

Deep feature learning for cover song identification

Jiunn-Tsair Fang¹ · Chi-Ting Day² · Pao-Chi Chang²

Received: 2 October 2015 / Revised: 27 October 2016 / Accepted: 31 October 2016 /
Published online: 13 November 2016
© Springer Science+Business Media New York 2016

Abstract The identification of a cover song, which is an alternative version of a previously recorded song, for music retrieval has received increasing attention. Methods for identifying a cover song typically involve comparing the similarity of chroma features between a query song and another song in the data set. However, considerable time is required for pairwise comparisons. In this study, chroma features were patched to preserve the melody. An intermediate representation was trained to reduce the dimension of each patch of chroma features. The training was performed using an autoencoder, commonly used in deep learning for dimensionality reduction. Experimental results showed that the proposed method achieved better accuracy for identification and spent less time for similarity matching in both covers80 dataset and Million Song Dataset as compared with traditional approaches.

Keywords Cover song · Deep learning · Music retrieval · Sparse autoencoder

1 Introduction

Because of the rapid development of Internet and multimedia compression techniques, it is highly convenient to share video or songs through networks in daily life. Large music collections require effective methods for management and retrieval. Therefore, efficiently retrieving music from a vast database was the focus of the present study. In particular, cover song identification has become an active study area in the music information retrieval community.

A cover song is any new version, performance, rendition, or recording of a previously recorded track [38]. The cover version may differ from the original song in timbre, tempo, structure, key, or arrangement [30]. Accordingly, content-based music retrieval has been proposed for extracting features from the music content to effectively identify a cover song.

✉ Pao-Chi Chang
pcchang@ce.ncu.edu.tw

¹ Department of Electronic Engineering, Ming Chuan University, No.5, Deming Rd, Taoyuan 33348, Taiwan

² Department of Communication Engineering, National Central University, No.300, Jhongda Rd, Taoyuan 32001, Taiwan

Finding a musical part to fit another one is challenging for a computer. Computers require rules or procedures to distinguish music characteristics for similarity measurement [29]. In particular, a cover version often adopts various interpretations of the original song. The matching scheme should be robust to changes in tempo, key, timbre, and musical instruments [11]. Tonality and harmony are generally considered major features to represent music. The pitch-class profile is thus widely used for cover song identification [12]. The pitch-class profile, also called chroma, records 12 semitones (C, C#, D, D#, E, F, F#, G, G#, A, A#, and B), or describes a 12-bin octave-independent histogram of energy intensity [31].

Ellis and Poliner won the MIREX cover-song-identification task in 2006 [17]. They first tracked the beat to overcome variability in tempo, and then collected 12-dimensional chroma features to overcome the variations in timbre. Lee used the hidden Markov model to transform chroma features into chord sequences, and applied dynamic time warping (DTW) to determine the minimum alignment cost [11, 12, 16, 17, 31]. Sailer used a pitch line for aligning melody, and measured the melodic similarity [25]. Riley et al. used a vector quantization method to quantize the chroma features for similarity calculation [24].

In 2007, Ellis et al. improved their system and collected 80 pairs of songs and covers to form a data set, called covers80 [10]. Then, Serra and Gomez used 36-dimensional chroma features and a dynamic programming algorithm to measure enhanced chroma similarity [28]. In 2009, Ravuri et al. used support vector machines and multilayer perceptions to perform general classification [22]. Recently, Tralie and Bendich used model shape in musical audio for version identification [34]. A new method called two dimensional Fourier transform magnitude (2D-FTM) has been used for classification. 2D-FTM maintains the key transposition and phase shift invariance [15, 18].

Feature selection in information retrieval is often determined according to experience and knowledge. Manual feature selection is often time-consuming, and the selected feature is typically subjective [37]. To reduce the subjectivity and increase the efficiency, machine learning has been used. Machine learning includes a class of algorithms that learn by using examples to build a model, from inputs, which makes data-driven predictions or decisions.

Deep learning refers to a class of machine learning techniques with many layers of information processing for feature learning. In 2006, Hinton et al. proposed deep belief networks (DBN) to extend artificial neural networks into deeper layers [14]. DBNs are composed of stacked restricted Boltzmann machines (RBM) [13]. RBMs can be expressed using bipartite graphs with visible and hidden variables, forming two layers of vertices, and no connection between units of the same layer. With this restriction, each RBM demonstrates a conditional distribution property over the hidden unit factorization, given the visible variables [33].

Autoencoder (AE) is a deep architecture for unsupervised learning. The special character of AE is that its learning output target is just the same as the input values, and the constraint is that the number of hidden variables is controlled to be less than that of the input [1]. Some variants of AEs have been proposed such as denoising AE [35] and contractive AE [23], in particular, the sparse AE (SAE). SAE has the advantage of over complete representation, where the hidden layer can be larger than the input [20].

In this study, chroma features was patched to preserve the melody. Since considerable time was required for pairwise comparisons, SAE was applied to train chroma features to be intermediate representation. Each patch could thus be reduced in dimension.

The remainder of this paper is organized as follows. Section II introduces deep learning. The proposed method for the patch of chroma features transformation by using deep learning methods is described in Section III. Experimental results are described in Section V. Finally, Section VI provides a conclusion.

2 Introduction to deep learning

The introduction to deep learning is mainly focused on AE. To understand the principle of AE, the principle of an RBM is first introduced, and then stacked RBMs are explained.

2.1 Deep belief networks

To explain an RBM model, this problem can be treated as to find a function for the relation between input data and output representation. Instead of doing this directly, a hidden layer is applied. The question is how to find these latent variables in the hidden layer to represent the output.

An RBM is as a particular energy-based model, and its joint distribution of visible variables, x , and latent variables, h , are expressed as follows [2, 3]:

$$p(\mathbf{x}, \mathbf{h}) = \frac{1}{z_\theta} \exp(-E_\theta(\mathbf{x}, \mathbf{h})) \quad (1)$$

where E_θ is the energy function, and θ is the model parameter. Visible variables represent the data, and latent variables mediate dependencies between the visible variables through their mutual interaction. These two variables are binary. The normalizer $z_\theta = \sum_{(\mathbf{x}, \mathbf{h})} \exp(-E_\theta(\mathbf{x}, \mathbf{h}))$,

called the partition function, was added to obtain 1 as the sum of all of the possibilities of the joint distribution. In addition, an RBM requires no interaction between hidden variables, and no interaction between the visible variables. The energy function can thus be represented by:

$$E_\theta(\mathbf{x}, \mathbf{h}) = -\mathbf{x}^T \mathbf{W} \mathbf{h} - \mathbf{b}^T \mathbf{x} - \mathbf{d}^T \mathbf{h} \quad (2)$$

where $\theta = \{\mathbf{W}, \mathbf{b}, \mathbf{d}\}$. The model parameter w_{ij} represents the interaction between the visible variable x_i and hidden variable h_j , and b_i and d_j are the weight for the bias terms of the visible and hidden layers, respectively.

Because there is no connection between units in the hidden layers, if sample values are given, each latent variable becomes an independent random variable. In other words, the conditional probability of $p(h_i|x)$ is calculated independently. Similarly, each conditional probability $p(x_j|h)$ is calculated independently.

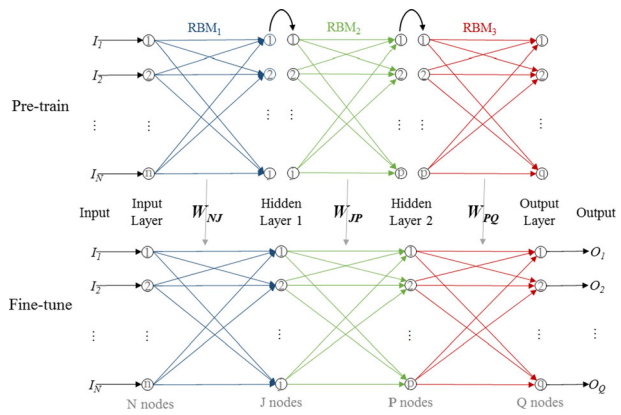
Figure 1 plots an example to show the structure of a three-layer DBN, including the pre-training and fine-tuning. From the RBM₁, in the left-top of Fig. 1, each sample of training data connecting with weight, W_{NJ} , to the Hidden Layer 1. With the backpropagation procedure, the weight, W_{NJ} , is modified to reduce the probability of classification error. This procedure is iteratively processed until the error converges. Then, weight W_{NJ} and output signals can be obtained.

Output data from the RBM₁ becomes the input data to the RBM₂. Similar procedures are applied to learn the weight, W_{JP} for the Hidden Layer 2 in RBM₂, and weight W_{PQ} for Hidden Layer 3 in RBM₃, respectively. Finally, the backpropagation method is applied to fine-tune weight in each hidden layer. After fine-tuning, weight in each layer can be built up.

2.2 Autoencoder

The main character of an AE is that the output representation in the AE is the same as the input signals. The hidden layers can thus be treated to compress the input data, and decoded for the output to be the same as input data. The encoder hidden layer variable can be expressed as follows:

Fig. 1 An example to show the DBN structure



$$h_i = \text{sigmoid} \left(\sum_j w_{i,j}x_j + d_i \right) \tag{3}$$

The reconstructed output variable can be expressed as follows:

$$\begin{aligned} x'_{i,j} &= \text{sigmoid} \left(\sum_j w'_{i,j}h_i + b_i \right) \\ &= \text{sigmoid} \left(\sum_j w'_{i,j} \left(\text{sigmoid} \left(\sum_j w_{i,j}x_j + d_i \right) + b_i \right) \right) \end{aligned} \tag{4}$$

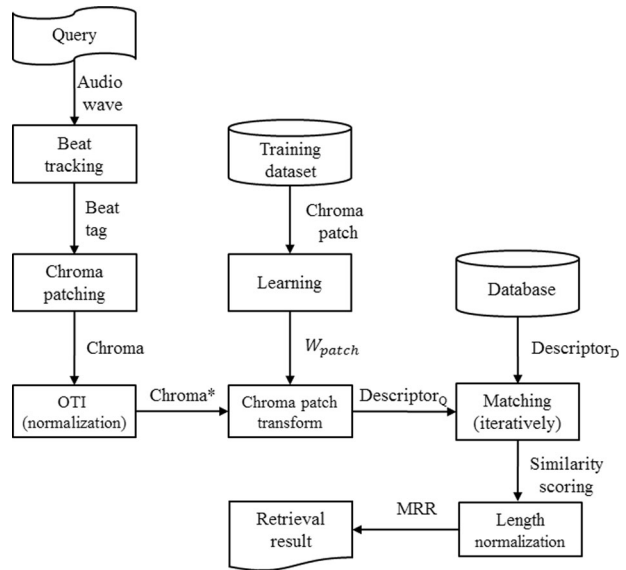
where w and w' can be the same. Latent variables, w and w' can be set up by minimizing the error function, which is the square value of the difference between x and x' . The dimensions of hidden layers are smaller than those of input data. In other words, smaller dimensions of variables are used to represent input data. Equations (3) and (4) have the similar forms as RBMs. Thus, encoder and decoder of AE can be stacked into a deep architecture.

Sparsity can be applied in AE. Sparsity in the representation can be achieved by penalizing the hidden unit biases to make these additive offset parameters more negative, or by directly penalizing the output of the hidden unit activations to make them closer to 0, far from their saturating value [2]. SAE has the advantage of selecting the least possible basis from the large basis pool to recover the given signal under a small reconstruction error constraint [2, 3, 20, 21].

3 Proposed method for cover song identificaion

The proposed method is mainly based on two concepts. First, chroma features are patched. The segment of chroma features can preserve the melody to improve the similarity matching [6]. However, the matching time is increased. Second, SAE is applied to train chroma features for intermediate representation. The dimension of chroma features can be reduced, and therefore the matching time for similarity comparison can be saved. The flow chart of the proposed method is plotted in Fig. 2. The detailed procedures of Fig. 2 are described in the following subsections.

Fig. 2 Chroma features extraction



3.1 Chroma features patch

The proposed method can be separated into three parts. That is, chroma feature patch, deep chroma features learning, and matching and scoring. Figure 3 describes the process of chroma feature extraction. The first step is beat tracking. The dynamic programming method was used to track beats [7, 17]. The second step is to transform audio signals into frequency domains. The discrete Fourier transform (DFT) was applied to transform audio signals into the frequency domain. Signals in the frequency domain are passed through log-scaled filters to separate 12 semitones. The third step is to cluster the frequencies into 12 semitones. Finally, frames of chroma features are patched.

Figure 4 plots an example of chroma features patching from the chromagram where the x -axis is for the beat number, and the y -axis is for the bin number (i.e., 12 semitones). At the right-hand side of Fig. 4, a color bar represents the intensity of semitones. Red color represents high energy and blue color represents low energy. Figure 4 displays two types of patches. If a patch contains too few beats, it may lose the melody of music. However, if a patch contains excessive beats, measuring similarities is difficult. The patch length is a variable parameter to be determined in the experiment. In addition, Fig. 4 shows the hop, which is the number of beats jump between two patches. Hops can preserve the continuity between two patches. The number of beats to be hopped is also a variable parameter to be determined in the experiment.

In the proposed method, the bin energy normalization was applied for each bin (semitone). For each frame, total energy of 12 semitones was normalized to 1. The bin energy normalization is written as follows:

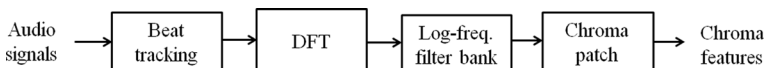


Fig. 3 Flow chart for the proposed method

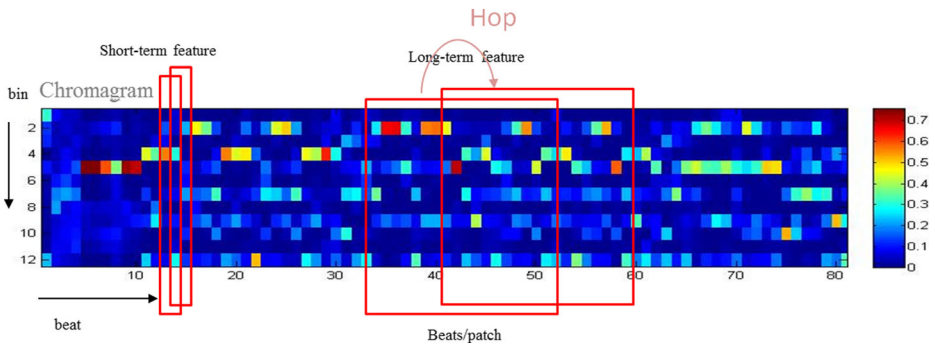


Fig. 4 An example for chroma features patch

$$E^*(b_i, t) = \frac{E(b_i, t)}{\sum_{i=1}^{12} E(b_i, t)} \tag{5}$$

where $E(b_i, t)$ is chroma energy in the i th bin at the t th frame. Figure 5 illustrates the comparison result for bin energy normalization. After normalization, the color difference (energy difference) for each frame (the vertical line) is enhanced regardless of in the chromagram or descriptor representation.

3.2 Deep chroma features learning

Each patch of chroma features is trained by SAE for intermediate representation. The intermediate representation reduces the dimension for similarity comparison. The dimensionality and weights of the hidden layer in the SAE are variable parameters to be determined. To determine these parameters, different pop songs, which were not part of covers80, were collected as the training set, for the parameter determination. As these parameters determined, each patch of chroma features, no matter from the cover song or data set, can be transformed to be the same dimension of descriptors.

Figure 6 illustrates an example for SAE to transform chroma features into descriptors. The input data for each patch, for example, are with dimensionality 480, which composes of 12 bins by 40 beats. This dimension is equal to the dimension of the first hidden layer as shown at the bottom left of Fig. 6. Weights in the final hidden layer are descriptors whose dimensions are 1×40 . In other words, the compression rate is 1/12. Finally, the reconstruction data is plotted on the right side, and each patch has the dimension of 12×40 , which is the same as that of input data.

The SAE is usually applied to project input data into higher dimensional output representations. However, in this example, the input chroma features are of high-dimensionality, 12×40 , and the output descriptors are of low-dimensionality, 1×40 , as shown in Fig. 6, where the compression rate reaches to 1/12.

3.3 Matching and scoring

The final step of the proposed method, plotted in the Fig. 2, is the matching and scoring. Descriptors in the query song are compared with descriptors of songs in the data set. The

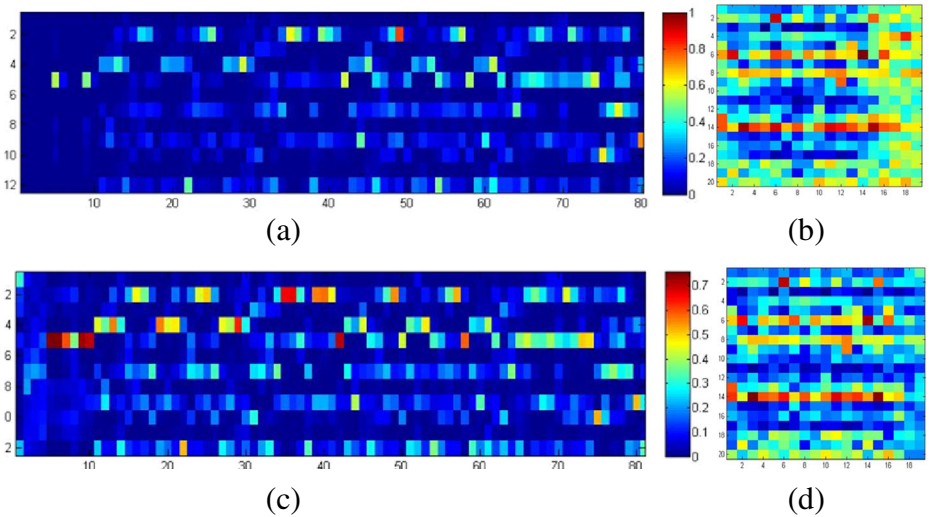


Fig. 5 Bin energy normalization: **a** chromagram and **b** descriptor are before normalization; **c** chromagram and **d** descriptor are after normalization

matching procedure for the cover song is the pairwise comparison to each song in the data set. That is, each cover song needs to compare all of the descriptors in a song, and this matching procedure is iteratively repeated until all songs in the data set have been compared.

The DTW method is applied for similarity scoring, which accumulates the score of all of the descriptors in a song. However, each song may have a different playing time. Considering

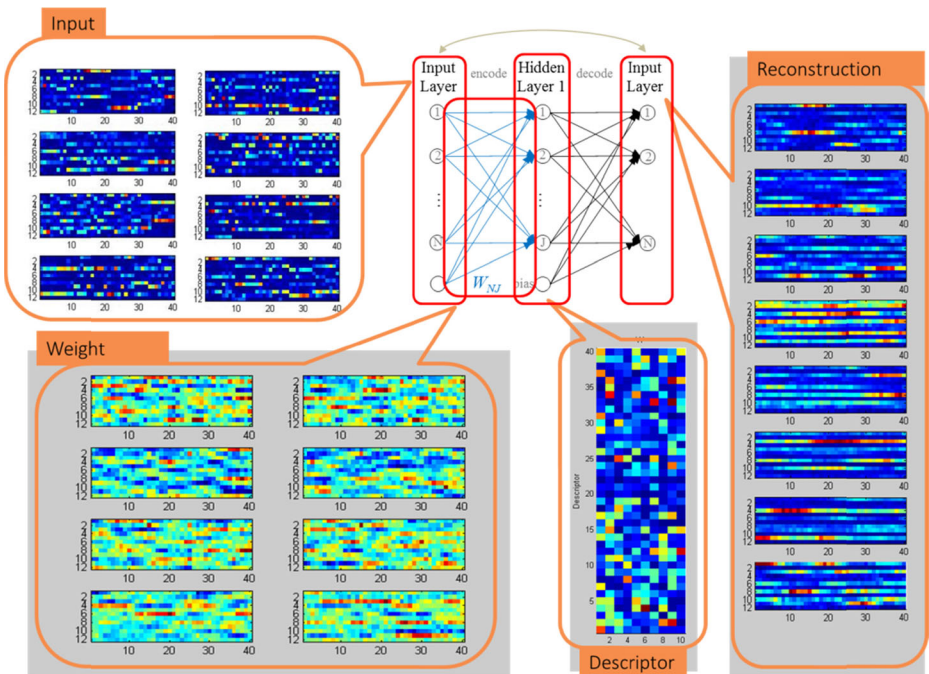


Fig. 6 An example to illustration of the chroma feature transformation by using SAE

DTW, songs of different lengths may have different numbers of patches, and this accumulates different similarity scores. Hence, a postprocessing procedure is added to normalize the length of each song. In other words, the similarity score is divided by a length of the reference song, which can be represented as follows:

$$DTW(Q, D)^* = \frac{DTW(Q, D)}{\text{length}(D)} \quad (6)$$

where Q represents the query song, and D represents the reference song.

Mean reciprocal rank (MRR) [36] and binary decision (BD) [4] were used to evaluate the performance of music retrieval. For MRR, the reference songs in the database are classified into different ranks, or top N lists. Songs with different ranks are then assigned different scores. The scoring is the reciprocal value of its rank. For example, if a reference song is recorded in rank n , the correct retrieval for this song can obtain $1/n$ score. The MRR scoring is obtained using (7):

$$\text{MRR} = \frac{1}{Q} \sum_{q=1}^Q \frac{1}{\text{rank}(Query_q)} \quad (7)$$

where Q is the retrieval number and $\text{rank}(query_q)$ is the rank list of the q th retrieved song. The BD is that given a query song A and two songs B and C, find which one, B or C, is a cover song of A [4].

4 Experimental results

Experiments were conducted to evaluate the performance of the proposed method. System parameters were determined, and the system performances were compared with other methods. Two databases, covers80 [8] and Million Song Dataset (MSD) [5], were applied for testing.

Experimental environment is described as follows. We used a personal computer with CPU, Intel Core i7-3770 (8 cores), 3.4Hz and RAM, 8GB Ram DDR3-1666 MHz. The software was Matlab2013a, and the deep learning tools [19] were applied. Furthermore, the sampling rate was 16,000 Hz. The size of the DFT was 2048 points, and the size of the window was 1024 points, with 512 points of overlap [32]. The DTW program was referred to [9]. To select suitable resolution of semitones, frequency was restricted, ranging from 130.81 Hz ~ 987.77 Hz, i.e. C3 ~ B5 [6]. Table 1 lists parameters for deep learning. Parameters for deep learning can refer to [26, 27].

4.1 Identification for covers 80

Experiments were first conducted to determine system parameters. Parameters include descriptor length, tempo, patch length (beats), and hops. The covers80 dataset contains 80 pop songs and 80 cover songs. Because no training songs were available in covers80, 698 pop songs, which were not part of covers80, were used in this study as training data. Figure 7 plots the MRR performance under different length of descriptors. Figures 7a and b are for Tempos 60 and 120, respectively. This experiment was selected based on 40 beats per patch and hop

Table 1 Parameters for deep learning

Hidden layer dimensionality	480-480-20
Learning rate	1
momentum	0.4
Sparsity	0.1

20. The optimal length for the descriptor is 1×20 , and was thus applied for SAE to determine other parameters. Table 2 lists the highest score of MRR, 0.658, and its corresponding matching time, for all descriptors, is 36.7 s. This performance was obtained under the following settings: 40 beats per patch, Tempo 120, and 20 beats for the hop.

If the chroma features for the same song are extracted using different tempos, they may represent different chroma features in a patch. Figure 8 shows the chroma features from the same song but extracted by Tempos 120 and 60, respectively. It shows that the same position of patch may contain different energy densities. We thus applied multitempo descriptors at the data set to improve the matching performance. Multitempo descriptors refer to chroma features of a reference song in the database that are created according to different tempos, and the similarity scoring involves selecting the highest scores from the similarity measure among these multitempo descriptors. In this study, tempos 120 and 60 were selected as the estimators, and their MRR and matching time are plotted in Fig. 9.

The performance of the proposed method was compared with that of other systems, including LabROSA'06 (Tempos 120 and 60), and LabROSA'07 (Tempos 120 and 60) from the covers80. Figure 9 plots the performances of MRR and similarity matching time. It shows that the proposed method could achieve an MRR of 0.658 with a matching time of 36.7 s, and an MRR of 0.695 with a matching time 46 s. by using two temporal descriptors. The proposed method improved the MRR performance and reduced the matching time by approximately 80 % compared with LabROSA'07 (Tempo 120). Consequently, deep chroma feature learning achieves better representation and retrieval performance than the traditional method.

The performance of proposed method was compared with Tralia and Bendich's [34], a recent work for cover song identification. We followed their procedure to evaluate the identification performance. From the covers80 dataset, given a song from set A, compute

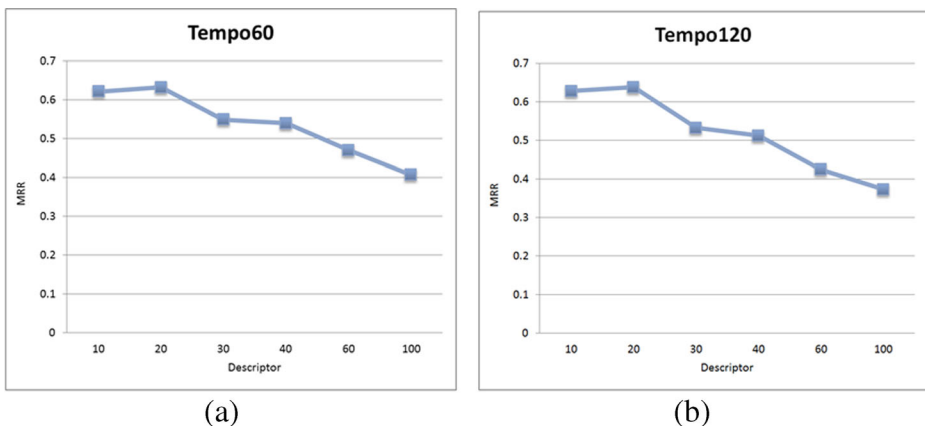
**Fig. 7** The descriptor length decision: **a** tempo120 **b** tempo60

Table 2 Performance of MRR and matching time for covers80

Descriptor 20		Hop 10 (beats)		Hop 20 (beats)		Hop 30 (beats)	
Tempo	Beats/patch	MRR	Time (sec.)	MRR	Time (sec.)	MRR	Time (sec.)
120	120	0.604	55.4	0.658	28.3	0.62	19.4
	80	0.638	63.3	0.639	32.3	0.591	22.1
	40	0.638	71.6	0.658	36.7	0.563	25
60	120	0.493	11.6	0.568	6.2	0.552	4.4
	80	0.521	15.4	0.588	8.1	0.585	5.7
	40	0.58	19.7	0.632	10.3	0.597	7.2

the score for all songs from set B and declare the cover song to be the one with the maximum score. The score for the proposed method was 40/80, and the score for theirs was 42/80 [34]. The performance of the proposed method is close to their performance.

4.2 Identification for million song dataset

The MSD collects metadata and audio features, composed of one million songs. Its subset, SecondHand Songs Dataset (SHSD) is used for cover songs test. SHSD contains training set, test set, and binary set, three datasets. Binary set collects 1,500 songs, including 500 original songs, 500 cover songs, and 500 random selected songs. Unlike the covers80, the MSD has provided the tempo information of each song.

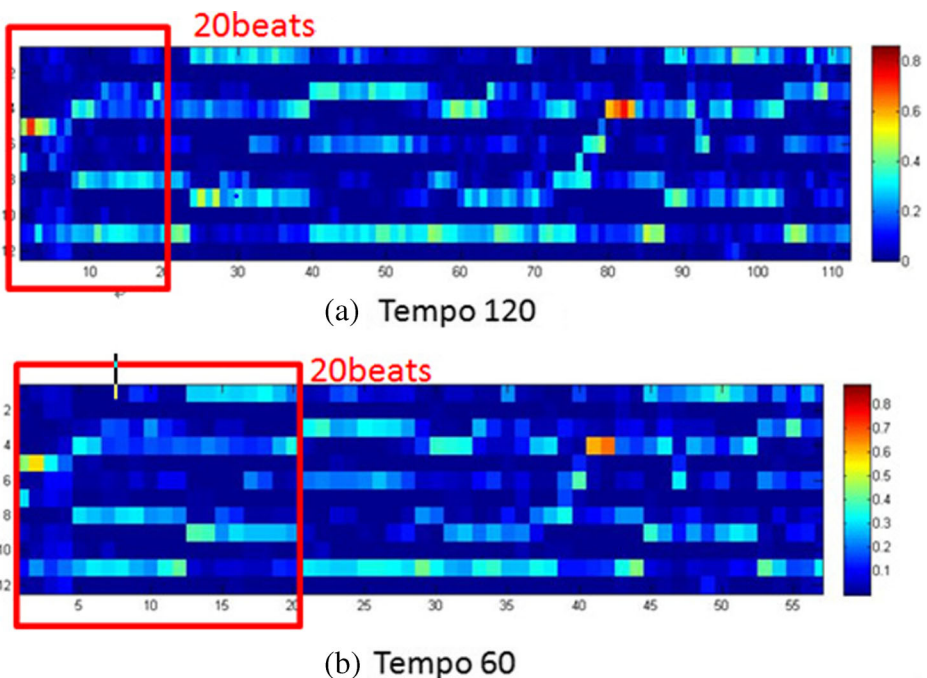


Fig. 8 Chroma features under different tempo extraction

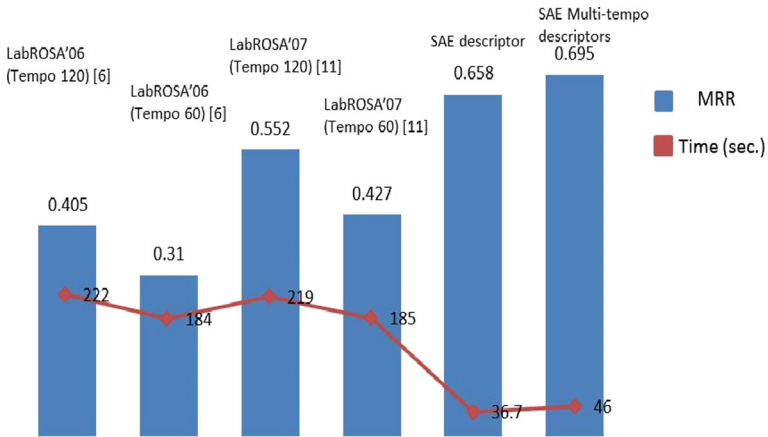


Fig. 9 Performance comparison with other systems from covers80

Table 3 lists the performance levels of BD and matching time of proposed method under different parameters setting for the binary set. It shows that the descriptor trained by hop 20 and 40 patches of chroma features has the best BD performance. The performance of the proposed method was compared with that of LabROSA'07, whose BD performance was 0.51 and the matching time was 38 s. It shows that the proposed method achieves better BD performance, and reduces the matching time by approximately 90 %.

5 Conclusion

Traditional methods for cover song identification are to extract chroma feature data, and make pairwise comparisons. Patching chroma features can add temporal information to preserve the melody of a song. The matching performance can thus be improved. However, this method increases the dimension for similarity matching, and thus increases the matching time. In this paper, deep learning was applied. An intermediate representation was trained to reduce the dimension of each patch of chroma features. The training was performed using a sparse autoencoder. Experimental results showed that the propose method achieved better accuracy for identification and spent less time for similarity matching in both covers80 dataset and Million Song Dataset as compared with traditional approaches.

Although the proposed method has better performance than traditional methods, the accuracy of identification still cannot reach to the best reward in recent MIREX evaluation

Table 3 Performance of BD and matching time for the MSD

		Hop10		Hop20		Hop30	
patch	descriptor	BD	time (sec.)	BD	time (sec.)	BD	time (sec.)
120	20	0.74	5.2	0.75	2.9	0.75	2.0
80	20	0.75	6.3	0.77	3.5	0.76	2.8
40	20	0.74	7.2	0.772	4.0	0.75	3.1

campaigns. In the future, the deep learning can be applied to recent methods of cover song identification, and the best representation by deep learning needs to be explored.

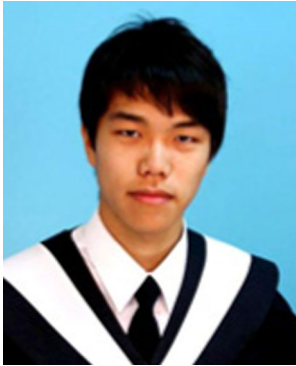
References

1. Al-Shareef AJ, Mohamed EA, Al-Judaibi E (2008) One hour ahead load forecasting using artificial neural network for the western area of Saudi Arabia. *Int J Elec Compu Eng* 3(13):834–840
2. Bengio Y (2009) Learning deep architectures for AI. *Found Trends Mach Learning* 2(1):1–127
3. Bengio Y, Courville A, Vincent P (2013) Representation learning: a review and new perspectives. *IEEE Trans Pattern Anal Mach Intell* 35(8):1798–1828
4. Bertin-Mahieux T, Ellis D (2012) Large-scale cover song recognition using the 2D-Fourier transform magnitude The 13th ISMIR Conference
5. Bertin-Mahieux T, Ellis D, Whitman B, Lamere P. (2011) The million song dataset In Proceedings of ISMIR
6. Chang TM, Hsieh CB, Chang PC (2014) An enhanced direct chord transformation for music retrieval in the AAC domain with window switching. *Multimed Tools and Appl* 74(18):7921–7942
7. Ellis DPW (2006) Beat tracking with dynamic programming MIREX 2006 Audio Beat Tracking Contest system description
8. Ellis DPW (2007) The “covers80” cover song data set. [Online]. Available: <http://labrosa.ee.columbia.edu/projects/coversongs/covers80/>
9. Ellis D. Dynamic Time Warp (DTW) in Matlab. [Online]. Available: <http://labrosa.ee.columbia.edu/matlab/dtw/>
10. Ellis DPW, and Cotton C (2006) The 2007 LABROSA cover song detection system. Music Information Retrieval Evaluation eXchange (MIREX) extended abstract
11. Ellis DPW, Poliner GE (2007) Identifying cover songs with chroma features and dynamic programming beat tracking. *IEEE Int. Conf. Acoustic, Speech and Signal Processing (ICASSP)*, Honolulu, HI, 1429 – 1432
12. Fujishima T (1999) Realtime chord recognition of musical sound: a system using common lisp music. *Int. Comput. Music Conf.*, Beijing 464–467
13. Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. *Neural Comput* 18(7):1527–1554
14. Hinton GE, Salakhutdinov RS (2006) Reducing the dimensionality of data with neural networks. *Science* 313(5786):504–507
15. Humphrey EJ, Nieto O, Bello JP (2013) Data driven and discriminative projections for large-scale cover song identification. The 14th ISMIR Conference: 149–154
16. Keogh E, Ratanamahatana CA (2005) Exact indexing of dynamic time warping. *Knowl Inf Syst* 7(3):358–386
17. Lee K (2006) Identifying Cover Songs from Audio Using Harmonic Representation. Music Information Retrieval Evaluation eXchange (MIREX) extended abstract
18. Nieto O, Bello JP (2014) Music segment similarity using 2D-Fourier magnitude coefficients. *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*: 664–668
19. Palm RB (2012) Deep learning toolbox, [Online]. Available: <http://www.mathworks.com/matlabcentral/fileexchange/38310-deep-learning-toolbox>
20. Ranzato M, Boureau Y, LeCun Y (2007) Sparse feature learning for deep belief networks. *Advances in Neural Information Processing Systems* 20 (NIPS)
21. Ranzato M, Poultney C, Chopra S, LeCun Y (2006) Efficient learning of sparse representations with an energy-based model NIPS
22. Ravuri S, Ellis DPW (2010) Cover song detection: From high scores to general classification. *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Dallas, Texas, U.S.A. 65–68
23. Rifai S, Vincent P, Muller X, Glorot X, Bengio Y (2011a) Contractive auto-encoders: Explicit invariance during feature extraction ICML

24. Riley M, Heinen E, Ghosh J (2008) A text retrieval approach to content-based audio retrieval. *Int. Conf. on Music Information Retrieval*, Philadelphia, Pennsylvania, U.S.A. 295–300
25. Sailer C, Dressler K (2006) Finding cover songs by melodic similarity. *Music Information Retrieval Evaluation eXchange (MIREX) extended abstract*
26. Salakhutdinov R (2009) Learning deep generative models doctoral dissertation. University of Toronto, Toronto
27. Salakhutdinov R Nonlinear dimensionality reduction using neural networks. Available: http://www.cs.toronto.edu/~rsalakhu/talks/NLDR_NIPS06workshop.pdf
28. Serrà J, Gómez E (2008) Audio cover song identification based on tonal sequence alignment. *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Las Vegas, Nevada, U.S.A. 61–64
29. Serrà J, Gómez E, Herrera P, Serra X (2008) Chroma binary similarity and local alignment applied to cover song identification. *IEEE Trans Audio Speech Lang Process* 16(6):1138–1151
30. Serrà J, Gómez E, Herrera P (2010) Audio cover song identification and similarity: background, approaches, evaluation, and beyond. *Adv Music Inf Retr* 274(14):307–332
31. Shepard RN (1982) Structural representations of musical pitch. In Deutsch, D, editor, *The Psychology of Music*, First Edition. Swets & Zeitlinger
32. Signal processing toolbox, time-dependent frequency analysis (specgram). [Online]. Available: <http://faculty.petra.ac.id/resmana/private/matlab-help/toolbox/signal/specgram.html>
33. Smolensky P (1986) Information processing in dynamical systems: foundations of harmony theory. in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, D. E. Rumelhart, J. L. McClelland, and C. PDP Research Group, Eds. Cambridge, MA, USA: MIT Press 194–281
34. Tralie CJ, Bendich P (2015) Cover song identification with timbral shape sequences. arXiv preprint arXiv:1507.05143
35. Vincent P, Laroche H, Bengio Y, Manzagol, PA. (2008) Extracting and composing robust features with denoising autoencoders *ICML*
36. Voorhees EM (1999) Proceedings of the 8th Text Retrieval Conference. *TREC-8 question answering track report*. 77–82
37. Wang R, Han C, Wu Y, Guo T (2014) Fingerprint classification based on depth neural network. arXiv preprint arXiv:1409.5188
38. Witmer R, Marks A (2006) In: Macy L (ed) *Cover*, grove music online. Oxford Univ. Press, Oxford



Jiunn-Tsair Fang received the Ph.D. degree in electrical engineering from National Chung-Cheng University, Taiwan in 2004. Currently, he is an assistant professor in the Department of Electronic Engineering at Ming Chuan University, Taiwan. His researching interest includes video/image coding, audio processing, and joint source and channel coding.



Chi-Ting Day received the B.S. degree in electrical engineering and the M.S. degree in communication engineering from NCU, Taiwan, in 2012 and 2014, respectively. His research interest is audio coding.



Pao-Chi Chang received the B.S. and M.S. degrees from National Chiao Tung University, Taiwan, and the Ph. D. degree from Stanford University, California, 1986, all in electrical engineering. From 1986 to 1993, he was a research staff at IBM T.J. Watson Research Center, New York. In 1993, he joined the faculty of NCU, Taiwan, where he is presently a professor in the Department of Communication Engineering. His main research interests include speech/audio coding, video/image compression, and multimedia retrieval.