Robust Downbeat Tracking Using an Ensemble of Convolutional Networks

Simon Durand, Juan Pablo Bello, Bertrand David, and Gaël Richard

***Abstract*—In this paper, we present a novel state of the art system for automatic downbeat tracking from music signals. The audio signal is first segmented in frames which are synchronized at the tatum level of the music. We then extract different kind of features based on harmony, melody, rhythm and bass content to feed convolutional neural networks that are adapted to take advantage of the characteristics of each feature. This ensemble of neural networks is combined to obtain one downbeat likelihood per tatum. The downbeat sequence is finally decoded with a flexible and efficient temporal model which takes advantage of the assumed metrical continuity of a song. We then perform an evaluation of our system on a large base of 9 datasets, compare its performance to 4 other published algorithms and obtain a significant increase of 16.8 percent points compared to the second best system, for altogether a moderate cost in test and training. The influence of each step of the method is studied to show its strengths and shortcomings.**

***Index Terms*—Downbeat tracking, Convolutional Neural Networks, Music Information Retrieval, Music Signal Processing.**

I. Introduction

The time structure of a music piece is often conceived as a superposition of multiple hierarchical levels, or time-scales, interacting with one another [1]. Automatically detecting those different temporal layers is thus of significant importance for music analysis and understanding. When listening to a song, most people naturally synchronize to a specific level called the tactus or beat level [2]. Depending on the duration, the loudness, the pitch of the events or even on the local prosody of the musical phrase, these beats are differently accented. It eventually leads to a grouping over a larger scale. This scale is, at least in the western music context, that of the bar or measure which thus defines the metrical structure of the piece. The purpose of this work is to automatically estimate the locations of the first beat of each bar, the so-called downbeat, with the help of multiple features and deep neural networks especially designed for the task.

Downbeats are often used by composers and conductors to help musicians read and navigate in a musical piece. Their automatic estimation is useful for various tasks such as automatic music transcription [3], genre recognition [4], chord recognition [5] and structural segmentation [6]. It is also useful for computational musicology [7], measuring rhythm pattern similarity [8], and synchronizing two musical excerpts [9] or a musical piece with another media segment such as a virtual dancer, a drum machine, virtual books or a light show [10].

S. Durand, B. David, and G. Richard are with the LTCI, CNRS, Télécom ParisTech, Université Paris-Saclay, 75013, Paris, France.

J. P. Bello is with the Music and Audio Research Laboratory (MARL), New York University – USA

Downbeat tracking is a challenging task because it often relies on other subtasks such as beat tracking, tempo and time signature estimation and also because of the lack of an unambiguous ground truth[1]. Current approaches are thus limited in their scope. For example, methods that are implicitly limited to percussive instruments only [12], [13] are not applicable to a wide range of music styles. Other systems choose to strongly limit the possible time-signature [10], [12]–[18], require downbeats to be at onset positions [19], or make use of restrictive prior knowledge [16], [17]. Approaches that estimate some necessary information beforehand are naturally prone to error propagation. A number of current methods characterize downbeats with the help of one feature type only, while a more diverse description has proven to be more useful [10], [13], [18], [20]. Interestingly, there is an analogy with the multi-faceted aspect of human downbeat perception, which takes into account rhythm, but also harmony, musical structure and melodic lines [1]. Lastly, downbeat detection functions are often computed from low-level features, without taking into account higher-level representations. When this is not the case, as in [17] the estimations depend on prior decision-making, e.g. chord classification, which can be prone to errors.

We therefore propose an approach that:

- Minimizes assumptions in feature, classifier and temporal model design, and therefore is more generalizable.
- Does not require any prior information such as genre or time signature.
- Combines different kinds of features to cover different aspects of the musical content.
- Uses deep learning, particularly convolutional nets, to obtain higher-level representations that fully characterize the complexity of the problem but are hard to design by hand.

The model is evaluated on a large number of songs from various music styles and shows a significant improvement over the state of the art. This article significantly expands on our previous publications in several ways. First, it groups information from multiple publications into a single discussion, providing a much more detailed and systematic analysis of the system, its strength and shortcomings, and an expanded comparison with other published methods. More specifically it introduces a new deep network configuration, the bass content neural network, and an improved temporal model including more states. This paper also explores and evaluates alternative strategies for data fusion using random forests and boosting algorithms, as

[1]Indeed, metrical structures in music are not always well-formed but ambiguous by design [11], which can in turn cause disagreement amongst listeners.

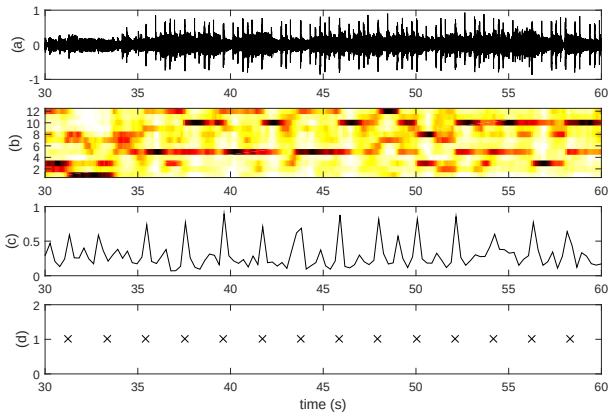

Fig. 1. Illustration of the three-step process of downbeat tracking for thirty seconds of audio. (a) Audio signal $\mathbf{x}$. (b) A feature $\mathbf{F}_i$, in this case chroma vectors. (c) Downbeat detection function $\mathbf{d}$, in this case downbeat likelihood. (d) Discrete downbeat sequence $\mathbf{s}$.

well as more configurations for the segmentation and temporal modeling stages. Lastly, the proposed convolutional nets are compared against simpler machine learning solutions based on Support Vector Machines (SVM) and fully-connected deep nets from chroma and spectral flux features.

The paper is organized as follows. Section II states the problem of downbeat tracking and presents some of the related work. Section III provides an overview of the method and its main steps Section IV describes the neural networks and their tuning together with the learning strategies. Finally, section V provides the experimental results and a comparison with four other published downbeat tracking systems.

II. Related Work

Algorithms for downbeat tracking are designed to estimate a discrete sequence of time instants corresponding to the bar positions in a musical piece. In most cases, they can be divided into three mains steps, as illustrated in figure 1:

1) From raw data compute feature vectors or matrices.
2) From features derive a function representing the downbeat likelihood (the so-called downbeat detection function).
3) Obtain the downbeat sequence with the help of a temporal model.

A wide range of techniques and ideas have been tested for each step, which are summarized in the following.

A. Feature extraction

Past approaches use domain-specific knowledge of music for feature design, often from only a single attribute. For example, chromas [13], [17], [18], [20] and spectral component histograms [10] have been utilized as harmonic features, with the assumption that the harmonic content is more likely to change around the downbeat positions.

Other features, more generally considered as rhythm markers, are based on the onset detection function (ODF) [10], [13], [20]–[22]. It is worth noting here that, even if isolated onsets are not always present at downbeat positions, onset patterns will often be synchronous with the bar changes. ODF are often extracted across frequency bands [12], [23] in order, for instance, to separate the events corresponding to kick drums or bass from those of the snare.

A third category concerns timbre inspired features [13], [19], [24]. Alterations of the timbre content occur more likely at the start of a new section and near a downbeat position. This feature extraction step can be done in conjunction with an onset [19], tatum [14] or beat [16] segmentation. A tatum is the lowest regular pulse train that is generally perceived from the musical events. As there are often three or four beats per bar, we can expect between six and sixteen tatums per bar even though it strongly depends on the music genre.

B. Downbeat detection function

The goal of the second step is to map the features into a downbeat detection function. This can be done with heuristics. When harmony features are used, most systems focus on measuring a change in the feature properties [16], [18], [20]. This can be done with the cosine distance, the Kullback-Leibler divergence or a difference of the principal harmonic components. If rhythm-based features are considered, i.e. features specifically designed to represent the rhythmic content in the music signal, comparison with pre-stored rhythm patterns [10], [15], [22], beat enhanced onset detection function [13] and relative spectral balance between high and low energy content at different beat positions [18] have also been proposed.

Machine learning systems can also be considered. Generic approaches such as SVMs with an auditory spectrogram [13], [19] have been used. Other systems focus on recognizing rhythm patterns in the data to find the downbeats. It can be done with a Conditional Deep Belief Network (CDBN) [25] or a Gaussian Mixture Model (GMM) [12] coupled to a k-means clustering [23]. These rhythm-related methods are more adapted to the problem, but they often make strong style-specific assumptions and require music with a very distinctive rhythm style to work well [12], [23], [26].

C. Downbeat sequence extraction

The goal of the last step is to discretize the downbeat detection function into a downbeat sequence. Considering that the temporal distance between two downbeats varies slowly, it can be useful to estimate the bar periodicity beforehand. To do so, it is possible to segment the audio into frames of different lengths and find the length that makes those segments the most similar [21]. We can also take advantage of the repetitive aspect of the onset detection function with a comb filter or a short time Fourier transform to find different periodicity levels: *tatum*, *tactus*, and *measure*. Assuming an integer ratio between these levels can help estimating them jointly [15] or successively [14].

To take into account the slow bar length variation over time, induction methods are often used. The estimated downbeat sequence will be the one that maximizes the values of the downbeat detection function and minimizes the bar length

variation. It can be done in a greedy way one downbeat after another, starting at the first downbeat [10], [21] or at the start of the first characteristic rhythm pattern [14]. To avoid being stuck in a local minimum, most algorithms search a more global downbeat sequence path. It can be done with dynamic programming [13] or Hidden Markov Models (HMM) [15], [17], [18] that can sometimes be coupled with a learned language model [20]. A particularly interesting temporal model is the dynamic bar pointer model [22]. It consists of a dynamic Bayesian network jointly modeling the downbeat positions, the tempo and the rhythm pattern. This method was refined to improve the observation probabilities [12] and reduce the computational complexity of the inference [27]. Most of the aforementioned systems don't handle varied time signatures during the downbeat sequence extraction.

III. Proposed model

In this section, we will describe the five parts of the proposed model as well as a general overview.

A. Model overview

The model overview is illustrated in figure 2. The signal's time-line is quantized to a set of bar subdivisions so-called tatums. The purpose of the system is to classify those tatums as being or not at a downbeat position. Features related to harmony, rhythm, melody and bass content are computed from the signal. Inputs centered around each candidate tatum are extracted. Each input is then fed to an independent deep neural network (DNN). The DNNs classify the tatums as being at a downbeat position or not and output a downbeat likelihood function. Networks outputs are fused to obtain a single downbeat likelihood per tatum. Finally, a HMM is used to estimate the most likely downbeat sequence.

B. Tatum synchronous segmentation

We adapt the local pulse information extractor proposed in [28] to achieve a useful tatum segmentation. The first step is to compute the tempogram of the musical audio from the short-term Fourier transform (STFT) of the novelty curve used in [28], and to keep only periodicities above 1 Hz to avoid bar-level frequencies. We then track the best periodicity path by dynamic programming with the same kind of local constraints as in [29][2]. The resulting system can find a fine-grained subdivision of the downbeats at a rate that is locally regular. We reduce the tempogram to 10 periodicities around the decoded path. We finally construct the predominant local pulse (PLP) function as in [28] based on this modified tempogram, and detect tatums using peak-picking on the PLP. The resulting segmentation period is typically twice as fast as the beats period, while it can be up to four times faster.

[2]They are more restrictive in our case, [0.5 0.7 1 0.7 0.5] instead of [0.95 0.98 1 0.98 0.95], to avoid jumping from one periodicity level to another. These local constraints state that it is only possible to move from one periodicity to its 5 neighbors {+2,+1,0,-1,-2} between time steps. It can be seen as a transition matrix from a state to the next one. The weight to stay at the same periodicity is 1, the weight to jump to the next periodicity is 0.7 and so on. See [29] for more detail on the dynamic programming computation.

TABLE I
STFT analysis parameters for each representation. s_r is the sampling rate used in Hz. Sizes are given in ms.

	Window size	Hop size	s_r
Chroma	743	92.9	5512.5
LFS	64	8	500
ODF	23.2	11.6	44100
MCQT	185.8	11.6	11025

It is interesting to use tatums[3] as a first segmentation step for several reasons. First, it is a musically meaningful segmentation that adapts to the local tempo variations of the song and introduces tempo invariance in the input representation. Like any type of invariance encoded by the features, this allows for the robust characterization of events with less training data and a lower capacity network. Finally, it is possible to design a tatum segmentation method with a high recall rate, meaning that almost all possible downbeats are candidates for detection.

C. Feature extraction

In this work, the aim of feature extraction is to represent the signal as a function of four musical attributes contributing to the grouping of beats into a bar, namely harmony, rhythmic pattern, bass content, and melody. This multi-faceted approach is consistent with well-known theories of music organization [1], where the chosen attributes contribute to the perception of downbeats. In section III-C, we discussed why harmony and rhythm features are useful for downbeat tracking. The bass or low-frequency content contains mostly bass instruments or kick drum, both of which tend to emphasize the downbeat. For melody, some notes tend to be more accented than others, and both pitch contour and note duration play an important role in our interpretation of meter [30], [31]. In the following, harmony will be represented by chromas, rhythm by an onset detection function (ODF), bass content by a low-frequency spectrogram (LFS) and melody by melodic constant-Q transform (MCQT) features. Each representation, illustrated in figure 3, is computed from a STFT using a Hann window of varying size applied to a resampled input signal as given in Table I. More details on their implementation are provided below.

1) Chroma: The chroma computation is done as in [32]. We apply a constant-Q transform (CQT) with 36 bins per octave, starting from 73 Hz to 584 Hz, to the STFT coefficients as in [33] and convert the constant-Q spectrum to harmonic pitch class profiles. Octave information is aggregated by accumulating the energy of equal pitch classes. The chromagram is tuned and then smoothed in time by a median filter of size 8 [743 ms]. It is finally mapped to a 12 bins representation by averaging.

2) Onset detection function: We compute a three band spectral flux ODF. To do so, we apply μ-law compression as in [15] with $\mu = 10^6$, to the STFT coefficients. We then sum the discrete temporal difference of the compressed signal

[3]It is to note that we are not formally looking for tatums, but more precisely for a regular and fast downbeat subdivision. The result is close to tatums and is called this way for convenience.

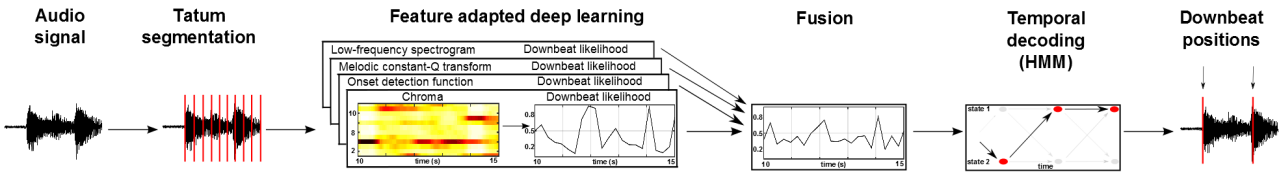

Fig. 2. Model overview.

on three bands for each temporal interval, subtract the local mean and half wave rectify the resulting signal. The frequency intervals of the low, medium and high frequency bands are [0, 150], [150, 500] and [500, 11025] Hz respectively as we believe low frequency bands carry a lot of weight in our problem. To limit the variation of this feature, the signal is clipped so that all values above the 9^{th} decile are equal.

3) Low-frequency spectrogram: We only keep spectral components of the STFT representation below 150 Hz (the first 10 bins). To limit the variation of this feature, as for the ODF, the signal is clipped so that all values above the 9^{th} decile are equal.

4) Melodic constant-Q transform: We apply a CQT with 96 bins per octave to the STFT coefficients as in [32], starting from 196 Hz to the Nyquist frequency, and average each CQT bin $q(w)$ with the following octaves:

$$q_a(w) = \frac{\sum_{j=0}^{J_w} q(w+96j)}{J_w+1} \tag{1}$$

with J_w such that $q(w+96J_w)$ is below the Nyquist frequency. We then only keep 304 bins from 392 Hz to 3520 Hz that correspond to three octaves and two semitones. This summation of the CQT bins is useful to highlight harmonic components. Also, it is noteworthy that we are summing the octaves of a note and not its partials[4]. Doing so allows us to have equally spaced versions of the melodic line at the cost of losing information about its absolute pitch. In this case, melodic lines are one octave apart from each other. Equally spaced melodic lines are useful to design fixed length convolutional filters later on, and absolute pitch is not useful here since we are looking for melodic contour. The frequency range of this feature and of the chroma feature are significantly different. This increases the diversity between features which is beneficial when they are used in combination. Besides, the frequency range of this feature is wide enough to capture most of the melodic lines while keeping a relatively low computational cost. We use a logarithmic representation of q_a to represent the variation of the energy more clearly:

$$lq_a = \log(|\hat{q_a}| + 1) \tag{2}$$

where $\hat{q_a}$ is the restriction of q_a between 392 Hz and 3520 Hz. Additionally, we zero all values below the third quartile Q_3 of a given temporal frame to get our melodic CQT:

$$m_{CQT} = \max(lq_a - Q_3(lq_a), 0) \tag{3}$$

Keeping only the highest values allows us to remove most of the noise and onsets so we can see some contrast and obtain features that are significantly different from the previous rhythm feature.

5) Temporal quantization: The four aforementioned features are then mapped to a grid with subdivisions lasting one fifth of a tatum using interpolation. We therefore have tempo independent features with a temporal precision higher than the tatum level. Their temporal dimension is therefore 5 per tatum and their spectral dimension is respectively 12, 3, 10 and 304. We segment these features so they start at the beginning of each tatum and have a fixed length of 9 or 17 tatums depending on the feature context as explained in section IV. Each instance is finally scaled between 0 and 1 and acts as input to the deep neural networks as described in the following section.

D. Feature learning

Downbeats are high-level events defined by complex patterns in the feature sequence. We propose that the probability of a downbeat can be estimated using feed-forward DNN $F(X_0|\Theta, P)$, where $X_0 \in [0,1]^{N_0 \times M_0 \times L_0}$ is our input tensor of temporal, spectral and feature map size N_0, M_0 and L_0 respectively. Θ and P are the learned and designed parameters of the network. In our implementation, F is a cascade of $K = 4$ non-linear functions $f_k(X_k|\theta_k, p_k)$:

$$F(X_0|\Theta, P) = f_{K-1}(...f_0(X_0|\theta_0, p_0)|\theta_{K-1}, p_{K-1}) \tag{4}$$

where X_k is the input of layer $k \in [\![0, K-1]\!]$[5] and $\theta_k = \{W_k, b_k\}$, with W_k a collection of L_{k+1} three-dimensional filters and b_k a vector of biases. p_k is a set of parameters allowing a compact description of the network and f_k is a cascade of a convolution c and of one or several non linear functions h_k:

$$f_k(X_k|\theta_k, p_k) = h_k(c(X_k|\theta_k, p_{1_k}), p_{2_k});\ \forall k \in [\![0, K-1]\!] \tag{5}$$

In our case, $p_{1_k} = \{t_{1_k}, v_{1_k}, L_k, L_{k+1}\}$ gathers the shape of W_k. The variables t_{1_k}, v_{1_k} and L_k are the temporal, spectral and feature map sizes of W_k. c is defined as:

$$c(X_k|\theta_k, p_{1_k}) = W_k * X_k + b_k \tag{6}$$

where $*$ represents a valid convolution. h_k is in our case a set of one or several cascaded nonlinear functions including rectified linear units $r(x) = \max(0, x)$, sigmoids $\sigma(x) = \frac{1}{1-e^{-x}}$, max pooling m, softmax normalization $s(x) = \frac{e^x}{\sum_{j=1}^{J} e^{x[j]}}$ and dropout regularization d [34]. $p_{2_k} = \{t_{2_k}, v_{2_k}\}$ is the designed set of parameters of h_k corresponding in our case to the temporal and spectral reduction factors of the max pooling. X_K is the final output and acts as a downbeat likelihood.

[4]The octaves of a note of frequency f have a frequency of $2^n f$. Its partials have a frequency of $(n+1)f$

[5]$[\![\ ,\ ]\!]$ denotes an integer interval.

In this work, we consider each feature type independently and we train one DNN per feature. This is a convenient way to work with features of different dimension and assess the effect of each of them. Moreover, we want to adapt feature learning to the feature type. Harmonic features will mostly exhibit change while rhythm features will often show characteristic patterns as downbeat cues. Low-frequency and melodic features will exhibit a bit of both, and melodic features need an adapted dimensionality reduction process for example. Besides these differences, there is no reason that the optimal DNN hyper-parameters have to be the same in each case. The specific adaptations are described in section IV. Each network is trained on binary ground truth targets representing tatums being or not at a downbeat position. We use the MatConvNet toolbox to design and train the networks [35].

E. Network combination

For each tatum we have four intermediary downbeat likelihoods. We need to fuse this information into a single robust downbeat likelihood to feed our temporal model. To do so, we use the average of the four downbeat likelihoods. The average—or sum rule—is generally quite resilient to estimation errors [36]. However, it is not robust to redundant information, and the network will need to produce complementary information.

F. Temporal modeling

We use a first order left-to-right HMM to map the continuous downbeat likelihood function $\mathbf{d}$ into the discrete sequence of downbeats $\mathbf{y}$. The inference is done with the Viterbi algorithm and the temporal interval of our model is the tatum. Our model contains:

- The state space $H = \{1...N_h\}$ with N_h the number of hidden states i.
  Since the downbeat likelihood depends on the bar length and the position inside a bar, we will define a state for each possible tatum in a given bar. For instance, considering two possible bars of two and three tatums, there would be five different states in the model. One state would represent the first tatum of the two-tatum bar, another would represent the second tatum of the two-tatum bar and so on. In that case, the states representing the second tatum of a two-tatum bar and the second tatum of a three-tatum bar are different. More precisely, the second tatum of a two-tatum bar should most likely transition to the first tatum of a two-tatum bar, and the second tatum of a three-tatum bar should transition to the third tatum of a three-tatum bar. In practice, we allow time signatures of {3,4,5,6,7,8,9,10,12,16} tatums per bar, for a total of $N_h = 3 + 4 + ... + 16 = 80$ states. Furthermore, modeling bars of different length independently allows for consistency in the decoding stage. We can also emphasize that bars with the same number of tatums are found for example.
- The most likely state sequence $\mathbf{y}' = [y'(1); ...; y'(T)]$ with $y'(\tau) \in H,\ \forall\tau \in [\![1, T]\!]$ and T the length of the sequence.
- The initial probability $\boldsymbol{\pi} = [\pi(1); ...; \pi(N_h)]$ with $\pi(i) = P(y'(1) = i)$ of being in a state i initially.
  We use an equal distribution of the initial probabilities: $\pi(i) = \frac{1}{N_h}$, $\forall i \in H$, for robustness[6].
- The observation sequence $\mathbf{o} = [o(1); ...; o(T)]$ with $o(\tau) \in [0, 1],\ \forall\tau \in [\![1, T]\!]$.
  It is equal to the downbeat likelihood: $\mathbf{o} = \mathbf{d}$.
- The emission probabilities $\mathbf{e}_i = [e_i(1); ...; e_i(T)]$ with $e_i(\tau) = P(o(\tau)|y'(\tau) = i)$ the probability of the observation $o(\tau)$ given a state i, $\forall\tau \in [\![1, T]\!]$.
  For the emission probabilities we will distinguish two cases, either the state corresponds to a tatum at the beginning of a bar: $i \in H_1 \subset H$, or it corresponds to another position inside a bar: $i \in \overline{H}_1 \subset H$. In the first case the emission probability is equal to the downbeat likelihood, and in the second case to its complementary probability:

$$\mathbf{e}_i = \begin{cases} \mathbf{d} & \text{if } i \in H_1 \\ 1 - \mathbf{d} & \text{if } i \in \overline{H}_1 \end{cases} \tag{7}$$

- The transition probabilities $\mathbf{A} = \{a_{i,j},\ (i,j) \in [\![1, N_h]\!]^2\}$ with $a_{i,j} = P(y'(\tau) = j|y'(\tau - 1) = i)$ the probability of transitioning from state i to state j, $\forall\tau \in [\![2, T]\!]$. The ${N_h}^2 = 6400$ parameters of our transition matrix could be trained entirely automatically, but this is left for future work. We set the majority of transition matrix parameter values by simply counting the number of occurrences for a specific transition and giving a minimum value if an occurrence didn't sufficiently happen. If a transition from i to j occurs n times out of a total of N transitions from i to any state, then $a_{i,j} = \max\left(\frac{n}{N}, 0.02\right)$. Putting non null values to $a_{i,j}$ allows for better adaptability to the downbeat likelihood from the model. Finally, transitions responsible for a change in the time signature are manually fine-tuned by maximizing the F-measure of the training set. It appeared that over-fitting was not an issue for this problem, probably because of a relatively wide range of close to optimal values. To give an idea, the transition matrix coefficients can be summarized in three categories. There are high probabilities to advance circularly in a given bar, medium probabilities to go from the end of one bar to the beginning of another, and low probabilities to go elsewhere.

In the end, decoded states corresponding to a tatum at the beginning of a bar will give the downbeat sequence: $\mathbf{y} = \{y'(\tau),\ \tau \in [\![1, T]\!] \ / \ y'(\tau) \in H_1\}$.

IV. Feature adapted deep neural networks

To take advantage of the specificities of the different extracted features, we first exploit their local structure by using convolutional neural networks (CNN). In CNNs, convolutional layers [37], [38] share weights across the inputs within a spatio-temporal region so each input unit is not considered

[6]It means that the first detected tatum can be at any given position inside the bar with equal probability.

independent from its neighbors. We then adapt the architecture and the learning strategies to each input as described below. The network design choices will be described with the notations introduced in subsection III-D.

A. Melodic neural network (MCNN)

Our intention here is to train a network that learns to predict the downbeat based on melodic contours as they play a role in our meter perception regardless of their absolute pitch [39]. Considering that those patterns can be relatively long, we will use 17 tatum-long inputs. It roughly corresponds to two bars of audio in 4/4. We then have input features X_{m_0}[7] of spectral dimension $M_0 = 304$ and of temporal dimension of 17 times 5 temporal quantization: $N_0 = 85$. Our network architecture is presented in figure 3(a). For example, the first layer:

$$f_0 = m(r(c(X_0|\theta_0, \{46, 96, 1, 30\})), \{2, 209\}) \qquad (8)$$

means that we use filters of parameters θ_0 and shape $p_{1_0} = \{46, 96, 1, 30\}$[8] on input X_0 for convolution c, and then rectified linear units r and max pooling m with a reduction factor of $\{2, 209\}$ as non linearity[9]. The first layer filter is relatively large, $t_{1_0} = 46$ and $v_{1_0} = 96$, so we are able to characterize pitch sequences. The reduction factor in frequency of the following max pooling, $v_{2_0} = 209$, is equal to the input size after the convolution. This way, we only keep the maximal convolution activation in the whole frequency range and lead the network to focus on patterns regardless of their absolute pitch center. The fourth layer, $f_3 = s(c(X_3|\theta_3, \{1, 1, 800, 2\}))$, can be seen as a fully connected layer: $t_{3_0} = 1$ and $v_{3_0} = 1$, that will map the preceding hidden units into 2 final outputs. Those outputs represent the likelihood of the center of the input to be at a downbeat position and its complementary. To do so, we train the network with the logarithmic loss between the outputs and binary ground truth targets corresponding to the tatum at the center of the input matrix.

B. Rhythmic neural network (RCNN)

We want to design a network that will estimate the downbeat positions by learning rhythmic patterns of specific length. Since rhythm patterns can be long we also use 17 tatum long inputs. Contrary to melodic patterns, the length of a rhythmic pattern and the length of a bar are strongly correlated and the pattern boundaries are likely to be located at a downbeat position. To characterize this pattern length, the network should give different outputs if patterns of different length are observed. We choose for that multi-label learning [40]. In this case, if there is a downbeat position at the first and ninth tatum of our 17 tatum-long input, the output of our network should be $X_4 = [1; 0; 0; 0; 0; 0; 0; 0; 0; 1; 0; 0; 0; 0; 0; 0; 0; 0]$. Since there might be multiple downbeats per input, it is not

[7]The index indicating the type of network, here m for melodic network, won't be explicitly mentioned in the following for simplicity.

[8]According to subsection III-D, this means that W_0 is a collection of 30 three-dimensional filters of temporal size 46, spectral size 96, feature map size 1.

[9]Therefore, the output of this layer, $X_1 \in \mathbb{R}_+^{20\times1\times30}$.

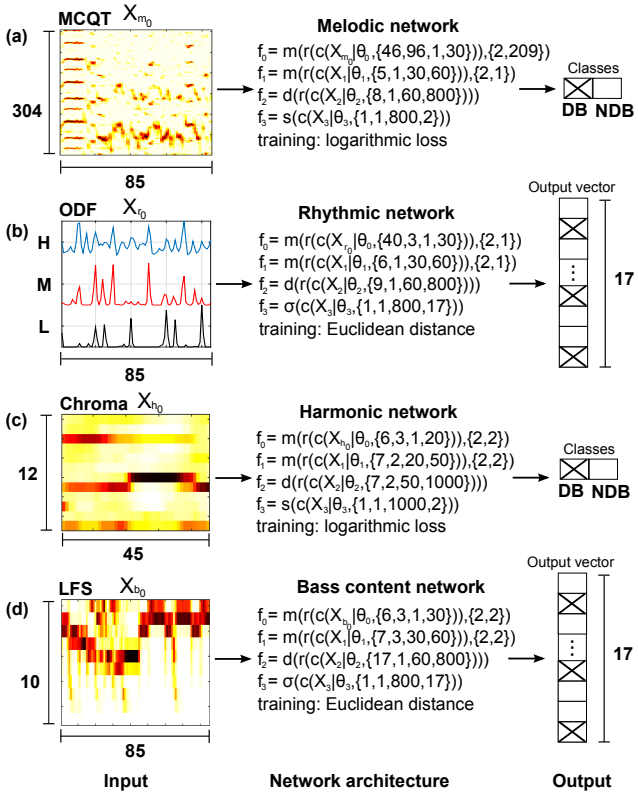

Fig. 3. Convolutional networks architecture, inputs and outputs. The notation is the same as in subsection III-D. DB and NDB stand for downbeat and no downbeat respectively.

appropriate to normalize the network output X_4. Instead, we want each coefficient of X_4 to be close to 0 if there isn't a downbeat and 1 if there is a downbeat at the corresponding tatum position. Therefore, we first use a sigmoid activation σ unit in our last layer to map the results into probabilities: $f_3 = \sigma(c(X_3|\theta_3, \{1, 1, 800, 17\}))$. We then train the network with an Euclidean distance between the output and the ground truth $\mathbf{g}$ with the same shape as X_4: $g(\tau) = 1$ if there a downbeat at the τ^{th} tatum of the input and $g(\tau) = 0$ otherwise. Each tatum is then considered independent from one another. Our network architecture is presented in figure 3(b). To detect bar-long rhythm patterns, our first convolutional layer uses relatively large temporal filters of about the length of a bar, $t_{1_0} = 40$ and $v_{1_0} = 3$. Besides, since we are using the Euclidean distance to ground truth vectors to train the network, we are not explicitly using classes such as 'downbeat' and 'no downbeat'. The output is then of dimension 17 and represents the downbeat likelihood of each tatum position in X_{r0}. Since we have 17 tatum-long inputs but a hop size of 1 tatum, overlap will occur. We will reduce the dimension of our downbeat likelihood to 1 by averaging the results corresponding to the same tatum.

Figure 4 shows the ODF input transformation through the rhythmic network until the downbeat likelihood is obtained after the averaging step. It can be seen that in the first layer some units tend to be activated around rhythmic patterns or events, highlighted here in part by the orange circles in figure 4b). As we go deeper into the network, some units tend to

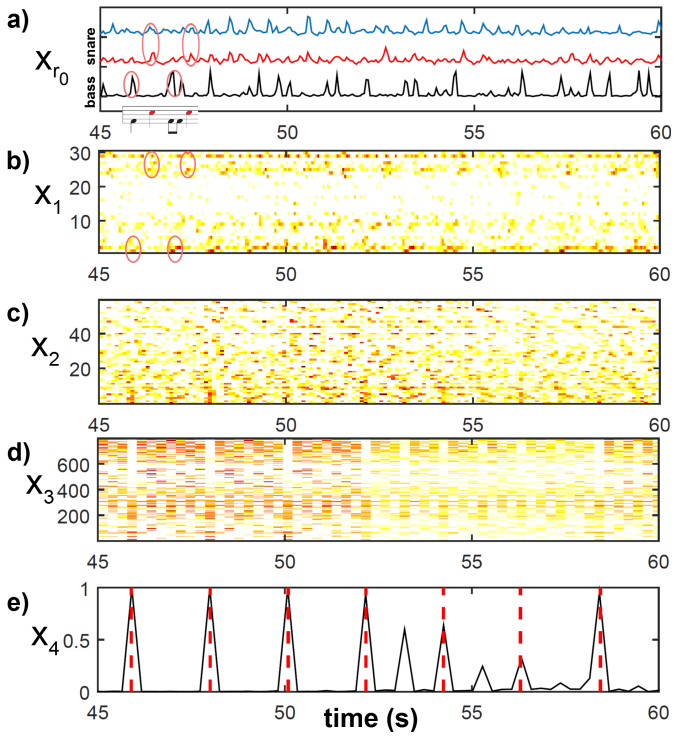

Fig. 4. RCNN input and intermediary and final outputs. a): Onset detection function input. b-c): Output of all the filters on top of each other, from the first and second layer respectively before the max pooling. d): Output of all filters from the third layer. e): Output of the network, that acts as a downbeat likelihood. The red dotted lines are the annotated downbeat positions. Since inputs overlap in time, the figures are averaged to get one bin per filter per time frame when necessary.

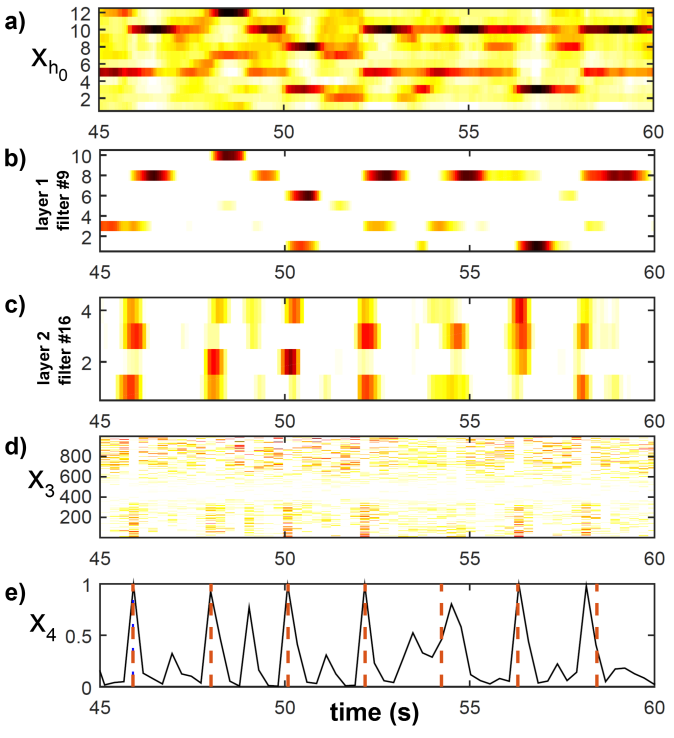

Fig. 5. HCNN input and intermediary and final outputs. a): Chroma input. b): Output of the filter number 9 of the first layer before max pooling. c): Output of the filter number 16 of the second layer before max pooling. d): Output of all the filters of the third layer. e): Output of the first class of the network, that acts as a downbeat likelihood. The red dotted lines are the annotated downbeat positions. Since inputs overlap in time, the figures are averaged to get one bin per filter per time frame when necessary.

be activated more clearly around downbeat positions and some other units around no-downbeat positions. We finally obtain a rather clean downbeat likelihood function, in figure 4e), that is high around the red dotted lines representing the annotated downbeats. The network is a bit more indecisive around the fifty fifth second, as there is a drum fills leading to a chorus.

C. Harmonic neural network (HCNN)

We aim at designing a network that will predict the downbeat based on harmonic changes and salient harmonic events in the input [10]. The temporal and spectral size of the first layer filters $t_{1_0} = 6$ and $v_{1_0} = 3$ respectively, as well as the input size, $N_0 = 45$, are chosen to be rather small. Here, contrary to the RCNN, we do not need to characterize the length of a pattern and we then choose the same kind of non-linear functions as in the MCNN for the four network layers and the logarithmic loss to train the network. The ground truth targets are the same as in the MCNN. The size of the filters differs to be adapted to the input size though. Additionally, it is desirable for this network to be robust to key transposition, as it changes the input but not our downbeat perception. To that aim, max pooling on the whole frequency range as in the melodic network will not work because chroma vectors are circular. Instead we choose to augment the training data with the 12 circular shifting combinations of the chroma vectors.

The network input, layers output and final output can be seen in figure 5. The first layer of the network transforms the input to remove some of the noise as in figure 5b) or the highlight some of its properties such as its onsets or offsets for example. The second layer takes advantage of this new representation to reduce its dimension or compute some sort of harmonic change. The third layer outputs 1000 units of dimension 1 and we can see that some of them act as downbeat detectors, lower figure 5d), and some of them as no-downbeat detectors, upper figure 5d). Finally, the obtained downbeat likelihood, although a little noisy, is rather close to the ground truth downbeats.

D. Bass content neural network (BCNN)

The bass content feature contains melodic and percussive bass instruments as can be seen in the figure 3(d) by the horizontal and vertical lines respectively. Our network architecture is also presented in figure 3(d). Detecting patterns is useful here but instantaneous events are more directly related to downbeats than for the melodic feature for example as bass notes or bass drums tend to be played at a downbeat position. We therefore use filters of moderate temporal size for our first layer to emphasize these events, $t_{1_0} = 6$. As

[10] See figure 3(c) for the input size, network parameters and output.

bass events may be repeated each bar, we want to be able to characterize the pattern length like with the rhythmic network. We therefore use the same last layer architecture: $f_3 = \sigma(c(X_3|\theta_3, \{1, 1, 800, 17\}))$, and multi-label procedure with the Euclidean distance to a ground truth vector $\mathbf{g}$ of zeros around no downbeat and ones around downbeats as a cost function to minimize. The dimension of our downbeat likelihood function will also be reduced to one by averaging.

V. Evaluation and results

The proposed system is compared to 4 other published downbeat tracking algorithms on a total of 9 datasets presented below. We also assess each step of our method and present some of its limitations and strengths.

A. Methodology

1) Evaluation metrics and procedure: We use the F-measure, expressed in percent, to evaluate the performance of our system. This widely used metric [17], [18], [20] is the harmonic mean of precision and recall rates. We use a tolerance window of $\pm$70 ms and do not take into account the first 5 seconds and the last 3 seconds of audio as annotations are sometimes missing or not always reliable. To assess statistical significance, we perform a Friedman's test and a Tukey's honestly significant criterion (HSD) test with a 95% confidence interval. The best performing method and the ones with non statistically significant decreases will be highlighted in bold in the result Tables. Finally, to be fair to systems that were not originally trained on all datasets, we use a leave-one-dataset-out approach as recommended in [41], whereby in each iteration we use all datasets but one for training, and the holdout dataset for testing. 90% of the training dataset is used to train the networks and 10% to set the hyper parameters.

2) Datasets: We evaluate our system on nine different real audio recording datasets, presented in Table II. The letter "e" in the "# Tracks" column means that the tracks are excerpts of about 1 minute, compared to full songs elsewhere. Those datasets, while being mainly focused on western music, cover a wide range of styles, including pop, classical, jazz, choral, hiphop, reggae, disco, blues, electro, latin, rock, vocal and world music. They include more than 1500 audio tracks ranging from 30 seconds excerpts to 10-minute long pieces for a total of about 43 hours of music and 78171 annotated downbeats. Long excerpts will be more sensitive to an adapted temporal model while an instantaneous downbeat estimation is more important for short excerpts. Some datasets focus on a certain music style or an artist while others include a wider musical spectrum and are labelled as "Various" in the Table II. The subset of the Klapuri dataset gathers 40 randomly selected tracks among four music styles - Blues, Classical, Jazz and Electro/Dance - given two constraints : the time signature is fixed for each excerpt and there are 10 tracks per genre. Besides variety in content, using several datasets creates variety in annotation. Indeed, downbeats can be ambiguous and different annotators will make different annotations.

TABLE II
Datasets overview.

Name	# Tracks	# DB	Length	Genre
RWC Class [42]	60	10148	5h 24m	Classical
Klapuri subset [15]	$40 - e$	1197	0h 38m	Various
Hainsworth [43]	$222 - e$	6180	3h 19m	Various
RWC Jazz [42]	50	5498	3h 44m	Jazz
RWC Genre [44]	92	11053	6h 22m	Various
Ballroom [45]	$698 - e$	12219	6h 04m	Ballroom dances
Quaero [46]	70	7104	2h 46m	Pop, rap, electro
Beatles [47]	179	13937	8h 01m	Pop
RWC Pop [42]	100	10835	6h 47m	Pop
Total	1511	78171	43h 05m	

TABLE III
F-measure results for several published downbeat trackers.

	Peeters et al. [18]	Davies et al. [16]	Papadopoulos et al. [17]	Krebs et al. [48]	**ACNN**
RWC Class	29.9	21.6	32.7	33.5	**51.0**
Hainsworth	42.3	47.5	44.2	51.7	**65.0**
RWC Genre	43.2	50.4	49.3	47.9	**66.1**
Klapuri	47.3	41.8	41.0	**50.0**	**67.4**
RWC Jazz	39.6	47.2	42.1	51.5	**70.9**
Ballroom	45.5	50.3	50.0	52.5	**80.1**
Quaero	57.2	69.1	69.3	71.3	**81.2**
Beatles	53.3	66.1	65.3	72.1	**83.8**
RWC Pop	69.8	71.0	75.8	72.1	**87.6**
Mean	47.6	51.7	52.2	55.8	**72.6**

B. Comparative analysis

We compare our system, called here ACNN[11], to the downbeat trackers of Peeters et al. [18], Davies et al. [16], Papadopoulos et al. [17] and Krebs et al. [48] using the same evaluation measure. [18], [16] and [17] are unsupervised methods. [48] is supervised and is also trained with a leave-one-dataset-out approach[12] with all the above sets but the Klapuri and Quaero datasets and with the addition of the Böck [49], [50], Rock [51] and Robbie Williams [52] datasets. It is worth noting that the algorithm of [16] was manually fed with the ground truth time signatures since it was needed to run it. Results are shown in Table III and highlight a better performance for our system on every dataset and an overall improvement of 16.8 percentage points (pp) compared to the second best system. The relative difference for the pop music datasets is the lowest at 11.1 pp. The harmonic change, spectral energy distribution and rhythmic pattern assumptions made by the other systems seem appropriate in this case. However, in the other music datasets where the downbeats are harder to estimate, the overall increase in performance is much higher, at 18.9 pp. A possible explanation is that our more sophisticated feature extraction and learning model is working well on some excerpts where downbeat cues are harder to obtain.

[11] A for all features used, CNN for the learning method

[12] Except for the Hainsworth dataset that is also contained in the training set.

A particularly interesting case is the ballroom dataset. There is indeed a substantial 27.6 pp difference in performance compared to the second best system. Our system has a similar performance with this dataset and the pop music datasets while the compared algorithms have a much lower performance here. The difference may then be explained by the fact that the explicit rhythm-related assumptions of [18] and [48] are not borne by the data. The performance of [48] increases drastically if it uses rhythmic patterns really close to the one used in this dataset [12]. Our system also uses training data with different rhythmic patterns, but seems to highlight a better robustness to unseen data. It may be because we use a larger set of rhythmic patterns or because of the usually better generalization power of the CNNs compared to the GMMs used in [48].

Besides, the assumption of [16], [17] and [18] that the harmonic content is different before and after a downbeat position may not be well verified here. To assess this, we used the chroma input only and replaced our harmonic network with a heuristic function as in [16] to obtain the downbeat likelihood. The performance of this heuristic function on the ballroom dataset is 8 pp poorer compared to its average performance across sets. The performance of [16], [17] and [18] is also lower for the ballroom dataset than overall as can be seen in Table III. On the other hand, the performance of the harmonic network is 5 pp better for the ballroom dataset than overall. Qualitative analysis of the results highlights that the harmonic network seems able to detect downbeats even without clear harmonic change, if there are salient harmonic events for example. Short melodic or harmonic events can be important to find downbeats and they tend to be diluted during the average process of the chroma vectors done in several methods, but the harmonic network can take them into account. Our network also seems less prone to noisy events because the content of a 9 tatum window is taken into account to estimate a downbeat position.

The system by Davies et al. [16] seems to work well on songs with correct beat estimation but it uses pre-estimated beat positions and an hypothesis of constant time signature that can propagate errors easily. Its performance increases significantly with the use of a more powerful beat tracker such as the one by Degara et al. [53] with an overall F-measure of 56.7%. Conversely, [17] is adapted to changes in the time signature but may be a bit too sensitive to these changes. [48] includes a sophisticated temporal model but uses rhythmic features only. [18] performs a global estimation of the meter with an efficient visualization of the output and may be improved with a feature deep learning stage to be less dependent on certain downbeat assumptions.

The performance of our algorithm can be summarized in three categories, highlighted by horizontal lines in the Table III. First is the RWC Class dataset with a rather low F-measure of about 50%. In this case, the tatum estimation is uncertain and it is therefore difficult to estimate the downbeat likelihood or use the same temporal model as for the other styles. Besides, annotation is more difficult to perform, especially with soft onsets and romantic pieces, and the $\pm$70 ms tolerance window of the evaluation measure may be too short in many cases. A better tatum segmentation will significantly improve the results, as will be shown in section V-C1. Finally, listening tests show that some classical music pieces are inherently more difficult to estimate without additional information such as the ground truth beats even for an expert listener. Second, the Ballroom, Quaero, Beatles and RWC Pop datasets can be regrouped with a F-measure between 80% and 90%. These datasets contain a stable tempo easy to be tapped and are often not surprising in terms of their metrical structure or cues in order to infer the downbeat position and are well estimated by our system. Finally, a third set can be composed of the Klapuri subset, Hainsworth, RWC Jazz and RWC Genre datasets with a F-measure between 65% and 70%. Most of them are composed of a mixture of genres that are either easy or difficult to estimate and therefore have a performance in between. The RWC Jazz dataset also belongs in this category because on the one hand the songs there contain, for the most part, clear rhythmic events and a relatively stable tempo. On the other hand, some instrumental lines are rhythmically more expressive, including rhythm pattern variations especially during solos. Finally many tempo errors occur because it is harder to define the correct metrical level. The F-measure is indeed increased by 10 pp if double or half tempo variations are allowed. It is the highest increase of all datasets.

The standard deviation of the F-measure across songs for this task is high but varies across databases. In our case, the mean of the standard deviation across songs, datasets and algorithms is 30%. The F-measure can indeed easily go from 100% to 0% if the third beat of the bar is taken for the downbeat for example, which happens frequently. The standard deviation is lower for the RWC Class dataset because the time signature changes more regularly and the downbeat estimation is less consistent, limiting the 100% and 0% occurrences. At the opposite, the standard deviation is higher on the datasets composed of short excerpts, up to 40%, because missing a downbeat has a higher effect on the performance.

C. Detailed analysis of the proposed system

An analysis of each step of the proposed system is given below.

1) Segmentation: How much do the limitations of our tatum segmentation affect the performance? The tatum segmentation step, with the advantage of segmenting the data into a reduced set of rhythmically meaningful events, has a mean downbeat recall rate of 92.9% across datasets considering a $\pm$70 ms tolerance window and therefore occasionally misses an annotated downbeat[13]. How much can it affect the overall performance? To assess this, we kept the system the same and only replaced the closest estimated tatum position to a ground truth downbeat by its annotated position

[13]We estimate that half of the missed downbeats come from inconsistencies in the segmentation step and half from a subjective annotation.

in the segmentation step to obtain a perfect recall rate. Mean results across datasets are shown in the row (a).1 of the figure 6 and we see an improvement of 3.9 pp compared to the reference model. This is statistically significant. Results in the RWC Pop, Quaero and Ballroom datasets of systems with or without a perfect downbeat recall are fairly close with a relative difference of about 1.4 pp. It highlights that our tatum segmentation step is very reliable to detect downbeats in music with a clear rhythmic structure. Improvement of 4.1 pp in the Beatles dataset may be contradictory, but a subjective error analysis shows that most of the difference in performance there is due to a questionable annotation. However, for other genres, a bigger difference of 5.4 pp appears, highlighting some of the limitations of the timing of estimated tatums near downbeat positions.

The segmentation step can have other issues than imprecise timing around downbeat positions. For example, two consecutive bars may contain a different number of estimated tatums. It also happens that tatums don't have the right periodicity or do not match downbeats at all. To assess this effect without changing the rest of the system, we want to use the best tatum segmentation for a given metrical level. Since we have access to annotated beats but not to annotated tatums, we replace the tatums with an interpolation by a factor 2 of the annotated beats in the segmentation step. This is the most common factor between beats and tatums in our datasets. Results are shown in the row (a).2 of the figure 6 and we see an improvement of 11.4 pp. This is statistically significant. The overall increase in performance is much higher here because all tatum positions are modified to match the downbeat sequence instead of only those near downbeats in the former experiment. The segmentation is therefore a lot cleaner and octave error are a lot less common. Compared to the perfect downbeat recall case, the improvement is particularly striking on the RWC Jazz dataset (9.4 pp) and the RWC Classical dataset (20.4 pp). Many octave errors were present in the first case and are now solved. In the second case, many spurious events and a lack of tatum consistency inside a bar were greatly disturbing the system. The hard part in the Classical music dataset is mostly to have a proper sub-downbeat segmentation. It is to note that the overall performance when beats are known is a bit overfitted to the annotations and may not be a precise estimate of the performance of our system with a better segmentation step. Indeed, besides some human errors in annotation, this task is sometimes subjective in terms of downbeat timing and metrical level. However, it appears that designing a more precise, clean and consistent segmentation step may have a significant effect on the overall performance.

2) Feature adapted neural networks: Are all feature adapted neural networks useful?

Using several complementary features and adapted learning strategies is the core of our method. However, there is a limit to the added value of a new feature compared to its redundancy with the others and not all features may be worth adding in our model, especially when using the average as a feature combination.

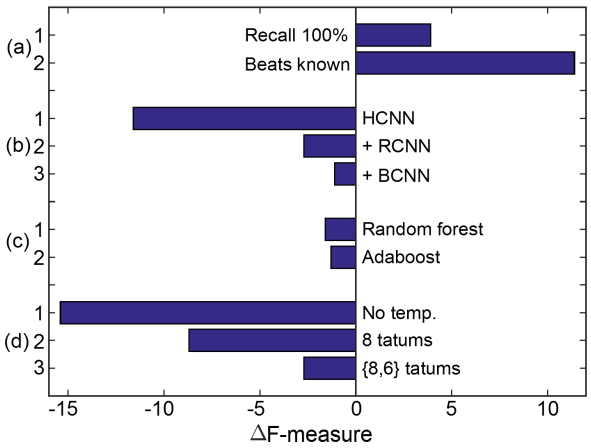

Fig. 6. Relative F-measure for various configurations of our system compared to the standard model (ACNN). Mean across all datasets. (a): Segmentation variation. (b): Network variation. (c): Network combination variation. (d): Temporal model variation.

To assess the effect of each feature adapted neural network efficiently, we show the performance of the best performing network, followed by the performance of the best combination of two networks, and then the performance of the best combination of three networks compared to our four networks system in figure 6(b). The harmonic neural network, figure 6(b).1, is quite powerful by itself with an overall F-measure of 61.0%, that is already significantly better than the compared algorithms in subsection V-B. On a dataset strongly depending on harmony such as the RWC Classical dataset, the performance of the harmonic network only is close to the one obtained with the whole model with a 4.7 pp difference. However, using the four networks is more robust on a variety of datasets and leads to a 11.6 pp increase overall. The harmonic and rhythmic networks is the best two-network configuration with an overall F-measure of 69.9%. Harmonic and rhythmic features are indeed quite complementary in a wide range of music. However, a statistically significant improvement is achieved by adding the bass content network with an overall F-measure of 71.5% and also by using all the networks with an overall F-measure of 72.6%.

While the system complexity increases with each new network, it remains relatively low. The mean processing time for 60 seconds of musical audio is respectively of 1.0, 1.5, 2.0 and 12.5 seconds as we add features and their corresponding network as in figure 6(b) on a PC with an Intel Xeon e5-1620 CPU with 3.6 GHz. The melodic network increases the processing time by a larger margin because of its bigger input. The training time of all four networks is about one day on a Geforce GTX TITAN Black GPU.

3) Feature learning: Is deep learning useful?

Considering that the combination of the harmonic and rhythmic networks provides a very good performance already and for complexity and clarity constraints, we will restrain the experiments in this part to these two networks. We will then consider several ways of getting a downbeat likelihood from

TABLE IV
MEAN F-MEASURE RESULTS FOR VARIOUS FEATURE PROCESSING STEPS.

Inputs used	SVM	DNN	CNN
ODF	49.0	53.9	**57.0**
Chroma	43.6	52.2	**61.1**
ODF + Chroma	56.6	64.5	**69.9**

our chroma and onset detection function inputs individually and in combination. We will compare a shallow learning method, a deep learning method and finally the hereby presented feature adapted and locally dependent deep learning.

We first use linear Support Vector Machine (SVM) with a penalty parameter C to the error term found on the validation set as our shallow learning method. As SVM predicts only class labels, the probability of each class will be estimated following the second method of [54] that is based on Platt scaling. As for the harmonic network, we used the same number of downbeat and no downbeat examples in our training data to have a balanced training set. There is also one classifier per input type and the fusion of several classifiers is done by averaging. Results, presented in Table IV show that the rhythmic SVM is significantly better than the harmonic SVM and that their combination is competitive with the algorithms presented in subsection V-B[14].

We will now take advantage of the relatively large number of training examples and the high level aspect of downbeats to apply a relatively simple deep learning method to our problem. We use DNN without convolutions with the exact same architecture as in [55]. The used networks consist of four fully connected layers of 40, 40, 50 and 2 sigmoid units respectively, and a softmax regularization output. We train the networks as in [55] using the chroma and ODF inputs described in subsection III-C. Each network is pre-trained with stacked Restricted Boltzmann Machines and 1-step Contrastive Divergence. 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. We randomly removed some of the features at non-downbeat position in such a way that our training set contains an equal amount of features computed at downbeat and non-downbeat positions. In our implementation we use early-stopping and dropout regularization. The two networks are independent. Results in Table IV highlight a significant improvement for both features and their combination. However, we see in Table IV that using the proposed convolutional networks is more suited to our problem and also significantly improves the results. Indeed, there are specific structures that shift in time and frequency that are properly captured and characterized by the CNN. Doing so without convolutions requires a network with significantly more capacity, since every shift in time and frequency need to be encoded as a separate pattern. Moreover, the DNN will need every shift of every pattern to appear in the training set a number of times, for it to be learned, while the CNN can learn the same structure out of a sparser set of expositions at different locations in the time-frequency plane. Besides, the CNNs here have more parameters than the DNNs[15], even though we didn't observe significantly better results with larger DNNs. In addition, the DNNs have the same architecture regardless of the input while we adapted the CNNs architecture to each feature characteristic. Finally using both inputs leads to better results as they can take into account different useful information for downbeat tracking.

4) Downbeat likelihood combination: Are more sophisticated combinations useful?

We are currently using the average of the downbeat likelihood computed with each of the four networks to obtain one single estimation. Since this method may seem too simple, we provide a comparison with two other feature combination techniques. We first compare our fusion method with the Adaboost algorithm [56]. A linear combination of the classifiers will be learned by emphasizing the ones that correctly classify an instance mis-classified by the other classifiers. This approach can work well in practice but it is better suited to a problem involving many weak classifiers. We use a learning rate of 0.10 and an ensemble of 100 trees.

Random forests are also tested. A multitude of decision trees are constructed by this ensemble learning method, to predict the class that is chosen by most individual trees [57]. The probability output is computed as the number of observations of the selected class in a tree leaf over the number of trees in the ensemble. Similar to deep learning algorithms, this method often requires a large number of training examples to work well. We use 30 trees and a leaf size of 50. For those two methods, as we did with the harmonic network, we randomly remove some training inputs in order to have a balanced training set.

Results are shown in figure 6(c). We can see a decrease of 1.3 pp with boosting and of 1.6 pp with random forest overall. This is statistically significant. This result may seem surprising at first, but may be explained by the fact that those two algorithms minimize classification error on the training set for each instance individually, while we evaluate the F-measure on full songs after the temporal model. Adaptation to the temporal modeling phase is therefore key and neither boosting nor random forest focus on this part. We found that the boosting method has a similar performance for the three Pop music datasets but is less robust to other sets. The average rule is indeed often more resilient and will not overfit a particular and more represented set. Besides, since we only have 4 features that are relatively strong, comparable in performance and complementary, an average of the result can give a good result already.

5) Temporal model: Is the temporal model useful?

The temporal model is an important part of our system as it allows an increase in performance of 15.4 pp as can

[14]A better SVM model, adapted feature, or probability estimate may be found but this is not the focus of this work.

[15]Since we chose to keep the architecture as in [55] to show the usefulness of a more refined deep learning system.

be seen in the figure 6(d). In the configuration without the temporal model, figure 6(d).1, a downbeat position is included in the final downbeat sequence if its likelihood is above a fixed oracle threshold. This oracle threshold $t = 0.88$ was manually selected to give the best F-measure result. It corresponds roughly to the ratio of downbeats and no downbeats in the datasets.

The important gap in performance can be explained by the fact the downbeat likelihood function is quite noisy as can be seen at the bottom of figure 5. Besides, 9 or 17 tatum long inputs are sometimes too short to give a reliable information about the downbeat position by themselves. However, longer inputs added a significant computational cost and didn't result in a better performance. They were especially counterproductive at the beginning and end of songs. Taking into account the heavily structured nature of music with a temporal model therefore enables the system to obtain a more sensible downbeat sequence.

However, considering that 94% of the songs in the datasets are mainly in 3 or 4 beats per bar and that the estimated tatum is mostly twice the beat[16], it may be interesting to reduce the number of states of the model. To investigate this, we first considered a simple temporal model with only 8 states (for an 8 tatum bar). Its F-measure performance, shown in figure 6(d).2 is of 63.9%. We then considered 14 states (for 8 and 6 tatum bars). The F-measure goes up to 69.9% as seen in figure 6(d).3. However, at the moderate cost of only 18 ms per one minute song, the 80 states temporal model gives a significantly better performance of +2.7 pp. Indeed, while using more states can lead to unlikely states being chosen, it allows enough flexibility to work in different scenarios. Especially when we can emphasize the most common case of two tatums per beat with appropriate transition matrix coefficients. Regarding the ability of the model to deal with different meters, downbeat sequences of three and four beats per bar tracks are fairly well estimated with a mean F-measure of 79.5% and 77.4% respectively. The performance of 2 beats per bar tracks is significantly lower, at 55.0% since they are mostly taken for 4 beats per bar tracks. This is a common ambiguity for algorithms and human listeners that goes beyond the scope of this work. The other meters represent less than 1% of the datasets.

VI. Conclusion and future work

In the present work, we have presented a system that robustly detects downbeat occurrences from various audio music files. Our work is based on the fact that a downbeat is a high level concept depending on multiple musical attributes. We therefore have designed complementary features based on harmony, melody, bass content and rhythm and adapted convolutional neural networks to each feature characteristics. Rhythm in music being highly structured, an efficient and flexible temporal model was also proposed to largely improve the instantaneous downbeat estimation. A comparative evaluation on a large database of 1511 audio files from various music styles shows that our system significantly outperforms four other published algorithms, while keeping a low computational cost. Each step of our system was analyzed to highlight its strengths and shortcomings. In particular, the combination of harmonic and rhythmic deep networks proved to be very good by itself.

While the proposed algorithm obtained the best results overall, it is to note that the recent MIREX campaign[17] highlighted some limitations in our method. The submitted system does not match exactly the one presented here but is based on the same framework. First, the temporal model doesn't deal well with 2 beats per bar songs as in Cretan music for example as it can easily be confused with 4 beats per bar songs. This issue can be fixed by adapting the temporal model to the music convention of a particular style. Second, while it can adapt to music of different traditions such as Turkish Usuls fairly well, music coming from a much more different genre such as Indian Carnatic or with particular rhythm conventions such as Hardcore, Jungle and Drum and Bass requires a training set containing some adapted examples for the system to work better. These remarks are rather expected as a human listener hearing these music styles for the first time will also tend to be lost before training [18]. It shows that bars are not intuitively understood for all music traditions and that designing a more adapted or exhaustive training set is important. However, the number of music tracks for which our system provides a good estimation is still important.

In the future, besides a more refined training set, a network combination procedure adapted to the temporal model and a more robust segmentation step seem promising to improve the current system.

References

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

[2] E. W. Large and M. R. Jones, "The dynamics of attending: how people track time-varying events." *Psychological review*, vol. 106, no. 1, p. 119, 1999.

[3] M. Mauch and S. Dixon, "Simultaneous estimation of chords and musical context from audio," vol. 18, no. 6. IEEE, 2010, pp. 1280–1289.

[4] E. Tsunoo, G. Tzanetakis, N. Ono, and S. Sagayama, "Beyond timbral statistics: Improving music classification using percussive patterns and bass lines," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 1003–1014, 2011.

[5] A. Shenoy and Y. Wang, "Key, chord, and rhythm tracking of popular music recordings," *Computer Music Journal*, vol. 29, no. 3, pp. 75–86, 2005.

[6] N. C. Maddage, "Automatic structure detection for popular music," *IEEE Multimedia*, vol. 13, no. 1, pp. 65–77, 2006.

[7] M. Hamanaka, K. Hirata, and S. Tojo, "Musical structural analysis database based on GTTM," 2014.

[8] J. Paulus and A. Klapuri, "Measuring the similarity of rhythmic patterns," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2002.

[9] J. A. Hockman, J. P. Bello, M. E. P. Davies, and M. D. Plumbley, "Automated rhythmic transformation of musical audio," in *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, 2008, pp. 177–180.

[16]The estimated tatum is at the beat level 16.4% of the time, twice the beat 62.4% of the time, three times the beat 3.5% of the time and four times the beat 17.5% of the time. Other configurations occur less than 0.2% of the time.

[17]http://www.music-ir.org/mirex/wiki/2015:Audio_Downbeat_Estimation_Results

[18]Several audio excerpts are available at http://www.music-ir.org/mirex/wiki/2015:Audio_Downbeat_Estimation for the interested listener.

[10] 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.

[11] J. London, *Hearing in time: Psychological aspects of musical meter*. Oxford University Press, 2012.

[12] 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.

[13] 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.

[14] D. Gärtner, "Unsupervised learning of the downbeat in drum patterns," in *Proceedings of the AES International Conference on Semantic Audio*, 2014.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] M. Gainza, D. Barry, and E. Coyle, "Automatic bar line segmentation," in *Audio Engineering Society Convention 123*. Audio Engineering Society, 2007.

[22] 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.

[23] A. Holzapfel, F. Krebs, and A. Srinivasamurthy, "Tracking the "odd": Meter inference in a culturally diverse music corpus," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2014.

[24] 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.

[25] E. Battenberg, "Techniques for machine understanding of live drum performances," Ph.D. dissertation, Electrical Engineering and Computer Sciences University of California at Berkeley, 2012.

[26] L. Nunes, M. Rocamora, L. Jure, and L. W. P. Biscainho, "Beat and downbeat tracking based on rhythmic patterns applied to the uruguayan candombe drumming," in *Proceedings of the 16th Int. Conference on Music Information Retrieval (ISMIR)*, 2015.

[27] F. Krebs, A. Holzapfel, A. T. Cemgil, and G. Widmer, "Inferring metrical structure in music using particle filters," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 23, no. 5, pp. 817–827, 2015.

[28] P. Grosche and M. Müller, "Tempogram Toolbox: MATLAB tempo and pulse analysis of music recordings," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR), late breaking contribution*, 2011.

[29] M. A. Alonso-Arevalo, "Extraction d'information rythmique à partir d'enregistrements musicaux," Ph.D. dissertation, École Nationale Supérieure des Télécommunications, 2006.

[30] E. Hannon, J. Snyder, T. Eerola, and C. Krumhansl, "The role of melodic and temporal cues in perceiving musical meter," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 30, no. 5, p. 956, 2004.

[31] P. Pfordresher, "The role of melodic and rhythmic accents in musical structure," *Music Perception*, vol. 20, no. 4, pp. 431–464, 2003.

[32] 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)*, vol. 41, 2005, pp. 304–311.

[33] J. C. Brown and M. S. Puckette, "An efficient algorithm for the calculation of a constant q transform," *The Journal of the Acoustical Society of America*, vol. 92, no. 5, pp. 2698–2701, 1992.

[34] 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.

[35] A. Vedaldi and K. Lenc, "Matconvnet – convolutional neural networks for matlab," *CoRR*, vol. abs/1412.4564, 2014.

[36] 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.

[37] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[38] Y. LeCun, K. Kavukcuoglu, and C. Farabet, "Convolutional networks and applications in vision," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2010, pp. 253–256.

[39] J. Thomassen, "Melodic accent: Experiments and a tentative model," *Journal of the Acoustical Society of America*, vol. 71, p. 1596, 1982.

[40] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," *International Journal of Data Warehousing and Mining*, vol. 3, pp. 1–13, 2007.

[41] 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.

[42] 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)*, vol. 2, 2002, pp. 287–288.

[43] 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.

[44] 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)*, vol. 3, 2003, pp. 229–230.

[45] "www.ballroomdancers.com."

[46] "http://www.quaero.org/."

[47] "http://isophonics.net/datasets."

[48] F. Krebs, S. Böck, and G. Widmer, "An efficient state-space model for joint tempo and meter tracking," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2015, pp. 72–78.

[49] J. P. Bello and J. Pickens, "A robust mid-level representation for harmonic content in music signals," vol. 19, 2005.

[50] S. Böck, F. Krebs, and M. Schedl, "Evaluating the online capabilities of onset detection methods." in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2012.

[51] D. Temperley and T. d. Clercq, "Statistical analysis of harmony and melody in rock music," *Journal of New Music Research*, vol. 42, no. 3, pp. 187–204, 2013.

[52] B. D. Giorgi, M. Zanoni, A. Sarti, and S. Tubaro, "Automatic chord recognition based on the probabilistic modeling of diatonic modal harmony," in *Proceedings of the International Workshop on Multidimensional Systems (nDS)*, 2013.

[53] N. Degara, E. Argones Rua, A. Pena, S. Torres-Guijarro, M. E. P. Davies, and M. D. Plumbley, "Reliability-informed beat tracking of musical signals," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 290–301, 2012.

[54] T. F. Wu, C. J. Lin, and R. C. Weng, "Probability estimates for multi-class classification by pairwise coupling," *The Journal of Machine Learning Research*, vol. 5, pp. 975–1005, 2004.

[55] S. Durand, J. P. Bello, B. David, and G. Richard, "Downbeat tracking with multiple features and deep neural networks," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[56] Y. Freund and R. Schapire, "A short introduction to boosting," *Journal-Japanese Society For Artificial Intelligence*, vol. 14, no. 771-780, p. 1612, 1999.

[57] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.