DOWNBEAT TRACKING WITH MULTIPLE FEATURES AND DEEP NEURAL NETWORKS

Simon Durand, Juan P. Bello†, Bertrand David*, Gaël Richard**

* Institut Mines-Telecom, Telecom ParisTech, CNRS-LTCI, 37/39, rue Dareau, 75014 PARIS − France
† Music and Audio Research Laboratory (MARL), New York University − USA

ABSTRACT

In this paper, we introduce a novel method for the automatic estimation of downbeat positions from music signals. Our system relies on the computation of musically inspired features capturing important aspects of music such as timbre, harmony, rhythmic patterns, or local similarities in both timbre and harmony. It then uses several independent deep neural networks to learn higher-level representations. The downbeat sequences are finally obtained thanks to a temporal decoding step based on the Viterbi algorithm. The comparative evaluation conducted on varied datasets demonstrates the efficiency and robustness across different music styles of our approach.

Index Terms— Downbeat Tracking, Music Information Retrieval, Music Signal Processing, Deep Networks

1. INTRODUCTION

Music is commonly organized through time as a function of pseudo-periodic pulses or *beats*. These beats can in turn be grouped into *bars*, depending on regular patterns of timing and accentuation which altogether define the music's metrical structure. The first beat of a given bar, is known as the *downbeat*. The automatic estimation of downbeat positions from music signals, a task known as downbeat tracking, is thus a fundamental problem in music signal processing with applications in automatic music transcription, music structure analysis, computational musicology, and computer music. The level of current interest in this problem amongst the community is illustrated by its recent inclusion in the MIReX evaluation initiative.

Yet, despite significant research effort, downbeat tracking remains an open and challenging problem. A number of previous approaches are limited in their scope: either they depend on hand-annotated beat positions, which is not readily available for most music recordings [1, 2]; or are applicable to only a few simple metrical structures [3,4] and given musical styles [5,6]. Most of these systems seek to characterize downbeats as a function of single attributes such as chord/harmonic changes [7,8], rhythmic patterns [9,10], or the explicit presence of drums and other percussive sounds [11, 12]. Only a few systems consider two or more features in tandem [6, 13–15], usually in the form of standard spectral or loudness features. Furthermore, the likelihood of downbeats is often estimated directly from low-level features, without further refining into higher-level representations. When this is not the case, as in [16, 17], the estimations depend on prior-decision-making, e.g. chord classification, which can be prone to errors. Finally, almost all past approaches use probabilistic dynamic models to exploit the sequential structure of music, generally resulting in a more robust estimation [4–11, 15, 16].

Based on our understanding of the strengths and shortcomings of past approaches, in this paper we propose a novel downbeat tracking method that:

- Independently analyzes downbeat occurrences using six different feature representations of harmony, timbre, low-frequency content, rhythmic patterns, and local similarities in both timbre and harmony.
- Uses deep neural networks (DNN) to learn higher-level representations from which the likelihood of downbeats can be robustly estimated.
- Implements a simple, yet-powerful model that leverages knowledge of metrical structure to decode the correct sequence of beats and downbeats.

The resulting method is fully automated (needs no prior information), and applicable to musical recordings covering a wide range of styles and metrical structures. It significantly extends our previous work in [2], which depends on knowledge of hand-annotated beat positions, uses a smaller and less robust set of features, and relies on heuristics for downbeat classification. To the best of our knowledge, this is the first use of deep networks for downbeat tracking in polyphonic music.

Figure 1 serves as an overview of the system and the structure of this paper. Then, section 2 presents our feature extraction strategies, including the synchronization of feature sequences to a grid of musical pulses. Section 3 discusses the use of DNN for feature learning and for assigning each pulse a probability of being a downbeat. Feature-specific estimations are aggregated before passing to the Viterbi algorithm in section 4, which can robustly classify downbeat positions. Section 5 shows, via an evaluation on several public datasets, the relative benefit of our main design choices, and how our system clearly outperforms the current state of the art. Finally, section 6 presents our conclusions and ideas for future work.

2. FEATURE EXTRACTION

The first task of our system is to represent the signal as a function of six musical attributes contributing to the grouping of beats into a bar, namely harmony, timbre, low-frequency content, rhythmic pattern, and local similarity in timbre and harmony. This multi-faceted approach is consistent with well-known theories of music organization [18], where the attributes we chose contribute to the perception of downbeats. Change in harmony or timbre content, for example chord changes, section changes or the entrance of a new instrument is often related to a downbeat position. The low-frequency content contains mostly bass instruments or bass drum, both of which tend to be used to emphasize the downbeat. Rhythmic patterns are frequently repeated each bar and are therefore useful to obtain the bar boundaries. Finally, by looking at timbre or harmony content in term of similarity, we can observe longer-term patterns of change and novelty that are invariant to the specific set of pitch values or spectral shape. The similarity in harmony, for example, has the interesting property of being key invariant and can therefore model

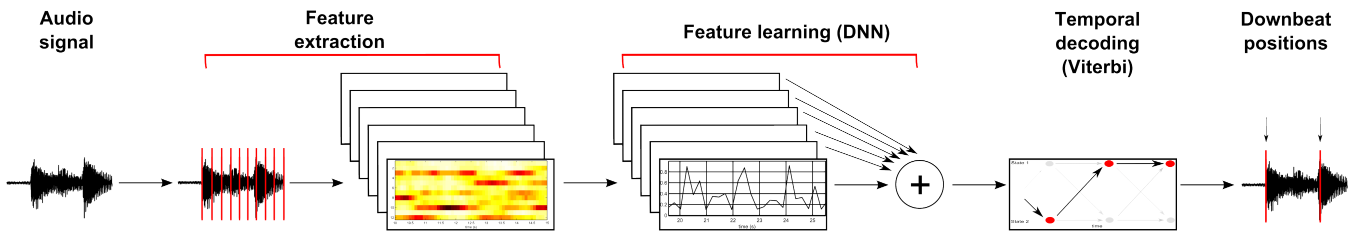

Fig. 1. Model overview. The signal's timeline is quantized to a set of downbeat subdivisions herein called *pulses*. Six different features are extracted from the signal and mapped to this temporal quantization: chroma, MFCC, ODF, LFS, CS and MS. Each feature is then used as input to an independent DNN which gives each pulse the probability of being a downbeat after averaging the output of all 6 networks. Finally, a Viterbi decoder is used to estimate the most likely downbeat sequence.

cadences and other harmonic patterns related to downbeat positions. These attributes will be represented by chroma, mel-frequency cepstral coefficients (MFCC), low-frequency spectrogram (LFS), onset detection function (ODF), MFCC similarity (MS) and chroma similarity (CS) features respectively. To make this process invariant to local and global tempo changes, we segment the signal into small subdivisions of musical time and synchronize features accordingly. More details are provided in this section.

Segmentation: To obtain an appropriate temporal quantization of the audio signal, we need to have a high recall representation, including the vast majority of downbeat pulses. We also want to avoid variations in the inter-pulses duration when it is possible. Finally, it is useful to be tempo independent.

To achieve these objectives we extend the local pulse information extractor in [19]. We first use this technique to obtain a tempogram of the musical audio. We then use dynamic programming with strong continuity constraints, and emphasis towards high tempi. It can therefore track abrupt tempo changes and find a fast subdivision of the downbeat at a pulse rate that is locally regular. We finally use the decoded path to recover instantaneous phase and amplitude values, construct the predominant local pulse (PLP) function as in [19], and detect pulses using peak-picking. The recall rate for downbeat pulses is above 95% for each dataset with this method, using a 100 ms tolerance window.

The chroma, MFCC, LFS and ODF, described below, are first computed frame by frame. They are then mapped to a grid with subdivisions lasting one fifth of a pulse using interpolation. The CS and the MS features are computed pulse by pulse because we believe a higher temporal precision than the pulse level is not useful here.

Chroma: The chroma computation is done as in [20]. We down-sample the audio signal at 5512.5 Hz and use a Hanning analysis window of size 4096 and a hop size of 512 to compute the short-term Fourier transform (STFT). We then apply a constant-Q filter-bank with 36 bins per octave and 108 bins and then convert the constant-Q spectrum to harmonic pitch class profiles. Circular shifting is done to obtain the 36 bins chroma. The chromagram is tuned by finding bias on peak locations and is smoothed by a median filter of length 8. It is finally mapped to a 12 bins representation by averaging.

MFCC: We compute the first 12 Mel-frequency cepstral coefficients using the Voicebox Toolbox [21], with a Hamming window of size 2048, a hop size of 1024 and 32 Mel filters on a signal sampled at 44100 Hz.

LFS: We down-sample the signal at 500 Hz, use a Hanning window of size 32 and a hop size of 4 to compute the STFT and the spectrogram. We then remove spectral components above 150 Hz. The signal is finally clipped so that all values on the 9^{th} decile are equal.

ODF: We use four band-wise onset detection functions (ODF) as computed by [4]. We compute the STFT using a Hanning window of size 1024 and a hop size of 256 for a signal sampled at 44100 Hz. We then compute the spectrogram and apply a 36-bands Bark filter. We use μ-law compression, with μ=100, and down-sample the signal by a factor of two. An order 6 butterworth filter with a 10 Hz cutoff is used for envelope detection. A weighted sum of 20% of the envelope and 80% of its difference is done to compute the ODF that are then mapped to 4 equally distributed bands.

CS and MS: The chroma are computed as before, but they are then averaged to obtain a pulse synchronous chroma. We then compute the cosine similarity of the pulse synchronous chroma. The same process is used with the MFCCs to get our last feature.

3. FEATURE LEARNING

Downbeats are high-level constructs depending on complex patterns in the feature sequence. We propose that the probability of a downbeat can be estimated using a DNN $F(X_1|\Theta)$, where X_1 is the input feature vector, and Θ are the parameters of the network. In our implementation, F is a cascade of K simpler layer functions of the form:

$$f_k(X_k|\theta_k) = \frac{1}{1+e^{-(X_k W_k + b_k)}},\ \theta_k = [W_k, b_k] \qquad (1)$$

where W_k is a matrix of weights, b_k is a vector of biases, and X_k is the output of layer $k-1$ for $k > 1$, and the input feature vector for $k = 1$. Furthermore, we apply softmax regularization to the output of the last layer, thus resulting in:

$$P(X_1|\Theta) = \frac{e^{f_K}}{\sum_{m=1}^{M} e^{f_K[m]}} \qquad (2)$$

The dimensionality of the output layer, M, corresponds to the number of classes we want to detect, in this case two: downbeat and no-downbeat. Thus the network's output represent the conditional probability of a one pulse of being a downbeat and its complement, while the output of intermediate layers can be seen as feature detectors. In our implementation we use a DNN composed of four fully connected layers ($K = 4$) of 40, 40, 50 and 2 sigmoid units respectively [22]. The network is pre-trained with stacked Restricted Boltzmann Machines and 1-step Contrastive Divergence [23]. The fine tuning is done with backpropagation by minimizing the cross entropy error using mini-batch gradient descent. The pre-training learning rate is 0.1 and the fine tuning learning rate is 1. The mini-batch size is 1000 and we use momentum. For the first 5 epochs it is of 0.5 and then it is 0.9. Our training set contains an equal amount of

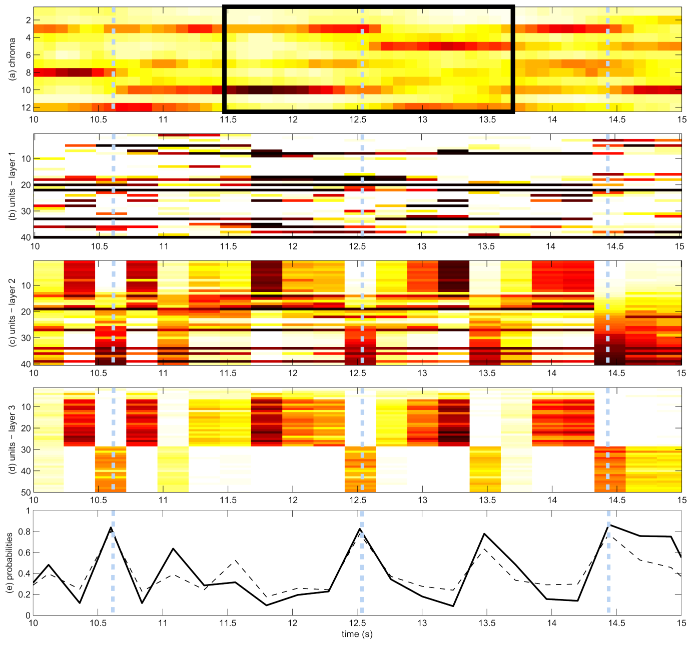

Fig. 2. Chroma feature representation through the network. (a): The chroma feature. The black rectangle represents the temporal context for one pulse (at 12.5 sec). (b-c-d): Units of layers 1, 2 and 3. (e). Output of the last layer, for the chroma feature (continuous bold curve) and all the features (dotted curve). The annotated downbeats are represented by the light-blue dotted lines.

features computed at downbeat and non-downbeat position. In our implementation we use early-stopping and dropout regularization, in order to prevent overfitting and feature co-adaptation [24]. We use 6 independent networks, each trained on one feature used as input. The input temporal size is of 9 pulses around the one to classify for the chroma, ODF, LFS and MFCC features and of 12 pulses before and 12 pulses after the one to classify for both the MS and the CS features.

As it is important for the following decoding process to reduce estimation errors, we use an average of the 6 observation probabilities obtained by the 6 independent networks. The average, or sum rule is indeed in general quite resilient to estimation errors [25].

Figure 2 illustrates the ability of DNN to learn powerful features and produce a robust downbeat estimation. The sequence of chroma feature vectors as well as the output of each layer of the network are displayed for five seconds of audio. The chroma representation, figure 2.(a), is clearly refined into a set of downbeat detectors as we advance trough the layers of the network. In the third layer, figure 2.(d), we can observe that there is a clear distinction between outputs activated at downbeat positions and outputs activated at non-downbeat positions. The downbeat positions are indicated by dotted light-blue lines. The strength of the resulting features allows the last layer of the network to robustly estimate downbeat probabilities by means of a simple regression. Finally, the output of the chroma-specific network is averaged with that of the other five networks, resulting in the dotted curve in figure 2.(e). It can be seen how this aggregation generally de-emphasizes the probability of false positives, while maintaining the high probability of correct estimations, thus increasing the robustness of the representation, and facilitating the decoding process in the next section.

4. TEMPORAL DECODING

As previously done in the literature, we use the Viterbi algorithm to decode the sequence of downbeats. This algorithm is mostly a function of four parameters: the state space, the probability of an observation given a certain state and the initial and transition probabilities. We use an equal distribution of initial probabilities; there are two distinct states: *downbeat* and *non-downbeat*; and the probability of observations given those states are the aggregated outputs of the DNN, as explained in section 3. The main focus is then the transition matrix which encodes the temporal model we use. We attempt to take into account that changes in time signature are possible albeit unlikely and that there can be downbeat observation errors or pulse tracking inconsistencies.

In our model, states correspond to downbeats and non-downbeats in a specific metrical position. For example, the downbeat in 4/4 and in 5/4 correspond to different states. Likewise, the first non-downbeat in 3/4 is different from its second non-downbeat, and different to any other non-downbeat in a different meter.

Then we assign high transition probabilities to moving sequentially across beats of the same meter, medium probabilities to moving from the last beat of a given meter to the downbeat of another, and low probabilities for non-consecutive changes within and without meter groups.

We allow time signatures of {2,3,4,5,6,7,8,9,10} beats per bar. Since there are mainly 3 or 4 beats per bar in most datasets, we assign a higher transition probability to moving from the last beat of a given meter to the downbeat of these two meters.

5. EVALUATION AND RESULTS

5.1. Methodology

We use the F-measure, the most used evaluation metric for downbeat tracking [8,9,16] computed from the evaluation toolbox in [26]. This measure is the harmonic mean of precision and recall rates. Correct detections occur when estimated downbeat positions fall within a tolerance window of +- 70 ms from an annotated downbeat position.

We evaluate our system on nine different datasets, presented in table 1. In our evaluation we use a leave-one-dataset-out approach, whereby in each of 9 iterations we use 8 datasets for training and validation, and the holdout dataset for testing. This evaluation method is considered more robust [27]. 90% of the training datasets is used for training the network and the 10% is used to set the parameters value.

5.2. Results and discussion

In our tests, we start by evaluating different configurations of the proposed system, in order to asses the effectiveness of our feature extraction, decoding and feature learning strategies. In figure 3 and throughout the discussion, these configurations are numbered to facilitate reference.

Is it important to use several features? To focus on the effect of feature design, we ran a simplified version of our system without Viterbi decoding. Instead we perform simple peak picking on the aggregated output of the networks. For this experiment we add one feature at a time. The order of features added, described in figure 3(a), is not of crucial importance to assess if they have a significant impact on the performance. The F-measure results, evaluated on the whole data, are shown as the first 6 boxplots in figure 3(b). We can see a consistent increase in the F-measure as we add features. We

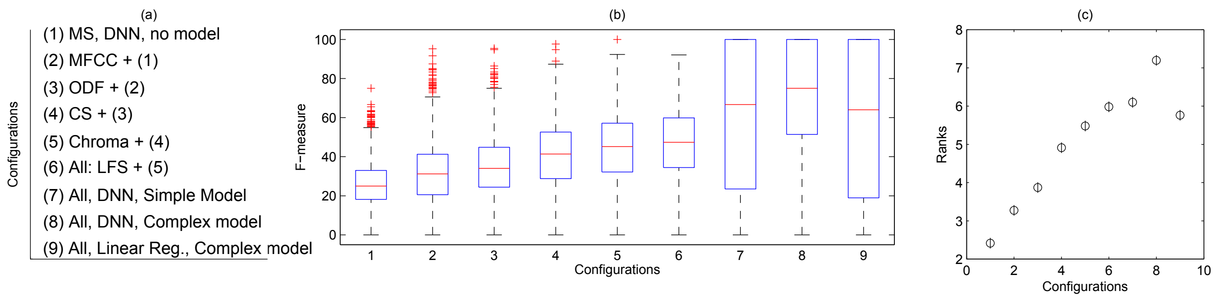

Fig. 3. Model comparison. (a): Description of the 9 compared configurations. (b): F-measure boxplots of the configurations. (c): Tukey's HSD of the configurations. Higher ranks correspond to higher results.

performed a Friedman's test and a Tukey's honestly significant criterion (HSD) test with a 95% confidence interval. As shown in the first 6 configurations of figure 3(c), the increase in performance is significant in each case. Additional Student T-tests were performed also indicating statistical significance for each feature added, illustrating the importance of each feature and their complementarity. Adding all features results in a staggering average increase over 18 points in F-measure, when compared to any individual feature, and of over 9 points when compared to any combination of 2 features.

Is temporal modelling useful? For this experiment, we use all features with peak picking (configuration 6), with the Viterbi algorithm and a simple model of only one possible time signature of 8 pulses per bar (configuration 7), and with the Viterbi algorithm and the more complex assumptions detailed in section 4 (configuration 8). Figure 3(b) shows that the simple temporal assumption in configuration 7 gives an important boost in performance but more complex assumptions give both an increase in average performance and a decrease in variance. Figure 3(c) shows that this improvement is statistically significant. This clearly indicates the importance of temporal modelling for downbeat tracking, whereby estimations including information from the local context of a downbeat outperform the instantaneous estimations of configuration 6.

Is feature learning really necessary or helpful? For this experiment, we keep all the features and the Viterbi algorithm and compare feature learning by the DNN (configuration 8) with a linear-regression method (configuration 9). Using DNN enable a twelve points increase in performance, which is statistically significant. This illustrates the power of learning higher-level feature representations in a data-driven fashion. The shallow architecture is less able to classify the pulse and some of the (perceptively) correct results (100% or 66,7%) tend to move towards phase error or inconsistent segmentations.

We then compare our system to 3 reference downbeat tracking algorithms [1][1], [15] and [16]. These systems do not require cross-validation. Results are shown in table 1. We can see a significant increase in performance with our method across all datasets. The difference in the overall result is statistically significant. [1] seems efficient but may be held back by the hypothesis of constant time signature that can propagate beat estimation errors easily. [16] is able to deal with change in time signature with a tradeoff coefficient between flexible and constant meter, but it also means that sometimes the results can be a little inconsistent with constant meter songs. Incorporating more cues may improve performance. Finally, [15] performs a global estimation of the meter of the song with an efficient visualisation of the output but may be best used by manually adjusting the parameter values while looking at the estimated downbeats.

[1]Since it needs the meter, we feed it with the annotated number of beats per bar

Datasets		F-measure				
Reference	NT	[15]	[1]	[16]	#8	Mean
RWC Class [28]	60	29.9	21.6	32.7	45.4	32.4
Klapuri [4]	40	47.3	41.8	41.0	61.8	48.0
Hainsworth [29]	222	42.3	47.5	44.2	59.4	48.4
RWC Jazz [28]	50	39.6	47.2	42.1	65.4	48.6
RWC Genre [30]	92	43.2	50.4	49.3	62.2	51.3
Ballroom [31]	698	45.2	50.3	50.0	73.6	54.8
Quaero [32]	70	56.9	69.1	69.3	76.5	68.0
Beatles [33]	179	65.3	66.1	65.3	82.2	69.8
RWC Pop [28]	100	69.8	71.0	75.8	81.3	74.2
Mean		**48.7**	**51.7**	**52.2**	**67.5**	55.0

Table 1. Downbeat detection results. F-measure with a 70 ms precision window. For [15], [1], [16] and our system (#8 for configuration 8 in figure 3). Results are shown per dataset and as a mean across datasets or algorithms. NT stands for the number of tracks per dataset.

We can see that there are relatively less increase in F-measure in the Popular music datasets (Pop and Quaero), about 10 points, because a simple feature model can already give good results. But when we face more complicated datasets, where there are less clues, more change in time signature, soft onsets or where there is not always percussion, such the as Classical, Jazz or Klapuri subset datasets, the increase is bigger, about 19 points. The biggest boost is obtained with the Ballroom dataset, about 25 points, because in this music the rhythm part, along with the timbre content, give very important cues for the downbeat estimation while the harmonic part is a little less informative than in the other datasets. The results overall are relatively lower for the classical music dataset and for songs where there are expressive timings. In this case it is difficult to distinguish clear and objective bar boundaries, and even to robustly estimate pulses. To illustrate this point, we evaluated our system on the RWC Classical dataset with the ground truth beats as input and the F-measure is considerably higher at around 85%.

6. CONCLUSION

In the present work, we have proposed a new approach to downbeat tracking. It is shown that deep neural networks trained on multiple musically inspired features can take advantage of the multi-faceted and high-level aspect of downbeats. The comparative evaluations over a large number of musical audio files have demonstrated that this novel paradigm significantly outperforms the previous state of the art. Future work will consider more sophisticated temporal models such as Conditional Random Fields to incorporate temporal context more freely and automatically infer an optimal combination of features.

7. REFERENCES

[1] M. E. P Davies and M. D. Plumbley, "A spectral difference approach to extracting downbeats in musical audio," in *Proceedings of the European Signal Processing Conference (EUSIPCO)*, 2006.

[2] S. Durand, B. David, and G. Richard, "Enhancing downbeat detection when facing different music styles," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 3132–3136.

[3] D. Gärtner, "Unsupervised learning of the downbeat in drum patterns," in *53rd International Conference on Semantic Audio (AES)*, 2014.

[4] 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, 2006.

[5] F. Krebs and S. Böck, "Rhythmic pattern modeling for beat and downbeat tracking in musical audio," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2013, pp. 227–232.

[6] J. Hockman, M. E. P. Davies, and I. Fujinaga, "One in the jungle: downbeat detection in hardcore, jungle, and drum and bass.," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2012, pp. 169–174.

[7] J. Weil, T. Sikora, J. L. Durrieu, and G. Richard, "Automatic generation of lead sheets from polyphonic music signals.," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2009, pp. 603–608.

[8] M. Khadkevich, T. Fillon, G. Richard, and M. Omologo, "A probabilistic approach to simultaneous extraction of beats and downbeats," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 445–448.

[9] F. Krebs, F. Korzeniowski, M. Grachten, and G. Wildmer, "Unsupervised learning and refinement of rhythmic patterns for beat and downbeat tracking," in *Proceedings of the European Signal Processing Conference (EUSIPCO)*, 2014.

[10] N. Whiteley, A. T. Cemgil, and S. J. Godsill, "Bayesian modelling of temporal structure in musical audio," in *Proceedings of International Conference on Music Information Retrieval (ISMIR)*, 2006, pp. 29–34.

[11] E. Batternberg, *Techniques for machine understanding of live drum performances*, Ph.D. thesis, Electrical Engineering and Computer Sciences University of California at Berkeley, 2012.

[12] D. Ellis and J. Arroyo, "Eigenrhythms: Drum pattern basis sets for classification and generation," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2004, pp. 101–106.

[13] T. Jehan, "Downbeat prediction by listening and learning," in *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2005, pp. 267–270.

[14] M. Goto, "An audio-based real-time beat tracking system for music with or without drum-sounds," *Journal of New Music Research*, vol. 30, no. 2, pp. 159–171, 2001.

[15] G. Peeters and H. Papadopoulos, "Simultaneous beat and downbeat-tracking using a probabilistic framework: Theory and large-scale evaluation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 6, 2011.

[16] H. Papadopoulos and G. Peeters, "Joint estimation of chords and downbeats from an audio signal," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 19, no. 1, pp. 138–152, 2011.

[17] T. Cho, *Improved techniques for automatic chord recognition from music audio signals*, Ph.D. thesis, New York University, 2013.

[18] F. Lerdahl and R. Jackendoff, *A generative theory of tonal music*, Cambridge, MA: The MIT Press, 1983.

[19] P. Grosche and M. Muller, "Extracting predominant local pulse information from music recordings," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 6, pp. 1688–1701, 2011.

[20] J. P. Bello and J. Pickens, "A robust mid-level representation for harmonic content in music signals.," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2005, vol. 41, pp. 304–311.

[21] "http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html," .

[22] G. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[23] M. A. Carreira-Perpinan and G. E. Hinton, "On contrastive divergence learning," in *Proceedings of the tenth international workshop on artificial intelligence and statistics*, 2005, pp. 33–40.

[24] G. Hinton, N. Srivastava, A. Krizhevsky, I. Suskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *The Computing Research Repository (CoRR)*, vol. abs/1207.0580, 2012.

[25] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 226–239, 1998.

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

[27] A Livshin and X. Rodet, "The importance of cross database evaluation in sound classification," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2003, pp. 241–242.

[28] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "Rwc music database: Popular, classical and jazz music databases," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2002, vol. 2, pp. 287–288.

[29] S. Hainsworth and M. D. Macleod, "Particle filtering applied to musical tempo tracking," *EURASIP Journal on Applied Signal Processing*, vol. 2004, pp. 2385–2395, 2004.

[30] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "Rwc music database: Music genre database and musical instrument sound database," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2003, vol. 3, pp. 229–230.

[31] "www.ballroomdancers.com," .

[32] "http://www.quaero.org/," .

[33] "http://isophonics.net/datasets," .