International Journal of Computational Intelligence Systems, Vol. 5, No. 4 (August, 2012), 668-678

Embedded Feature Selection for Multi-label Classification of Music Emotions

Mingyu You, Jiaming Liu, Guo-Zheng Li*, Yan Chen
The MOE Key Laboratory of Embedded System and Service Computing
Department of Control Science and Engineering, Tongji University,
Shanghai, 201804, China,
**Corresponding author E-mail: gzli@tongji.edu.cn*

Received 12 April 2011
Accepted 16 May 2012

Abstract

When detecting of emotions from music, many features are extracted from the original music data. However, there are redundant or irrelevant features, which will reduce the performance of classification models. Considering the feature problems, we propose an embedded feature selection method, called Multi-label Embedded Feature Selection (MEFS), to improve classification performance by selecting features. MEFS embeds classifier and considers the label correlation. Other three representative multi-label feature selection methods, known as *LP-Chi, max* and *avg,* together with four multi-label classification algorithms, is included for performance comparison. Experimental results show that the performance of our MEFS algorithm is superior to those filter methods in the music emotion dataset.

Keywords: Embedded feature selection, Multi-label learning, Music emotion

1. Introduction

In daily life, music plays an important role. It influences people emotional by nature, makes people feel happy or sad, angry or relaxed. In the past, the problem of automatically categorizing music into emotions was modeled as single-label classification[1,2] or regression[3]. However, as we experience in our daily life, more than one emotion may be evoked by music simultaneously. In this case, classification and regression with single-label can hardly model the multiplicity in music emotion studies. Thus, multi-label approaches are more appropriate in modeling music emotions[4,5].

Besides music emotion classification, various applications, like text categorization, video annotation, clinic diagnosis, etc., all relate to multi-label learning problems[6]. The goal of music emotion tagging is to correctly predict which emotion tags should be associated with a song. Multi-label problem attracts the attention of scholars all over the world. Previous works provide different algorithms solving the multi-label problem[7].

These algorithms are grouped into two categories, problem transformation methods and algorithm adaptation methods. One of the most famous problem transformation algorithms, known as binary relevance, learns a binary classifier for each class independently, and then predicts each of the labels separately. Another well-known problem transformation method is label power set transformation. This method takes each unique combination of labels that exists in a multi-label training set as one single-label multi-value classification task. Other representative problem transformation methods include random k-labelsets (RAkEL)[8], ECC[9], and LEAD[10]. As algorithm adaptation, Rank-SVM[11] trains a collection of SVMs, minimizing the ranking loss, a multi-label evaluation criterion. Other adaptation methods contain ML-KNN[12], BPML[13], Adaboost.MH[14], etc.

The curse of dimensionality still exists in multi-label learning as well as in single-label task. Feature extraction and feature selection are usually employed to solve the dimensionality curse problem.

Many scholars tend to use feature extraction to solve the curse of dimensionality in multi-label tasks. Besides unsupervised feature extraction methods, like PCA[15], many multi-label feature extraction methods are proposed, such as MDDM[16], LSI[17] and LDA[18], etc. These methods are effective to improve classification performance. However, the extracted features fuse the information of original features, and lose the distinct physical meanings. Hence the dimension reduction results cannot be explained and easily comprehended.

Unlike feature extraction, feature selection will remain the physical meaning of features when reduce the dimensionality. It's essential in many applications. To cope with the feature selection task on multi-label problem, Yang et al[19] propose a filter framework to evaluate features for each label separately under some statistic evaluation metrics, and combine the results by average or max approaches. This framework is an extension of single label filter feature selection methods. It considers the labels separately, which ignores the correlations between labels. Trohidis et al propose another filter method on multi-label feature selection[4]. In their work, Multi-label dataset is transformed into single label dataset with LP method, and then a common attribute evaluation statistic is used to evaluate the feature's correlation with the transformed single label. This method considers label correlation, which is important in multi-label learning. Other scholars proposed wrapper methods to improve classification performance along with dimensionality reduction. Zhang et al. use genetic algorithm to improve the performance of multi-label Naïve Bayes classifier[20]. In their work, genetic algorithm is used to select the feature subset after PCA feature extraction. With PCA process, the original meaning of features is discarded. And the authors only investigate the performance of the multi-label Naïve Bayes classifier, and more classifiers need to be further investigated. Shao et al[21] propose a hybrid optimization multi-label feature selection method called HOML. In their work, simulated annealing, genetic algorithm and hill climb strategies are combined to select the best feature subset. The results show great improvement on performance. However, as a wrapper method, the computational complexity is too high.

As feature selection methods, wrapper methods are classifier specified feature selection methods. They select different optimal feature subset for different classifier, and measure the feature subsets with the classification performance directly. Wrapper methods can improve the performance of classifiers in a large range However, their computational complexity are always too high. Filter methods have linear computation cost, but their selection results are always rough. They consider the relevance between labels and each feature, while ignoring the power when features combine together. Moreover, filter methods provide a unique feature rank for different kind of classifier. The selected feature subset is always not the most suitable subset for a certain classifier. When we try to improve classification performance with feature selection, the time cost of wrapper methods are always too high, while filter methods can not fit certain classifier. To select the classifier specified features without the high time cost like wrapper methods, we propose a tradeoff method by introducing an embedded feature selection method into multi-label classification. Especially, we apply the new embedded method on music emotion classification. Less works concentrate on the study of music emotion classification.

The contribution of this paper is twofold: to present a new embedded feature selection method, called MEFS, on multi-label datasets, and to improve the performance when tagging music emotions, with the help of MEFS method.

The remaining of this paper is organized as the following. Section 2 introduces the music emotion dataset employed in experiments. Section 3 presents details of the proposed embedded feature selection methods. Section 4 explains the multi-label learning algorithms and multi-label evaluation metrics included in performance comparison. Section 5 reports our experiment results, and conclusions and future work are drawn in Section 6.

2. Music Emotion Classification Task

The music emotion dataset used in this work was firstly published by Konstantinos Trohidis et al[4] There are 593 chosen records in this dataset. Each of them belongs to the following 7 different genres: Classical, Reggae, Rock, Pop, Hip-Hop, Techno and Jazz. 72 features were extracted from each song. The extracted features fell into two categories: 8 rhythmic features and

64 timbre features. Detailed feature list and the computing method can be referred to the literature[4].

The emotion labels come from the Tellegen-Watson-Clark model.[4] 6 main labels are associated with the samples. The labels are "amazed-surprised", "happy-pleased", "relaxing-calm", "quiet-still", " sad-lonely" and "angry-fearful". The number of samples having these labels is 173, 166, 264, 148, 168 and 189 respectively.

For a multi-label dataset, more statistic indexes can be studied to give a deep understanding.[6]Common measurements for a multi-label dataset are cardinality, density and distinct. Cardinality means the average number of labels of a sample. Density is the average number of labels of a sample divided by the total number of labels. Distinct represents the number of different distinct label combinations appeared in the dataset. The statistic measurements of the music emotion dataset are shown in Table 1.

Table 1. Statistic indexes of the music emotion dataset studied

Measurement	Value
Cardinality	1.869
Density	0.311
Distinct	27

With the information shown in Table 1, we analyze the dataset roughly. The cardinality is 1.869, which means each sample is associated with about 2 labels in average. Physically, a clip of music contains two kinds of emotions on average. 27 of distinct show a strong correlation among the emotion labels.

3. Multi-label Embedded Feature Selection (MEFS)

Embedded feature selection evaluates feature subsets with the metrics extracted from some certain classifiers. Classification target is embedded naturally into the selection metrics in embedded feature selection approach. With the metrics, selected features are more direct to improve the classification performance. Embedded feature selection methods can achieve comparable selection results with the wrapper model and have the similar efficiency with filter way. Considering these benefits, embedded feature selection methods have been paid close attentions in areas of machine learning, data mining and bioinformatics in recent years.

Inspired by single label embedded feature selection methods, we propose a multi-label embedded feature selection method, called MEFS. In MEFS, prediction risk criterion[22] is adopted for the evaluation of features, and backward search strategy is used for the search of feature subset. In MEFS, feature selection process cooperates with multi-label classifiers. The feature selection results mainly depend on the used classifier, and the feature extraction ability of MEFS relies on the learning ability of the classifier.

Prediction risk is to evaluate the expected performance in classification of new observed data. During the process of data modeling, prediction risk is used to assess prediction accuracy of the models and select suitable models. The principle of minimization of prediction risk is often used for the selection of the optimal feature subset in single label problems. Prediction risk criterion evaluates each feature by calculating the change of training accuracy when the value of a certain feature is replaced by its mean value in all the samples, defined as:

$$S_i = ERR(X^i) - ERR \quad (1)$$

Here, ERR (error) stands for the prediction error of training model on training dataset. $ERR(X^i)$ stands for the prediction error of training model when the value of the ith feature is replaced by its mean value in all the samples of training dataset.

Let $X \in R^{N\times D}$ denote the dataset with N samples and D features, and $Y \in R^{N\times L}$ be the label set associated with X. $x^i \in R^{N\times 1}$ is the value of ith feature in all samples. The output of a classifier $f(x^1, \dots, x^D)$ is $\hat{Y}$. Let $L(\hat{Y}, Y)$ denotes a multi-label loss function, in which Y is the real label set associated with samples. Then $ERR(X^i)$ is defined as:

$$ERR(X^i) = L(f(x^1, \dots, \bar{x}^i, \dots, x^D), Y) \quad (2)$$

in which, $\bar{x}^i$ is the mean value of the ith feature and $f(x^1, \dots, \bar{x}^i, \dots, x^D)$ is the prediction value of all the samples with the ith feature replaced by its mean value.

The feature with the least value of S_i will be deleted, because the impact on the result by the change of the feature's value is the least. The effects of the deleted feature for distinguishing labels is the least and even negative.

When we apply the prediction risk criterion to the dimension reduction in multi-label learning, we take the evaluation measure of multi-label learning as the loss function in prediction risk. Five metrics, i.e. *hamming* *loss*, *one-error*, *average precision*, *coverage*, *ranking loss* are included. Especially, when take *average precision* as the loss function in Eq. (2), we need to calculate *1- average precision* as a *ERR* measurement.

The pseudo code of MEFS (Multi-label Embedded Feature Selection) algorithm is shown in Table 2, whose main idea is to make use of prediction risk to evaluate the features in feature subset, and use a backward search algorithm to delete the worst feature from feature subset step by step. In each loop, a classifier model is trained with the remained features, and evaluates each feature with Eq. (1). The worst feature is saved in the feature rank and removed from feature subset. Repeat above step until each feature is stored into the feature rank. The output is the feature rank and corresponding trained models. In testing step, the test data is restricted to a certain number of features based on the feature rank. Then we find the corresponding training model, and use it directly on the low dimensional test data to evaluate performance.

4. Experiment

4.1. *Feature selection methods*

For experiment, we include three other filter feature selection methods, *max*, *avg*[19] and *LP-Chi*[4], for comparison.

- *max*: *max* is a framework extended from single label feature selection methods. It calculates the dependency score with an attribute evaluation statistic, like χ^2, between a feature and a label separately. The maximal dependency score of a certain feature across all labels stands for the importance of the feature.
- *avg*: *avg* is similar to *max*. Dependency scores for some feature on all labels are averaged to form the final weight for that feature.
- *LP-Chi*: *LP-Chi* algorithm aims to select the best features for music emotion classification task[4]. In *LP-Chi*, the multi-label problem is transformed into multiclass problem by the transformation of LP method firstly. Then a common attribute evaluation statistic, like χ^2, is used to evaluate each feature on multiple classes. Finally, features are ranked by the statistic values. *LP-Chi* showed a better result than *max* and *average* approaches, because it takes the label correlation into account[4].

4.2. *Multi-label classifiers*

After selecting appropriate features, multi-label classifiers would participate in the classification tasks. In order to eliminate the bias of classifiers, four multi-label classifiers, which are LEAD[10], MLNB[20], Rank-SVM[11] and ML-KNN[12], are employed in the experiment. LEAD and MLNB are problem transformation methods, which transform the multi-label classification problem into one or more single-label classification, regression or ranking tasks. Rank-SVM and ML-KNN are algorithm adaptation methods, which extend specific learning algorithms in single label problem to handle multi-label data directly.

- LEAD means multi-label Learning by Exploiting label Dependency. In LEAD, a Bayesian network is built to characterize the joint probability of all labels, conditioned on the feature set. Then BR (Binary Relevance) classifiers are trained to predict each label by taking its parental labels in the learned Bayesian network as additional input features.

  The SVM model used in LEAD is trained with a linear kernel and the complexity constant C equals to 1. We use the LIBSVM package[25], which involves the training and testing algorithms of SVM models. The BDAGL (Bayesian DAG learning) package is used, which implemented the dynamic programming-based algorithm for computing the marginal posterior probability of every edge in a Bayesian network.[26]
- MLNB stands for Multi-Label Naïve Bayes. It uses the Bayesian rule and adopts the assumption of class conditional independence among features as classic naive Bayes classifiers do, then uses Bayes rule to calculate the posterior probability of each label. The labels with the largest posterior probabilities are labeled to the unlabeled instances.

  In our experiment, the Gaussian probability density model is used to estimate the conditioned probability.
- Rank-SVM tries to train SVMs for each label. The objective function in Rank-SVM is minimizing the ranking loss, which is one of the main targets of multi-label learning. The SVM model used in Rank-SVM is trained with a linear kernel and the complexity constant C equals to 1. The tolerance value for λ, for difference between α^{p+1} and α^{p} are set to their default value. Maximum number of

Table 2. Pseudo code of MEFS

```
MEFS (X_t, Y_t, L)
input
X_t, Y_t are the training data and training label
L is the loss function in Eq. (2)
---------------------------------------------------------------
M ← ∅; // empty trained model list
r ← ∅; // empty feature ranking list
u ← [1,2,…,D] // u is the remained feature set, initialize it by the universal set

while (u ≠ ∅)
  S ← (0,…,0) with the dimensionality |u| // initialize S
  X̂_t ← X_t(:,u) //restrict all training samples to having the remained feature indexes
  model ← trainclassifier(X̂_t,Y_t) // train a classifier with the restricted dataset
  ERR ← testclassifier(model, X̂_t,Y_t) // test the trained classifier and get the tanning error

  for each feature i in u
     compute ERR(X̂_t^i) as Eq.(2) showed
     S[i] ← ERR(X̂_t^i) − ERR
  end // evaluate each feature's importance according to the prediction risk criterion

  insert model to the end of M // save the classifier models in list M
  h ← argmin_{i∈u} S // find the index of worst feature
  insert u[h] to the head of r // update the feature rank
  remove u[h] from u // remove the worst feature

end //complete the feature selection process

output the classifier list M and the feature rank r.
```

iterations is set to 50. Detail information can be found in Ref.11.

- ML-KNN is a high-performance problem adaptation method. ML-KNN brings the idea from KNN classifier, but it adopts maximum a posteriori (MAP) principle instead of the simple number counting to predict the label for new instances.

  For ML-KNN, the number of nearest neighbors considered is set to 10.

4.3. *Evaluation metric*

The evaluation measure of multi-label learning is more complex than that of single label. Five popular measures specially designed for multi-label learning are used in this paper, i.e. hamming loss, one-error, coverage, ranking loss and average precision.[14] Suppose X is the instance set, Y is the label set. T is the training set.and $T = \{(x_1, Y_1), (x_2, Y_2), \dots, (x_m, Y_m)\}$ $(x_i \in X, Y_i \in Y)$. S is the test set, and $S = \{(x_1, Y_1), (x_2, Y_2), \dots, (x_p, Y_p)\}$. The target of the learning process is to output a function $h{:} = X \to 2^Y$ in order to get a multi-label classifier which can optimize the evaluation measure. However in most cases, the classifiers produce real value function: $f{:}X \times Y \to R$. For a given instance x_i and its label set Y_i, a good classifier tends to produce a greater value for the label in Y_i compared with those instances without label Y_i, so there is $f(x_i, y_1) > f(x_i, y_2)$ for any $y_1 \in Y_i$ and $y_2 \notin Y_i$. Real function $f\ (x_i,\cdot)$ can be transformed to be ranking function $rank(x_i,\cdot)$, which is a one-to-one mapping onto $\{1, \dots, |Y|\}$. These two functions have the following relations. When $f(x_i, y_1) > f(x_i, y_2)$, there is $rank(x_i, y_1) < rank(x_i, y_2)$. Still real function $f\ (x_i,\cdot)$ also can be transformed to be a multi-label function $h\ (x_i)$,$h(x_i) = \{y \mid f(x_i, y) > t(x_i), y \in Y\}$. $t\ (x_i)$ is a threshold function (0 by default). Based on the above descriptions, five measures are defined as follows.

- Hamming loss

  Hamming loss is used to evaluate the times when the label of the instance is predicted wrongly, i.e. when $\mathrm{hloss}_S(h) = 0$. The smaller the hamming loss, the better the classifier.

$$hloss(h) = \frac{1}{p}\sum_{i=1}^{p}\frac{1}{|Y|}|h(x_i)\Delta Y_i| \quad (3)$$

where p is the size of testing set, Δ finds the difference between predicted label set and the actual label set. When $|Y_i| = 1$ on all the instances, this becomes one label problem.

- One error

One error counts the number of instance whose first predicted label is not one of its real labels. The smaller the one error, the better the classifier.

$$one - error(f) = \frac{1}{p}\sum_{i=1}^{p} argmax_{y\in Y} f(x,y) \notin Y_i \quad (4)$$

- Coverage

Coverage is the number of labels we need to search along the label rank when finding all the labels of one instance in the label set. The smaller the coverage, the better the classifier.

$$\text{coverage(f)} = \frac{1}{p}\sum_{i=1}^{p} \max_{y\in Y} rank(x_i, y) - 1 \quad (5)$$

- Ranking loss

Ranking loss is the number of label pairs disordered in the label list. The smaller the ranking loss, the better the classifier.

$$rloss(f) = \frac{1}{p}\sum_{i=1}^{p}\frac{1}{|Y_i||\widehat{Y_i}|}|O_i| \quad (6)$$

where O_i is the disordered label pair, defined by:

$$O_i = \{(y_j, y_k)|f(x_i, y_j) \le f(x_i, y_k), (y_j, y_k) \in Y_i \times \widehat{Y_i}\}$$

- Average precision

Average precision represents the average fraction of pairs that are not correctly ordered. The bigger average precision, the better classifier.

$$avgprec = \frac{1}{p}\sum_{i=1}^{p}\frac{1}{|Y_i|}\sum_{y\in Y_i}\frac{|W_i(y)|}{rank(x_i,y)} \quad (7)$$

where $W_i(y)$ is the predicted label set which have a higher ranking than the true label y, defined as:

$$W_i(y) = \{(y_j|rank(x_i, y_j) \le rank(x_i, y), y_j \in \widehat{Y_i})\}$$

4.4. *Experimental setup*

In the experiment, average precision (Eq. (7)) and hamming loss (Eq. (3)) function are chosen as the measurement functions $L(\hat{y}, y)$ in Eq.(2), respectively. MEFS is compared with three other feature selection methods *LP-Chi*, *max* and *avg*. 4 classifiers, i.e. LEAD, Rank-SVM, MLNB and ML-KNN, are all implemented in the experiment for an exhaustive assessment. 5 evaluation criterions, average precision, hamming loss, ranking loss, coverage and one error, are investigated in the results comparison. In all of the experiments, we used 10-fold cross validation. The whole dataset is segmented into 10 groups with equal number of samples. In each experiment, nine of the groups are used to select features and train a model that is evaluated on the remaining group. This procedure is then repeated for all 10 possible choices for the held-out group.

We design the methods comparison from three aspects:

1) The performance variation against increasing selected feature number.

   In this part, feature subset is expanded by the feature from the rank list, one by one. Evaluation metrics for each classifier with different feature subsets are recorded and compared in detail.

2) The best performance can each feature selection method get.

   Among the expanded feature sets from the first experiment aspect, the best performance of each feature selection method is extracted and compared. On five evaluation criterions, best results are inspected, respectively.

3) Processing time.

   The processing time for different feature selection methods will be compared. In this case we can find time cost to improve performance.

5. Results and discussion

In this section, MEFS (AP) represents the MEFS with average precision as the prediction risk criterion, while MEFS (HL) represents the MEFS with hamming loss.

5.1. *Performance comparison against feature number on hamming loss*

We demonstrate the hamming loss comparison of five feature selection methods on four different classifiers in Figure 1-4. The representation of each line is figured out in the legends. The horizontal axis represents the number of features retained, and the vertical axis stands for the corresponding evaluation metric. From Figure 1, we can observe that with the number of feature increasing, hamming loss on LEAD classifier seems is monotone decreasing. That is, more features bring better performance of LEAD. When feature set is larger than 40, the hamming loss on LEAD seems changeless. It may attribute to the strong learning ability of LEAD. LEAD can make good use of features, and its performance may not be heavily damaged by redundant features.

At the beginning part of Figure 1, result from

MEFS decreases more quickly, showing that MEFS can quickly select better features than LP-Chi, max, and avg does. And in most of the feature subsets, MEFS achieves lower hamming loss than those filter methods.

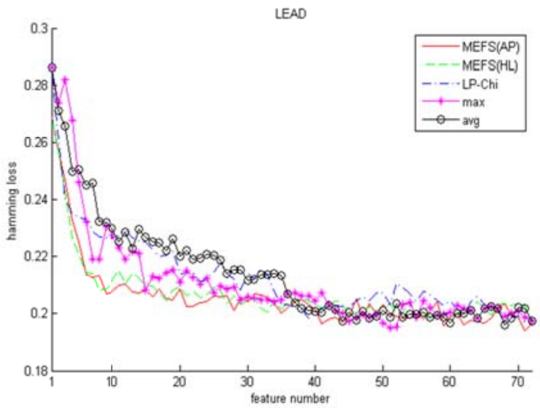

Figure 1. Hamming loss of five methods using LEAD classifier

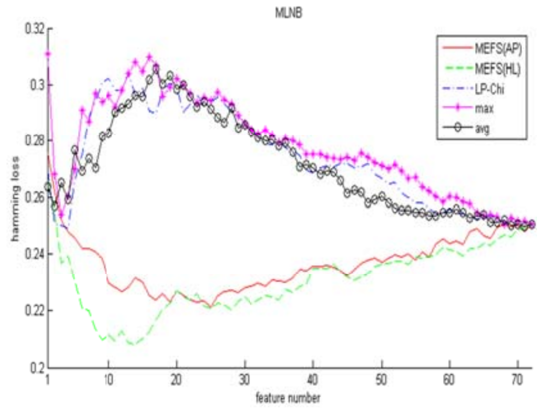

Figure 2. Hamming loss of five methods using MLNB classifier

On MLNB and Rank-SVM classifiers in Figure 2 and 3, the concave curves of MEFS indicate the necessary of feature selection. That is, involving all the features will be unexpectedly harmful to the classification.

In Figure 2 and 3, models based on MEFS get prominent improvements, compared with other three feature selection techniques. The great difference between MEFS and others presents the significant advantage of the proposed MEFS method. However, MEFS performs worse on ML-KNN classifier, which may due to the incompatibility between ML-KNN and the intrinsic mechanism of MEFS. In MEFS, the feature's importance is evaluated by the error change when the feature value on all instances is replaced by their mean value. When a certain feature is replaced by its mean value, the distance between two samples would be changed in a small range, because the effect of the mean value feature will be averaged by other features. As a result, the neighborhoods may not be changed significantly, and MEFS can hardly find the worst features based on the error change. In this case, we call MEFS and ML-KNN may be incompatible.

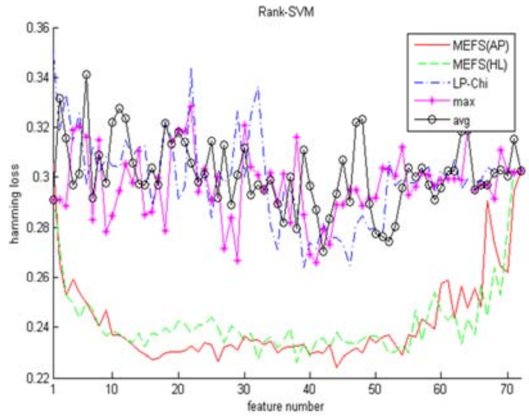

Figure 3. Hamming loss of five methods using Rank-SVM classifier

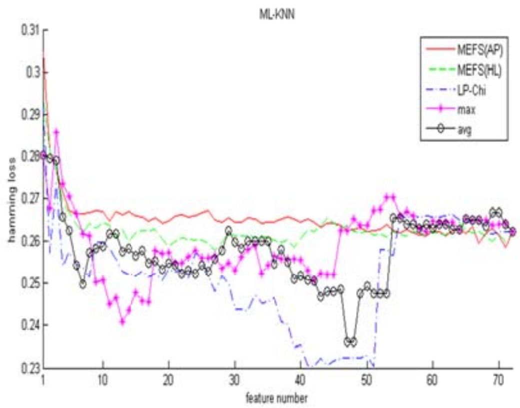

Figure 4. Hamming loss of five methods using ML-KNN classifier

5.2. *Best performance comparison*

The best results of each feature selection method with four classifiers are shown in Table 3-6. The upper line in each block is the averaged best performance on each evaluation metric in 10 fold cross validation. The best performance across all feature selection methods on each evaluation criterion is highlighted in bold. The lower line is the number of features when the classifier reaches its best performance. The number of selected features gives more information for comparison when the best performances are similar. In the last column, performances of the classifiers without feature selection are demonstrated.

The mean and standard deviation are shown in Tables 3-6 with the format "mean±std" "↓" indicates "the smaller the better" while "↑" indicates "the bigger the better".

From Table 3-6, we can find out that:

1) The proposed MEFS with average precision or hamming loss as its prediction risk criterion can get the best performance with three of the classifiers, LEAD, Rank-SVM and MLNB, on all the five evaluation metrics. But with ML-KNN, *LP-Chi* obtains the best performance. All the classifiers' performance can be improved by feature selection methods.
2) As shown in Table 3, MEFS (AP) slightly outperforms other feature selection methods and MEFS (HL). However, all the feature selection methods can only improve the performance of LEAD in a small range. It may because of the strong learning ability of LEAD. Classifier LEAD can learn sufficient information from the instance features, even when most of the features are redundant or irrelevant. Feature selection will contribute little to LEAD classification.
3) In Table 3, the best result is obtained by MEFS (AP), which improves LEAD by 8.57% in average. MEFS can not only achieve the highest performance improvement, but also with the smallest feature number. The best performance can be got with 40 of 72 features in the music emotion dataset, by MEFS method.
4) From Table 3-6, we can observe that with different evaluation metric in Eq. (2), such as average precision, hamming loss in the experiment, MEFS have different performance. It is always essential to choose the best fit metric for each classifier. In this paper, we only tried two of the metrics and find a better one for each classifier. The relationship between the evaluation metric and classifier needs to be further studied.

In Table 7, we present how much improvement can be obtained when employing different feature selection methods. There are 5 multi-label evaluation metrics investigated, we calculate the improvement of hamming loss metric as a representative. The percentage of reduced hamming loss is shown in Table 7, with the format "mean±std".

When the chosen classifier is Rank-SVM or MLNB, the difference between MEFS and other methods begin to emerge. According to Table 7, MEFS can improve classifier Rank-SVM by more than 32.6%, and improve MLNB by 22.6% in average, while the other methods can only improve Rank-SVM by about 20%, and improve MLNB by about 6%. These classifiers may be damaged by redundant features. With the classifier specified features chosen, the performance of these classifiers have been significantly improved.

Table 3. Comparative results with classifier LEAD

	MEFS(AP)	MEFS(HL)	LP-Chi	max	avg	All features
average precision↑	**0.8358±0.0377** 39.9±17.6	0.8299±0.0372 44±19.2	0.8286±0.0352 58.8±11.2	0.8292±0.0309 61.8±5.9	0.8322±0.0372 60.2±9.2	0.8045±0.0362
hamming loss↓	**0.1801±0.0181** 45.8±20.6	0. 1854±0.0151 34.7±15.6	0.1835±0.0168 46.4±13.4	0.1807±0.0141 53.5±9.3	0.1848±0.0165 55.8±14.2	0.1967±0.0184
one error↓	**0.2005±0.0619** 35.5±17.5	0.2072± 0.0631 35.3±36.1	0.2190±0.0513 56.9±14.9	0.2157±0.0527 51.7±19.6	0.2207±0.0648 47.2±15.5	0.2696±0.0638
coverage↓	**0.2718±0.1830** 41.7±15.4	0.2735±0.0307 44.2±24.1	0.2746±0.0319 53±12.1	0.2752±0.0312 57.1±8.1	0.2732±0.0319 54.±12.5	0.2948±0.0337
ranking loss↓	**0.1349±0.0282** 51.9±18.5	0.1384±0.0244 49.5±22.7	0.1402±0.0265 55.5±11.8	0.1384±0.0242 57.4±9.1	0.1353±0.0269 57.4±8.5	0.1597±0.0271

Table 4. Comparative results with Rank-SVM

	MEFS(AP)	MEFS(HL)	LP-Chi	max	avg	All features
average precision↑	0.8032±0.0293 49.3±9.7	**0.8040±0.0336** 53.3±18.3	0.7641±0.0372 37.2±11.3	0.7667±0.0455 29.5±16.3	0.7584±0.0389 33.2±13.0	0.6786±0.0447
hamming loss↓	0.2088±0.0080 43.1±17.6	**0.2052±0.0157** 39.4±25.4	0.2357±0.0211 44.8±8.3	0.2338±0.0231 25.2±17.6	0.2413±0.0169 44.1±17.6	0.3024±0.0324
one error↓	0.2577±0.0578 41.9±15.8	**0.2493±0.0647** 39.8±23.4	0.3252±0.0747 34.9±11.4	0.3169±0.0778 33.0±21.4	0.3185±0.0702 43.8±6.9	0.4621±0.0688
coverage↓	0.2940±0.0317 55.1±9.8	**0.2867±0.0252** 52.5±19.1	0.3162±0.0273 41.5±5.1	0.3162±0.0223 24.0±15.1	0.3246±0.0270 36.8±18.3	0.3997±0.0422
ranking loss↓	0.1647±0.0220 54.8±8.2	**0.1578±0.0207** 54.9±17.8	0.1961±0.0235 38.8±10.7	0.1922±0.0336 26.3±15.1	0.2053±0.0279 36.8±12.1	0.2988±0.0511

Table 5. Comparative results with MLNB

	MEFS(AP)	MEFS(HL)	LP-Chi	max	avg	All features
average precision↑	**0.8148±0.0254** 30.0±14.0	0.8139±0.0309 28.5±14.8	0.7845±0.0259 53.1±26.7	0.7826±0.0273 55.1±25.3	0.7865±0.0321 55.6±20.6	0.7689±0.0342
hamming loss↓	0.2062±0.0155 20.4±13.1	**0.1955±0.0234** 13.7±7.9	0.2323±0.0240 23.1±33.0	0.2366±0.0226 23.2±32.7	0.2340±0.0231 32.1±29.7	0.2507±0.0186
one error↓	**0.2257±0.0443** 29.8±20.7	0.2307±0.0562 31.2±17.9	0.2915±0.0554 53.6±19.1	0.2966±0.0518 54.0±26.8	0.2914±0.0632 45.3±20.7	0.3134±0.0629
coverage↓	0.2884±0.0251 31.0±15.5	**0.2827±0.0332** 31.3±21.2	0.3036±0.0261 48.0±25.6	0.3053±0.0275 58.4±19.6	0.3013±0.0285 50.1±19.5	0.3134±0.0265
ranking loss↓	0.1532±0.0156 33.4±12.9	**0.1488±0.0257** 22.1±10.5	0.1817±0.0202 53.3±27.0	0.1833±0.0197 54.6±25.3	0.1800±0.0218 51.2±19.9	0.1906±0.0214

Table 6. Comparative results with ML-KNN

	MEFS(AP)	MEFS(HL)	LP-Chi	max	avg	All features
average precision↑	0.7271±0.0242 47.5±28.7	0.7253±0.0235 44.8±29.1	**0.7752±0.0367** 34.4±16.8	0.7599±0.0335 25.2±13.6	0.7633±0.0351 38.4±18.1	0.7117±0.0259
hamming loss↓	0.2511±0.0179 46.4±25.0	0.2473±0.0191 33.6±27.4	**0.2186±0.0186** 37.4±13.7	0.2245±0.0218 28.5±19.8	0.2299±0.0131 28.3±16.3	0.2623±0.0155
one error↓	0.3559±0.0448 21.0±21.1	0.3509±0.0453 22.0±26.6	**0.3033±0.0545** 30.1±19.1	0.3067±0.0447 26.1±17.2	0.3117±0.0529 35.0±14.7	0.3913±0.0570
coverage↓	0.3663±0.0355 51.4±26.8	0.3666±0.0335 49.2±29.1	**0.3145±0.0228** 27.8±19.1	0.3269±0.0263 26.9±15.6	0.3244±0.0283 27.0±18.7	0.3787±0.0357
ranking loss↓	0.2456±0.0282 40.9±28.1	0.2472±0.0268 48.5±27.4	**0.1866±0.0291** 32.1±20.0	0.2032±0.0271 29.0±20.7	0.2004±0.0294 36.4±20.3	0.2600±0.0255

Table 7. Improvements obtained by employing each feature selection method

	MEFS	LP-Chi	max	avg
LEAD	**8.57%±6.2%**	6.6%±1.7%	7.7%±7.4%	5.8%±5.1%
Rank-SVM	**32.6%±9.0%**	21.3%±10.7%	22.0%10.7%	19.5%±8.9%
MLNB	**22.6%±7.0%**	7.9%±7.0%	5.6%±6.2%	6..6%±6.6%
ML-KNN	6.25%±4.4%	**16.4%±8.5%**	14.1%10.7%	12.1%±7.4%

5.3. *Computing time*

In this part, we compare the time cost of the proposed MEFS method and LP-Chi method when searching the optimal feature subset in training steps.

The experiments execute on a personal computer with an Intel Core (TM) 2 Duo CPU E7400 @ 2.80GHz processor, and 2990MB RAM.

In Table 8, the time costs are shown in the format "mean±std", and the measurement unit of time cost is "seconds".

The result shows that MEFS spent more time than LP-Chi method in most of the cases. Intuitively, if there are F features, MEFS needs to train F modelsand test $\frac{F(F+1)}{2} + F$ times to sort the features into a rank list. While LP-Chi method is a filter technology, which needs to train F models, and only test F times. So, in the train part, MEFS method costs much time to delicately search for the best feature subset. But it will not harm the advantage of MEFS method. It is as fast as other filter functions in the test part, which is more important for practical applications. Exceptively, we surprisingly find that in Rank-SVM, MEFS runs even faster than LP-Chi. It looks like MEFS can converge on Rank-SVM more quickly, just as Figure 3 shows.

6. Conclusions

The feature selection problem for multi-label musical emotion classification is investigated, where a novel multi-label feature selection algorithm MEFS is proposed. Experimental evaluation is performed by using four multi-label classification algorithms on a collection of 593 songs. Results show that MEFS performs better than the state-of-arts works like *LP-Chi* in most cases. This would benefit the automated annotation of large musical collections with multiple emotions.

We consider further to improve the efficiency of feature selection, which we believe has great potential in this domain.

Table 8. Comparison of computing time

	MEFS	LP-Chi
LEAD	19114±3461	6837±368
Rank-SVM	8498±168.9	9965±55.5
MLNB	1293±3.2	5±0.1
ML-KNN	4227±15.3	61±0.1

Acknowledgements

This work was supported by the Natural Science Foundation of China under grant no. 61005006, and the Fundamental Research Funds for the Central Universities.

References

1. Y. H. Yang, C. C. Liu, and H. H. Chen, Music emotion classification: a fuzzy approach, in *Proceedings of the 14th annual ACM international conference on Multimedia*, 2006, pp. 81–84.
2. L. Lu, D. Liu, and H. J. Zhang, Automatic mood detection and tracking of music audio signals, *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 14, no. 1, pp. 5–18, 2006.
3. Y. H. Yang, Y. C. Lin, Y. F. Su, and H. H. Chen, Music emotion classification: A regression approach, in *Multimedia and Expo, 2007 IEEE International Conference on*, 2007, pp. 208–211.
4. K. Trohidis, G. Tsoumakas, G. Kalliris, and I. Vlahavas, Multilabel classification of music into emotions, in *Proc. 9th International Conference on Music Information Retrieval (ISMIR 2008), Philadelphia, PA, USA*, 2008, vol. 2008.
5. A. Wieczorkowska, P. Synak, and Z. Raś, Multi-label classification of emotions in music, *Intelligent Information Processing and Web Mining*, pp. 307–315, 2006.
6. G. Tsoumakas, I. Katakis, and I. Vlahavas, Mining multi-label data, *Data mining and knowledge discovery handbook*, pp. 667–685, 2010.
7. G. Tsoumakas and I. Katakis, Multi-Label Classification, *International Journal of Data Warehousing & Mining*, vol. 3, no. 3, pp. 1–13, 2007.
8. G. Tsoumakas, I. Katakis, and I. Vlahavas, Random k-Labelsets for Multilabel Classification, *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 7, pp. 1079-1089, 2011.
9. J. Read, B. Pfahringer, G. Holmes, and E. Frank, Classifier chains for multi-label classification, *Machine Learning and Knowledge Discovery in Databases*, pp. 254–269, 2009.
10. M. L. Zhang and K. Zhang, Multi-label learning by exploiting label dependency, in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2010, pp. 999–1008.
11. J. Weston and others, A Kernel Method for Multi-Labelled Classification, 2008.
12. M. L. Zhang and Z. H. Zhou, ML-KNN: A lazy learning approach to multi-label learning, *Pattern Recognition*, vol. 40, no. 7, pp. 2038–2048, 2007.

13. M. L. Zhang and Z. H. Zhou, Multilabel neural networks with applications to functional genomics and text categorization, *Knowledge and Data Engineering, IEEE Transactions on*, vol. 18, no. 10, pp. 1338–1351, 2006.
14. R. E. Schapire and Y. Singer, BoosTexter: A boosting-based system for text categorization, *Machine learning*, vol. 39, no. 2, pp. 135–168, 2000.
15. C. Wang, S. Yan, L. Zhang, and H. J. Zhang, Multi-label sparse coding for automatic image annotation, in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 2009, pp. 1643–1650.
16. Y. Zhang and Z. H. Zhou, Multilabel dimensionality reduction via dependence maximization, *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 4, no. 3, p. 14, 2010.
17. S. Gao, W. Wu, C. H. Lee, and T. S. Chua, A MFoM learning approach to robust multiclass multi-label text categorization, in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 42.
18. C. H. Park and M. Lee, On applying linear discriminant analysis for multi-labeled problems, *Pattern recognition letters*, vol. 29, no. 7, pp. 878–887, 2008.
19. Y. Yang and J. O. Pedersen, A comparative study on feature selection in text categorization, in *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, 1997, pp. 412–420.
20. M. L. Zhang, J. M. Peña, and V. Robles, Feature selection for multi-label naive Bayes classification, *Information Sciences*, vol. 179, no. 19, pp. 3218–3229, 2009.
21. H. Shao, G. Z. Li, G. P. Liu, and Y. Wang, "Symptom Selection for Multi-label Data of Inquiry Diagnosis in Traditional Chinese Medicine", *SCIENCE CHINA Information Sciences*, p. p.in press, 2011.
22. G.-Z. Li, J. Yang, G.-P. Liu, L. Xue. Feature selection for multi-class problems using support vector machines, In: Proceedings of 8th Pacific Rim International Conference on Artificial Intelligence (PRICAI-04), Sheraton, Auckland, August 9 to August 13, 2004, LNCS3157, 292-300
23. R. E. Thayer and R. J. McNally, The Biopsychology of Mood and Arousal, *Cognitive and Behavioral Neurology*, vol. 5, no. 1, p. 65, 1992.
24. G. Tzanetakis, G. Essl, and P. Cook, Automatic Musical Genre Classification Of Audio Signals, *Organised Sound*, vol. 10, no. 5, pp. 143-146, 2001.
25. C. C. Chang and C. J. Lin, LIBSVM: a library for support vector machines, *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.
26. M. Koivisto, Advances in exact Bayesian structure discovery in Bayesian networks, in *Proc. of the 22nd Conference on Uncertainty in Artificial Intelligence (UAI)*, 2006, pp. 241–248.