Deep Belief Networks for Automatic Music Genre Classification

Xiaohong Yang, Qingcai Chen, Shusen Zhou, Xiaolong Wang

Department of Computer Science and Technology
Key Laboratory of Network Oriented Intelligent Computation
Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China
{yxh2008, qingcai.chen}@gmail.com, zhoushusen@hitsz.edu.cn, wangxl@insun.hit.edu.cn

10.21437/Interspeech.2011-633

Abstract

This paper proposes an approach to automatic music genre classification using deep belief networks. Based on the restricted Boltzmann machines, the deep belief networks is constructed and takes the acoustic features extracted through content-based analysis of music signals as input. The model parameters are initially determined after the deep belief network is trained by greedy layer-wise learning algorithm with feature vectors that are comprised of short-term and long-term features. Then the parameters are fine-tuned to local optimum according to back propagation algorithm. Experiments on GTZAN dataset show that the performance of music genre classification using deep belief networks is superior to those of widely used classification methods such as support vector machine, K-nearest neighbor, linear discriminant analysis and neural network.

Index Terms: deep belief networks, music genre classification, timbral texture feature, modulation spectral analysis

1. Introduction

Nowadays a large number of digital music can be accessed and downloaded from the Internet due to the rapid progress of communication networks and storage technologies. However, how to effectively manage the large scale digital music database that contains millions of music tracks is a difficult but important work. At present, most music search engines organize their music collections according to the categories such as artist, album, genre, mood, instrument, theme and rhythm etc. This work focuses on the automatic music genre classification problem which labels the genres of music tracks. Music genre classification is considered to be a great practical component for music retrieval and recommendation systems. Therefore various classification techniques have been used for automatic music genre classification [1]-[11].

Automatic music genre classification mainly consists of two parts: feature extraction and music classification with classifiers. At first various discriminative features are extracted through content-based analysis of music signal, and can be categorized into short-term and long-term features. Short-term features primarily represent the characteristics of music spectrum, and are computed in every short time window or frame which is denoted as "analysis window". The generally used short-term features include spectral centroid, spectral flux, spectral rolloff, zero crossings, Mel-frequency cepstral coefficients (MFCC) [1][2][5][7] and octave-based spectral contrast (OSC) [7][8]. The long-term features, that usually describe the variation of spectral shape and calculated in long time window namely "texture window", include low energy [1][2], Daubechies wavelet coefficients histograms (DWCHs) [2][3], octave-based modulation spectral contrast (OMSC) [7][8], modulation spectral crest measure (MSCM), modulation spectral flatness measure (MSFM) [4][8], and modulation spectral analysis of spectral and cepstral features [7].

After feature extraction, many classifiers or classification methods can be employed to recognize the genres of given music tracks. So far, the widely used classifiers or classification methods include Gaussian mixture models (GMM) [1], K-nearest neighbor (KNN) [1], linear discriminant analysis (LDA) [2][7], support vector machine (SVM) [2], regularized least-square [5], sparse representation-based classification (SRC) [6], compressive sampling [8], non-negative matrix factorization (NMF), non-negative tensor factorization (NTF) [10][11] and non-negative multilinear principal component analysis (NMPCA) [9]. Although most classifiers or methods based on combination of different features may obtain good results, they are not clear on how to learn and optimize the weights of different types of features.

In this work we propose an approach to the genre classification of music tracks using deep belief networks (DBN). DBN is composed of directed belief nets with many layers of hidden variables. Since the fast greedy layer-wise unsupervised learning algorithm is introduced by Hinton et al. [12] and generalized by Bengio et al. [13], deep belief networks has obtained much attention and applied to multiple domains such as document retrieval, audio classification [14][15] and image classification [16]. Firstly the deep belief network is constructed based on Restricted Boltzmann Machines (RBM). The model parameters are initially determined after the deep belief network is trained by greedy layer-wise learning algorithm with feature vectors that are comprised of short-term and long-term features. Then the parameters are fine-tuned to local optimum according to back propagation algorithm. Experimental results show that the performance of music genre classification based on DBN is superior to those of widely used classification methods such as SVM, KNN, LDA and neural network (NN).

The rest of this paper is organized as follows. The short-term and long-term features are described in section 2. Section 3 introduces the deep belief networks in detail. The experimental results are reported and analyzed in section 4. Finally our conclusions are summarized in section 5.

2. Feature extraction

Feature extraction is to draw the discriminative characteristics that can primarily represent the music tracks. In this study, the used acoustic features consist of timbral texture features, octave-based spectral contrast (OSC), modulation spectral flatness measure (MSFM), modulation spectral crest measure (MSCM), modulation spectral analysis of cepstral feature MFCC and spectral feature OSC.

2.1. Timbral texture features (TTF)

Timbral texture features that can differentiate the audios with same or similar rhythm include MFCC, spectral centroid, spectral flux, spectral rolloff, zero crossings and low energy. The first five are short-term features, and only low energy is long-term feature. MFCC represent the spectrum of music signal in a compact form. Spectral centroid describes the gravity of magnitude spectrum. Spectral flux is the squared difference between normalized magnitudes of successive spectrum. Spectral rolloff indicates the frequency blow which 85% of the magnitude distribution is concentrated. Zero crossings is the number of zero crossings of the signal in time domain. Low energy is the percentage of frames whose energies are less than the average energy over the whole signal. Detailed introduction of timbral texture features can be found in [1] and [2].

To extract the timbral texture features, the music signals are segmented into short time frames, namely "analysis window" during which the signals are assumed to be stationary. The signals of each frame are multiplied by Hamming window function, then short time Fourier transformation (STFT) is conducted to calculate corresponding feature value of each timbral texture feature. Especially we use Dan Ellis' implementation[1] to calculate the 13 MFCC coefficients. In this paper, the frame size is 23ms (1024 samples at 44.1kHz sampling rate) with 50% overlap between every two successive frames.

2.2. Octave-based spectral contrast (OSC)

OSC represents the spectral characteristics of music signal and considers the spectral peak, valley and contrast in each subband. After the spectrum of each frame is divided into octave-based subbands, the spectral peaks and valleys are calculated by the logarithm of the average values across the small neighborhood around maximum and minimum values of the magnitude spectrum respectively [8]. The difference between spectral peak and spectral valley is denoted by spectral contrast. The frequency ranges of all octave-based subbands for 44.1kHz sampling rate are 0-100Hz, 100Hz-200Hz, 200Hz-400Hz, 400Hz-800Hz, 800Hz-1600Hz, 1600Hz-3200Hz, 3200Hz-8000Hz, 8000Hz-22050Hz. The spectral valley and spectral contrast in all octave-based subbands are used as the features. The detailed implementation for calculating OSC can refer to [7].

2.3. Modulation spectral analysis of spectral features

Besides above short-term frame-based features, we also consider the long-term features, namely modulation spectral analysis of spectral flatness measure (SFM), spectral crest measure (SCM), MFCC and OSC to capture the time-varying behavior of music signals. These features are abbreviated to MSFM, MSCM, MMFCC and MOSC respectively, and extracted from "texture window" whose length is 512 frames (about 6s) with 50% overlap between two consecutive texture windows.

MSFM and MSCM are computed by applying discrete Fourier transformation (DFT) on the power spectrum of music signals in texture window based on the octave-based subbands. MSFM is defined as the ratio of the geometric mean to the arithmetic mean of the modulation magnitude spectrum, and MSCM is defined as the ratio of the maximum value to the arithmetic mean of the modulation magnitude spectrum.

In order to calculate MMFCC and MOSC, fast Fourier transformation (FFT) is applied independently on each feature value along the frame direction within each texture window to obtain the modulation spectrum. The averaged modulation spectrum of each feature value of all texture windows is decomposed into $J(J = 8)$ logarithmically spaced modulation subbands [7]. Then modulation spectral peak (MSP), modulation spectral valley (MSV) and modulation spectral contrast (MSC) within each modulation subband can be obtained, where MSC is defined as the difference between MSP and MSV. Then the mean and standard deviation along each row and each column of the MSV and MSC matrices are used as the feature values. The detailed methods to extract MSFM, MSCM and MMFCC, MOSC features are introduced in [4] and [7] respectively.

[1] http://labrosa.ee.columbia.edu/matlab/rastamat

2.4. Feature vector and normalization

To represent the whole music track, the mean and standard deviation of spectral centroid (1), spectral flux (1), spectral rolloff (1), zero crossings (1), MFCC (13) and OSC (16) of all frames in a music track are used as features. The number in parentheses denotes the dimension of corresponding feature extracted from each frame. In addition, the low energy has one value over the whole music track. The long-term features MSFM and MSCM are computed based on the eight octave-based subbands mentioned before, thus their dimensions are both 8. The dimensions of MMFCC and MOSC depend on the dimensions (D) of MFCC and OSC, and the number of logarithmically spaced modulation subbands J. According to approach described in [7], MMFCC and MOSC have $(4 \times D + 4 \times J)$, namely 84 and 96 feature values respectively. Therefore a music track can be represented by a 263-dimensional feature vector.

All feature vectors of all music tracks are normalized along the dimensional direction, in other words, normalization is independently implemented to each feature value of all feature vectors as the following formula:

$$f_{norm}(i) = \frac{f(i) - f_{min}(i)}{f_{max}(i) - f_{min}(i)} \quad (1)$$

where $f_{norm}(i)$ denotes the normalized i-th feature value, $f_{max}(i)$ and $f_{min}(i)$ are the maximum and minimum values of the i-th feature value of all feature vectors respectively [7].

3. Deep belief networks

Deep belief networks is a multi-layer architecture with one visible layer and several hidden layers. The number of stochastic units in visible layer is equal to the dimension of input feature vector, namely one unit corresponds to one feature value. And the output layer provides one unit for each category that music tracks will be classified. Whereas the number of units included in each hidden layer are predefined empirically. The units between two adjacent layers are fully connected, and there are no inter-layer connections. The model parameters of the DBN are learned and optimized via greedy layer-wise unsupervised learning and fine-tuning processing.

3.1. Layer-wise unsupervised learning

Each pair of two adjacent layers of DBN can be regarded as a Restricted Boltzmann Machine (RBM) [12]. RBM is a two-layer network in which the stochastic binary visible vector $\mathbf{v}$ is connected to stochastic binary hidden vector $\mathbf{h}$ with symmetrically weighted connections. The energy of the state $\{\mathbf{v}, \mathbf{h}\}$ is:

$$E(\mathbf{v}, \mathbf{h}) = -\sum_{i=1}^{N_v} b_i v_i - \sum_{j=1}^{N_h} c_j h_j - \sum_{i=1}^{N_v}\sum_{j=1}^{N_h} w_{ij} v_i h_j \quad (2)$$

where v_i and h_j are binary states, $(\mathbf{w}, \mathbf{b}, \mathbf{c})$ are the model parameters: w_{ij} is the symmetric interaction term between visible unit i and hidden unit j; b_i and c_j are bias terms. Then the probability that DBN assigns to the visible vector $\mathbf{v}$ can be obtained through energy function:

$$p(\mathbf{v}) = \frac{\sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}}{\sum_{\mathbf{u}} \sum_{\mathbf{g}} e^{-E(\mathbf{u},\mathbf{g})}} \tag{3}$$

The conditional distributions over hidden vector $\mathbf{h}$ and visible vector $\mathbf{v}$ are as follows:

$$p(\mathbf{h}|\mathbf{v}) = \prod_j p(h_j|\mathbf{v}), \quad p(\mathbf{v}|\mathbf{h}) = \prod_i p(v_i|\mathbf{h}) \tag{4}$$

Given a visible vector $\mathbf{v}$, the hidden vector $\mathbf{h}$ obeys subsequent conditional distribution:

$$p(h_j = 1|\mathbf{v}) = \sigma(c_j + \sum_i w_{ij} v_i) \tag{5}$$

where $\sigma = 1/(1 + e^{-x})$ is the logistic function. Then new visible vector can be generated using above hidden vector:

$$p(v_i = 1|\mathbf{h}) = \sigma(b_i + \sum_j w_{ij} h_j) \tag{6}$$

The derivative of the log-likelihood with respect to the model parameter $\mathbf{w}$ can be got using contrastive divergence approximate method:

$$\frac{\partial \log p(\mathbf{v})}{\partial w_{ij}} = \langle v_i h_j \rangle_{P_0} - \langle v_i h_j \rangle_{P_T} \tag{7}$$

where $\langle v_i h_j \rangle_{P_0}$ represents an expectation with respect to the data distribution and $\langle v_i h_j \rangle_{P_T}$ denotes a distribution of samples from running the Gibbs sampler that is initialized at the data, for T full steps [16]. Then the model parameters $(\mathbf{w}, \mathbf{b}, \mathbf{c})$ are updated according to the following equations:

$$w_{ij} = \eta w_{ij} + \varepsilon(\langle v_i h_j \rangle_{P_0} - \langle v_i h_j \rangle_{P_T}) \tag{8}$$

$$b_i = b_i + \varepsilon(\langle v_i \rangle_{P_0} - \langle v_i \rangle_{P_T}) \tag{9}$$

$$c_j = c_j + \varepsilon(\langle h_j \rangle_{P_0} - \langle h_j \rangle_{P_T}) \tag{10}$$

where ε is the learning rate, η is the momentum.

To summarize, DBN can be constructed by training one RBM at a time with the feature vectors introduced in section 2, using greedy layer-wise unsupervised learning method in a bottom-up manner. The model parameters of the DBN will be learned and optimized after DBN is trained layer by layer for certain times repeatedly. And the trained DBN will capture the deep correlations between the visible vectors, in other words input feature vectors.

3.2. Fine-tuning the parameters

After greedy layer-wise unsupervised learning, the model parameters of DBN is initially determined and represented by $(\mathbf{w}, \mathbf{b}, \mathbf{c})$. For different applications, there are different optimizing objectives about the parameters. One is to maximize the probability to generate the input feature vectors, the other is to minimize the classification error to improve the classification performance. In this work, the deep belief network is employed as a classifier. Therefore in order to improve the discriminative ability of the classifier, the model parameters can be refined by minimize the classification error via the labeled feature vectors. The procedures to refine the parameters are as follows: firstly we calculate the gradient and use the model parameters $(\mathbf{w}, \mathbf{b}, \mathbf{c})$ to initialize the back propagation algorithm through the whole DBN until predefined requirements are satisfied. Then the back propagation is further employed to fine-tune the model parameters to minimize the classification error.

4. Experiments

4.1. Experimental dataset

The experiments are conducted on the GTZAN corpus which is a widely used dataset for automatic music genre classification and collected by G. Tzanetakis [1]. This dataset consists of ten genre classes: Blues, Classical, Country, Disco, HipHop, Jazz, Metal, Pop, Reggae and Rock. Each genre class contains 100 recordings of music segments of 30 second duration. All the recordings are stored as 22050Hz, 16 bits, monophonic audio files with au format. Moreover, the audio signals are normalized by dividing themselves by their maximum values before feature extraction. The classification performance on the GTZAN dataset is evaluated based on the ten-fold cross validation.

4.2. Experimental results

At present, the number of hidden layers and the number of hidden units in each hidden layer are preset empirically when constructing a deep belief network. We firstly conduct a experiment to study the influence of different network structures. The DBN is initially trained by greedy layer-wise learning with 50 epochs, and the learning rate ε is defined as 0.1. The initial momentum η is set to 0.5 and modified to 0.9 after 5 epochs. When fine-tuning the model parameters via back propagation, the conjugate gradient method with line search is implemented within 50 epochs.

Table 1: *Classification accuracy using DBN with different structures.*

DBN structure	Acc.	DBN structure	Acc.
263-50-10	71.5%	263-50-20-10	63.2%
263-100-10	73.3%	263-50-50-10	66.1%
263-200-10	72.7%	263-50-100-10	66.9%
263-100-50-10	66.6%	263-200-100-10	69.3%
263-100-100-10	68.4%	263-200-200-10	67.7%
263-100-200-10	68.1%	263-200-400-10	63.2%

Despite the number of hidden layers, the input layer has 263 units which is equal to the dimensions of feature vectors, and the number of units in output layer is preset to 10 that is the number of distinct music genres to be classified. The classification results using DBNs with three or four layers are showed in Table 1 from which we can find that the DBNs with three layers outperform those with four layers, and the DBN with 263-100-10 structure gets the best classification accuracy 73.3%. The reason is that it doesn't need too much layers and units to learn and optimize the model parameters of DBN for small scale GTZAN dataset, just 1000 music tracks.

The second experiment is to investigate the influence of different epochs when initial training and fine-tuning the DBN. The experimental results in Table 2 show that the classification accuracies of both DBNs (263-100-10 and 263-100-100-10) are improved with the increase of epochs. When the epochs at two stages are both preset to 400, the accuracies of DBNs (263-

Table 2: *Classification accuracy using DBN with different structures and epochs.*

DBN structure	Epochs at two stages		Acc.
	Initial training	Fine-tuning	
263-100-10	50	50	73.3%
	100	100	75.0%
	200	200	78.3%
	300	300	79.5%
	400	400	80.1%
263-100-100-10	50	50	68.4%
	100	100	74.9%
	200	200	77.5%
	300	300	78.7%
	400	400	79.6%

100-10 and 263-100-100-10) can be up to 80.1% and 79.6% respectively. However, the increment of computation cost is much larger when the epochs reach 300 and 400.

Table 3: *Classification accuracy using different classifiers.*

Features	DBN_1	DBN_2	SVM	KNN	LDA
TTF+OSC	72.2%	69.5%	72.9%	63.3%	67.8%
MSA	75.1%	74.9%	73.2%	63.1%	70.0%
ALL	79.5%	78.7%	75.9%	65.4%	74.6%

We compare the classification performances of DBNs using different features with three widely used classifiers: SVM, KNN and LDA. The structures of DBN_1 and DBN_2 are 263-100-10 and 263-100-100-10. The epochs at both stages are predefined to 300 for balancing the accuracy and computation cost. Other setups are the same to the first experiment. SVM with linear kernel ($C = 3$) based on the LIBSVM[2] that is a library for support vector classification, KNN ($K = 10$) and LDA are implemented by using the open-source MATLAB package MATLABArsenal[3]. The performances using classifiers SVM, KNN and LDA are also assessed based on ten-fold cross validation.

The results in the 3rd row of Table 3 indicate that both DBNs with all features (TTF, OSC and MSA) outperform the other three classifiers. In addition, we use part of the features to compare the performances again. Although the accuracies of both DBNs with TTF plus OSC features showed in the 1st row of Table 3 are little smaller than that of SVM, other results still show that DBN is superior to SVM, KNN and LDA. Meanwhile, the results indicate that modulation spectral analysis of spectral and cepstral features have more discriminative ability to music genre classification than TTF plus OSC features.

Furthermore, DBN can be regarded as a neural network. Therefore we compare it with the back propagation neural network which includes one hidden layer with 100 units. When epochs at the two training stages are set to 400, the classification accuracy of DBN_1 is 80.1% which is much better than 72% of the back propagation neural network.

5. Conclusions

This paper proposes an approach to automatic music genre classification based on deep belief networks. The DBN is trained by greedy layer-wise unsupervised learning algorithm with short-term and long-term features, and fine-tuned via back propagation algorithm. Experimental results show that the performance of DBN outperforms four widely used classifiers SVM, KNN, LDA and NN. Moreover, the following two aspects are worth to study for future work: 1) feature selection, namely fuse other representative features to represent the music tracks comprehensively; 2) DBN adjustment or optimization. Since DBN is primarily applied with binary features at present, the DBN should be adjusted or optimized to fit the non-binary features (like music features) better so that the performance of DBN for music genre classification can be improved further .

[2]LIBSVM: http://www.csie.ntu.edu.tw/∼cjlin/libsvm

[3]http://finalfantasyxi.inf.cs.cmu.edu/tmp/MATLABArsenal.zip

6. Acknowledgements

This investigation is supported in part by National Natural Science Foundation of China (No. 60703015 and No. 60973076).

7. References

[1] Tzanetakis G. and Cook P., "Musical genre classification of audio signals", IEEE Trans. Speech and Audio Processing, 10(5):293–302, 2002.

[2] Li, T., Ogihara, M. and Li, Q., "A comparative study on content-based music genre classification", in Proc. of SIGIR, 282-289, 2003.

[3] Li, T. and Ogihara, M., "Toward intelligent music information retrieval", IEEE Trans. Multimedia, 8(3):564-574, 2006.

[4] Jang, D., Jin, M.H. and Yoo, C.D., "Music genre classification using novel features and a weighted voting method", in Proc. of ICME, 2008.

[5] Song, Y.Q. and Zhang C.S., "Content-based information fusion for semi-supervised music genre classification", IEEE Trans. Multimedia, 10(1):145-152, 2008.

[6] Panagakis, Y. and Kotropoulos, C., "Music genre classification via sparse representations of auditory temporal modulations", in Proc. of 17th European Signal Processing Conference, 2009.

[7] Lee, C.H., Shih, J.L., Yu, K.M. and Lin, H.S., "Automatic music genre classification based on modulation spectral analysis of spectral and cepstral features", IEEE Trans. Multimedia, 11(4):670-682, 2009.

[8] Chang, K.K., Jang, J-S.R. and Iliopoulos, C.S., "Music genre classification via compressive sampling", in Proc. of ISMIR, 387-392, 2010.

[9] Panagakis, Y., Kotropoulos, C. and Arce, G.R., "Non-Negative multilinear principal component analysis of auditory temporal modulations for music genre classification", IEEE Trans. Audio, Speech and Language Processing, 18(3):576-588, 2010.

[10] Panagakis, Y. and Kotropoulos, C., "Music genre classification via topology preserving non-negative tensor factorization and sparse representations", in Proc. of ICASSP, 249-252, 2010.

[11] Benetos, E. and Kotropoulos, C., "Non-negative tensor factorization applied to music genre classification", IEEE Trans. Audio, Speech and Language Processing, 18(8):1955-1967, 2010.

[12] Hinton, G.E., Osindero, S. and Teh, Y.W., "A fast learing algorithm for deep belief nets", Neural Computation, 18(7):1527-1554, 2006.

[13] Bengio, Y., Lamblin, P., Popovici, D. and Larochelle, H., "Greedy layer-wise training of deep networks. in Proc. of NIPS, 2006.

[14] Ballan, L., Bazzica, A., Bertini, M., Bimbo, A.D. and Serra, G., "Deep networks for audio event classification in scoccer videos", in Proc. of ICME, 2009.

[15] Lee, H., Largman, Y., Pham, P. and Ng, A.Y., "Unsupervised feature learning for audio classification using convolutional deep belief networks", in Proc. of NIPS, 1-9, 2009.

[16] Liu, Y., Zhou, S.S and Chen, Q.C., "Discriminative deep belief networks for visual data classification", Pattern Recognition, 2010.