TAG INTEGRATED MULTI-LABEL MUSIC STYLE CLASSIFICATION WITH HYPERGRAPH

Fei Wang, Xin Wang, Bo Shao, Tao Li
Florida International University
{feiwang,xwang009,bshao001,taoli}@cs.fiu.edu

Mitsunori Ogihara
University of Miami
ogihara@cs.miami.edu

ABSTRACT

Automatic music style classification is an important, but challenging problem in music information retrieval. It has a number of applications, such as indexing of and searching in musical databases. Traditional music style classification approaches usually assume that each piece of music has a unique style and they make use of the music contents to construct a classifier for classifying each piece into its unique style. However, in reality, a piece may match more than one, even several different styles. Also, in this modern Web 2.0 era, it is easy to get a hold of additional, indirect information (e.g., music tags) about music. This paper proposes a multi-label music style classification approach, called *Hypergraph integrated Support Vector Machine* (*HiSVM*), which can integrate both music contents and music tags for automatic music style classification. Experimental results based on a real world data set are presented to demonstrate the effectiveness of the method.

1. INTRODUCTION

Music styles (e.g., Dance, Urban, Pop, and Country) are one of the top-level descriptions of music content. Consequently, automatic *Music Style Classification* (*MSC* for short) is a key step for modern music information retrieval systems [7]. There has already been some work toward automatic music style classification. For example, Qin and Ma [10] introduce an MSC system that takes MIDI as data source and mines frequent patterns of different music. Zhang and Zhou [18] present a study on music classification using short-time analysis along with data mining techniques to distinguish among five music styles. Zhou *et al.* [19] propose a Bayesian inference based decision tree model to classify the music into pleasurable and sorrowful music. Although these methods are highly successful, two major limitations exist.

- These are single-label methods in that they can assign only one style label, but many pieces of music may map to more than one style.
- They only make use of the music content information. However, with the rapid development of web technologies, we can easily obtain much richer information of the music (e.g., tags, lyrics, and user comments). How to incorporate these pieces of information into the MSC process effectively is a problem worthy of researching.

In this paper, we propose a multi-label MSC method that can integrate three types of information: (1) audio signals (MFCC coefficients, STFT, DWCH); (2) music style correlations; (3) music tag information and correlations. Specifically, we construct two hyper-graphs, one on music style labels and the other on music tags, where the vertices on the hypergraphs correspond to the data points, and the hyperedges correspond to the music styles and the tags, respectively. We first integrate those two hypergraphs to obtain a unified hypergraph. Next, assuming that similar music tends to have similar style labels on the hypergraph, we propose a new, SVM-like multilabel ranking algorithm. The algorithm uses a hypergraph Laplacian regularizer and can be efficiently solved by the dual coordinate descent method. Finally, we propose a predictor of the size of label set to determine the number of labels assigned to for each piece of music independently. To demonstrate the efficiency and effectiveness of our proposed method, we conducted a set of experiments applying the method to a real world data.

The rest of this paper is organized as follows. In Section 2 we briefly introduce preliminaries on our key concept, the hypergraph. In Section 3 we describe our HiSVM algorithm. We describe in Section 4 the audio features extracted from the data set as well as the style and tag information of the data set. We present the results of experiments in Section 5 and conclude the paper in Section 6.

2. PRELIMINARIES

A *hypergraph* is a generalization of a graph, in which edges, called hyperedges, may connect any positive number of vertices [1, 11]. Formally, a hypergraph $\mathcal{G}$ is a pair $(\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is a set of *vertices* and $\mathcal{E} \subseteq 2^{\mathcal{V}} - \emptyset$ is a set of *hyperedges*. An *edge-weighted hypergraph* is one in which each hyperedge is assigned a weight. We use $w(e)$ to denote the weight given to e. The *degree* of a hyperedge e, denoted as $\delta(e)$, is the number of vertices in e. For a

THE WORK IS PARTIALLY SUPPORTED BY NSF GRANTS IIS-0546280, DMS-0844513, AND CCF-0830659.

standard graph (sometimes called a "2-graph") the value of δ is 2 for all edges. The degree $d(v)$ of a vertex v is $d(v) = \sum_{v \in e, e \in \mathcal{E}} w(e)$. The *vertex-edge incidence matrix* $\mathbf{H} \in \mathbb{R}^{|V| \times |E|}$ is defined as: $h(v, e) = 1$ if $v \in e$ and 0 otherwise. We thus have

$$d(v) = \sum\nolimits_{e \in \mathcal{E}} w(e) h(v, e) \tag{1}$$

$$\delta(e) = \sum\nolimits_{v \in \mathcal{V}} h(v, e). \tag{2}$$

Let $\mathbf{D}_e$ (respectively, $\mathbf{D}_v$ and $\mathbf{W}$) be the diagonal matrix whose diagonal entries are $d(v)$ (respectively, $\delta(e)$, and $w(e)$).

The *graph Laplacian* is the discrete analog of the Laplace-Beltrami operator on compact Riemannian manifolds [12]. The *graph Laplacian* has been widely used in unsupervised learning (e.g., spectral clustering [9]) and semi-supervised learning (e.g. [16, 20]). Below we will sketch a commonly used algorithm by Chung [3], called the *Clique Expansion Algorithm*, for constructing the hypergraph Laplacian.

The Clique Expansion Algorithm constructs a traditional 2-graph $\mathcal{G}_c = (\mathcal{V}_c, \mathcal{E}_c)$ from the original hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and views the Laplacian of $\mathcal{G}_c$ to be the Laplacian of $\mathcal{G}$. Suppose $\mathcal{V}_c = \mathcal{V}$ and $\mathcal{E}_c = \{(u, v) | u, v \in e, e \in \mathcal{E}\}$. The edge weight $w_c(u, v)$ of $\mathcal{G}_c$ is defined by

$$w_c(u, v) = \sum\nolimits_{u, v \in e, e \in E} w(e) \tag{3}$$

An interpretation of this definition is that the edge weight matrix, $\mathbf{W}_c$, of $\mathcal{G}_c$ can be expressed as

$$\mathbf{W}_c = \mathbf{H}\mathbf{W}\mathbf{H}^T \tag{4}$$

Let $\mathbf{D}_c$ be the diagonal matrix such that

$$\mathbf{D}_c(u, u) = \sum_v w_c(u, v).$$

Then the combinatorial Laplacian, $\mathbf{L}_c$, of $\mathcal{G}_c$ is given by

$$\mathbf{L}_c = \mathbf{D}_c - \mathbf{W}_c = \mathbf{D}_c - \mathbf{H}\mathbf{W}\mathbf{H}^T \tag{5}$$

and the normalized Laplacian, $\mathbf{L}_n$, is given by

$$\mathbf{L}_n = \mathbf{I} - \mathbf{D}_c^{-1/2} \mathbf{H}\mathbf{W}\mathbf{H}^T \mathbf{D}_c^{-1/2}. \tag{6}$$

From Eq. (5) and (6), we have

$$\mathbf{L}_n = \mathbf{D}_c^{-1/2} \mathbf{L}_c \mathbf{D}_c^{-1/2}. \tag{7}$$

In our music style classification, we construct two hypergraphs: the style hypergraph $\mathcal{G}_s$ and the tag hypergraph $\mathcal{G}_t$. The vertices of $\mathcal{G}_s$ and $\mathcal{G}_t$ are simply the data points. The hyperedges of $\mathcal{G}_s$ correspond to the style labels, i.e., each hyperedge in $\mathcal{G}_s$ contains all the data points that belong to a specific style category. Similarly, each hyperedge of $\mathcal{G}_t$ contains all the data points that own the corresponding tag. Figure 1 shows an intuitive example on the music style and tag hypergraphs.

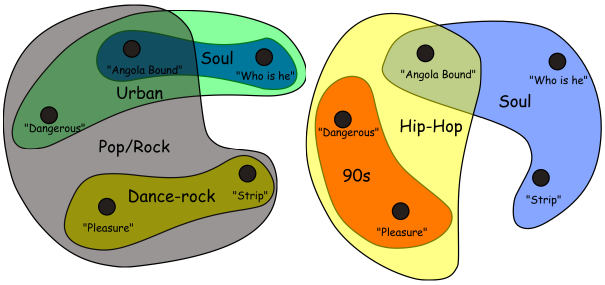

Figure 1. An example of the music style (left) and tag (right) hypergraph. The nodes on the hypergraphs correspond to the music "Angola Bond", "Who is he", "Dangerous", "Pleasure", and "Strip". The regions of different colors correspond to the different hyperedges. The hyperedges correspond to music styles in the left panel and to music tags in the right panel.

3. MULTI-LABEL LEARNING WITH HYPERGRAPH REGULARIZATIONS

In this section we will present in detail our proposed multi-label classification algorithm with hypergraph regularization. Suppose there are n training samples $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, where each instance $\boldsymbol{x}_i$ is drawn from some domain $\mathcal{X} \subseteq \mathbb{R}^m$ and its label y_i is a subset of the output label set $\mathcal{Y} = \{1, \cdots, k\}$. For example, if x_i belongs to categories 1, 3, and 4, then $y_i = \{1, 3, 4\}$. We use $\mathbf{X} = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n)^T$ to represent the data feature matrix.

Our basic strategy is to solve the multi-label learning by combing a label ranking problem and a label number prediction problem. That is, for each instance we produce a ranked list of all possible labels, estimate the number of labels for the instance, and then select the predicted number of labels from the list.

Label ranking is the task of inferring a total order over a predefined set of labels for each given instance [5]. Generally, for each category, we define a linear function $f_i(x) = \langle \boldsymbol{w}_i, \boldsymbol{x} \rangle + b_i \quad (i = 1, \cdots, k)$, where $\langle \cdot, \cdot \rangle$ is the inner product and b_i is a bias term. One often deals with the bias term by appending to each instance an additional dimension

$$\boldsymbol{x}^T \leftarrow [\boldsymbol{x}^T, 1], \quad \boldsymbol{w}_i^T \leftarrow [\boldsymbol{w}_i^T, b_i] \tag{8}$$

then the linear function becomes $f_i(\boldsymbol{x}) = \langle \boldsymbol{w}_i, \boldsymbol{x} \rangle$. The goal of label ranking is to order $\{f_i(\boldsymbol{x}), i = 1, \cdots, k\}$ for each instance $\boldsymbol{x}$ according to some predefined empirical loss function and complexity measures. Elisseeff and Weston [6] apply the large margin idea to multi-label learning and present an SVM-like ranking system, called Rank-SVM, given as follows:

$$\begin{aligned} \min \quad & \frac{1}{2} \sum_{i=1}^{k} \|\boldsymbol{w}_i\|^2 + C \sum_{i=1}^{n} \frac{1}{|y_i||\overline{y}_i|} \sum_{(p,q) \in y_i \times \overline{y}_i} \xi_{ipq} \\ \text{s.t.} \quad & \langle \boldsymbol{w}_p - \boldsymbol{w}_q, \boldsymbol{x}_i \rangle \geq 1 - \xi_{ipq}, (p, q) \in y_i \times \overline{y}_i \\ & \xi_{ipq} \geq 0 \end{aligned} \tag{9}$$

where $C \geq 0$ is a penalty coefficient that trades off the empirical loss and model complexity, $\overline{y}_i$ is the complementary set of y_i in $\mathcal{Y}$, $|y_i|$ is the cardinality of the set y_i, i.e., the number of elements of the set y_i, and $\xi_{ipq}(i = 1, \cdots, n; (p,q) \in y_i \times \overline{y}_i)$ are slack variables. The margin term $\sum_{i=1}^{k} \|\boldsymbol{w}_i\|^2$ controls the model complexity and improves the model generalization performance. Although this approach performs better than Binary-SVM in many cases, it still does not model the category correlations clearly. Next, we will describe how to construct a hypergraph to exploit the category correlations and how to incorporate the hypergraph regularization into the problem in the form of Eq. (9).

3.1 Basic Framework

To model the correlations among different categories effectively, a hypergraph is built where each vertex corresponds to one training instance and a hyperedge is constructed for each category which includes all the training instances relevant to the same category. Here, we apply the Clique Expansion algorithm to construct the similarity matrix of the hypergraph. It means that the similarity of two instances is proportional to the sum of the weights of their common categories, thereby captures the higher order relations among different categories. This kind of hypergraph structure was used in the feature extraction by spectral learning [13]. However, we consider how to apply the relation information encoded in the hypergraph to directly design the multi-label learning model. Intuitively, two instances tend to have a large overlap in their assigned categories if they share high similarity in the hypergraph. Formally, this smoothness assumption can be expressed using the hypergraph Laplacian regularizer, trace($\widehat{\mathbf{F}}^T \mathbf{L} \widehat{\mathbf{F}}$). Therefore we can introduce the smoothness assumption into problem Eq. (9) and obtain

$$\begin{aligned} \min \quad & \frac{1}{2} \sum_{i=1}^{k} \|\boldsymbol{w}_i\|^2 + \frac{1}{2} \lambda \mathrm{trace}(\widehat{\mathbf{F}}^T \mathbf{L} \widehat{\mathbf{F}}) + \\ & C \sum_{i=1}^{n} \frac{1}{|y_i||\overline{y}_i|} \sum_{(p,q) \in y_i \times \overline{y}_i} \xi_{ipq} \\ \text{s.t.} \quad & \langle \boldsymbol{w}_p - \boldsymbol{w}_q, \boldsymbol{x}_i \rangle \geq 1 - \xi_{ipq}, (p,q) \in y_i \times \overline{y}_i \\ & \xi_{ipq} \geq 0 \end{aligned} \tag{10}$$

Here $\widehat{\mathbf{F}}$ is the matrix of label prediction; that is, it is the $n \times k$ matrix $(f_j(\boldsymbol{x}_i))$, $1 \leq i \leq n$, $1 \leq j \leq k$. Also, $\mathbf{L}$ is the $n \times n$ hypergraph Laplacian and $\lambda \geqslant 0$ is a constant that controls the model complexity in the intrinsic geometry of input distribution.

3.2 Optimization Strategy

Problem (10) is a linearly constrained quadratic convex optimization problem. To solve it, we first introduce a dual set of variables, one for each constraint, i.e., $\alpha_{ipq} \geq 0$ for $\langle \boldsymbol{w}_p - \boldsymbol{w}_q, \boldsymbol{x}_i \rangle - 1 + \xi_{ipq} \geq 0$ and η_{ipq} for $\xi_{ipq} \geq 0$. After some linear algebraic derivation, we obtain the dual of Problem (10) as

$$\begin{aligned} \min g(\boldsymbol{\alpha}) \quad = \quad & \frac{1}{2} \sum_{p=1}^{k} \sum_{h,i=1}^{n} \beta_{ph} \beta_{pi} \boldsymbol{x}_h^T (I + \lambda X^T L X)^{-1} \boldsymbol{x}_i \\ & - \sum_{i=1}^{n} \sum_{(p,q) \in y_i \times \overline{y}_i} \alpha_{ipq} \\ \text{s.t.} \quad & 0 \leq \alpha_{ipq} \leq \frac{C}{|y_i||\overline{y}_i|} \end{aligned} \tag{11}$$

where $\boldsymbol{\alpha}$ denotes the set of dual variables α_{ipq} and I is the $(m+1) \times (m+1)$ identity matrix.

Once the variables α_{ipq} that minimize $g(\boldsymbol{\alpha})$ are obtained, we can compute w_p by

$$\boldsymbol{w}_p = (I + \lambda X^T L X)^{-1} \sum_{i=1}^{n} \beta_{pi} \boldsymbol{x}_i \tag{12}$$

where

$$\beta_{pi} = \sum_{(j,q) \in y_i \times \overline{y}_i} t_{ijq}^{p} \alpha_{ijq} \tag{13}$$

$$t_{ijq}^{p} = \begin{cases} 1 & j = p \\ -1 & q = p \\ 0 & \text{if } j \neq p \text{ and } q \neq p \end{cases} \tag{14}$$

Compared to the primal optimization problem, the dual has $k(m+1)$ less variables and includes simple box constraints. The dual can be solved by the dual coordinate descent algorithm shown in Algorithm 1.

Algorithm 1 A dual coordinate descent method for HiSVM

```
Start with $\boldsymbol{\alpha} = \mathbf{0} \in \mathbb{R}^{n_\alpha}$ ($n_\alpha = \sum_{i=1}^{n} |y_i||\overline{y}_i|$), and the corresponding $\boldsymbol{w}_i = \mathbf{0}$ $(i = 1, \cdots, k)$
while 1 do
    for $i = 1, \cdots, n$ and $(j,q) \in y_i \times \overline{y}_i$ do
        1. $G = (\boldsymbol{w}_p - \boldsymbol{w}_q)^T \boldsymbol{x}_i - 1$
        2. $PG = \begin{cases} G & \text{if } 0 < \alpha_{ipq} < \frac{C}{|y_i||\overline{y}_i|} \\ \min(0,G) & \text{if } \alpha_{ipq} = 0 \\ \max(0,G) & \text{if } \alpha_{ipq} = \frac{C}{|y_i||\overline{y}_i|} \end{cases}$
        3. If $|PG| \neq 0$,
            $\alpha_{ipq}^* \leftarrow \min\left(\max\left(\alpha_{ipq} - \frac{G}{2A_{ii}}, 0\right), \frac{C}{|y_i||\overline{y}_i|}\right)$
            $\boldsymbol{w}_p \leftarrow \boldsymbol{w}_p + (\alpha_{ipq}^* - \alpha_{ipq})(I + \lambda X^T L X)^{-1} \boldsymbol{x}_i$
            $\boldsymbol{w}_q \leftarrow \boldsymbol{w}_q - (\alpha_{ipq}^* - \alpha_{ipq})(I + \lambda X^T L X)^{-1} \boldsymbol{x}_i$
    end for
    if $\|\boldsymbol{\alpha}^* - \boldsymbol{\alpha}\| / \|\boldsymbol{\alpha}\| < \epsilon$(e.g. $\epsilon = 0.01$) then
        Break
    end if
    $\boldsymbol{\alpha} = \boldsymbol{\alpha}^*$
end while
```

3.3 Predicting the Size of Label Set

So far we have only provided a label ranking algorithm. To identify the final labels of data, we need to design an appropriate threshold for each instance to determine the size of its corresponding label set. Here, we adopt the strategy proposed by Elisseeff and Weston [6], which treats threshold designing as a supervised learning problem. More concretely, for each instance $\boldsymbol{x}$, define a threshold function $h(\boldsymbol{x})$ and the size of label set $s(\boldsymbol{x}) = \|\{j \mid f_j(\boldsymbol{x}) > h(\boldsymbol{x}), j = 1, \cdots, k\}\|$. Our goal is to obtain $h(\boldsymbol{x})$ through a supervised learning method. For the training data $\boldsymbol{x}_i$, its label ranking values, $f_1(\boldsymbol{x}_i), \cdots, f_k(\boldsymbol{x}_i)$, can be given by the foregoing ranking algorithm, and its corresponding threshold $h(\boldsymbol{x}_i)$ is simply defined by

$$h(\boldsymbol{x}_i) = \frac{1}{2}(\min_{j \in y_i}\{f_j(\boldsymbol{x}_i)\} + \max_{j \in \overline{y}_i}\{f_j(\boldsymbol{x}_i)\})$$

Once the training data $(\boldsymbol{x}_1, h(\boldsymbol{x}_1)), \cdots, (\boldsymbol{x}_u, h(\boldsymbol{x}_u))$ are generated, we can use off-the-shelf learning methods to learn $h(\boldsymbol{x})$. In this paper, Linear Support Vector Regression [15] has been adopted to solve $h(\boldsymbol{x})$. We note there that all the label ranking based algorithms toward multi-label learning can use this postprocessing approach to predict the size of label set.

4. DATA DESCRIPTION

For experimental purpose, we created a data set consisting of 403 artists. For each artist, we include a representative song and also obtain the style and tag description.

4.1 Music Content Features

For each song, a single vector of 80 components is extracted. The single vector contains the following audio features:

1) Mel-Frequency Cepstral Coefficients (MFCC): Mel-Frequency Cepstral Coefficients (MFCC) is a feature set that is very popular in speech processing. MFCC is designed to capture short-term spectral based features. The features of MFCC are computed as follows: First, for each frame, the logarithm of the amplitude spectrum based on short-term Fourier transform is calculated, where the frequencies are divided into thirteen bins using the Mel-frequency scaling. Next, this vector is decorrelated using discrete cosine transform. The resulting vector is called the MFCC vector. In our experiments, we compute the mean and variance of each bin over the frame for the two vectors (before and after decorrelation). Thus, for each sample, MFCC occupies 52 components.

2) Short-Term Fourier Transform Features (STFT): This is a set of features related to timbral textures and is not captured using MFCC. It consists of the following types of features: Spectral Centroid, Spectral Rolloff, Spectral Flux, Zero Crossings, and Low Energy. More detailed descriptions of STFT can be found in [14]. In our experiments, we compute the mean for all types and the variance for all but zero crossings. STFT thus occupies 12 components.

3) Daubechies Wavelet Coefficient Histograms (DWCH): Daubechies wavelet filters are a set of filters that are popular in image retrieval (see [4]). The Daubechies Wavelet Coefficient Histograms, proposed in [8], are features extracted in the following manner: First, the Daubechies-8 (db8) filter with seven levels of decomposition (or subbands) is applied to 30 seconds of monaural audio signals. Then, the histogram of the wavelet coefficients is computed for each subband. Then the first three moments of each histogram, i.e., the average, the variance, and the skewness, are calculated from each subband. In addition, the subband energy, defined as the mean of the absolute value of the coefficients, is computed from each subband. More details of DWCH can be found in [8].

4.2 Music Tag Information

Music tags are descriptions given by visitors or music tag editors from the website to express their idea on the music artists. Tags can be as simple as a word or as complicated as a whole sentence. Popular tags are terms like “rock,” “black metal,” and “indie pop.” Long tags are like “I love you baby can I have some more.” The tags are not as formal as style description created by music experts, but they give us ideas of how large population music listeners think about the music artists. In our experiments, tag data was collected from the popular music website http://www.last.fm. In order to understand how important a tag is, and how accurately it reflects the characteristics of an artist, the frequencies of all the tags to describe the artists (tag counts) were also used in the experiments.

A total of 8,529 tags were collected. Each artist has at most 100 tags and at least 3 tags. On average, each artist is associated with 89.5 tags. Note that, each artist may be described by some tags for more than once, for example, Michael Jackson has been tagged with “80s” for 453 times.

4.3 Music Style Information

Style data were collected from All Music Guide (http://www.allmusic.com). These data are created by music experts to describe the characteristics of music artists. Style terms are nouns like Rock & Roll, Greek Folk, and Chinese Pop as well as adjectives like Joyous, Energetic, and New Romantic. Styles for each artist/track are different from the music tags described in the above, since each style name for one artist appears only once.

A total of 358 styles were found. Each artist has at most 12 and at least one style type. On average, every artist is associated with 4.7 style labels.

5. EXPERIMENTS

We performed experiments on HiSVM and four real-world multi-label learning models arising in text categorization, image classification, video indexing and gene function prediction. Comparisons are made with Binary-SVM and Rank-SVM [6].

5.1 Methods and Experimental Setup

The data set information we used to evaluate our proposed approach has been introduced in the previous section, where we use 70% of the data for training (282 pieces total), and the remaining 30% for testing (121 pieces total). Here, the four models used for multi-label learning are as follows:

- Binary-SVM. In this model, first, for each category, train a linear SVM classifier independently. Then, the labels for each test instance can be obtained by aggregating the classification results from all the binary classifiers. Here, we use LIBSVM [2] to train the linear SVM classifiers.

- Rank-SVM [6]. In this model, first, using Eq. (9), we implement Algorithm 1 ($\lambda = 0$) to train a linear label ranking system. We then apply the prediction method for the size of label set described in Section 3.3 to design the threshold model. Finally, for each test instance, we combine the label ranking and threshold models, thereby infer its labels.

- HiSVM. This is our proposed algorithm. The algorithm is composed of three steps: (1) we implement Algorithm 1 to achieve a linear label ranking system; (2) we apply the method in Section 3.3 to design the threshold model; (3) for each test instance, we combine the label ranking and threshold models to infer its labels.

- HSVM. HSVM is the style Hypergraph regularized SVM method, which is the same as the HiSVM method except that it only makes use of the style hypergraph and does not use the tag hypergraph.

- GSVM. GSVM is similar to HiSVM except we construct a traditional 2-graph where each vertex represents one training instance in GSVM rather than a hypergraph. In order to compute the Laplacian, the weight w_{ij} of the edge between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is defined as follows

  $$w_{ij} = \exp(-\rho \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2) \qquad (15)$$

  where ρ is a nonnegative constant. Apparently, the category correlation information is not used during the construction of 2-graph in GSVM.

Some details of HiSVM are in order. We use Eq. (5) to construct both the style hypergraph Laplacian L_s and the tag hypergraph Laplacian L_t, where the weight $w(e)$ of the hyperedge is calculated by

$$w(e) = \exp(-\nu \bar{d}_e) \qquad (16)$$

Here ν is a nonnegative constant, and $\bar{d}_e$ is the average intra-class distance (N.B. Each hyperedge corresponds to one specific style or tag):

$$\bar{d}_e = \frac{\sum_{u,v \in e} \|\boldsymbol{x}_u - \boldsymbol{x}_v\|^2}{\delta(e)(\delta(e) - 1)} \qquad (17)$$

The smaller the average intra-class distance, the larger the corresponding hyperedge weight. Finally we combine L_s and L_t to obtain a unified hypergraph Laplacian L by

$$L = \frac{1}{2}(L_s + L_t)$$

which is used in the rest of the inferences and experiments.

Table 1. A contingency table

	YES is correct	NO is correct
Assigned YES	a	b
Assigned NO	c	d

In the above four models, it is necessary to identify the best value of model parameters such as C, λ and ν on the training data. Here, the grid search method with 5-fold cross validation is used to determine the best parameter values. For the penalty coefficient C in the Linear SVM, we tune it from the grid points $\{10^{-6}, 10^{-5}, \cdots, 10^0, 10^1, \cdots, 10^6\}$; for the tradeoff parameter λ, we tune it from the grid points $\{10^{-6},\ 10^{-5}, \cdots, 10^0, 10^1, \cdots, 10^6\}$; the scale parameter ν and ρ are tuned from the grid points $\{2^{-6}, 2^{-5}, \cdots, 2^0, 2^1, \cdots, 2^6\}$.

5.2 Evaluation Metrics

We choose two measures, F_1 *Macro* and F_1 *Micro* [17], as the evaluation metrics for multi-label learning. Suppose there are a total of S style categories. Then for each category, we can construct a contingency table as follows: Let a (respectively, b) be the number of pieces that are correctly assigned (respectively, not correctly assigned) to this style category, and let c (respectively, d) be the number of pieces that are incorrectly rejected (respectively, correctly rejected) by this style category (see Table 1). Let $r = a/(a + c)$ and $p = a/(a + b)$, where the former is called the recall and the latter the precision. Then the F_1 score of this style category can be computed as

$$F_1 = \frac{2pr}{p + r} \qquad (18)$$

The F_1 *Macro* can be computed by first calculating the F_1 scores for the per-category contingency tables and then averaging these scores to compute the global means. F_1 *Micro* can be computed by first constructing a global contingency table, each of whose cell value is the sum of the corresponding cells in the per-category contingency tables, and then use this global contingency table to compute the Micro F_1 score.

5.3 Experimental Results

Table 2 illustrates the experimental results on our HiSVM algorithm along with the four other methods on the data set. The values in Table 2 are the F_1 Micro values and F_1 Macro values averaged over 50 independent runs together with their standard deviations. From the table we can clearly observe the following:

- Multi-label methods perform better than the simple Binary-SVM method.

- The consideration of label correlations is helpful for the final algorithm performance.

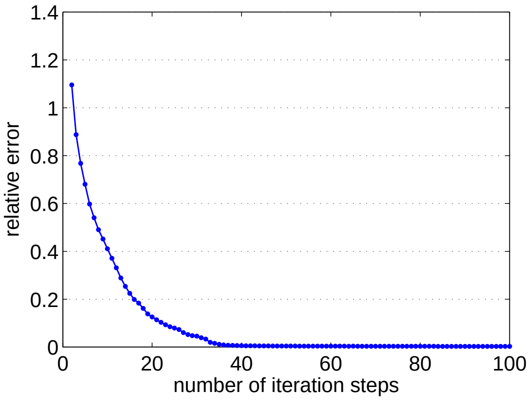

Figure 2. The relative error $\|\boldsymbol{\alpha}^* - \boldsymbol{\alpha}\|/\|\boldsymbol{\alpha}\|$ vs. iteration step plot of our proposed dual coordinate descent algorithm for solving HiSVM.

Table 2. Performance comparisons of four models on the Last.fm dataset

Methods	F1 *Macro*	F1 *Micro*
Binary-SVM	0.4231 ± 0.0025	0.4317 ± 0.0103
Rank-SVM	0.4526 ± 0.0114	0.4733 ± 0.0036
GSVM	0.5018 ± 0.0054	0.5244 ± 0.0103
HSVM	0.5365 ± 0.0120	0.5509 ± 0.0072
HiSVM	$\mathbf{0.5613 \pm 0.0069}$	$\mathbf{0.5802 \pm 0.0116}$

- Hypergraph regularization is better than flat two-graph regularization because it can incorporate the high-order label relationships naturally.

- The incorporation of tag information is helpful for the final classification performance.

Figure 2 shows how the relative error $\|\boldsymbol{\alpha}^* - \boldsymbol{\alpha}\|/\|\boldsymbol{\alpha}\|$ varies with the process of iteration using the dual coordinate descent method introduced in Algorithm 1. From the figure we clearly see that with the process of coordinate descent, the relative error will decrease and it takes approximately 30 steps to converge. This validates the correctness of our algorithm experimentally.

6. CONCLUSION

We propose a novel multi-label classification method called Hypergraph integrated SVM (HiSVM) for music style classification. Our method can not only take into account the music style correlations, but also the music tag correlations. We also propose an efficient dual coordinate descent algorithm to solve it, and finally experimental results on a real world data set are presented to show the effectiveness and correctness of our algorithm.

7. REFERENCES

[1] C. Berge. Graphs and Hypergraphs. Elsevier, 1973.

[2] C.-C. Chang and C.-J. Lin, LIBSVM: a library for support vector machines, 2001, software available at `http://www.csie.ntu.edu.tw/cjlin/libsvm`.

[3] F. R. K. Chung. The laplacian of a hypergraph. In Expanding Graphs, DIMACS Series, Vol. 10, AMS, 1993.

[4] I. Daubechies. Ten lectures on wavelets. SIAM, 1992.

[5] O. Dekel, C. D. Manning, and Y. Singer. Log-linear models for label ranking. In Proc. of NIPS, 2003.

[6] A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. In Proc. of NIPS, 2001.

[7] T. Li and M. Ogihara. Towards intelligent music information retrieval. IEEE Transactions on Multimedia 8(3):564-575, 2006.

[8] T. Li, M. Ogihara, and Q. Li. A comparative study on content-based music genre classification. In Proceedings of SIGIR, pages 282-289, 2003.

[9] U. von Luxburg. A tutorial on spectral clustering. Max Planck Institute for Biological Cybernetics, Tech. Rep., 2006.

[10] D. Qin and G. Z. Ma. Music style identification system based on mining technology. Computer Engineering and Design. 26, 3094-3096. 2005.

[11] S. Chen, F. Wang and C. Zhang: Simultaneous heterogeneous data clustering based on higher order relationships. ICDM Workshops 2007: 387-392.

[12] S. Rosenberg. The Laplacian on a Remannian manifold. London Math. Soc., 1997.

[13] L. Sun, S. Ji, and J. Ye. Hypergraph spectral learning for multilabel classification. In Proc. KDD, 2008.

[14] G. Tzanetakis and P. Cook. Music genre classification of audio signals. IEEE Transactions on Speech and Audio Processing, 10(5):293-302, 2002.

[15] V. Vapnik. The Nature of Statistical Learning Theory. Springer, 1995.

[16] F. Wang and C. Zhang. Label Propagation Through Linear Neighborhoods. In Proc. ICML, pages 985-992, 2006.

[17] Y. Yang. An evaluation of statistical approaches to text categorization. Information Retrieval 1:69–90, 1999.

[18] Y. B. Zhang and J. Zhou. A study on content-based music classification. In Proc. IEEE Signal Processing and Its Applications, 2003.

[19] Y. Zhou, T. Zhang, and J. Sun. Music style classification with a novel Bayesian model. In Proc. Advanced Data Mining and Appliations, Springer, 2006.

[20] X. Zhu. Semi-supervised learning literature survey. University of Wisconsin-Madision, Technical Report TR 1530, 2006.