the work by Zapata et al. [22] in which a confidence measure that can be used for tempo estimation is described (see below), we are not aware of other works directly targeted at this issue. Among these few, Grosche and Müller [11] describe a confidence measure for their tempo estimation algorithm based on the amplitude of a predominant local pulse curve. By analysing tempo estimation accuracy and disregarding the regions of the analysis with bad confidence, the overall accuracy significantly increases. Alternatively, Percival and Tzanetakis [15] suggest that beat strength [18] can be used to derive confidence for tempo candidates, but no further experiments are carried out to asses its impact on the accuracy of tempo estimation. Finally, a very recent work by Quinton et. al. [16] proposes the use of rhythmogram entropy as a measure of reliability for a number of rhythm features, and report a statistical correlation between measured entropy and the resulting accuracies for different tasks.

[1] https://github.com/ffont/ismir2016

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

3. CONFIDENCE MEASURE

Assuming that we obtain a BPM estimate for a given audio signal, the confidence measure that we propose is based on comparing the duration of the whole audio signal with a multiple of the duration of a single beat according to the estimated BPM. If the actual duration of the signal is close to a multiple of the duration of a single beat, we hypothesise that the BPM estimation is reliable. The first thing we do to compute the confidence measure is to round the estimated tempo value to its nearest integer. The reasoning behind this is that it is very unlikely that loops are created with less than 1 BPM resolution tempo (see Sec. 4.1), and thus we consider the best BPM estimate of a tempo estimation algorithm to be its nearest integer. Given the sample rate SR of an audio signal and its estimated tempo BPM^e, we can estimate the duration (or length) of an individual beat in number of samples l^b as

$$l^b = \frac{60 \cdot SR}{BPM^e}.$$

Then, potential durations for the audio signal can be computed as multiples of the individual beat duration, $L[n] = n \cdot l^b$, where $n \in \mathbb{Z}^+$. In our computation, we restrict n to the range $1 \leq n \leq 128$. This is decided so that the range can include loops that last from only 1 beat to 128 beats, which would correspond to a maximum of 32 bars in 4/4 meter. In practice, what we need here is a number big enough such that we won't find loops longer than it. Given L, what we need to see at this point is if any of its elements closely matches the actual length of the original audio signal. To do that, we take the actual length of the audio signal l^a (in number of samples), compare it with all elements of L and keep the minimum difference found:

$$\Delta l = \min\{|L[n] - l^a| : n \leq 128\}.$$

A value of Δl near 0 means that there is a close match between one of the potential lengths and the actual length of the audio signal. Having computed Δl, we finally define our confidence measure as

$$confidence(L, l^a) = \begin{cases} 0 & \text{if } \Delta l > \lambda \\ 1 - \frac{\Delta l}{\lambda} & \text{otherwise} \end{cases},$$

where λ is a parameter set to half the duration of a single beat ($\lambda = 0.5 \cdot l^b$). In this way, if l^a exactly matches one of the multiples of l^b, the confidence will be 1. If Δl is as long as half the duration between beats, the confidence will be 0 (see Fig. 1, top).

The reasoning behind this simple confidence measure is that it is very unlikely that, only by chance, an audio signal has a duration which closely matches a multiple of the beat duration for a given estimated BPM. This means that we assume that there is a relation between the duration of the signal and its BPM, and therefore our proposed confidence will fail if the audio signal contains silence (either at

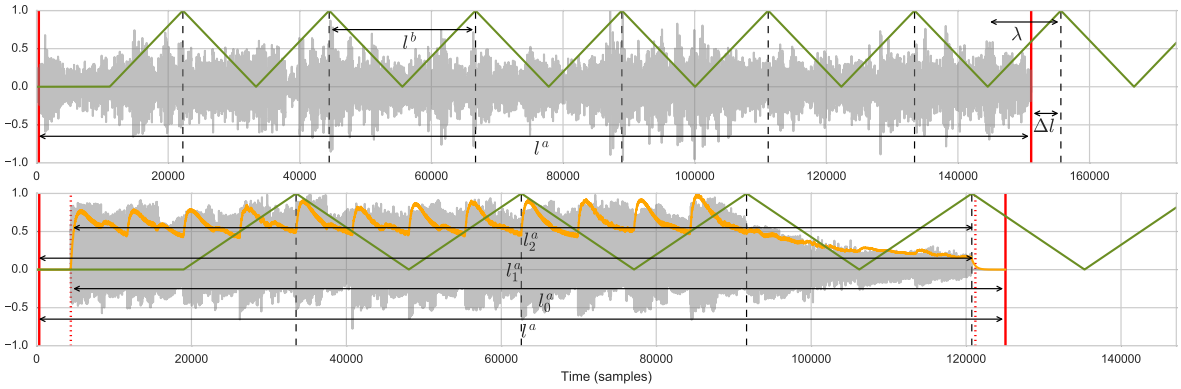

Figure 1. Visualisation of confidence computation output according to BPM estimation and signal duration (green curves). The top figure shows a loop whose annotated tempo is 140 BPM but the predicted tempo is 119 BPM. The duration of the signal l^a does not closely match any multiple of l^b (dashed vertical lines), and the output confidence is 0.59 (i.e., $1 - \frac{\Delta l}{\lambda}$). The figure at the bottom shows a loop that contains silence at the beginning and at the end, and for which tempo has been correctly estimated as being 91 BPM. The yellow curve represents its envelope and the vertical dashed red lines the estimated effective start and end points. Note that l_2^a closely matches a multiple of l^b, resulting in a confidence of 0.97. The output confidence computed with l^a, l_0^a and l_1^a produces lower values.

the beginning or at the end) which is not part of the loop itself (i.e., the loop is not *accurately cut*). To account for this potential problem, we estimate the duration that the audio signal would have if we removed silence at the beginning, at the end, or both at the beginning and at the end. We take the envelope of the original audio [3] and consider the effective starting point of the loop as being the point in time t_s where the envelope amplitude raises above 5% of the maximum. Similarly, we consider the effective end t_e at the *last point* where the envelope goes below the 5% of the maximum amplitude (or at the end of the audio signal if envelope is still above 5%). Taking t_s, t_e, and l^a (the original signal length), we can then compute three alternative estimates for the duration of the loop (l_0^a, l_1^a and l_2^a) by *i)* disregarding silence at the beginning ($l_0^a = l^a - t_s$), *ii)* disregarding silence at the end ($l_1^a = t_e$), and *iii)* disregarding silence both at the beginning and at the end ($l_2^a = t_e - t_s$). Then, we repeat the previously described confidence computation with the three extra duration estimates l_0^a, l_1^a and l_2^a. Note that these will produce meaningful results in cases where the original loop contains silence which is not relevant from a musical point of view, but they will not result in meaningful confidence values if the loop contains silence at the beginning or at the end which is in fact part of the loop (i.e., which is needed for it seamless repetition). Our final confidence value is taken as the maximum confidence obtained when using any of l^a, l_0^a, l_1^a and l_2^a estimated signal durations (see Fig. 1, bottom).

Because the confidence measure that we propose only relies on a BPM estimate and the duration of the audio signal, it can be used in combination with any existing tempo estimation algorithm. Also, it is computationally cheap to compute as the most complex operation it requires is the envelope computation. However, this confidence measure should not be applied to content other than music loops as it only produces meaningful results under the assumption that tempo is completely steady across the whole signal.

4. EVALUATION

4.1 Datasets

Our evaluation is conducted using 4 different datasets collected from different sources and containing a total of more than 35k loops. Table 1 shows basic statistics of each dataset. We now briefly describe each of the datasets:

- **FSL4**: This dataset contains user-contributed loops uploaded to Freesound [7]. It has been built in-house by searching Freesound for sounds with the query terms *loop* and *bpm*, and then automatically parsing the returned sound filenames, tags and textual descriptions to identify tempo annotations made by users. For example, a sound containing the tag *120bpm* is considered to have a ground truth of 120 BPM. Detailed instructions on how this dataset was created and on how can be reproduced are found in the source code repository (see Sec. 1).

- **APPL**: This dataset is composed of the audio loops bundled in Apple's Logic Pro [4] music production software. We parsed the metadata embedded in the audio files using source code available in a public repository [5], and extracted in this way tempo annotations for all the loops.

- **MIXL**: This dataset contains all the loops bundled with Acoustica's Mixcraft 7 music production software [6]. Tempo annotations are provided in its loop

[3] We use the Envelope algorithm from the open-source audio analysis library Essentia [2], which applies a non-symmetric lowpass filter and rectifies the signal.

[4] http://apple.com/logic-pro

[5] http://github.com/jhorology/apple-loops-meta-reader

[6] http://acoustica.com/mixcraft

Dataset	N instances	Total duration	Mean loop duration	Duration range	Tempo range	Source
FSL4	3,949	8h 22m	7.63s	0.15s - 30.00s	32 - 300	Freesound
APPL	4,611	9h 34m	7.47s	1.32s - 40.05s	53 - 140	Logic Pro
MIXL	5,451	14h 11m	9.37s	0.32s - 110.77s	55 - 220	Mixcraft 7
LOOP	21,226	50h 30m	8.57s	0.26s - 129.02s	40 - 300	Looperman

Table 1. Basic statistics about the datasets used for evaluation. Additional information and plots can be found in the paper's source code repository (see Sec. 1).

browser and can be easily exported into a machine-readable format.

- **LOOP**: This dataset is composed of loops downloaded from Looperman [7], an online loop sharing community. It was previously used for research purposes in [17]. Tempo annotations are available as metadata provided by the site.

Because of the nature of how the datasets were collected, we found that some of the loops do not have a BPM annotation that we can use as ground truth or have a BPM annotation which is outside what could be intuitively considered a reasonable tempo range. To avoid inconsistencies with the annotations, we clean the datasets by removing instances with no BPM annotation or with a BPM annotation outside a range of $[25, 300]$. Interestingly, we see that all the loops in our datasets are annotated with integer tempo values, meaning that it is not common for music loops to be produced with tempo values with less than 1 BPM resolution. For analysis purposes, all audio content from the dataset is converted to linear PCM mono signals with 44100 Hz sampling frequency and 16 bit resolution.

4.2 Tempo estimation algorithms

In our evaluation we compare six existing tempo estimation algorithms. These have been chosen based on their availability and to represent different approaches to the tempo estimation task. We now briefly describe each of the algorithms, further details on how the algorithms work can be found in corresponding papers.

- **Gkiokas12**: Gkiokas et. al. [9] propose a tempo estimation algorithm based on the separation of the audio signal into its percussive and harmonic components. Periodicity analysis is carried out by convolving extracted features (filterbank energies for the percussive component and chroma features for the harmonic component) with a bank of resonators. Output tempo value is computed by applying heuristics based on metrical relations knowledge (meter, tactus, tatum) to the periodicity vector. We use a Matlab implementation of the algorithm kindly provided to us by the authors.

- **Degara12**: Degara et. al. [4] describe a probabilistic approach for beat tracking based on inter-onset-interval times and a salience measure for individual beat estimates. This method builds from previous probabilistic beat tracking methods such as Klapuri et. al. [12]. We use the implementation provided in Essentia, where final estimated tempo is given based on the mean of estimated beat intervals (see RhythmExtractor2013 algorithm [8]).

- **Zapata14**: Zapata et. al. [20] propose a beat tracking algorithm which estimates beat positions based on computing the agreement of alternative outputs of a single model for beat tracking using different sets of input features (i.e., using a number of onset detection functions based on different audio features). Again, we use the implementation provided in Essentia, which outputs a single BPM estimate based on estimated beat intervals.

- **Percival14**: Percival and Tzanetakis [15] describe a tempo estimation algorithm optimised for low computational complexity that combines several ideas from existing tempo estimation algorithms and simplifies their steps. The algorithm computes an onset strength function based on filtered spectral flux from which tempo lag candidates are estimated using autocorrelation. The most prominent tempo lag is selected and a simple decision tree algorithm is used to chose the octave of the final BPM output. We use a Python implementation of the algorithm provided by the authors in their original paper [9].

- **Böck15**: Böck et. al. [1] propose a novel tempo estimation algorithm based on a recurrent neural network that learn an intermediate beat-level representation of the audio signal which is then feed to a bank of resonating comb filters to estimate the dominant tempo. This algorithm got the highest score in ISMIR 2015 Audio Tempo Estimation task. An implementation by the authors is included in the open-source Madmom audio signal processing library [10].

- **RekBox**: We also include an algorithm from a commercial DJ software, Rekordbox [11]. Details on how the algorithm works are not revealed, but a freely downloadable application is provided that can analyse a music collection and export the results in a machine-readable format.

[7] http://looperman.com

[8] http://essentia.upf.edu/documentation/algorithms_reference.html

[9] http://opihi.cs.uvic.ca/tempo

[10] http://github.com/CPJKU/madmom

[11] http://rekordbox.com

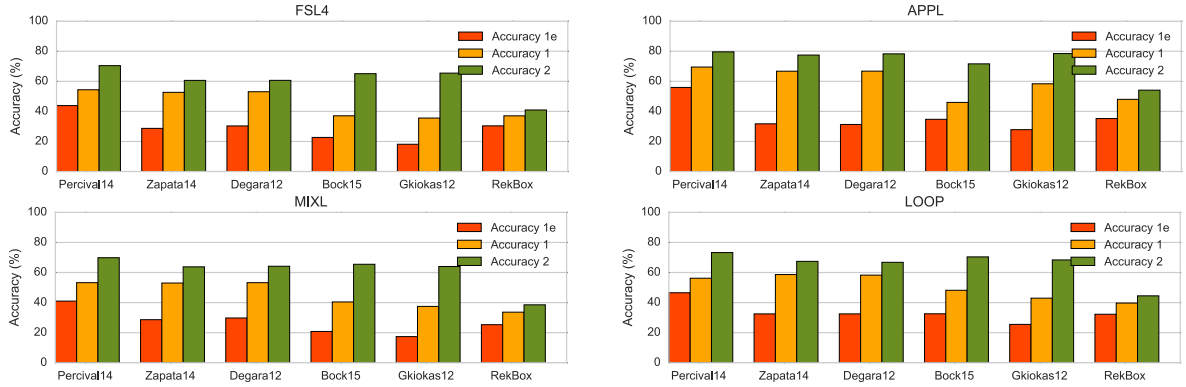

Figure 2. Overall accuracy for the six tempo estimation algorithms tested against the four datasets.

4.3 Methodology

For testing the above algorithms against the four collected datasets we follow standard practice and adopt the methodology described by Gouyon et al. [10]. In addition to the standard *Accuracy 1* and *Accuracy 2* measures [12], we add an extra measure that we call *Accuracy 1e* and that represents the percentage of instances whose estimated BPM is exactly the same as the ground truth after rounding the estimated BPM to the nearest integer. Accuracy 1e is therefore more strict than Accuracy 1. The reason why we added this extra accuracy measure is that, imagining a music creation context where loops can be queried in a database, it is of special relevance to get returned instances whose BPM exactly matches that specified as target.

Besides the overall accuracy measurements, we are also interested in observing how accuracy varies according to the confidence values that we estimate (Sec. 3). We can intuitively imagine that if we remove instances from our datasets whose estimated BPM confidence is below a certain threshold, the overall accuracy results will increase. However, the higher we set the minimum confidence threshold, the smaller the size of filtered dataset will be. Hence, we want to quantify the relation between the overall accuracy and the total number of music loops that remain in a dataset after filtering by minimum confidence. To do that, given one of the aforementioned accuracy measures, we can define a minimum confidence threshold γ and a function $A(\gamma)$ that represents overall BPM estimation accuracy when only evaluating loop instances whose estimated confidence value is above γ for a given dataset and tempo estimation algorithm. Similarly, we can define another function $N(\gamma)$ which returns the percentage of instances remaining in a dataset after filtering out those whose estimated confidence value (for a given tempo estimation method) is below γ. $A(\gamma)$ and $N(\gamma)$ can be understood as standard precision and recall curves, and therefore we can define a combined score measure $S(\gamma)$ doing the analogy with an f-measure computation:

$$S(\gamma) = 2 \cdot \frac{A(\gamma) \cdot N(\gamma)}{A(\gamma) + N(\gamma)}.$$

An overall score for a given dataset, tempo estimation algorithm and accuracy measure can thus be given by taking the mean of $S(\gamma)$, $\bar{S}$.

5. RESULTS

5.1 Overall accuracy

Overall accuracy results show that Percival14 obtains the highest accuracy scores for all accuracy measures and all datasets except for the LOOP dataset, in which highest score for Accuracy 1 is obtained by Zapata14 (Fig. 2). Considering the data from all datasets at once, mean accuracy values for Percival14 range from 47% (Accuracy 1e) to 73% (Accuracy 2), with an average increase of 7% accuracy when compared with the second best-scored method. With a few exceptions, pairwise accuracy differences between Percival14 and the second best-scored method in all datasets and accuracy measures are statistically significant using McNemar's test and a significance value of $\alpha = 0.01$ (i.e., $p \ll 0.01$). We also observe that accuracies for the APPL dataset tend to be higher than for other datasets. This can be explained by the fact that APPL contains professionally created and curated loops, while the other datasets contain user contributed content, not necessarily created by professionals (Mixcraft's loop library also contains content gathered from online repositories).

5.2 Accuracy vs confidence measure

Fig. 3 shows the accuracy of the three best-scoring tempo estimation algorithms and the number of instances remaining in the dataset when filtering by different values of a confidence threshold γ (Sec. 4.3). As we expected, we can see how accuracy increases with γ but the number of instances decreases. Interestingly, we observe that the number of instances decays later for estimations performed with Percival14 algorithm than for the other algorithms. This reflects the fact that Percival14 produces better BPM estimates. Filtering by the confidence measure, a potential

[12] Accuracy 1 is the percentage of instances whose estimated BPM is within 4% of the annotated ground truth. Accuracy 2 is the percentage of instances whose estimated BPM is within a 4% of $\frac{1}{3}$, $\frac{1}{2}$, 1, 2, or 3 times the ground truth BPM.

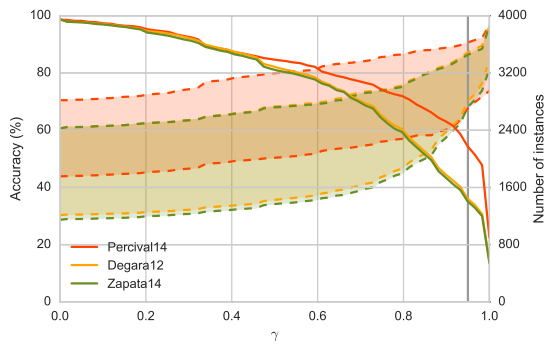

Figure 3. Accuracy vs confidence measure for FSL4 dataset. Lower bounds of the filled areas correspond to Accuracy 1e, while upper bounds correspond to Accuracy 2. Solid lines represent the number of instances remaining in the dataset.

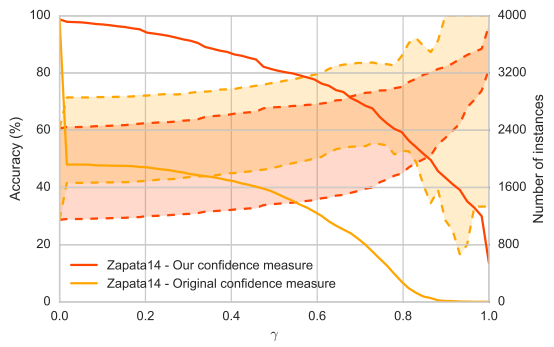

Figure 4. Comparison of our proposed confidence measure with the confidence measure proposed in [22] for FSL4 dataset and Zapata14 tempo estimation algorithm.

user searching for loops in a dataset could define a minimum threshold to get more accurate results at the expense of getting less loops returned. For instance, if we set a hard confidence threshold of $\gamma = 0.95$ (vertical line in Fig. 3), we find that the accuracies for Percival14 method range, on average, from 67% (Accuracy 1e) to 92% (Accuracy 2) while preserving an average of 52% of the instances. Interestingly enough, we observe that when setting that hard threshold, reported RekBox accuracies outperform these of Percival14 in all datasets, with an average increase ranging from 2% for Accuracy 2 to 14% for Accuracy 1e (all statistically significant with $p \ll 0.01$). We attribute this to the fact that RekBox seems to have a built-in confidence measure thresholding step in which the algorithm outputs 0 BPM when the analysis does not meet certain confidence requirements. Therefore, once filtering the datasets by γ (even with small values), all those instances whose BPM estimation is 0 BPM get discarded. Nevertheless, it is also important to note that filtering with the hard threshold, RekBox only preserves an average of 31% of the instances (lower than the 52% reported above by Percival14).

If we look at the combined accuracy and confidence measure $\bar{S}$ described in Sec. 4.3, we again find that Percival14 obtains the best score in all datasets and for all accuracy measures (i.e. for $A(\gamma)$ computed with Accuracy 1e, Accuracy 1 or Accuracy 2). This means that Percival14 offers the overall best balance between estimation accuracy and number of preserved instances in the dataset when filtering by a minimum confidence threshold.

5.3 Comparison of confidence measures

Zapata et. al. [22] propose a confidence measure that can be used for tempo estimation and that is based on computing the mutual agreement between an ensemble of tempo estimation algorithms. To make this confidence measure numerically comparable to the one we propose, we normalise the confidence output of Zapata et. al. to take values from 0 to 1. Similarly to Fig. 3, we plot the estimation accuracy and the number of remaining instances as a function of a minimum confidence threshold γ (Fig. 4). We observe that Zapata's confidence measure allows to achieve accuracies which are around 15% higher than when using our confidence. However, the number of remaining instances in the dataset is drastically reduced, and accuracy values for $\gamma > 0.75$ become inconsistent. Looking at the $\bar{S}$ score, we find that Zapata14 in combination with our confidence measure gets better results than when using the original measure, with an average $\bar{S}$ increase of 17%, 29% and 31% (for the three accuracy measures respectively). This indicates that our confidence measure is able to better maximise accuracy and number of remaining instances.

6. CONCLUSION

In this paper paper we have compared several tempo estimation algorithms using four datasets of music loops. We also described a simple confidence measure for tempo estimation algorithms and proposed a methodology for evaluating the relation between estimation accuracy and confidence measure. This methodology can also be applied to other MIR tasks, and we believe it encourages future research to put more emphasis on confidence measures. We found that by setting a high enough minimum confidence threshold, we can obtain reasonably high tempo estimation accuracies while preserving half of the instances in a dataset. However, these results are still far from being optimal: if we only consider exact BPM estimations (Accuracy 1e), the maximum accuracies we obtain are still generally lower than 70%. We foresee two complementary ways of improving these results by *i)* adapting tempo estimation algorithms to the case of music loops (e.g. taking better advantage of tempo steadiness and expected signal duration), and *ii)* developing more advanced confidence measures that take into account other properties of loops such as the beat strength or the rate of onsets. Overall, the work we present here contributes to the improvement of the reusability of unstructured music loop repositories.

7. ACKNOWLEDGMENTS

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688382.

8. REFERENCES

[1] Sebastian Böck, Florian Krebs, and Gerhard Widmer. Accurate Tempo Estimation based on Recurrent Neural Networks and Resonating Comb Filters. In *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, 2015.

[2] Dmitry Bogdanov, Nicolas Wack, Emilia Gómez, Sankalp Gulati, Perfecto Herrera, Oscar Mayor, Gerard Roma, Justin Salamon, Jose Zapata, and Xavier Serra. ESSENTIA: An audio analysis library for music information retrieval. In *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, pages 493–498, 2013.

[3] Matthew EP 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.

[4] Norberto Degara, Enrique Argones Rua, Antonio Pena, Soledad Torres-Guijarro, Matthew EP Davies, and Mark D Plumbley. Reliability-Informed Beat Tracking of Musical Signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):290–301, 2012.

[5] Simon Dixon. Automatic extraction of tempo and beat from expressive performances. *Journal of New Music Research*, 30:39–58, 2001.

[6] Anders Elowsson, Anders Friberg, Guy Madison, and Johan Paulin. Modelling the Speed of Music using Features from Harmonic/Percussive Separated Audio. In *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, pages 481–486, 2013.

[7] Frederic Font, Gerard Roma, and Xavier Serra. Freesound technical demo. In *Proc. of the ACM Int. Conf. on Multimedia (ACM MM)*, pages 411–412, 2013.

[8] Mikel Gainza and Eugene Coyle. Tempo detection using a hybrid multiband approach. *IEEE Transactions on Audio, Speech and Language Processing*, 19(1):57–68, 2011.

[9] Aggelos Gkiokas, Vassilis Katsouros, George Carayannis, and Themos Stafylakis. Music Tempo Estimation and Beat Tracking By Applying Source Separation and Metrical Relations. In *Proc. of the Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, volume 7, pages 421–424, 2012.

[10] Fabien Gouyon, Anssi Klapuri, Simon Dixon, Miguel Alonso, George Tzanetakis, Christian Uhle, and Pedro Cano. An Experimental Comparison of Audio Tempo Induction Algorithms. *IEEE Transactions on Audio, Speech and Language Processing*, 14(5):1832–1844, 2006.

[11] Peter Grosche and Meinard Müller. Extracting Predominant Local Pulse Information from Music Recordings. *IEEE Transactions on Audio, Speech and Language Processing*, 19(6):1688–1701, 2011.

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

[13] Meinard Müller. *Fundamentals of Music Processing*. Springer, 2015.

[14] João L Oliveira, Fabien Gouyon, Luis Gustavo Martins, and Luis Paolo Reis. IBT: A real-time tempo and beat tracking system. In *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, pages 291–296, 2010.

[15] Graham Percival and George Tzanetakis. Streamlined tempo estimation based on autocorrelation and cross-correlation with pulses. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(12):1765–1776, 2014.

[16] Elio Quinton, Mark Sandler, and Simon Dixon. Estimation of the Reliability of Multiple Rhythm Features Extraction from a Single Descriptor. In *Proc. of the Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 256–260, 2016.

[17] Gerard Roma. *Algorithms and representations for supporting online music creation with large-scale audio databases*. PhD thesis, Universitat Pompeu Fabra, 2015.

[18] George Tzanetakis, Georg Essl, and Perry Cook. Human Perception and Computer Extraction of Musical Beat Strength. In *Proc. of the Int. Conf. on Digital Audio Effects (DAFx)*, pages 257–261, 2002.

[19] Fu-Hai Frank Wu and Jyh-Shing Roger Jang. A Supervised Learning Method for Tempo Estimation of Musical Audio. In *Proc. of the Mediterranean Conf. on Control and Automation (MED)*, pages 599–604, 2014.

[20] Jose R Zapata, Matthew EP Davies, and Emilia Gómez. Multi-Feature Beat Tracking. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(4):816–825, 2014.

[21] Jose R Zapata and Emilia Gómez. Comparative Evaluation and Combination of Audio Tempo Estimation Approaches. In *Proc. of the AES Int. Conf. on Semantic Audio*, pages 198 – 207, 2011.

[22] Jose R Zapata, André Holzapfel, Matthew EP Davies, João L Oliveira, and Fabien Gouyon. Assigning a Confidence Threshold on Automatic Beat Annotation in Large Datasets. In *Proc. of the Int. Conf. on Music Information Retrieval (ISMIR)*, pages 157–162, 2012.