Multimedia Systems
https://doi.org/10.1007/s00530-019-00613-z

SPECIAL ISSUE PAPER

Toward efficient indexing structure for scalable content-based music retrieval

Jialie Shen[1] · Mei Tao[2] · Qiang Qu[3] · Dacheng Tao[4] · Yong Rui[5]

Abstract
With advancement of various information processing and storage techniques, the scale of digital music collections has been growing at very fast speed during recent decades. To support high-quality content-based retrieval over such a large volume of music data, how to develop indexing structure with good effectiveness, efficiency and scalability becomes an important research issue. However, existing techniques mainly focus on improving query efficiency. Very few approaches have been proposed to address issues related to scalability and accuracy. In this study, we address the problem via introducing a novel indexing technique called effective music indexing framework (EMIF) to facilitate scalable and accurate music retrieval. It is designed based on a "classification-and-indexing" principle and consists of two main functionality modules: (1) music classification—a novel semantic-sensitive classification to identify an input song's category and (2) indexing module—multiple local indexing structures, one for each semantic category to reduce query response time significantly. In particular, the classification model combining linear discriminative mixture model (LDMM) and advanced score fusion scheme has been applied to estimate category of music accurately. Layered architecture enables EMIF to enjoy superior scalability and efficiency. To evaluate the approach, a set of experimental studies has been carried out using two large music test collections and the results demonstrate various advantages of EMIF over state-of-the-art approaches including efficiency, scalability and effectiveness.

Keywords Multimodal · Indexing · Content-based music retrieval · Efficiency · Scalability

1 Introduction

Recent years have witnessed a fast growth in digital multimedia data from various real application domains (e.g., online streaming service, education and entertainment) [1–7]. To achieve fast and reliable access on such large volume of multimedia data, efficiency becomes an important issue and an intelligent indexing structure is essential to scale the data space. Particularly, advances in technologies such as networks, cloud storage and mobile device boosted volume increase of enormous music data in different formats. For example, according to Nelsen market report, on-demand song streaming volume is up 45%, having already exceeded 268 billion in 2018. In response to the needs for tools to fast access such large size of music information, different kinds of indexing methods have been recently proposed to support efficient content-based music information retrieval (CBMIR) and analysis during the last decades [8–14]. The specific examples include the CM*F [15], QUC-tree [16], LSH-based approaches [17–19] and so on. In general, most

✉ Qiang Qu
qiang@siat.ac.cn

Jialie Shen
j.shen@qub.ac.uk

Mei Tao
tmei@live.com

Dacheng Tao
dacheng.tao@sydney.edu.au

Yong Rui
yongrui@lenovo.com

[1] Queen's University, Belfast, UK

[2] JD AI Research, Beijing, China

[3] Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Beijing, China

[4] UBTECH Sydney Artificial Intelligence Centre and School of Information Technologies, Faculty of Engineering and Information Technologies, University of Sydney, Darlington, NSW, Australia

[5] Lenovo, Beijing, China

Published online: 27 April 2019

of them are designed based on the principle called "feature transformation", which has been emerging as an important search paradigm. The basic idea is to extract the low-level acoustic features (usually in the form of a multidimensional feature vector) from each music document in the database and then to map the features into points in a high-dimensional feature space as signature. The distance between two feature points is frequently used as a measure of similarity between two audio files. Once the distance or similarity function is defined in the feature space, a nearest neighbor search can be used to retrieve the objects that satisfy the criteria specified in a given query.

1.1 Motivation

While existing approaches are efficient in some specialized music IR and database applications [19, 20], many open problems still remain unsolved. First, good scalability and low cost-of-maintenance are essential to modern music information retrieval systems whose contents could easily be huge and updated frequently. Notice that the rebuilding cost for existing indexing structure is directly related to the data size. Unfortunately, relatively little attention has been paid on improving performance in this direction and associated update operation generally results in very expensive computational cost. Further, efficiency of query processing (e.g., response time or system reconstruction time) based on the existing approaches could be decreased dramatically when the size of music collections becomes larger and larger. Moreover, recently proposed indexing structures (e.g., M-tree, Hybrid tree, Δ-tree, QUC-tree and LSH) focus primarily on improving query efficiency but generally ignore the quality of retrieval results. In fact, due to well-known "semantic gap", accurate query processing cannot be achieved using indexing structure constructed based on low-level features only [21]. In developing comprehensive music content descriptors for accurate similarity retrieval, we need to combine low-level feature to produce more effective music signature. This introduces two correlated subproblems: (1) how should the various low-level features be fused for particular search task and (2) how can the combined feature be compact enough to enable fast search and classification using existing indexing algorithms or machine learning methods. Naturally, raw acoustic feature vectors have high dimensions (e.g., some of them can have up to 100 dimensions) and creating a generalized high-dimensional index that can handle hundreds of dimensions is still an unsolved problem to date [22]. This is because many existing indexing methods have an exponential time and space complexity as the number of dimensions increases. When indexing high-dimensional vectors, they will not perform better than sequential scanning of the database. Moreover, existing study generally ignores scalability issue of indexing structure, which is crucial for retrieval and management of large-scale music databases. This is due to the fact that such systems can potentially contain thousands of audio files for retrieval and the contents of the data collections could be changed frequently. The associated cost could be extremely high. Motivated by the concerns, several dimensional reduction methods were proposed to generate smaller content representation to improve efficiency and effectiveness. However, they still suffer from poor scalability—expensive update cost or/and low effectiveness—less comprehensive content representation [23].

1.2 Core technical contributions

In this article, we present a scalable and effective indexing framework called EMIF[1]. to facilitate fast, scalable and effective CBMIR. The main contributions to this technical advancement can be summarized as follows:

- We develop multiple-feature-based music class profiling model to characterize different music categories. In terms of functionality, it is a probabilistic classifier to estimate correct label of input music. The scheme can effectively combine multiple features to enhance categorization effectiveness and thus improve the overall retrieval accuracy greatly.
- Distinguished from previous approaches, EMIF'S architecture is designed based on a "Classify-and-Indexing" principle and applies a multiple-layer structure, which consists of two basic components—classification module and indexing module. This innovation enables superior scalability, efficiency and significantly reduces system reconstruction cost, which is a major overhead for existing solutions.
- We develop a novel deep learning-based music signature generation scheme called DMSG to compute compact and comprehensive music descriptor—deep music signature (DMS). The approach can effectively combine various kinds of acoustic features to produce small feature vector to enhance the indexing and retrieval based on the existing access methods.
- We conduct a set of detailed experimental studies and result analysis based on three large test collections. It demonstrates that EMIF enjoys superior scalability, effectiveness and efficiency over the existing approaches.

The rest of the paper is structured as follows: Sect. 2 gives background knowledge and literature review on the research. Section 3 presents the architecture of the EMIF system and associated learning algorithms. Section 4 gives a detailed

[1] **EMIF** stands for **E**ffective **M**usic **I**ndexing **F**ramework.

introduction about experimental configuration. Next, Sect. 5 describes a set of comprehensive experiments over three large music testbeds and gives a detailed analysis of the related results. Finally, Sect. 6 draws conclusions and indicates several future directions for the work.

2 Related work

In this section, we mainly focus on introducing previous work and background knowledge related to CBMIR. In Sect. 2.1, we survey the existing approaches of multidimensional indexing structures. Then, Sect. 2.2 briefly overviews the prior work about how to model music signal and generate music content descriptor.

2.1 Multidimensional indexing structure

The first relevant stream of literature is about developing high-dimensional access methods (e.g., indexing tree and dimension reduction). To support fast similarity search in high-dimensional databases, various schemes have been proposed in recent decades [24]. The typical examples include M-tree [25], the VA-file [26], Hybrid tree [27], the iDistance [28] and Hashing [17, 29–33]. In [25], the authors proposed the height-balanced M-tree to organize and search large datasets from a generic metric space, where object proximity is defined by a distance function satisfying the positivity, symmetry and triangle inequality postulates. The strength of the M-tree lies in maintaining the pre-computed distance in the index structure; however, it still suffers from the dimensionality curse. To solve the problem, representation of the data points using smaller and approximate signatures has been also proposed in recent years. The typical examples under this paradigm include the VA-file [26] and the IQ-tree [34]. The basic idea of VA-file is to divide the data space into 2^b rectangular cells, where b denotes a user-specified number of bits. The scheme allocates a unique bit-string of length b for each cell, and approximates data points that fall into a cell by that bit-string. The VA-file itself is simply an array of these approximations. KNN searches are performed by scanning the entire approximation file, and by excluding the vast majority of vectors from the search (filtering step) based on these approximations. When searching for the nearest neighbors, the entire approximation file is scanned and the upper and lower bounds on the distance to the query can easily be determined based on the rectangular cell represented by the approximations. After the filtering step, a small set of candidates are then visited and the actual distances to the query point Q are determined. The VA-file has been shown to perform well for disk-based systems as it reduces the number of random I/Os. The IQ-tree was proposed based on the concept of quantization [34]. The compressed index has a three-level structure: the first level is a regular (flat) directory consisting of minimum bounding boxes, the second level contains data points in a compressed representation, and the third level contains the actual data. On the other hand, the compressed MBRs can reduced the disk I/O during the search processing. One-dimensional transformations provide another direction for high-dimensional indexing. The iDistance [28] was presented as an efficient method for KNN search in a multidimensional space. iDistance partitions the data and selects a reference point for each partition. The data points in each cluster are transformed into a single-dimensional space based on their similarity with respect to a reference point. It then indexes the distance of each data point to the reference point of its partition. Since this distance is a simple scalar, with a small mapping effort to keep partitions distinct, it is possible to use a standard B^+-tree structure to index the data and KNN search be performed using one-dimensional range search. More recently, as a novel indexing structure to support fast approximate query processing, LSH has attracted a lot of research attentions. The first LSH-based music search system is developed by Yan [17]. It aims to apply LSH to speed up the nearest neighbor search and the acoustic feature used is short-time Fourier transform (STFT). Yu et al. develop dual-phase LSH-based algorithm to improve accuracy and scalability of content-based music information retrieval systems [29]. More recently, McFee and Lanckriet apply variants of the classical KD-tree to support content-based similarity search over the Million Song Dataset [35].

Due to the difficulty of indexing very high dimensional data space, a reasonable approach might be to reduce the dimensionality to a "reasonable" level (e.g., 10–12 dimensions), and then use an existing "high-dimensional" indexing scheme as an access method (e.g., M-tree or R-tree). In the past 2 decades, there have been a lot of research efforts on developing dimension reduction methods. The techniques can be classified into two independent categories: linear dimension reduction (LDR) and nonlinear dimension reduction (NLDR). Basic idea of LDR is to apply linear statistical analysis to map the original high-dimensional features to low-dimensional ones by eliminating the redundant information from the original feature space. The most well-known statistical approaches for doing this is the principal component analysis (PCA) and linear discriminative analysis (LDA). The fundamental of NLDR is the standard nonlinear statistical analysis and machine learning algorithm, which have been widely explored by various research communities in recent years. However, the drawbacks of NLDR are that the training of a learning algorithm requires high-quality training examples and that training can be computationally inefficient.

In recent years, advanced hashing has been playing more and more important role in support of fast and effective

multimedia information retrieval [30–32]. Consequently, a steady progress in the related field has been observed.

2.2 Music signature generation

The second stream of previous research is about how to model music contents and develop effective scheme to generate comprehensive music signatures. Indeed, various kinds of music features can be applied for categorizing and indexing music collections. They include text, acoustic features and symbolic signature of music melody. Here, our primary focus is on content-based acoustic features.

While there has been a long history of developing effective techniques for speech recognition and music–speech identification, much less attention has been paid on developing small and effective music signatures for effective and efficient retrieval. Many existing systems directly apply the low-level musical features adapted from signal processing communities. They include mel-frequency cepstral coefficients (MFCCs), spectral centroid, linear prediction coefficients, spectral flux, etc. [36]. One typical example using this approach is the scheme proposed by Nam and Berger [37]. In the study, three different kinds of low-level acoustic features (spectral centroid, short-time energy, and zero crossing rate) are extracted and combined as music descriptors to support automatic music genre classification. In [38], a set of nearest feature line methods are developed to facilitate content-based audio retrieval and classification. Lu et al. explore audio classification with nine different audio features including MFCCs, zero crossing rates (ZCR), short-time energy (STE), sub-band power distribution, brightness, bandwidth, spectrum flux (SF), band periodicity (BP) and noise frame ratio (NFR) [39]. In this study, support vector machine (SVM) is used as a classifier. Tzanetakis et al. propose the MARSYAS system to model music signals. It can extract a set of features to describe and represent various acoustic properties such as timbral texture, pitch content and rhythm [40]. Wavelet analysis enjoys superior capability to effectively estimate signals' probability distribution over time and frequency. Motivated by this observation, Daubechies wavelet histogram technique(DWCHs) is proposed to capture more discriminative information from local and global perspective and has been proven to be a very effective approach to generate music signatures [41]. Effective multiple acoustic feature fusion is important for music content modeling. In [42], Shen et al. develop a neural network-based music content descriptor generation scheme to combine various kinds of acoustic feature in nonlinear fashion. The experimental results over three music test collections show that music classification and retrieval based on the approach is a good way to improve the accuracy and robustness. More recently, Song and Zhang develop a regularized least-squares framework to generate music signature for semi-supervise music genre classification [43] (Table 1).

3 System architecture

This section gives a detailed introduction on overall system architecture of EMIF, its two basic components and related algorithms. EMIF, as illustrated in Fig. 1, consists of two major functionality layers—music classification module and indexing module. The main functionality of the first layer is to categorize input music accurately and music indexing module contains a group of deep learning music signature generation schemes and indexing trees, one for each

Table 1 Summary of symbols and definitions

Notation	Definition
c	Music class c
f	Feature type f
L	Loss function of deep learning framework
C	Number of classes in the database
B	Number of blocks for music segmentation
F	Number of acoustic features extracted
M	Number of training examples for logistic fusion function
CF	Score combination function
DMS	Deep music signature
Θ_f^s	Parameter set for GMM
$\boldsymbol{v}_{bf}$	Feature vector extracted from block b for feature type f
$\boldsymbol{V}_f$	Set of feature vectors extracted from different blocks for feature type f
L^c	Final score generated by logistic combination function for class c
l_f^c	Likelihood value generated by category c's profile model using feature type f
$\mathbf{W^c}$	Fusion weight vector of logistic fusion function for class c
L_{sqe}	Squared error loss
$Extract_f$	Feature extraction scheme for feature type f

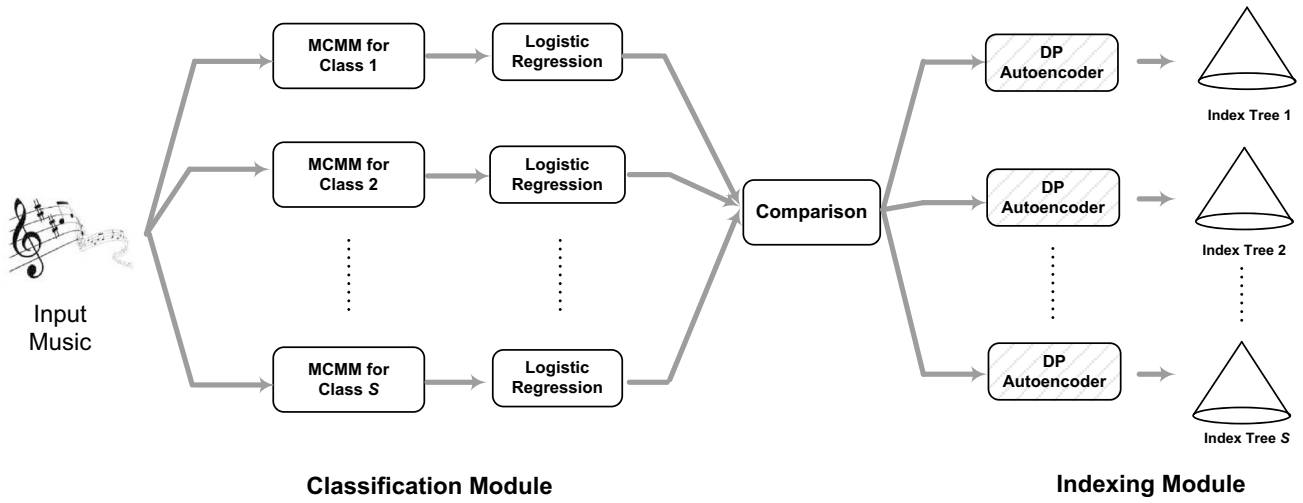

Fig. 1 The architecture of EMIF indexing framework

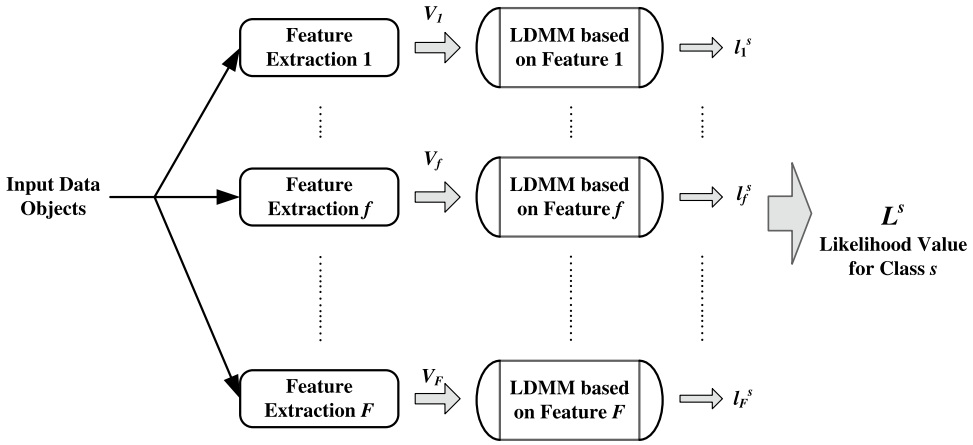

Fig. 2 Statistical class modeling module based on multiple features for category c

category. To search a set of similar music based on input example, different features are extracted first and then music category c can be identified. Finally, top k songs are returned after search using the local indexing tree for category c.

3.1 Multifeature-based music category modeling

In EMIF, the first layer consists of C music category modeling modules (MCMM), which aim to effectively model each music category (or class) and support classification. Each MCMM in EMIF corresponds to one music category in the database and C is the total number of music categories in the database. As illustrated in Fig. 2, MCMM is made up of two parts: (1) feature extraction, and (2) a set of LDMMs (linear discriminative mixture model) built for statistical modeling of music category based on various kinds of acoustic features. A LDMM is a stochastic model combining the advantages of LDA and Gaussian mixture model (GMM). The novelty of LDMM is its greatest capability and flexibility to support effective feature modeling. In MCMM, each LDMM corresponds to one acoustic feature type.

3.1.1 Feature extraction

Feature extraction is the computational process to calculate a numerical representation of music documents. In EMIF, the partition-based approach is applied to extract multiple local features. The basic idea is that an input signal is first segmented into small blocks and then different kinds of acoustic features are extracted from each block as basic content representation. Specifically, the features considered in this study include timbre, rhythm and pitch. The extraction process can be denoted as

$$\boldsymbol{V}_f = \text{Extract}_f(\text{MD}) = [\boldsymbol{v}_{1f}, \boldsymbol{v}_{2f}, \ldots, \boldsymbol{v}_{Bf}], \quad (1)$$

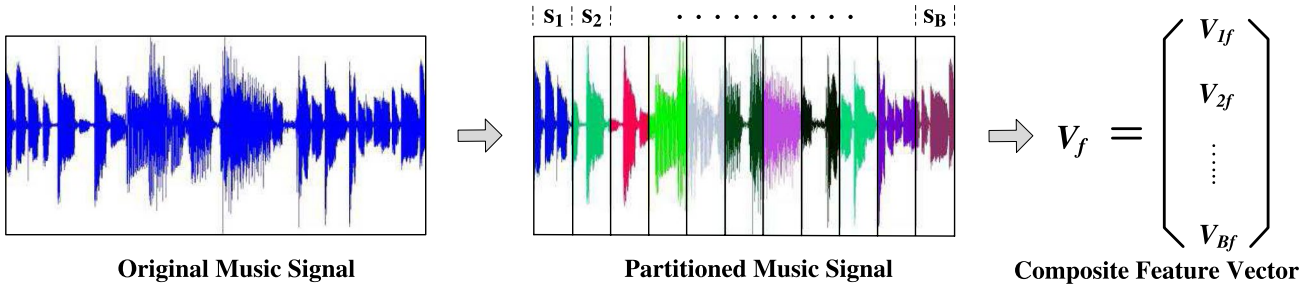

Fig. 3 Partition-based feature extraction scheme used in EMIF

where $\boldsymbol{V}_f$ is the set of vectors for a feature f extracted from the B blocks of the input music MD. For our system, GMM is used as a statistical processor to model feature distributions for the particular semantic concepts. Based on each kind of feature, a GMM-based category model can be trained separately for the task of class identification and detail information about acoustic features used in this study is as below (Fig. 3).

- Timbre feature Timbral texture is a global statistical music property used to differentiate a mixture of sounds. It has been widely applied to speech recognition and audio classification. The 33-dimensional feature vector representing timbre feature includes means and variance of spectral centroid, spectral flux, time domain zero crossings and 13 MFCC coefficients (32) plus low energy(1).
- Rhythm feature Rhythmic content indicates reiteration of musical signal over time. It can be represented as beat strength and temporal pattern. The beat histogram (BH) proposed by Tzanetakis et al. [40] is used to describe rhythmic content. The 18-dimensional feature vector is used to represent rhythmic information of music and includes relative amplitude of the first six histogram peaks (divided by the sum of amplitudes), ratio of the amplitude of five histogram peaks (from second to sixth) divided by the amplitude of the first one, period of the first six histogram peaks, and overall sum of the histogram.
- Pitch feature Pitch is an important acoustic feature used to characterize melody and harmony information in music file. It can be extracted via the multi-pitch detection techniques [44]. The 18-dimensional pitch feature vector includes the amplitude and periods of the maximum six peaks in the histogram, pitch interval between the six most prominent peaks and the overall sums of the histograms.

3.1.2 Statistical category profiling with linear discriminative mixture model

For the purpose of effective category identification, EMIF constructs a statistical model for each class using multiple multiple features. To achieve this, the individual feature of the music objects is extracted, and then individual profiling model for one class is built based on each feature. In our framework, category profiling aims to capture statistical properties of different features using linear discriminative mixture model (LDMM), which is a novel classification scheme combining the advantages of both LDA and GMMs. The main advantage of LDA over other linear subspace methods is to generate a discriminative feature space to maximize the ratio of between-class scatter against within-class scatter (Fisher's criterion). In the LDMM for ea

textittextitch acoustic feature, LDA is used as feature extraction that provides a linear transformation of raw features (n-dimensional) to m-dimensional subspace (m dimension, $m < n$). Consequently, the samples belonging to the same category are close together and the samples from different categories are far apart. At the same time, since LDA can significantly reduce the dimensionality of raw feature, LDMM's training and classification will be accelerated greatly. With GMM, the probability of class s can be modeled as a random variable drawn from a probability distribution for a particular feature f after LDA transformation. Given a parameter set Θ_f^s based on feature f, the probability distribution is present as a mixture of multivariate component densities:

$$P_f^c(\boldsymbol{V}_f|\{sTheta) = \prod_{b=1}^{B}\left\{\sum_{j=1}^{J} w_{fj}^c p_f^c(\boldsymbol{v}_{bf}|\,\boldsymbol{\mu}_{fj}^c, \boldsymbol{\Sigma}_{fj}^c)\right\}, \tag{2}$$

where $\boldsymbol{V}_f = \{\boldsymbol{v}_{1f}, \boldsymbol{v}_{2f}, \ldots, \boldsymbol{v}_{Bf}\}$. The Gaussian density is used as the multivariate component in this study, according to

GMM $\Theta_f^s = \{w_{fj}^c, \mu_{fj}^c, \sum_{fj}^c | \text{where } 1 < j < J\}$, where w_{fj}^c, $\boldsymbol{\mu}_{fj}^c$ and $\boldsymbol{\Sigma}_{fj}^c$ denote, respectively, mixture weights, mean vectors and covariance matrices. Also, $p_f^c(\boldsymbol{v}_{bf} | \boldsymbol{\mu}_{fj}^c, \boldsymbol{\Sigma}_{fj}^c)$ is the probability of a class label s based on feature f extracted from segment b and given data $\boldsymbol{v}_{bf}$, and can be easily calculated using the Gaussian density function and associated parameters $\{\boldsymbol{\mu}_{fj}^c, \boldsymbol{\Sigma}_{fj}^c\}$.

In EMIF, an EM algorithm is used to determine a set of model parameters [45]. The EM is an iterative optimization method to estimate some unknown parameters based on given data set. This process of estimation is an iterative hill-climbing procedure. The goal is to derive an optimal parameter set Θ_f^s via a maximum likelihood estimation. The training procedure is repeated until the log-likelihood value is increased by less than a predefined threshold from one iteration to the next. Since EMIF considers different features, the overall training procedure will be repeated multiple times, once for each feature. After the training process is completed, the likelihood value generated based on feature f for input feature vector $\boldsymbol{V}_f$ can be given as below:

$$l_f^c = \log(P_f^c(\boldsymbol{V}_f | \Theta_f^s)) = \sum_{b=1}^{B} \log\left(\left\{ \sum_{j=1}^{J} w_{fj}^c p_f^c(\boldsymbol{v}_{bf} | \boldsymbol{\mu}_{fj}^c, \boldsymbol{\Sigma}_{fj}^c) \right\} \right). \quad (3)$$

An overall likelihood value can be derived based on various features for category s, expressed as below:

$$L^c = C^c(\mathbf{l^c}, \mathbf{W^c}), \quad (4)$$

where $\mathbf{l^c} = \{l_1^c, l_2^c, \ldots, l_F^c\}$ and $\mathbf{W^c} = \{W_1^c, W_2^c, \ldots, W_F^c\}$ contain combination weights and scores from the category profiling model for class s. C^c is likelihood value combination function. L^c can be used to quantify the universal similarity distance between and input object and a class label s. In fact, the simplest way to determine combination weights would be to give all combination weights same value, no matter the score is generated based which feature. The key problem for this approach is that different feature might have varied impact on determine category of incoming objects. To alleviate this problem, in the next section, we introduce the logistic function for score fusion and the relative training algorithm to generate score fusion weights. The method can scale likelihood value L^c to [0,1].

3.2 Fusion weight estimation

To gain a comprehensive statistical model for each music category in EMIF, it is very important to develop effective fusion weight estimation scheme to compute likelihood score. In this article, we introduce two approaches, which are similar to the ones used in [46].

3.2.1 Logistic regression-based scheme

In this approach, logistic function is applied as a linear combination scheme CF^c to derive an overall likelihood score. Basic idea is very similar to ones presented in [46]. Logistic function has been widely used in many real applications and serves as key technical component in Logistic Regression (LR) [47, 48]. With logistic functions, formula 5 can be presented as below:

$$L^c = CF^c(\mathbf{l^c}, \mathbf{W^c}) = \frac{1}{1 + exp\left(-y_c \sum_{f=1}^{F} W_f^c l_f^c\right)}, \quad (5)$$

where $y_c = 1$ if this input object belongs to category s, $y_c = -1$ otherwise, W_f^c is the weight for category c's likelihood value generated based on feature f. F is the size of input score and equals the number of feature types extracted in the first layer. L^c denotes the overall relevancy score - conditional probability of class c. The main reason for using LR to estimate parameters is that less statistical assumptions are required and less computational cost is needed for training. More importantly, the output of logistic functions can be mapped to probabilistic value, ranging from 0 to 1. Based on Eq. 5, the likelihood value occurring in the learning samples is

$$\prod_{m=1}^{M} \frac{1}{1 + \exp\left(-y_c \sum_{f=1}^{F} W_f^c l_f^c\right)}, \quad (6)$$

where M is the number of training examples. The goal of the training process is to maximize the overall likelihood value and obtain $\mathbf{W^c}$ to minimize log loss of the model, as below:

$$\sum_{i=1}^{M} ln\left(1 + exp\left(-y_c \sum_{f=1}^{F} W_f^c l_f^c\right)\right). \quad (7)$$

To achieve this goal, an algorithm is developed based on a parallel-update optimization scheme proposed by Collins et al. [49][2] and Fig. 4 shows its details. Theoretically, the basic procedure aims to minimize *LogLoss*. During the training, on each iteration t, the distribution $d_{t,i}$ is updated to increase the weights of misclassified training examples in the previous round. To calculate the data distribution between positive and negative learning examples, the algorithm is revised to give $d_{t,i}$ weight.

[2] For more information, please refer to paper [49].

Fig. 4 Logistic regression training algorithm to determine weights of score fusion for class c

Input: Matrix $MA \in [-1,1]^{M \times F}$ where $MA_{mf} = y_m l_f^m$
N_p is number of the positive training example
N_n is number of the negative training example
Output: Weight vector $\mathbf{W^c}$

begin
$\mathbf{W^c} = (0, 0, ..., 0)$ and $q_0 = (0.5, 0.5, ..., 0.5)$
For $t = 1, 2, ..., T$
$d_{t+1,m} = d_{t,m}\, exp(-\sum_{f=1}^{N} \delta_{t,f} MA_{mf})$
for each positive training example
$$d_{t,m} = \frac{N_n d_{t,f}}{N_n + N_p}$$
for each negative training example
$$d_{t,m} = \frac{N_p d_{t,m}}{N_n + N_p}$$
For $f = 1, 2, ..., F$
$K_{t,f}^{+} = \sum_{i:sign(MA_{mf})=+1} d_{t,m} MA_{mf}$
$K_{t,f}^{-} = \sum_{i:sign(MA_{mf})=-1} d_{t,m} MA_{mf}$
$\delta_{t,f} = \frac{1}{2} ln(\frac{K_{t,f}^{+}}{K_{t,f}^{-}})$
end;
update with $\mathbf{W^c}^{t+1} = \mathbf{W^c}^{t} + \boldsymbol{\delta}^{t}$
end;
return $\mathbf{W^c} = (W_1^c, W_2^c, ..., W_F^c)$
end;

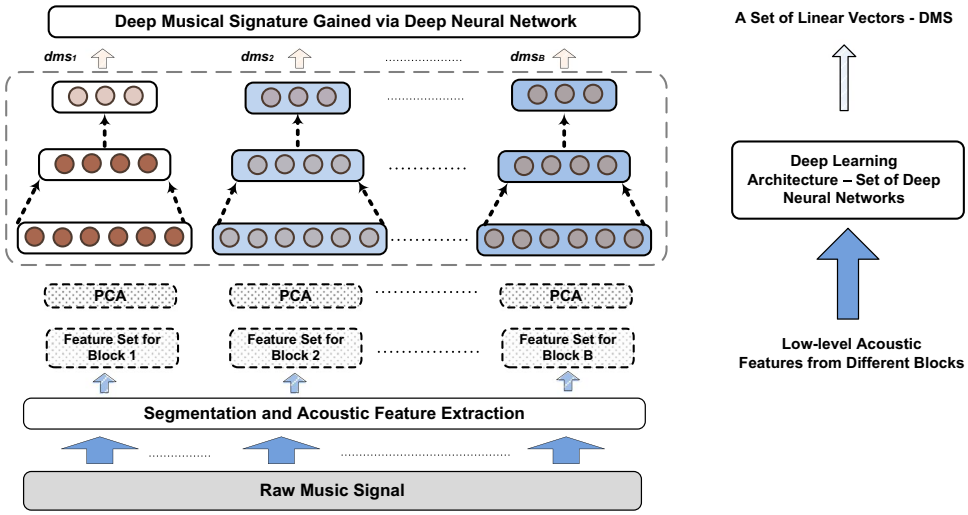

Fig. 5 DMSG—a deep learning-based music signature generation scheme

3.3 Deep music signature generation and music retrieval

The basic principle of EMIF is to identify the category of an input music in the first layer and then carry out query processing using the corresponding local indexing tree in the second layer. To achieve this goal, a deep learning-based music signature generation scheme (DMSG) is developed to combine various low-level acoustic features extracted from different segments into Deep Music Signature (DMS)—a set of linear vectors. Its physical representation can be given as

$$\text{DMS} = \{dms_1, dms_2, \ldots, dms_B\}, \tag{8}$$

where B is number of blocks in the music. Then linear similarity functions (e.g., Euclidean distance) can be applied to calculate the similarity between two music documents. DMSC is deep neural network architecture based on stacked denoising autoencoder (SDA) and principal components analysis (PCA). Figure 5 illustrates its detail structure. It performs learning task via

– PCA is used to preprocess raw input features from different blocks via linear transformation and speed up learning of SDA.

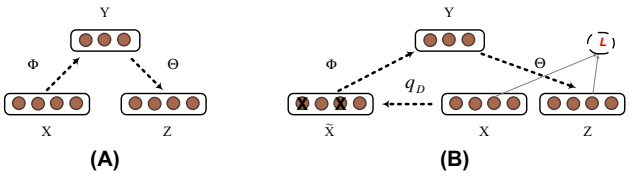

Fig. 6 **a** Autoencoder and **b** denoising autoencoder

- SDA is adopted to pretrain neural networks for each block with unlabeled data.
- For each block of input music documents, the parameters of SDA are optimized via stochastic gradient descent [50].

Both denoising autoencoder (DAE) and stacked denoising autoencoder (SDA) are developed based on autoencoder (AE) [51]. Generally, it consists of two key components—encoder and decoder. Encoder transforms an input X into hidden representation y and decoder maps it back to a reconstructed d-dimensional vector z. Figure 6a illustrates basic idea of AE. In this study, we consider that the hidden layer is encoded by a nonlinear one-layer neural network and the mapping can be $Y = \boldsymbol{\Phi}(X)$. The reconstruction from hidden representation Y can be computed using $Z = \boldsymbol{\Theta}(Y)$. There could be various kinds of distributional assumptions on the input given the code. Various loss functions can be applied to quantify reconstruction errors on the output side. In this study, since we assume the distribution $dist(X|Z)$ is Gaussian, squared error loss L_{sqe} can be used:

$$L_{\text{sqe}}(X, Z) = ||X - Z||^2. \tag{9}$$

In real world, the reconstruction criterion alone may not be able to guarantee the generation of effective representation of raw data. It might easily lead to the undesirable result—"simply copy the input". Thus, DAE is proposed to avoid this phenomenon by taking different strategy—training neural network locally to denoise noisy versions of initial inputs. Part (B) of Fig. 6 visualizes the basic idea of DAE. It is done by first constructing X's corrupted version $\widetilde{X}$ via a stochastic mapping $\widetilde{X} - q_D(\widetilde{X}|X)$. q_D is a function to corrupt X and the corrupted input $\widetilde{X}$ is then mapped to a hidden representation $Y = \boldsymbol{\Phi}_1(X)$, where Y is then used to reconstruct the initial version of X by $Z = \boldsymbol{\Theta}(Y)$. The reconstruction error $L(X, Z)$ instead of $L(\widetilde{X}, Z)$ is minimized in DAE. During the training, each round one training example X is given, a different version of corrupted X is generated based on function q_D.

In our approach, SDA is applied to build deep learning architecture for computing DMS as basic component, one for each music block. We initialize the deep neural network using the same strategy which stacking RBMs in deep belief networks apply. Figure 7 illustrates the procedure to gain multilayer DAE. First, the corrupted input is only used for training each layer at very beginning. This is very important to learn effective features. Right after the mapping function $\boldsymbol{\Phi}$ has been learnt successfully, it can be applied to process uncorrupted inputs. Then to train the neurons in the next layer, corrupted training examples will be used as inputs.

After a set of encoders are trained and stacked as SDAs, outputs from top layer serve as music content representation—DMS and inputs to different indexing structure for effective and efficient music search. In this study, we apply stochastic gradient descent to infer and optimize various parameters of the SDAs due to its good efficiency [50].

4 Experimental configuration

Before presenting experimental results, we first introduce the experimental configuration including the test music datasets, evaluation metrics, query tasks and competitors considered for performance comparison.

4.1 Music testbed

The testbed plays an important role in evaluating content-based music retrieval systems. To facilitate the evaluation, three separate music databases are used. The first one, called Dataset I, is used for testing performance of different methods on genre-based retrieval. It contains 5000 music data items covering ten genres with 500 songs per genre. This

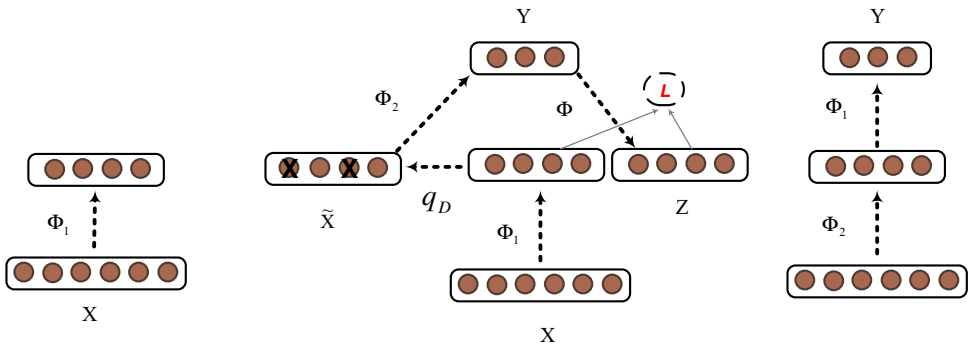

Fig. 7 Stacking denoising autoencoder

dataset is very similar to the test collection used in [40, 41]. To ensure variety of recording quality, the excerpts of this dataset were taken from radio, compact disks, and MP3 compressed audio files. It consists of ten music genre categories: Classical, Country, Dance, Hip-hop, Jazz, Reggae, Metal, Blues and Pop. The second dataset, called Dataset II, is used for evaluating performance of different methods on artist-based query. It contains 7000 songs covering 50 different artists. It includes 25 male singers (such as Van Morrison, Michael Jackson, Elton John) and 25 female singers (such as Kylie Minogue, Madonna, Jennifer Lopez). Thus, there are 140 songs for each singer in Dataset II. Dataset III contains 1000 sounds covering 10 different solo instruments such as piano, guitar and violin, and there are 100 music items for each category. This dataset is developed to test performance of instrument-based similarity search. The music in both Dataset II and Dataset III was collected from the CD collection of the authors and their friends.

4.2 Evaluation metrics and tasks

The efficacy of multimedia retrieval systems can be assessed by different performance metrics. The different kinds of measures can reflect different characteristics of each system. In this study, our goal is to demonstrate the effectiveness of our evaluation methodology under different kinds of measures. Thus, we test the methodology with various evaluation metrics. They include precision measured up to a certain rank ($P@k$) and mean average precision (MAP).

In situations where the results are ranked, $P@k$ is a common measure of precision based on the top-k matches:

$$P@k = \frac{\text{Number of relevant objects in top k}}{k}. \tag{10}$$

MAP is the most frequently used measure of ranked retrieval and can be defined as

$$\text{MAP} = \frac{\sum_{m=1}^{M}(P@m \times rel(m))}{\text{RE}}, \tag{11}$$

where M is the number of objects retrieved, RE is the number of relevant objects, $P@m$ is the precision at cutoff rank m, and $rel(m)$ is a binary function on the relevance of the rank m object. MAP is one of the most popular system-oriented measures, whereas precision measured at cutoff R is typically a user-oriented measure.

Content-based music retrieval can be informally defined as the user submits a query music clip and the system retrieves a list of music pieces from the database that are most similar; the list of "matching" pieces is displayed in order starting from the most similar. However, the meaning of music similarity can be defined over a board range and each notion of similarity corresponds to one kind of query. In this study, we consider the following three different music retrieval tasks:

- Type I: Search music that has similar genre from database constructed using Dataset I.
- Type II: Search music performed by the same artist from database constructed using Dataset II.
- Type III: Search music with the same instrument from database constructed using Dataset III.

4.3 Competitors

To demonstrate different advantages of EMIF, we compare EMIF with the following state-of-the-art:

- EMIF: In this study, a CBMIR system is built based on EMIF and Hybrid tree is selected as multidimensional indexing structure to speed up music search.
- DWCH + hybrid tree (DWCH+HT): Daubechies wavelet histogram technique (DWCH) is used to extract wavelet-based music signatures to describe music content. Similar to EMIF, Hybrid tree is the indexing structure for speeding up search process.
- MARSYAS + hybrid tree (MARSYAS+HT): MARSYAS framework is used to extract the signatures, which linearly combines three different acoustic features—timbral texture, pitch content and rhythm. Similar to EMIF and DWCH+HT, Hybrid tree is the indexing structure for speeding up search process.

All above methods have been implemented and tested on a Intel (R) Core (TM) i5, 2.40 GHz, PC running the Windows 7.0 operating system.

5 An empirical study

This section presents an experimental study to evaluate the proposed method and its competitive schemes. Our results demonstrate the superiority of EMIF against other state-of-the-art approaches over a range of different measures, including accuracy of retrieval, scalability to accommodate different sizes of data and handle update process, improvement on efficiency in terms of the query response time.

5.1 Effectiveness comparison

In the first experiment, we report a comparative study on the retrieval effectiveness of DWCH+HT, MARSYAS+HT, EMIF and EMIF without the decision module.[3] Tables 2, 3

[3] EMIF without decision module is denoted by EMIF-W.

Table 2 Query accuracy comparison of EMIF and other approaches for music retrieval based on Query Type I

Query methods	Query accuracy	
	P@10	MAP
EMIF	0.617	0.511
EMIF-W	0.527	0.452
DWCH+HT	0.372	0.302
MARSYAS+HT	0.297	0.275

Table 3 Query accuracy comparison of EMIF and other approaches for music retrieval based on Query Type II

Query methods	Query accuracy	
	P@10	MAP
EMIF	0.603	0.505
EMIF-W	0.515	0.452
DWCH+HT	0.361	0.292
MARSYAS+HT	0.285	0.266

Table 4 Query accuracy comparison of EMIF and other approaches for music retrieval based on Query Type III

Query methods	Query accuracy	
	P@10	MAP
EMIF	0.725	0.617
EMIF-W	0.605	0.526
DWCH+HT	0.435	0.382
MARSYAS+HT	0.365	0.291

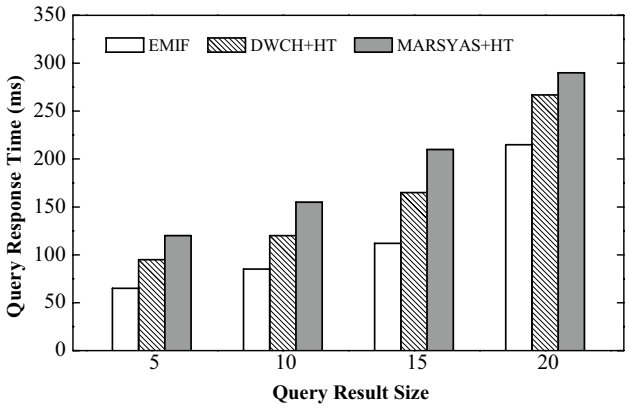

Fig. 8 Query response time comparison of EMIF and other approaches—Query I

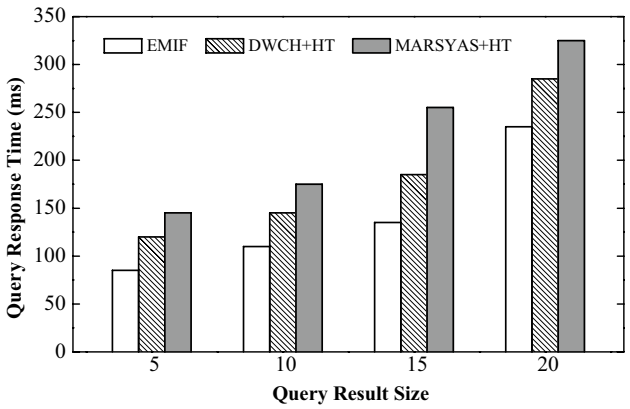

Fig. 9 Query response time comparison of EMIF and other approaches—Query II

and 4 illustrate the query precisions for three different music query types in terms of different measurements. Specifically, in each test, we randomly select query examples from the database and no overlap between query sets and training sets exists. It can be clearly seen that EMIF achieves significant improvement on query accuracies for all cases. In particular, the EMIF method improves the query effectiveness over three query types, on average, 11.2% for $P@10$ and by 13.2% for MAP. These results indicate that EMIF, whose structure integrates classification scheme, deep learning-based music signature generation scheme and multiple high-dimensional access methods into one framework, is more effective than other approaches. This superior effectiveness is due to the multiple layer structure, which contains category statistical profiling model and a likelihood score fusion scheme based on Logistic regression. Furthermore, the access methods supporting query process on data from individual category lead to a compact searching space and faster retrieval.

The decision module with logistic regression score fusion function plays an important role in enhancement of EMIF's performance. To investigate the effects of the decision module, we compare the difference between EMIF with and without decision module via experiments over three different query types. Tables 2, 3 and 4 present relative gains in query accuracy when the decision module is integrated for weight estimation. Integrating the decision module has a strong influence on the retrieval accuracy for all different query cases. We find that the corresponding performance improvement is fairly high (about 17%). The main reason behind this performance gain is that the misclassification can be captured using the weight of scores from different features with logistic-based learning. The misclassification by LDMM-based category models in the first layer of the system is further corrected by the inductive process of LR via an adaboost-like training algorithm. This implies that final classification accuracy can be improved significantly via the performance compensation in the decision module. Experimental results also validate this finding empirically.

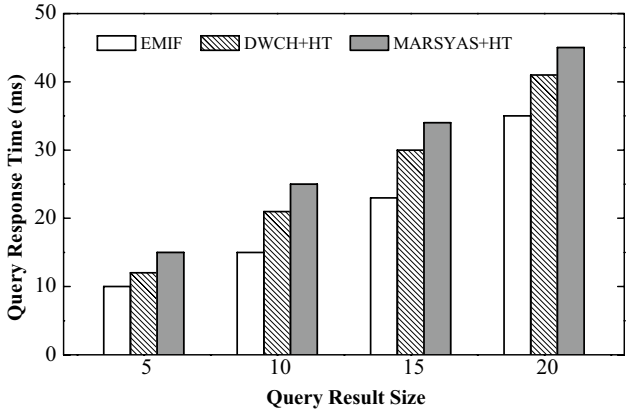

Fig. 10 Query response time comparison of EMIF and other approaches—Query III

Table 5 System reconstruction time comparison of different approaches with different number of classes

Total classes	System reconstruction time (s)		
	EMIF	DWCH+HT	MARSYAS+HT
1	407	206	210
2	409	400	408
3	367	720	705
4	390	900	890
5	398	1200	1250
6	402	1805	1890
7	387	1951	2100
8	364	2345	2580
9	309	2876	2900
10	399	3320	3421

5.2 Efficiency comparison

For large music databases, response time to query is another key indicator for system performance. Although the statistical concept model and decision module in EMIF lift the accuracy significantly, they might introduce extra query cost overhead. In this experiment, we show how it affects the time efficiency. A test was run with 1000 query examples randomly selected from the music datasets. Figures 8, 9 and 10 show the response time of three different queries for different methods with various size of the result sets. From the experimental results summarized in the figures, we can see that EMIF achieves great gain in terms of query speed against the other approaches for all sizes of result set. MARSYAS+HT performs worst among all different approaches tested. EMIF achieves the best response time over different query tasks and compared to other approaches, performance improvement is very significant, at least 14.6%. The main reason behind this is that EMIF's layered structure which facilitates retrieval processing based on index structure from individual class results in a more compact searching space (smaller indexing structure). Consequently, this improves the final query speed significantly. Further, another major advantage of our scheme is its simplicity. All the components in our framework (such as LDA, GMM, single layer neural network and logistic regression) are standard techniques which can be implemented efficiently.

In addition, the proposed EMIF system is very efficient in terms of space cost, and hence it can be applied to larger databases. During the first phase of retrieval process—categorization, we do not need to access the real data in the database, but only the discriminative information in LDMM and decision module. Such information is generated during the construction stage, and will not incur any cost overhead for identification. Those system parameters are only proportional to the number of classes. It is not affected significantly by database size as one class may have thousands of music objects. Thus, comparing to other approaches, potentially, our approach can achieve less search cost for the same size of data. Furthermore, the model can be very adaptive to insertion of new semantic class because scoring module and corresponding access method for each category are independent.

5.3 Scalability comparison

Scalability is particularly important for large music databases, because such systems can potentially contain thousands of audio files for retrieval and the content of the data collections could be updated frequently. In this section, we illustrate the behavior of our scheme under different sizes of data. EMIF is evaluated against other schemes using (1) datasets containing different number of classes and (2) datasets containing different number of music objects. Due to space limitation, we only present the empirical results using Dataset I.

In the first experiment, we compare the reconstruction cost and query accuracy of EMIF and other approaches when different classes of music are gradually inserted into the system. Note that the subset of classes and the order of class insertion is chosen randomly. In Table 5, the number of classes varies from 1 to 10. The results show that compared to other methods, EMIF consumes much less construction time. One thing worth noting is that when the number of classes is less than 2, all other methods use less time to complete construction than EMIF does. This is because besides building indexing tree and music signature generation scheme for music from new class, EMIF's construction cost also includes training time for relative LR analysis. This overhead could make EMIF less efficient in terms of construction cost when the number of classes is small. From Table 5, we also find that there is no significant increase for

Table 6 Query accuracy comparison of different approaches with different number of classes (Query Type I and Dataset I)

Total classes	Query accuracy (P@10)		
	EMIF	DWCH+HT	MARSYAS+HT
1	0.661	0.543	0.537
2	0.654	0.512	0.525
3	0.642	0.493	0.489
4	0.637	0.472	0.476
5	0.632	0.467	0.425
6	0.629	0.445	0.411
7	0.625	0.431	0.386
8	0.620	0.389	0.352
9	0.617	0.378	0.325
10	0.617	0.372	0.297

reconstruction time when the system includes more object classes. The main reason is that with "classify-and-indexing" approach, only one associated indexing structure needs to be built when a new class is integrated into the database. Also, the index's size is much smaller. Likewise, Table 6 illustrates the query accuracy as the number of classes is varied from 1 to 10. EMIF demonstrates much better stability in terms of query accuracy. In contrast, performance of all other methods deteriorates rapidly with the growth of the number of classes.

On the other hand, as the number of stored items increases, the performance of a CBMIR system may degrade due to noise and more similar objects in the database. Thus, we compare the query accuracy and response time of the EMIF system with other approaches using different sizes of data. Our methodology is as follows. First, we randomly pick 1000, 2000, 3000, 4000, and 5000 music. 20% of data are used for training and the rest is used for testing. Then we increase the size of music gradually from 1000, and measure the query response time and the query accuracy. Note that there are two cases for this evaluation as below:

- Case I—static: For the static case, the system is initially trained and tested with 1000 music. Then we increase the dataset to 2000 music, train the system again and evaluate it. This process is repeated until the size of music reaches to 5000 music.
- Case II—incremental: In this setting, the system is trained and evaluated with 1000 music at the first stage. Then, 1000 music is added into the system without rerunning the training process and we carry out the evaluation on the systems again. The process will be repeated until the size of music reaches to 5000 music.

Tables 7 and 8 summarize the results of query accuracy comparison on the above two cases, respectively. Note

Table 7 Query accuracy comparison of different approaches with different number of music—static case

Music size	Query accuracy (P@10)		
	EMIF	DWCH+HT	MARSYAS+HT
1000	0.657	0.531	0.502
2000	0.645	0.506	0.461
3000	0.635	0.488	0.413
4000	0.629	0.419	0.375
5000	0.617	0.372	0.297

Table 8 Query accuracy comparison of different approaches with different number of music—incremental case

Music size	Query accuracy (P@10)		
	EMIF	DWCH+HT	MARSYAS+HT
1000	0.657	0.531	0.502
2000	0.625	0.506	0.461
3000	0.609	0.488	0.413
4000	0.595	0.419	0.375
5000	0.590	0.372	0.297

that since both DWCH+HT and MARSYAS+HT are not learning-based methods, the results for the two cases are the same. As shown, EMIF outperforms its competitors greatly. Comparing EMIF with the other schemes in both contexts, several important observations can be gained. First, EMIF still outperforms the competitors in the static scenario. This makes it a very promising scheme since the static approaches actually optimize over the full datasets. Second, as expected, EMIF in the dynamic context is inferior to that in the static context. However, the degeneration is acceptable.

6 Conclusion

In this article, we introduce a novel approach called EMIF based on the "classify-and-indexing" design principle. To achieve a more scalable indexing framework, an independent LDMM-based profiling model for each music category is constructed using multiple features to generate likelihood score. To address robustness issues, a decision module with logistic regression-based score fusion function has been developed to further improve classification accuracy. Moreover, EMIF's layered architecture results in more compact indexing structure for each category and consequently achieves a significant reduction on query execution time and updating cost. Combination of two schemes further enhances the scalability of the whole system, while providing large-scale music search with effectiveness and efficiency. To

validate the approach, we have carried out comprehensive experiments and the results demonstrate the various advantages of EMIF over the existing state-of-the-art indexing methods.

The current study can be extended in several interesting directions for future investigation: first, at this stage, our method is only tested using music data. It would be interesting to apply the method to data on other application domains (e.g., image and video retrieval) and investigate corresponding experimental results. In addition, developing a framework for estimating cost model of indexing framework construction and maintenance is another promising direction. Last but not least, we plan to design advanced fusion scheme to combine scores.

References

1. Shen, J., Tao, D., Li, X.: Modality mixture projections for semantic video event detection. IEEE Trans. Circuits Syst. Video Tech. **18**(11), 1587–1596 (2008)
2. Gong, Y., Ke, Q., Isard, M., Lazebnik, S.: A multi-view embedding space for modeling internet images, tags, and their semantics. Int. J. Comput. Vision **106**(2), 210–233 (2014)
3. Chen, J., Zhang, H., He, X., Nie, L., Liu, W., Chua, T.: Attentive collaborative filtering: Multimedia recommendation with item- and component-level attention. In: *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7–11, 2017*, pp. 335–344 (2017)
4. He, X., He, Z., Du, X., Chua, T.: Adversarial personalized ranking for recommendation. In: *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pp. 355–364 (2018)
5. He, X., He, Z., Song, J., Liu, Z., Jiang, Y., Chua, T.: NAIS: neural attentive item similarity model for recommendation. IEEE Trans. Knowl. Data Eng. **30**(12), 2354–2366 (2018)
6. He, X., Gao, M., Kan, M., Wang, D.: Birank: Towards ranking on bipartite graphs, *to appear in IEEE Trans. Knowl. Data Eng.* (2017)
7. Murthy, Y .V .S., Koolagudi, S .G.: Content-based music information retrieval (CB-MIR) and its applications toward the music industry: a review. ACM Comput. Surv. **51**(3), 45:1–45:46 (2018)
8. Essid, S., Richard, G.: Fusion of multimodal information in music content analysis. In: *Multimodal Music Processing*, pp. 37–52 (2012)
9. Cheng, Z., Shen, J.: On effective location-aware music recommendation. ACM Trans. Inf. Syst. **34**(2), 13:1–13:32 (2016)
10. Cheng, Z., Shen, J., Hoi, S. C. H.: On effective personalized music retrieval by exploring online user behaviors. In: *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016*, pp. 125–134 (2016)
11. Cheng, Z., Shen, J., Zhu, L., Kankanhalli, M. S., Nie, L.: Exploiting music play sequence for music recommendation. In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pp. 3654–3660 (2017)
12. Schedl, M., Yang, Y., Herrera-Boyer, P.: Introduction to intelligent music systems and applications. ACM TIST **8**(2), 17:1–17:8 (2017)
13. Hsu, J., Zhen, Y., Lin, T., Chiu, Y.: Affective content analysis of music emotion through EEG. Multimedia Syst. **24**(2), 195–210 (2018)
14. Deldjoo, Y., Constantin, M. G., Ionescu, B., Schedl, M., Cremonesi, P.: MMTF-14K: a multifaceted movie trailer feature dataset for recommendation and retrieval. In: *Proceedings of the 9th ACM Multimedia Systems Conference, MMSys 2018, Amsterdam, The Netherlands, June 12-15, 2018*, pp. 450–455 (2018)
15. Shen, J., Shepherd, J., Ngu, A.: Towards effective content-based music retrieval with multiple acoustic feature combination. IEEE Trans. Multimedia **8**(6), 1179–1189 (2006)
16. Shen, J., Tao, D., Li, X.: Quc-tree: integrating query context information for efficient music retrieval. IEEE Trans. Multimedia **11**(2), 313–323 (2009)
17. Yang, C.: Efficient acoustic index for music retrieval with various degrees of similarity. In: *Proc. of ACM MM Conference* (2002)
18. Ryynänen, M., Klapuri, A.: Query by humming of midi and audio using locality sensitive hashing. In: *Proceedings of ICASSP*, pp. 2249–2252 (2008)
19. Yu, Y., Zimmermann, R., Wang, Y., Oria, V.: Scalable content-based music retrieval using chord progression histogram and tree-structure lsh. IEEE Trans. Multimedia **15**(8), 1969–1981 (2013)
20. Böhm, C., Berchtold, S., Keim, D.A.: Searching in high-dimensional spaces: index structures for improving the performance of multimedia databases. ACM Comput. Surv. **33**(3), 322–373 (2001)
21. Rui, Y., Huang, T.S., Ortega, M., Mehrotra, S.: Relevance feedback: a power tool for interactive content-based image retrieval. IEEE Trans. Circuits Syst. Video Techn. **8**(5), 644–655 (1998)
22. Gaede, V., Günther, O.: Multidimensional access methods. ACM Comput. Surv. **30**(2), 170–231 (1998)
23. Santini, S., Jain, R.: Similarity measures. IEEE Trans. Pattern Anal. Mach. Intell. **21**(9), 871–883 (1999)
24. Böhm, C., Berchtold, S., Keim, D.A.: Searching in high-dimensional spaces: index structures for improving the performance of multimedia databases. ACM Comput. Surv. **33**(3), 322–373 (2001)
25. Ciaccia, P., Patella, M., Zezula, P.: M-tree: An efficient access method for similarity search in metric spaces. In: *Proc. of the 23rd VLDB conference (VLDB'97)* (1997)
26. Weber, R., Schek, H. J., Blott, S.: A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In: *Proc. of 24th VLDB Conference (VLDB'98)* (1998)
27. Chakrabarti, K., Mehrotra, S.: The hybrid tree: an index structure for high dimensional feature spaces. In: *Proc. of ICDE Conference (ICDE'99)* (1999)
28. Yu, C., Ooi, B. C., Tan, K. L., Jagadish, H. V.: Indexing the distance: an efficient method to knn processing. In: *Proc. of 27th VLDB Conference (VLDB'01)* (2001)
29. Yu, Y., Crucianu, M., Oria, V., Damiani, E.: Combining multi-probe histogram and order-statistics based lsh for scalable audio content retrieval. In: *Proc. of ACM MM Conference* (2010)
30. Wu, G., Han, J., Guo, Y., Liu, L., Ding, G., Ni, Q., Shao, L.: Unsupervised deep video hashing via balanced code for large-scale video retrieval. *to appear in IEEE Trans. on Image Processing*
31. Guo, Y., Ding, G., Han, J.: Robust quantization for general similarity search. IEEE Trans. Image Process. **27**(2), 949–963 (2018)

32. Wu, G., Han, J., Lin, Z., Ding, G., Zhang, B., Ni, Q.: Joint image-text hashing for fast large-scale cross-media retrieval using self-supervised deep learning. *to appear in IEEE Trans. on Industrial Electronics*
33. Wu, G., Lin, Z., Han, J., Liu, L., Ding, G., Zhang, B., Shen, J.: Unsupervised deep hashing via binary latent factor models for large-scale cross-modal retrieval. In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, pp. 2854–2860 (2018)
34. Berchtold, S., Bohm, C., Kriegel, H., Sander, J., Jagadish, H.: Independent quantization: an index compression technique for high-dimensional data spaces. In: *Proc. of 16th ICDE Conference (ICDE'00)* (2000)
35. McFee, B., Lanckriet, G. R. G.: Large-scale music similarity search with spatial trees. In: *ISMIR* (2011)
36. Rabiner, L., Juang, B.-H.: *Fundamentals of speech recognition* (1993)
37. Nam, U., Berger, J.: Addressing the same but different-different but similar problem in automatic music classification. In: *Proc. of ISMIR* (2001)
38. Li, G., Khokhar, A.A.: Content-based indexing and retrieval of audio data using wavelets. In: *Proceedings of IEEE International Conference on Multimedia and Expo(II)*, pp. 885–888 (2000)
39. Lu, L., Zhang, H., Li, S.Z.: Content-based audio classification and segmentation by using support vector machines. Multimedia Syst. **8**(6), 482–492 (2003)
40. Tzanetakis, G., Cook, P.R.: Musical genre classification of audio signals. IEEE Trans. Speech Audio Process. **10**(5), 293–302 (2002)
41. Li, T., Ogihara, M., Li, Q.: A comparative study on content-based music genre classification. In: *Proc. of ACM SIGIR Conference (SIGIR'03)* (2003)
42. Shen, J., Shepherd, J., Ngu, A.H.H.: Towards effective content-based music retrieval with multiple acoustic feature combination. IEEE Trans. Multimedia **8**(6), 1179–1189 (2006)
43. Song, Y., Zhang, C.: Content-based information fusion for semi-supervised music genre classification. IEEE Trans. Multimedia **10**(1), 145–152 (2008)
44. Tolonen, T., Karjalainen, M.: A computationally efficient multipitch analysis model. IEEE Trans. Speech Audio Process. **8**(6), 708–716 (2000)
45. Dempster, N.L.A., Rubin, D.: Maximum likelihood from incomplete data via the em algorithm. J. R. Stat. Soc. Ser. B **39**(1), 1–22 (1977)
46. Shen, J., Shepherd, J., Cui, B., Tan, K.: A novel framework for efficient automated singer identification in large music databases. ACM Trans. Inf. Syst. **27**(3), 18 (2009)
47. Jordan, M.I.: Why the logistic function? a tutorial discussion on probabilities and neural networks. Massachusetts Institute of Technology, Tech. Rep. 9503 (1995)
48. Hastie, T., Tibshirani, R., Friedman, J.: The elements of statistical learning: data mining, inference, and prediction. Springer, New York (2001)
49. Collins, M., Schapire, R. E., Singer, Y.: Logistic regression, adaboost and bregman distances. In: *Proc. of the 13th Annual Conference on Computational Learning Theory (COLT'00)* (2000)
50. Bottou, L., Bousquet, O.: The tradeoffs of large scale learning. In: *NIPS* (2007)
51. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.-A.: Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. J. Mach. Learn. Res. **11**, 3371–3408 (2010)