Contents

General notes

- Goal of supervised learning: predict not yet observed input
- Motivation: "*It is probable that no universal beat tracking model exist which does not utilise a switching model to recognize style and context prior to application.*" [1, Collins2006]
- Instead of calculating the STFT as for input featuress discover a good set of features by representation learning. ("*Learned representations often result in much better performance than can be obtained with hand-designed representations.*" [2, Goodfellow2016])
- Common association between sequence modeling and recurrent networks: "*Given a new sequence modeling task or dataset, wich architecture should one use?*" [3, Bai2018]
- Certain convolutional architectures can reach state-of-the-art accuracy in audio synthesis (e.g. Google WaveNet [4, Oord2016])
- Do TCNs outperform LSTM architectures in this particular sequence modeling task?
- Dataset size: "*[...] a rough rule of thumb is that a supervised deep learning algorithm will gernerally achieve acceptable performance with around 5000 labeled examples per category and will match or exceed human performance when trained with a dataset containing at least 10 million labeled examples.*" [2, Goodfellow2016]
- Contrast of the two statements:
  - "*Feature extraction is an essential step towards efective and accurate beat/downbeat positions extraction.*" [5, Khadkevich2012]
  - "*The system shows state-of-the-art beat and downbeat tracking performance on a wide range of different musical genres and sryles. It does so ny avoiding hand-crafted features such as harmonic changes, or rhythmic patterns, but rather leans the relevant features directly from audio.*" [6, Boeck2016b]

1 Introduction

- A fundamental research topic in music information retrieval is the automatic extraction of beat locations from music signals.
- The aim of a beat tracker is to recover a sequence of time instants from a musical input that are consitent with the times when a human might tap their foot. [7]
- Beat tracking is an important initioal step in computer emulation of human musc understanding, since beats are fundamental to the perception of (Western) music.
- The goal of beat tracking is to construct a computational algorithm capable of extracting a symbolic representation which correspond to the phenomenal experience of beat or pulse in a human listener.

1.1 Difficulties of Beat Tracking

It is difficult to reliably extract high-level rhythm related features from musical excerpt having properties such as [Quinton2016]

- The principal reason that beat tracjing is intrinsically difficult is that is is the problem of inferring an original beat strucutre that is not expressed explicitly. The degree of beat tracking difficulty is therefore not determided simply by the number of musical instruents performing a musical piexe it depends on how explicitly the beat structure is expressed in the piece. However, it is very difficult to measure its explicitness because it is influenced from various aspects of the songs.
- The larger the number of syncopation, the greater the difficulty of beat tracking (quantitative measure: power-difference measure [8])
- The main reason that different tendencies with regard to the explicitness with which their beat structure is indicated.
- There is not necessarily a specific sound that directly indicates tho position of beats. In fact, a musical beat may not directly correspond to a real sound, there may even be no signal on a beat.
- absence of a clear rhythmic structure (Classical music)
- soft onsets
- blurred note transitions (e.g. classigal music dominated by string instruments)
- heavy syncopation
- expressive timing (e.g. rubato playing)
- the problem of 'octave errors' (detecting double or half time the rate of the ground truth)

- acoustiv signals consit of sound of various kinds of instruments
- the onsets of notes are difficult to obtain, unlike the case of MIDI signals, where there is no such problem.
- a beat may not directly correnspond to a real sound. It is a perceptal concept that human feels in music.
- Multiple interpretations of beats are possible at any given time
- There is no simple relationship beteen polyphonic complexity - the number and timbres of notes played at a single time - in apiece of music, and its rhythmic complexity or pulse complexity [9]. There are pieces and sryles of music which are texturally and timbrally complex, but have straighforward, perceptual simple rhythms; and ther also exist music which deal in less complex textures but are more difficult to rhythmically understand an describe.
- The complexity of grouping harmonic partials together to form notes, and determining the onst times of those.
-

1.2 Applications of Beat Tracking

- Beat tracking can be used to automate the time-consuming tasks that must be completed in order to synchronize events with music.
- Video and audio editing (visual track can be automatically synchronized with an audio track using beat tracking)
- Stage light control. In live performances, beat tracking is useful in the control of stage lighting by a computer. For instance, various properties of lighting such as color, brightness, direction, and effect can be changed in time to the music.
- Musical interaction systems [10]
- Content-based audio effects
- Beat-driven reat-time computer graphics.
- Temoral segmentation to higher level MIR tasks such as chord extraction
- Structural segmentaion of audio
- Music similarity
- Beat tracking can provide computers the ability to participate intelligently in live performances in real time and join the ensemble.

2 History of Beat Tracking

A brief summary of the related work that forstered the development of automatic beat tracking for musical audio.

- Overview: Signal processing → data-drivven approaches
- The task of beat tracking has been well established over the last twenty-five years.

In 1994, Goto and Muraoka [11] introduced the first considerable beat tracking system which could process music played on ensembles of a variety of instruments. However, they restricted their system to rock and pop music in which drums maintain the the beats. The system leverages the fact that for a large class of popular music, a bass drum and a snare drum usually occur on the strong and weak beats, respectively. It manages multiple agents that track beats according to different stategies in order to examine multiple hypothesis in parallel. All hypotheses are gathered and the most reliable one is selected as the output. This enables the system to follow beats without losing track if them, even if some hypothesis became wrong. Assumptions were made; between 65 and 185 BPM, time-signature is 4/4, tempo stays almost constant. (Development to a beat tracking system for music with or without drum-sounds [12]. Also uses chord changes.)

From psychoacoustic demonstration on beat perception, Scheirer [9] figured out that amplitude envelopes from a small number of broad frequency channels are sufficient information to rhythmically analyse musical content. He concludes that a rhythmic processing algorithm shoud treat frequency bands separately, combining results at the end, rather than attempting to perform beat tracking on the sum of filterbank outputs. This leads him to use a small number of bandpass filters and banks of parallel comb filters to function as tuned resonators to first analyse the tempo of a musical signal. Then, in the next processing step the phase of the musical siganl is extracted by examining the internal state of the delays of the comb filters. Finally, the pahes and the period is used to estimate the beat times as far into the future as desired.

3 Basic Principles of Human Rhythm Perception

- Many works in musicpsychology attempt to group musical onsets together into a rhythmic context, i.e., to construct a model which subsubes multiple onsets separated in time into a rhythmic clock, hierachy, grouping, or oscillatory model.
- Experiments on the tempo sensitivity on humans have shown that the ability to notice tempo changes is proportional to the tempo, with the JND (just noticable difference) beeing around 2-5% [13].
- Although in the brains of performers music is temporally organized according to its hieaechical beat structure, this structure is not explicitly expressed in music; it is implied in the relations among variuos musical elements which are not fixed and which are dependent on musical genres or poeces.
- A listener who cannot identify chord names can nevertheless percieve chord changes.
- The experience of rhythm involves movement, regularity, grouping, and yet accentutation and differentiation [14].
- There is no ground thuth for rhythm to be found in simple measurement of an acoustic signal. The only ground truth is what human listeners agree to be the rhythmic aspects of the musical contetent of that signal
- As contrasted with rhythm in general, beat and and pulse correspond inly th the sense of equally spaced temporal units [14]. Where meter and rhythm associate with qualities of grouping, hierarchy and a strong/weak dichitimy, pulses in apiece of music are only periodic at a simple level.
- The beat of a piece of music is the sequence of equally spaced phenomenal impulses which define a tempo for the music.
- The frequency pf the pulse in a rhythmic musical signal is the tempo or the rate of the rhythm, and the phase of the pulse indicates where the downbeat of the rhythm occurs. That is, the times at which a pulse occurs can be defined to have zero phase, and thus the points in time exactly in-between pulses hace phase of π radians.
- Human pitch recognition is only sensitive to signal phase under certain unusual conditions. rhythmic response is crucially aphased phenomenon - tapping on the beat is not at all the same as tapping against the beat. or slightly ahead of or behind the beat, even ig the frequency of tapping is accurate [9]
- From psychoacoustic demonstration on beat perception it can be shown that certain kinds of signal manipulations and simplifications can be performed without affecting the perceived pulse content of a musical signal. An amplitude-modulated noise constructed by vocoding a white noise signal with the subband envelopes

of a musical signal is sufficient to extract pulse and meter. The simplified signal is created by performing a frequency analysis of the original signal by processing it through a filterbank of bandpass filters, or grouping FFT bins together. Thus it seems that separating the signal into subbands and maintaining the subband envelopes separately is necessary to do accurate rhythmic processing. This fact leads to the hypotheses that some sort of cross-band ryhtmic integration, not simply summation across frequency bands, is performed by the auditory system to perceive rhythm [9].

- Studies of Povel and Essnes [15] have demonstrated that beat perceptin may be explianed with a model in which a perceptual clock is aligned with the accent structure of the input. The model relies heavily on structural qualities of the input, such as a sophisticated model of temporal accent, to function.

4 Foundations of Machine Learning

4.1 Sequence Modeling

Given an input sequence $x_{1:T} = x_1, \ldots, x_T$ with sequence length T, a sequence model is any function $f : \mathcal{X}^T \to \mathcal{Y}^T$ such that

$$y_{1:T} = y_1, \ldots, y_T = f(x_1, \ldots, x_T) \tag{1}$$

where y_t should only depend on $x_{1:t}$ and not on $x_{t+1:T}$, i.e., no leakage of information from the future. This causality constraint is essential for autoregressive modeling. We use $x_{1:T} = x_1, \ldots, x_T$ to denote a input sequence of length T, where vector $x_t \in \mathbb{R}^p$ is the input at time step t. Thus $x_{1:T} \in \mathbb{R}^{T \times p}$.

The goal of learning in the sequence modeling setting is to find a network f that minimizes some expected loss between the actual outputs and the predictions,

$$L\big(y_1, \ldots, y_T,\, f(x_1, \ldots, x_T)\big) \stackrel{!}{=} \min, \tag{2}$$

where the sequences and outputs are drawn according some distribution.

For most deep learning practitioners, sequence modeling is synonymous with recurrent networks. For example the sequence modeling chapter in a standard reference on deep learning is titled "Sequence Modeling: Recurrent and Recursive Nets" [2] capturing the common association of sequence modeling and recurrent architectures. Recent results indicate that convolutional architectures can outperform recurrent networks. Concluding from an empirical evaluation of generic convolutional and recurrent networks for sequence modeling, Bai et al. [3] assert that the common association between sequence modeling and recurrent networks should be reconsidered, and convolutional networks should be regarded as a natural starting point for sequence modeling tasks.

- Sequence modeling include polyphonic music modeling[4], word- and character-level language modeling.
- settings such as auto-regressive prediction, where we try to predict some signal given its past, by setting the target output to be simply the input shifted by ome time step

4.2 Data Representation

- Many artificial intelligence tasks can be solved by designing the right set of features to extraxt for that task, then providing theses features to a simple machine learning algorithm. [2]

4.3 Architectures

- Given a sequence modeling task or dataset, which architecture should one use?
- Bai et al. [3] show that a simple convolutional architectures outperforms canonical recurrent networks such as LSTMs across a diverse range of tasks and datasets, while demonstrating longer effective memory. Is it also the case in beat tracking?

4.3.1 Recurrent Neural Networks

In deep learning, a recurrent neural network (RNN) is a class of artificial neural networks where connections between nodes form a directed graph along a temporal sequence.

- Recurrent networks are dedicated sequence models that maintain a vector of hidden activations that are propagated throgh time [16, 17, 18].
- This family of architectures has gained tremendois popilarity die to prominent applications to language modeling and machine translation.
- The intuitive appeal of recurrent modeling is that the hidden state can act as a representations of everything that has been seen so far in the sequence.
- Basic RNN architectures are notoriously difficult to train and more elaborate architectures are commonly used instead, such as the LSTM [19] and the GRU [20].

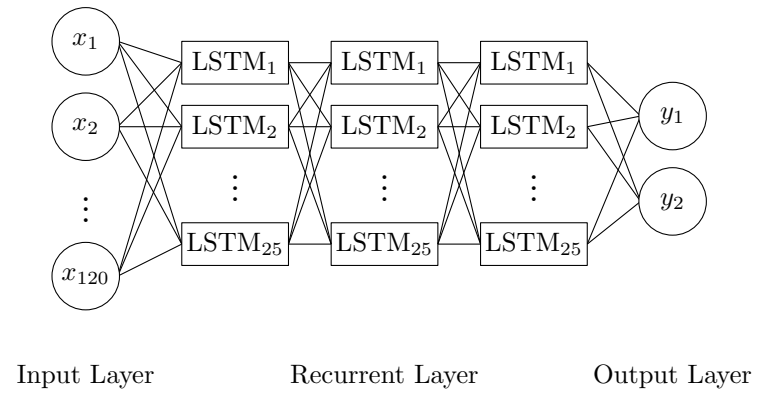

Figure 1: Model architecture in [21].

4.3.2 Convolutional Neural Networks

A convolutional neural network (CNN) is a class of deep neural networks, most commonly applied to analyzing visual imagery.

- Convolutional neural networks (CNNs) [22] have been applied to sequences for decades. They were used prominently for speech recognition in the 80s and 90s.

4.3.3 Temporal Convolutional Netwoks

A temporal convolutional network (TCN) represents a special kind of convolutional neural network and is informed by recent convolutional architectures for sequential data [3]. It is designed from first principles and combines simplicity, autoregressive prediction,

and very long memory. In comparison to WaveNet [4], the TCN does not employ skip connections across layers (no conditioning, context stacking, or gated activations).

The TCN is based upon two principles: 1) the convolutions are casual, i.e, no information leakage from future to past; 2) the architecture can take a sequence of any length and map it to an output sequence of the same length just as with an RNN. To achieve the first point, the TCN uses a 1D fully-convolutional network architecture [23], where each hidden layer is the same length as the input layer. To accomplish the second point, the TCN uses causal convolutions, i.e., convolutions where an output at time t is convolved only with elements from time t and earlier in the previous layer.

Simple causal convolutions have the disadvantage to only look back at history with size linear in the depth of the network. To circumvent this fact, the architecture employs dilated convolutions that enable an exponentially large receptive field. More formally, for a 1-D sequence input $\mathbf{x} \in \mathbb{R}^T$ and a filter $f : \{0, \dots, k-1\} \to \mathbb{R}$, the delated convolution operation F on element s of the sequence is defined as

$$F(s) = (\mathbf{x} *_d f)(s) = \sum_{i=0}^{k-1} f(i)\, \mathbf{x}_{s-d \cdot i} \tag{3}$$

where $d = 2^\nu$ is the dilation factor, with ν the level of the network, and k is the filter size. The term $s - d \cdot i$ accounts for the direction of the past. Dilation is equivalent to introducing a fixed step between every two adjacent filter taps, as it can be seen in Fig. 2. Using larger dilation enables an output at the top level to represent a wider range of inputs, thus effectivly expanding the receptive field of a CNN. There are two ways to increase the receptive field of a TCN: choosing lager filter sizes k and increasing the dilation factor d, since the effective history of one layer is $(k-1)\, d$.

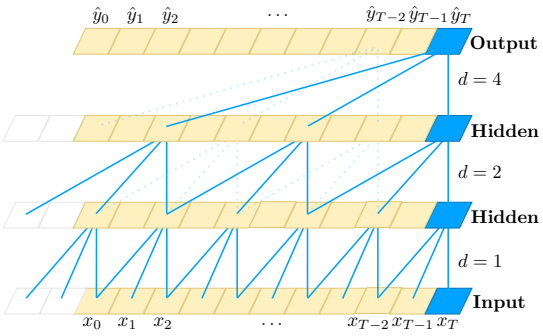

Figure 2: A dilated casual convolution with dilation factors $d = 1, 2, 4$ and filter size $k = 3$ [3].

Another architectual element of a TCN are residual connections. In place of a concolutional layer, TCNs employ a generic residual module. Each residual block contains a branch leading out to a series of transformations $\mathcal{F}$, whose outputs are added to the input $\mathbf{x}$ of the block,

$$o = \text{Activation}\big(\mathbf{x} + \mathcal{F}(\mathbf{x})\big). \tag{4}$$

This effectively allows layers to learn modifications to the identity mapping rather than the entire transformation, which has been shown to benefit deep neural networks [24]. Especially for very deep networks stabilization becomes important, for example, in the case where the prediction depends on a large history size ($> 2^{12}$) with a high-dimensional input sequence.

A residual block has two layers of dilated causal convolutions and recfied linear units (ReLU) as non-linearity, shown in Fig. 3. For normalization, weight normalization [25] is applied to the convolutional filters. In addition, a spatial dropout [26] is added after each dilated convolution for regularization, i.e., at each training step, a whole channel is zeroed out.

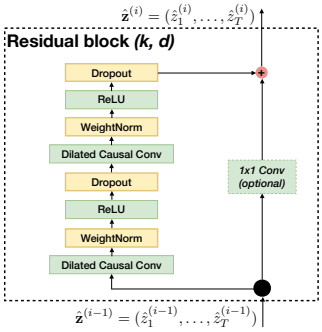

Figure 3: TCN residual block [3].

- the convolutions are casual, that means no information leakage from future to past (compare to bidirectional RNN).
- long effective history sizes, i.e., the ability for networks to look very far into the past to make a prediction (receptive field?), by using a combination of very deep networks (augmented with residual layers) and dilated convolutions
- this allows layers to learn modifications to the identity mapping rather than the entire transformation, which has been shown to benefit deep neural networks (where?)
- within a residual block there exist two layers of dilated causal convolution and a rectified linear unit (ReLU) [27, Nair2010] as a non-linearity
- for normalization a weight normalization [25, Salimans2016] is applied to the convolutional filters
- additionaly spatioal dropout [26, Srivastava2014] is added after each dilated convolution for regularization (at each training step, a whole channel is zeroed out)
- The TCN architecture appers not inly more accurate than canonical recurrent networks such as LSTMs and GRUs, but also simpler and clearer.

4.3.4 Trellis Networks

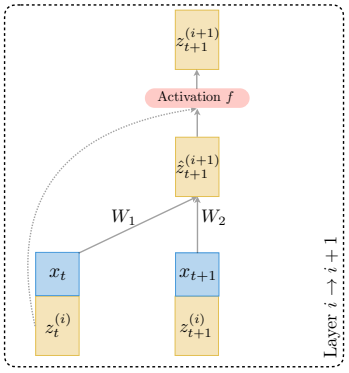

Figure 4: TrellisNet at an atomic level.

Input: $x_{1:T} = x_1, \ldots, x_T, \quad x_t \in \mathbb{R}^p, \quad x_{1:T} \in \mathbb{R}^{T \times p}$
with sequence length T input dimensionality p

Output: $y_{1:T} = y_1, \ldots, y_T = G(x_1, \ldots, x_T)$
with function $G : \mathcal{X}^T \to \mathcal{Y}^T$

Hidden: $z_t^{(i)} \in \mathbb{R}^q$

LogSoftmax: A LogSoftmax normalization is added the last layer of the neural network to obtain the log-probabilities for the differnent classes.

$$\text{LogSoftmax}(x_i) = \log\left(\frac{\exp(x_i)}{\sum_j \exp(x_j)}\right) \tag{5}$$

4.4 Performance Measure

Loss Function Cross entropy is defined for two probability distributions p and q as

$$H(p, q) = -\sum_{x \in \mathcal{X}} p(x) \log q(x). \tag{6}$$

In machine learning cross entropy can be used as a loss funciton to measures the performance of a classification model. The probability p_i is the true label (binary indicator 0 or 1), where as the distribution q_i is the predicted value of the current model.

4.5 Regularization Techniques

4.6 Optimization

- Regularization with Dropout or Weight Decay
- check out cyclical learning rates [28][Smith2017]

Estimate generalization error with a validation set during training. Stop training when the error on the validation set rises (early stopping).

4.7 Hyperparameter Optimization

- optimzie hyperparameters with nevergrad
- instrumentation: turn a piece of code with parameters into a function defined on an n-dimensional continous data space

4.8 Multi Task Learning

- It is the information in these extra training signals that helps the hidden layer learn a better internal representation for the door recognition domain, and this better representation in turn helps the net better learn to recognize [Caruana1997]
- Information in the extra tasks is helping the hidden layer learn a better internal representation
-

4.9 Valididation

Test set method Split the observarions into to disjunct subsets

$$\text{observations} \begin{cases} \text{training data } \left\{\left(\mathbf{x}^{(\alpha)}, \mathbf{y}_T^{(\alpha)}\right)\right\}, \alpha \in \{1, \ldots, p\} \\ \rightarrow E^T \text{ selects model parameters} \\ \\ \text{test data } \left\{\left(\mathbf{x}^{(\beta)}, \mathbf{y}_T^{(\beta)}\right)\right\}, \beta \in \{1, \ldots, q\} \\ \rightarrow \widehat{E}^G \text{ estimates generalization error} \end{cases}$$

n-fold cross-validation

5 Beat Tracking System

From an overall perspective, the proposed automatic beat tracking system comprises three major phrases: data preprocessing, feature learning and temporal decoding.

Figure 5: Beat tracking system signal flow

Seuqence-to-sequence prediction:

- the entire input sequence can be used to predict each output (not feasible in real time processing)

Binary classification problem:

- beat (class 1)
- no beat (class 0)

5.1 Dataset

For training and evaluation we use the datasets listed in Table 1.

- The ideal outcome of the annotation is an unambigous representation of the start points of musical events
- Due to uncertainties in the annotation process, for many types of inpit signals (especially multi-instriment exerpts) it may not possible to determine onset locations with greater precision than 50 ms. [29]

Table 1: Datasets used for training.

Dataset	**files**	**length**
Ballroom [30, 31]	685	5 h 57 m
GTZAN [32, 33]	1000	8 h 20 m
Hainsworth [34]	222	3 h 19 m
SMC [35]	217	2 h 25 m
Total		h m

5.2 Data Preprocessing

- "Is it important to use several features?" [Durand2015]
- Two different types of input features: Onset events and harmonic changes
- Assumtion: Most harmonic changes occur inside a piece of music are located at metric bars [5, Khadkevich2012]
- in complex polyphonic mixtures of music, simultaneously occurring events of high intensities lead to masking effects that prevent any observation of an energy increase of a low intensity onset. To circumbent thesse masking effects, detection functions were proposed that analyze the signal in a bandwise fashion to extract transients occurring in certain frequency regions of the signal [Grosche2011]

The Beat Tracking task requires annotations in the form of time instants of beats from a musical excerpt. Additionally, the tempo, metrical level of the beat and downbeat positions can also be annotated for related tasks like downbeat tracking/ meter tracking/ tempo tracking etc.

The method for obtaining ground truth annotations depends on whether the aim is to identify descriptive beat locations or to replicate a human tapping response. In the former case, an initial estimate of the beat locations can be obtained by recording tap times for a given musical excerpt and iteratively modifying and auditing the obtained beat positions while in the latter case, the ground truth can be completely defined by the tapping response [36].

The audio excerpt in time domain and its spectrogram can be visualised using tools like Sonic Visualizer. Beat locations can first be obtained by recording the tap locations the musical excerpt and then manually correcting these locations for exact onsets. Annotators can also follow a semi-automatic process, for example in [2], where beats and downbeats were first obtained through the ircambeat software and then errors were manually corrected by human annotators.

After preprocessing the audio we obtain the observation set $O = \left\{\mathbf{x}^{(\alpha)}, \mathbf{y}_T^{(\alpha)}\right\}_{\alpha=1}^{p}$, with p samples in total.

- Data augmentation (transformation or adding noise) $\Rightarrow$ prediction gets robust against transformation and noisy signals
- output: y probability of beat instant per time frame

The first step in the approach consists of preprocessing the original data. Data preprocessing refers to all transformations on the raw data before the resulting training set is fed to the machine learning algorithm. It includes different methods such as normalization, transformation and feature extraction.

The data set contains raw pulse code modulated (PCM) audio signals stored as WAV files. For the sake of consistency and also to reduce computational complexity the audio signal is resampled at a sampling rate $f_s = 44.1\,\text{kHz}$ and converted to a monaural signal by averaging both stereo channels.

- Slice audio $x(t)$ into frames $x_n(t)$, $n = 1, 2, \ldots, N$, where N is the number of frames
- Compute complex spectrogram $X(n, k)$ with FFT
- Complex spectrogram is converted to the power spectrogram $S(n, k) = |X(n, k)|^2$
- Mel-spectrogram $M(n, m) = \log\left(S(n, k) \cdot F(m, k)^T + 1.0\right)$

Short-time Fourier transform:

$$X(n, k) = \sum_{m=0}^{N-1} w(m)\, x(m + nh)\, e^{-2\pi jk/N} \tag{7}$$

Spectrogram:

$$S(n, k) = |X(n, k)|^2 \tag{8}$$

Separation of magnitude and phase:

$$X(n, k) = |X(n, k)|\, e^{j\phi(n,k)} \tag{9}$$

5.2.1 Chroma Representation

- Chords are more likely to change in beat times than on other positions. Chords are more likely to change at the beginnings of measures than at other positions of the beat.
- A chromagram comprises a time-series of chroma vectors. which represent harmonic content at a specific time in the audio.
- Chromagrams are concise descriptors of harmony because they encode tone quality and neglect tone height.
- CRP (chroma DCT-reduced log-pitch) features have significant amount of robustness to changes in timbre an instrumentation [37, Mueller2010]
- Employ data-driven approach to extract chromagrams that specifically encode content relevant to harmony
- Chroma features are noisy in their basic formulation be- cause they are affected by various interferences: musical instruments produce overtones in addition to the funda- mental frequency; percussive instruments pollute the spec- trogram with broadband frequency activations (e.g. snare drums) and/or pitch-like sounds (tom-toms, bass drums); different combinations of instruments (and different, possibly genre-dependent mixing techniques) create different timbres and thus increase variance [7,20] [38]

5.3 Feature Learning

5.4 Temporal Decoding

We use a probabilistic dynamic model to exploit the seqential structure of music, generally resulting in a more robust estimation.

As a postprocessing step we make use of a dynamic Bayesian network (DBN) which jointly infers tempo an phase of the beat.

Hidden variables:

- ω - the tempo
- ϕ - the position inside the beat period

In order to infer the hidden variables from an audio signal, we specify three entities:

- **transition model**: describes the transitions between the hidden variables
- **observation model**: takes the beat activations from the neural network
- **initial distribution**: encodes prior knowledge about the hidden variables
  - test probability model for given tempo estimation

5.4.1 Transition Model

- The number of observations M per beat at tempo T in BPM is defined as

$$M(T) = \frac{60}{T} f_r \quad \left[\frac{\text{frames}}{\text{beat}}\right] \tag{10}$$

- The beat position state is dependent on the tempo by using exactly one state per audio frame.
- $\Phi \in \{1, 2, \ldots, M(T)\}$ denotes the position inside the beat period (pib)
- The tempo space corresponds to integer valued beat positions in the interval $[M(T_{\max}), M(T_{\min})]$, with

$$N_{\max} = M(T_{\min}) - M(T_{\max}) + 1 \tag{11}$$

  different tempo states.

- The tempo in beat positions per time frame $\dot{\Phi} \in \{M(T_{\max}), M(T_{\max})+1, \ldots, M(T_{\min})\}$

- For example:

$$T_{\min} = 56\,\text{BPM} \;\rightarrow\; M(T_{\min}) = 107$$
$$T_{\max} = 215\,\text{BPM} \;\rightarrow\; M(T_{\max}) = 28$$

$$\Phi \in \{1, 2, \ldots, 107\}$$
$$\dot{\Phi} \in \{28, 29, \ldots, 107\}$$

- Transitions to each tempo is allowed but only at beat times

$$\omega_k = \begin{cases} \omega_{k-1}, & P(\omega_k|\omega_{k-1}) = 1 - p_\omega \\ \omega_{k-1} + 1, & P(\omega_k|\omega_{k-1}) = \frac{p_\omega}{2} \\ \omega_{k-1} - 1, & P(\omega_k|\omega_{k-1}) = \frac{p_\omega}{2} \end{cases} \tag{12}$$

- The probability of tempo change is heuristically set to $p_\omega = 0.002$
- Higher-level or domain specific knowledge could be used to set this parameter. For example in rock or pop music, the beat is usually quite steady, so a samll value for p_ω would be quite appropriatre, while for classical music, particularly styles including many tempo changes, a higher value would be more optimal.

5.4.2 Observation model

6 Evaluation

- annotations tapped at different metrical levels in beat tracking
- a evaluation method should adequately contend with the inherent uncertainty and/or ambiguity while providing a measurement of performance which is both meaningful and easy to interpret. [36]
- objective methods for beat tracking evaluation compare the output beat times from a beat tracking algorithm against one or more sequences of ground truth annotated beat times. [36]
-

6.1 Performance measures

Set of annotated beat times $\mathcal{A} = \{a_1, \ldots, a_A\}$
Set of predicted beat times $\mathcal{B} = \{b_1, \ldots, b_B\}$

True positive if beat is in ± 70 ms rage of annotated beat
tp = number of true positives
fp = number of false positives (extra detections)
fn = number of false negatives (missed detections)

Precision and recall:

$$\text{precision} = \frac{tp}{tp + fp}\,, \qquad \text{recall} = \frac{tp}{tp + fn}$$

F-measure:

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{2\,tp}{2\,tp + fp + fn}$$

P-score: Define two sequences T_a and T_b as

$$T_a(n) = \begin{cases} 1, & \text{if } n \in \mathcal{A} \\ 0, & \text{otherwise} \end{cases}\,, \qquad T_b(n) = \begin{cases} 1, & \text{if } n \in \mathcal{B} \\ 0, & \text{otherwise} \end{cases}$$

$$\text{P-score} = \frac{\sum_{m=-w}^{w} \sum_{n} T_a(n)\, T_b(n+m)}{\max(A, B)}\,, \qquad \text{where } w = 0.2\,\text{median}(\Delta_a)$$

6.2 Saliency maps

- Artificial neural networks often give good results, but it is difficult to understand what they learned, or on which basis they generate their output.
- compute saliency maps using guided back-propagation [25]

6.3 Network training

- To achieve better results, we could use DNN ensembles instead of a single DNN. We could ensure that the network sees data for which its pre- dictions are wrong more often during training, or similarly, we could simulate a more balanced dataset by showing the net super-frames of rare chords more often [38]

6.4 Labeling

- In an editor (Sonic Visualizer) that enables a user to mark beat positions in a digitalized audio signal while listening to the audio and watching its waveform.
- The positions can be finely adjusted by playing back the audio with click tones at beat times.

7 Conclusion

- Limitations in both the amount of training data and the variable quality of the annotations. Perform data augmentation.

8 Appendix

8.1 Hyperparameter

- window function (e.g. Hamming window)

References

[1] Nick Collins. Towards a style-specific basis for computational beat tracking. 2006.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[3] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[4] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[5] Maksim Khadkevich, Thomas Fillon, Gaël Richard, and Maurizio Omologo. A probabilistic approach to simultaneous extraction of beats and downbeats. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 445–448. IEEE, 2012.

[6] Sebastian Böck, Florian Krebs, and Gerhard Widmer. Joint beat and downbeat tracking with recurrent neural networks. In *ISMIR*, pages 255–261, 2016.

[7] Daniel P. W. Ellis. Beat tracking by dynamic programming. *Journal of New Music Research*, 36(1):51–60, mar 2007.

[8] Masataka Goto and Yoichi Muraoka. Real-time beat tracking for drumless audio signals: Chord change detection for musical decisions. *Speech Communication*, 27(3-4):311–335, 1999.

[9] Eric D. Scheirer. Tempo and beat analysis of acoustic musical signals. *Journal of the Acoustical Society of America*, 103(1):588–601, 1998.

[10] Andrew Robertson and Mark D. Plumbley. B-keeper: A beat-tracker for live performance. In *NIME*, 2007.

[11] Masataka Goto and Yoichi Muraoka. A beat tracking system for acoustic signals of music. In *Proceedings of the second ACM international conference on Multimedia*, pages 365–372. ACM, 1994.

[12] Masataka Goto. An audio-based real-time beat tracking system for music with or without drum-sounds. *Journal of New Music Research*, 30(2):159–171, 2001.

[13] Carolyn Drake and Marie-Claire Botte. Tempo sensitivity in auditory sequences: Evidence for a multiple-look model. *Perception & Psychophysics*, 54(3):277–286, 1993.

[14] Stephen Handel. Listening. *An introduction to the perception of auditory events, Cambridge, MA*, 1989.

[15] Dirk-Jan Povel and Peter Essens. Perception of temporal patterns. *Music Perception: An Interdisciplinary Journal*, 2(4):411–440, 1985.

[16] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

[17] Paul J Werbos et al. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[18] Alex Graves. Supervised sequence labelling. In *Supervised sequence labelling with recurrent neural networks*, pages 5–13. Springer, 2012.

[19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[20] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[21] Sebastian Böck and Markus Schedl. Enhanced beat tracking with context-aware neural networks. In *Proceedings of the 14th International Conference on Digital Audio Effects (DAFx-11*, 2011.

[22] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[23] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[25] Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems*, pages 901–909, 2016.

[26] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[27] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[28] Leslie N Smith. Cyclical learning rates for training neural networks. In *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*, pages 464–472. IEEE, 2017.

[29] Pierre Leveau and Laurent Daudet. Methodology and tools for the evaluation of automatic onset detection algorithms in music. In *In Proc. Int. Symp. on Music Information Retrieval*. Citeseer, 2004.

[30] Fabien Gouyon, Anssi Klapuri, Simon Dixon, Miguel Alonso, George Tzanetakis, Christian Uhle, and Pedro Cano. An experimental comparison of audio tempo induction algorithms. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(5):1832–1844, 2006.

[31] Florian Krebs, Sebastian Böck, and Gerhard Widmer. Rhythmic pattern modeling for beat and downbeat tracking in musical audio. In *ISMIR*, pages 227–232, 2013.

[32] George Tzanetakis and Perry Cook. Musical genre classification of audio signals. *IEEE Transactions on speech and audio processing*, 10(5):293–302, 2002.

[33] Ugo Marchand and Geoffroy Peeters. Swing ratio estimation. pages 423–428, 2015.

[34] Stephen W Hainsworth and Malcolm D Macleod. Particle filtering applied to musical tempo tracking. *EURASIP Journal on Advances in Signal Processing*, 2004(15):927847, 2004.

[35] A. Holzapfel, M. Davies, J. R. Zapata, J. Oliveira, and F. Gouyon. Selective sampling for beat tracking evaluation. *IEEE Transactions on Audio, Speech and Language Processing*, 20(9), 2012.

[36] Norberto Degara Matthew E. P. Davies and Mark D. Plumbley. Evaluation methods for musical audio beat tracking algorithms. Technical report, Centre for Digital Music, Queen Mary University of London, 2009.

[37] Meinard Muller and Sebastian Ewert. Towards timbre-invariant audio features for harmony-based music. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):649–662, 2010.

[38] Filip Korzeniowski and Gerhard Widmer. Feature learning for chord recognition: The deep chroma extractor. *arXiv preprint arXiv:1612.05065*, 2016.