Unsupervised feature learning for Music Structural Analysis

Michele Buccoli, Massimiliano Zanoni, Augusto Sarti, Stefano Tubaro
Dipartimento di Elettronica,
Informazione e Bioingegneria,
Politecnico di Milano,
piazza Leonardo da Vinci 32
Milano, Italy
Email: {name.surname}@polimi.it

Davide Andreoletti
Networking Laboratory
University of Applied Sciences
of Southern Switzerland
Manno, Switzerland
Email: davide.andreoletti@supsi.ch

***Abstract*—Music Structural Analysis (MSA) algorithms analyze songs with the purpose of automatically retrieving their large-scale structure. They do so from a feature-based representation of the audio signal (e.g., MFCCs, chromagram), which is usually hand-designed for that specific application. In order to design a proper audio representation for MSA, we need to assess which musical properties are relevant for segmentation purposes (e.g., timbre, harmony); and we need to design signal processing strategies that can be used for capturing them. Deep learning techniques offer an alternative to this approach, as they are able to automatically find an abstract representation of the musical content. In this work we investigate their use in the task of Music Structural Analysis. In particular, we compare the performance of several state-of-the-art algorithms working with a collection of traditional descriptors and by descriptors that are extracted with a Deep Belief Network.**

I. Introduction

In popular western music, songs typically consist of a sequence of functional *segments*, named *sections*, that are assigned with different labels depending on their role in the song, such as *intro*, *verse*, *chorus*, *bridge* and *outro*, and that may appear multiple times in the same song. Automatic *Musical Structural Analysis* (MSA) aims at identifying such sections and grouping them together, which can be useful for numerous applications, such as the generation of audio summaries [1] or the extraction of section-level descriptors for music retrieval purposes [2, 3]. This is why the Music Information Retrieval (MIR) community has recently focused on this problem and proposed several solutions [4, 5, 6].

MSA typically comprises two main steps: first the section boundaries are identified (*boundary detection*); and then the related regions are clustered together in order to identify the structure of the song (*clustering*). In this study we focus only on boundary detection.

Section boundaries typically occur in correspondence of the substantial variation of some musical (rhythmic, harmonic, etc.) or timbral properties [7]. In view of this, is crucial to provide an effective representation of music data able to capture these properties. At this purpose, most of representations adopted in the literature are based on specifically-designed descriptors such as chromagram and MFCC [7, 5]. However, structural analysis is a complex task that may involve other perceptual elements not captured by these descriptors. The best set of features to use is still not clear [7, 8]. To address these limitations, in this study we propose an unsupervised method based on a Deep Belief Network (DBN) in order to automatically learn the set of features that are more effective to represent the music data. This approach has proved to be effective in several MIR tasks, such as bootleg detection [9] and genre recognition [10].

In order to asses the validity of our approach, we compare the learned representation of data with the most used hand-crafted features (chromagram and MFCC) by using some of the state-of-the-art MSA techniques: [4, 5, 6] that use a measure of homogeneity in order to identify homogeneous regions in the Self-Similarity Matrix (SSM); [11] that focuses on the detection of repetition of patterns by using a novelty curve; [12] that uses graph theory techniques over a recurrence matrix.

Let the reader notice that deep learning techniques have also been used in the past to design a novel approach to MSA using Convolutional Neural Networks (CNN) in [13]. However, in our study we only focus on the feature extraction stage. For this reason the comparison with [13] is out of the scope of this work.

II. Deep Learning

Deep learning techniques refer to a broad class of machine-learning approaches that rely on the hierarchical organization of the information. In this section we give a brief overview of the technique used in our work, i.e the Deep Belief Network (DBN). We refer the reader to [14] for a more comprehensive dissertation.

A. Restricted Boltzmann Machine (RBM)

An RBM is a stochastic neural network in which it is possible to identify a visible layer $\mathbf{v} \in \mathbb{R}^{M\times 1}$ (i.e., the input data) and an hidden layer $\mathbf{h} \in \mathbb{R}^{N\times 1}$. Visible and hidden layers are fully connected, whereas no connection exists among neurons of the same layer.

978-0-9928-6265-7/16/$31.00 ©2016 IEEE

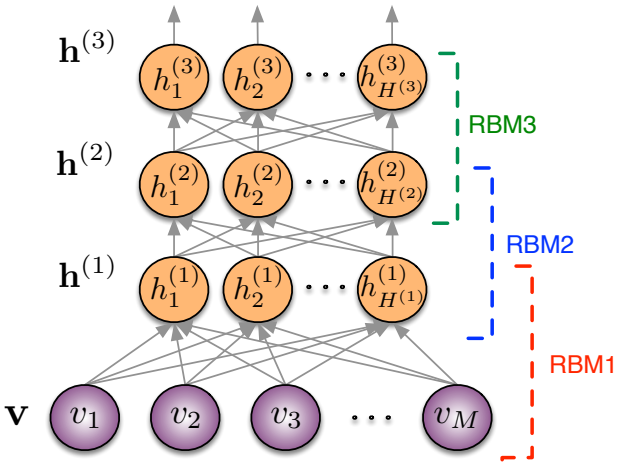


Fig. 1: Representation of a DBN, which is composed by several stacked RBM

The RBM is trained in order to model the probability distribution of the input, which is presented in the visible layers, by means of the configuration of the visible and hidden layers. To do so, an *energy* value is defined for each configuration of visible and hidden layers [14] as

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{b}^\top \mathbf{v} - \mathbf{c}^\top \mathbf{h} - \mathbf{h}^\top \mathbf{W} \mathbf{v}, \tag{1}$$

being $\mathbf{b} \in \mathbb{R}^{M \times 1}$, $\mathbf{c} \in \mathbb{R}^{N \times 1}$ and $\mathbf{W} \in \mathbb{R}^{N \times M}$ the parameters to compute, where $^\top$ indicates the transpose operator.

Low values of energy are assigned to high probable configuration and vice versa. In order to compute the marginal with respect of the visible layer, the *Free Energy* value is introduced and is defined as [14]

$$F(\mathbf{v}) = -\log \sum_{\mathbf{h} \in \mathcal{H}_\mathbf{v}} e^{-E(\mathbf{v}, \mathbf{h})}, \tag{2}$$

where $\mathcal{H}_\mathbf{v}$ indicates the set of possible configuration from the input vector $\mathbf{v}$.

Given a dataset $\mathcal{V}$ of inputs, a RBM is then trained in order to find the parameters such that:

$$\{\hat{\mathbf{W}}, \hat{\mathbf{b}}, \hat{\mathbf{c}}\} = \underset{\mathbf{W}, \mathbf{b}, \mathbf{c}}{\operatorname{argmin}} \prod_{\mathbf{v} \in \mathcal{V}} F(\mathbf{v}). \tag{3}$$

Once the RBM has been trained, the hidden layer can be used to provide a representation of new input data $\mathbf{v}$ as $\mathbf{h} = \tau\left(\hat{\mathbf{W}}\mathbf{v} + \hat{\mathbf{c}}\right)$, being τ the sigmoid function applied element-wise.

Since the hidden layer $\mathbf{h}$ is designed to extract the most salient properties of a given input, such properties can be used as a description of the input or processed to provide other more representative features.

B. Deep Belief Network

A DBN is a multi-layer architecture that is composed by the stacking of several RBMs [15], where each layer k takes as input the hidden vector $\mathbf{h}^{(k-1)}$ of the previous one (as in Figure 1) in order to produce the more abstract features $\mathbf{h}^{(k)}$. This kind of architecture allows the RBMs in the top to combine the properties learned in the lower layers in order to provide a more abstract representation of the input data. Given a new data input $\mathbf{v}$, the output of the DBN is its multi-layer representation, i.e., the collection of all the hidden layers $\mathbf{h}^{(k)}$ with $k = 1, ...K$, each one providing a different level of abstraction.

A DBN is trained by individually training the RBMs by which it is composed. This training is made to learn a representation of the input data, hence unlabeled data is sufficient in this step. The learned parameters can be *fine-tuned* in order to learn a target-oriented representation, i.e., a representation that is more useful for a given task. In this study, we investigate the use of unsupervised features only.

III. Music Structure Analysis

Our approach is composed by three main steps: *training phase*, *validation phase* and *test phase*. In the training phase we train a 3-layer DBN by using a training set $\mathcal{S}_{\text{DBN}}$. The DBN produces the set of learned features used to feed the boundaries detection algorithms. We consider a set of state-of-the-art methods (see section III-B) for the boundary detection. In the validation phase, we find the best parameters for the MSA algorithms over a validation set $\mathcal{S}_{\text{VALID}}$. The final test phase aims to retrieve the overall performance and it is performed over a test set $\mathcal{S}_{\text{TEST}}$. The three sets are completely disjoint in order to avoid over-fitting issues.

A. Feature Extraction

We train the DBN over an input represented by the normalized log-magnitude of the Fourier Transform of a frame-level representation of the audio files. More formally, we divide each song $\mathbf{x}_s$, $s \in \mathcal{S}_{\text{DBN}}$ into a sequence of F overlapping frames $\mathbf{x}_{s,f}$, $f \in \{1, ..., F\}$ of fixed duration and we compute the set of the DBN training input $\mathcal{V} = \{\mathbf{v}_{s,f}\}$ as

$$\mathbf{v}_{s,f} = \log_{10}(|\mathcal{F}(\mathbf{x}_{s,f})|^2), \tag{4}$$

where $\mathcal{F}$ represents the Fourier transform. In order to make data consistent, we normalize the input to zero mean and unit variance for each frequency bin as in [13].

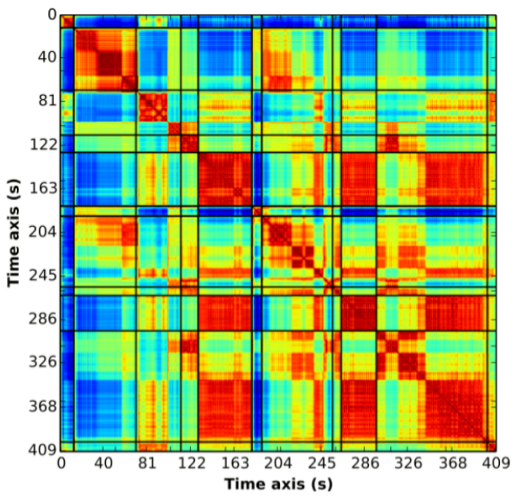
We train the DBN in a layer-wise fashion and we obtain the network parameters $\left\{\hat{\mathbf{W}}^{(\mathbf{k})}, \hat{\mathbf{b}}^{(\mathbf{k})}, \hat{\mathbf{c}}^{(\mathbf{k})}\right\}$ for each layer $k \in \{1, 2, 3\}$ as discussed in Section II-B. We use the network parameters to compute the learned features for unseen songs $\mathbf{x_s}$, $s \notin \mathcal{S}_{\text{DBN}}$ for each layer and each frame of $\mathbf{x_{s,f}}$ as:

$$\mathbf{h}_{s,f}^{(k)} = \tau(\hat{\mathbf{W}}^{(k)} \mathbf{h}_{s,f}^{(k-1)} + \hat{\mathbf{c}}^{(k)}), \tag{5}$$

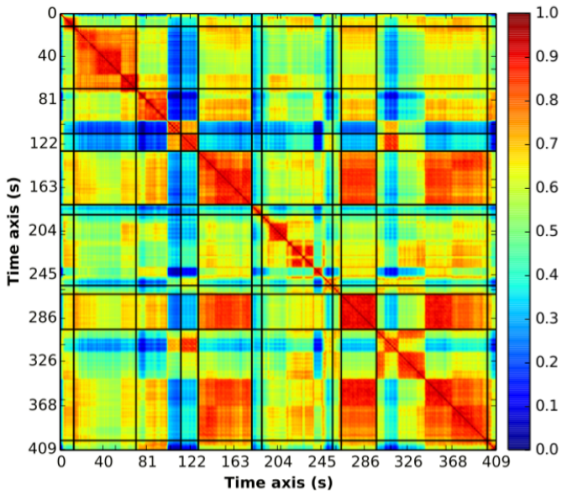
where $\mathbf{h}_{s,f}^{(0)} = \mathbf{v}_{s,f}$. We compose the final feature vector $\mathbf{h}_{s,f}^{(ALL)}$ by stacking $\mathbf{h}_{s,f}^{(1)}$, $\mathbf{h}_{s,f}^{(2)}$ and $\mathbf{h}_{s,f}^{(3)}$ together.

B. Music Segmentation

Several musicology studies in the literature highlight that in popular western music section boundaries occur typically on the beat. For this reason, in order do make the method more robust to noisy fluctuations and independent on tempo variations, it is a common practice in MSA literature [16] to



(a) The SSM computed with traditional features (MFCCs and chromagram).



(b) The SSM computed with learned features.

Fig. 2: Comparison of the SSMs for the song *Special Roll (6:59)* by Blacklite. The black lines indicate where the section boundaries occur. The figure is best seen in color.

synchronize feature extraction to the beat. In this study we extracted beat sequences by using the librosa framework [17]. Once the beat instants are obtained, we compute sequences of beat-synchronous feature vectors by averaging the feature vectors over the beat frames. The beat-synchronous feature vectors are used to compute the SSM that is the base of all MSA methods considered in this study.

In Figure 2 we show a comparison between two SSMs computed from traditional (Fig. 2a) and learned (Fig. 2b) features. The black lines indicate where the section boundaries are located. The best scenario for most of the MSA algorithms requires high and uniform self-similarity values in the blocks in the main diagonal and lower self-similarity values between contiguous blocks. We can notice that the SSM generated from the learned features is visually neater. As an example, the self-similarity values of the section around 40s presents a noisy structure with the traditional features, that may lead to over-segment that block, while with learned features is visually smoother, i.e., the values are more uniform within the block. For this reason, we expect that this might be a better representation for MSA.

The boundary detection algorithms used in this study are: i) Foote [4] correlates a Gaussian kernel along the main diagonal of the SSM to obtain a novelty curve from which peaks are extracted; ii) C-NMF [6] is based on a decomposition of the SSM by using a convex non-negative matrix factorization; iii) the algorithm proposed in [11] is a repetition-based approach that uses structure features (SF) obtained from a time-lag matrix computed from a recurrence plot of the SSM; iv) Spectral Clustering (SC) [12] associates a recurrence plot of the SSM with a graph and applies spectral clustering techniques; v) Jensen [5] retrieve boundaries as the shortest path of an adjacency matrix computed from the SSM. We refer the reader to the original papers for a more extensive explanation.

IV. Experiments and Results

A. Experimental setup

In order to keep $\mathcal{S}_{\text{DBN}}$, $\mathcal{S}_{\text{VALID}}$ and $\mathcal{S}_{\text{TEST}}$ disjoint we used three separate datasets as follows: i) $\mathcal{S}_{\text{DBN}}$, which is used to train the DBN, is composed by the 10,000 copyright-free songs from the Grand Challenge of User Experience (GCUX14) of MIREX[1]; ii) $\mathcal{S}_{\text{VALID}}$ is composed by a set of 90 annotated songs by Queen, Michael Jackson and Carole King, from the Centre for Digital Music (C4DM) at Queen Mary, University of London[2]; iii) $\mathcal{S}_{\text{TEST}}$ is composed by 140 freely-available songs of the well-known SALAMI Dataset [18]. Since in this study we propose an unsupervised approach to feature learning, it is not necessary for the $\mathcal{S}_{\text{DBN}}$ to be annotated for the specific task of boundary detection.

In order to reduce the computational time, we down-sampled songs to 11025 Hz, since we experimentally assessed that it does not effect the performance. We considered rectangular-windowed frames of 1024 samples, with 50% overlap for the feature extraction. We used the *Theano* Python library [19] to implement a three-layers DBN, with 75, 50 and 25 neurons for the first, second and third layer, respectively. We trained the DBN with a pre-training learning rate of 10^{-5} for 10 epochs (i.e., iterations [14]) for each layer. We empirically choose the above-mentioned parameters by considering previous experiments on unsupervised feature learning via DBN, such as in [9].

[1]http://www.music-ir.org/mirex/wiki/2014:GC14UX
[2]www.isophonics.net

TABLE I: Precision, Recall and F-Measure (P,F,R respectively) with 0.5 and 3 seconds of tolerance. The traditional features are composed by the stacking of MFCC and chromagram.

Algorithm	Feature	P@0.5s	R@0.5s	F@0.5s	P@3s	R@3s	F@3s
Foote	learned	0.2153	0.3415	0.2542	0.4715	0.7306	**0.5530**
Foote	traditional	0.2454	0.3867	**0.2874**	0.4055	0.6371	0.4740
SF	learned	0.1665	0.3083	0.2067	0.3021	0.6996	0.4741
SF	traditional	0.2166	0.3282	**0.2515**	0.4328	0.6485	**0.5013**
CNMF	learned	0.2392	0.2653	**0.2421**	0.4846	0.5451	**0.4937**
CNMF	traditional	0.1890	0.3029	0.2230	0.3747	0.6190	0.4470
average	learned	0.2070	0.3050	0.2343	0.4194	0.6584	**0.5070**
average	traditional	0.2170	0.3393	**0.2540**	0.4043	0.6349	0.4741

TABLE II: Precision, Recall and F-Measure (P,F,R respectively) with 0.5 and 3 seconds of tolerance for the Jensen and SC algorithm.

Algorithm	Feature	P@0.5s	R@0.5s	F@0.5s	P@3s	R@3s	F@3s
Jensen	learned	0.2203	0.3435	0.2559	0.4401	0.6810	0.5111
SC	learned	0.3282	0.2557	0.2658	0.5471	0.4440	0.4592

Most of the MSA algorithms used in this study are implemented in the Music Structure Analysis Framework (MSAF[3]), except for [5], which we implemented[4].

After a preliminary validation step, we observed that parameters used in the original papers for some of the MSA algorithms were well-suited also for the learned features. However, we experimentally tuned better parameters for the algorithm proposed by Foote [4] and for the one by Jensen [5] over the validation set $\mathcal{S}_{\text{VALID}}$. As far as [4] is concerned, we set the following parameters: $M = 48$ is the size, in beats, of the Gaussian kernel that is applied to the SSM; $m = 6$ is the size, in beats, of a median filter which is applied, along the time axis, over the feature matrix in the preprocessing; $L = 96$ is the size, in beats, of the windows over which the adaptive threshold for the novelty curve is computed. As far as [5] is concerned, we set $\alpha = 0.35$ the cost of a new segment in the adjacency matrix.

We performed the evaluation of our method over $\mathcal{S}_{\text{TEST}}$ by using the Python MIR evaluation framework[20] and we compared the results with those published in the MIREX 2014 evaluation task in Structural Segmentation[5] for the same set of songs. We considered the hit-measures Precision, Recall and F-measure with a tolerance of 0.5 and 3 seconds, which are common metrics and tolerance values in the literature and in the MIREX evaluation task. We present the results in Table I. The traditional features used in the algorithms are the chromagram and the MFCCs stacked together.

B. Comments on the results

As far as the Foote algorithm is concerned, the learned features perform slightly worse considering the F-measure with a tolerance of 0.5 seconds, while they clearly outperform the hand-crafted features for the tolerance of 3 seconds. In particular, the performance exhibits a higher recall than precision, which let us to suppose that the Foote algorithm might introduce a certain degree of over-segmentation, hence to be too sensible to variation with both kind of descriptors

As far as the C-NMF is concerned, the learned features outperform the traditional ones both in the case of 0.5s and 3s tolerance. Moreover, while with the traditional features the recall is much higher than the precision, they are rather balanced with the learned features. This implies that the learned features are more robust to slight variation of harmony or timbre, which can mislead the boundary detection.

The SF algorithm performs better with the traditional features. We suppose this is because SF is based on the analysis of the repetition of patterns, while learned features seem to perform better with homogeneity-based algorithms.

The SC algorithm and the one proposed by Jensen were not part of the MIREX2014 evaluation task, hence we could not compare their performance with the traditional features and their results are shown in the Table II. The SC algorithm obtains optimal performance with the tolerance of 0.5s, since it is able to outperform most of the algorithms, including the SF. However, it exhibits poor performance on the 3s tolerance. On the other hand, the algorithm proposed by Jensen exhibits average results with the 0.5s tolerance, while it outperforms all the other algorithms in the 3s, except for the one by Foote in the case of learned features.

As an overall consideration on the results, the traditional features exhibit higher performance when a lower tolerance is considered, while learned features stand out with the tolerance of 3s. In [13] it is shown that the same parameters setup for a CNN achieves highly different performance for the two tolerance values. The authors perform an analysis over the parameters' space for their MSA algorithm for the two tolerance levels. In this work, we aim at finding a generic feature representation for several MSA algorithms from the state of the art, hence we select a setup that achieves fairly

[3]https://github.com/urinieto/msaf

[4]https://github.com/mbuccoli/jensen-segmenter

[5]http://nema.lis.illinois.edu/nema_out/mirex2014/results/struct/sal/

high results for all the algorithms and all the tolerance levels.

We consider this is a very promising result since the MSA algorithms used here were originally designed for a traditional feature representation.

V. Conclusion and future works

We investigated the use of features learned by means of deep learning techniques for the task of Music Structural Analysis. We compared the performance of some state-of-the-art algorithms using traditional and learned features. The performance proves the validity of learned features in the MSA task when a higher tolerance is considered, even if the algorithms were originally designed for traditional features. As future works, we intend to exploit further deep architectures for the feature extraction, as well as to exploit fine-tuning techniques and to design specifically tailored MSA algorithms for learned features.

References

[1] G. Peeters, A. La Burthe, and X. Rodet, "Toward automatic music audio summary generation from signal analysis." in *Proc. of the 3rd International Society for Music Information Retrieval Conference (ISMIR)*, 2002.

[2] M. Buccoli, A. Gallo, M. Zanoni, A. Sarti, and S. Tubaro, "A dimensional contextual semantic model for music description and retrieval," in *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[3] L. Chiarandini, M. Zanoni, and A. Sarti, "A system for dynamic playlist generation driven by multimodal control signals and descriptors," in *Proc. of the 13th IEEE International Workshop on Multimedia Signal Processing (MMSP)*, 2011.

[4] J. Foote, "Automatic audio segmentation using a measure of audio novelty," in *IEEE International Conference on Multimedia and Expo (ICME)*, 2000.

[5] K. Jensen, "Multiple scale music segmentation using rhythm, timbre, and harmony," *EURASIP Journal on Applied Signal Processing*, vol. 2007, no. 1, pp. 159–159, Jan. 2007. [Online]. Available: http://dx.doi.org/10.1155/2007/73205

[6] O. Nieto and T. Jehan, "Convex non-negative matrix factorization for automatic music structure identification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2013, pp. 236–240.

[7] M. J. Bruderer, M. F. McKinney, and A. Kohlrausch, "Structural boundary perception in popular music." in *Proc. of the 7th International Society for Music Information Retrieval Conference (ISMIR)*, 2006.

[8] D. Turnbull, G. R. G. Lanckriet, E. Pampalk, and M. Goto, "A supervised approach for detecting boundaries in music using difference features and boosting." in *Proc. of the 8th International Society for Music Information Retrieval Conference (ISMIR)*, 2007.

[9] M. Buccoli, P. Bestagini, M. Zanoni, A. Sarti, and S. Tubaro, "Unsupervised feature learning for bootleg detection using deep learning architectures," in *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2014.

[10] P. Hamel and D. Eck, "Learning features from music audio with deep belief networks," in *Proc. of the 11th International Society for Music Information Retrieval (ISMIR)*, 2010.

[11] J. Serra, M. Muller, P. Grosche, and J. Arcos, "Unsupervised music structure annotation by time series structure features and segment similarity," *IEEE Transactions on Multimedia*, vol. 16, no. 5, pp. 1229–1240, August 2014.

[12] B. McFee and D. P. W. Ellis, "Analyzing song structure with spectral clustering," in *Proc of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, 2014, pp. 405–410.

[13] K. Ullrich, J. Schlüter, and T. Grill, "Boundary detection in music structure analysis using convolutional neural networks," in *Proc. of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, 2014.

[14] Y. Bengio, "Learning Deep Architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009. [Online]. Available: http://www.nowpublishers.com/product.aspx?product=MAL\&doi=2200000006

[15] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Journal Neural Computation (JNC)*, vol. 18, no. 7, pp. 1527–1554, July 2006.

[16] O. Nieto and J. Bello, "Music segment similarity using 2d-fourier magnitude coefficients," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014.

[17] B. McFee, M. McVicar, C. Raffel, D. Liang, O. Nieto, J. Moore, D. Ellis, D. Repetto, P. Viktorin, J. F. Santos, and A. Holovaty, "librosa: v0.4.0," 2015. [Online]. Available: http://dx.doi.org/10.5281/zenodo.18369

[18] J. B. L. Smith, J. A. Burgoyne, I. Fujinaga, D. D. Roure, and J. S. Downie, "Design and creation of a large-scale database of structural annotations," in *Proc. of the 12th International Society for Music Information Retrieval Conference (ISMIR)*, 2011.

[19] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a CPU and GPU math expression compiler," in *Proc. of the Python for Scientific Computing Conference (SciPy)*, 2010.

[20] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, and D. P. W. Ellis, "mir eval: A transparent implementation of common mir metrics," in *Proc. of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, 2014.