Multi-Label Selective Ensemble

Nan Li, Yuan Jiang and Zhi-Hua Zhou

National Key Laboratory for Novel Software Technology, Nanjing University
Collaborative Innovation Center of Novel Software Technology and Industrialization
Nanjing 210023, China
{lin, jiangy, zhouzh}@lamda.nju.edu.cn

Abstract. Multi-label selective ensemble deals with the problem of reducing the size of multi-label ensembles whilst keeping or improving the performance. In practice, it is of important value, since the generated ensembles are usually unnecessarily large, which leads to extra high computational and storage cost. However, it is more challenging than traditional selective ensemble, because real-world applications often employ different performance measures to evaluate the quality of multi-label predictions, depending on user requirements. In this paper, we propose the MUSE approach to tackle this problem. Specifically, by directly considering the concerned performance measure, we develop a convex optimization formulation and provide an efficient stochastic optimization solution for a large variety of multi-label performance measures. Experiments show that MUSE is able to obtain smaller multi-label ensembles, whilst achieving better or at least comparable performance in terms of the concerned performance measure.

Key words: multi-label classification, ensemble pruning, selective ensemble

1 Introduction

Multi-label learning deals with the problem where each instance is associated with multiple labels simultaneously, and it has wide applications in different domains, for example, document categorization where a document may belong to multiple topics [16, 27], multi-media annotation where an image or a music can be annotated with more than one tags [1, 26]. During the past few years, it has become an active research topic [20, 8, 31, 11, 10, 2, 19, 7, 28], and a recent comprehensive survey can be found in [32].

In multi-label learning, label correlations have been widely accepted to be important [3, 30], and many approaches have been proposed to exploit label correlations. Amongst them, multi-label ensemble methods which construct a group of multi-label classifiers and combine them for prediction have drawn much attention. For examples, random k-labelsets (RAKEL) is an ensemble of classifiers, each taking a small random subset of labels and learning on the power set of this subset [25]; ensemble of pruned sets (EPS) combines a set of pruned sets classifiers, each mapping sets of labels to single labels while pruning infrequent ones [18]; ensemble of classifier chains (ECC) combines multiple classifier chain classifiers, each learning an extended binary relevance classifier based on a random label ordering [19], and EPCC is a probabilistic extension of ECC [3]. From the perspective of ensemble learning, these multi-label

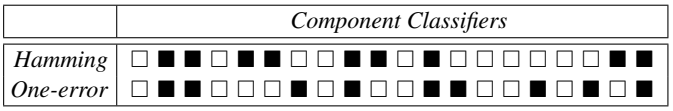

Fig. 1. Comparison of the selected classifiers when considering different multi-label performance measures (i.e., Hamming loss and one-error), where the component classifiers are classifier chains with random label ordering trained on the *cal500* dataset, the selective ensemble chooses 9 out of 20 component classifiers via exhaustive search, and '■'/'□' indicates it is selected/unselected.

ensemble methods try to construct diverse multi-label classifiers [22], mostly by smart heuristic randomization strategies.

In general, by combining more diverse multi-label classifiers, the ensemble performance tends to improve and converge. However, one issue is that the constructed ensembles tend to be unnecessarily large, requiring large amount of memory and also decreasing the response time of prediction. In traditional single label learning, selective ensemble (*a.k.a.* ensemble pruning or ensemble selection) addresses this issue by choosing a subset of component classifiers to form a subensemble, this has achieved success and became an active research topic in ensemble learning (see [34, chapter 6]). In this paper, we study the selective ensemble problem in the multi-label learning setting, that is, we try to reduce the size of a multi-label ensemble, such that compared with the original ensemble, the memory requirement and the response time can be reduced while similar or better prediction performance can be achieved.

In contrast to traditional supervised learning which usually takes accuracy as the performance measure, multi-label learning systems often employ different performance measures to evaluate the quality of multi-label predictions, depending on the application and user requirements [20]. These measures are designed from different aspects, and a classifier performing well in terms of one measure does not necessarily achieve good performance in terms of other measures, and it has been shown that a multi-label classifier tailored for one specific performance measure can perform poorly in terms of other measures [4]. This will make multi-label selective ensemble quite different from traditional selective ensemble, that is, we need to take the concerned performance measure fully into account when generating multi-label selective ensemble. As an example, we can see in Fig.1 that the classifiers selected for Hamming loss are quite different from those for one-error. Essentially, this makes the task of multi-label selective ensemble more challenging, because most multi-label performance measures are quite complicated and difficult to optimize, for example, most of them are non-decomposable over labels, non-convex, and non-smooth. To deal with this issue, we propose the MUSE approach to optimize the concerned performance measure. Specifically, by focusing on two groups of multi-label performance measures, we formulate the problem into a convex optimization problem with ℓ_1-norm regularization, and then present an efficient stochastic optimization solution. Experiments on real-world datasets show the effectiveness of the proposed MUSE approach.

The remainder of the paper is organized as follows. Section 2 presents our proposed MUSE approach. Section 3 reports the experiment results. Section 4 makes some brief discussion with related work, which is followed by the conclusion in Section 5.

2 The MUSE Approach

Let $\mathcal{X}$ be the instance space and $\mathcal{L}$ be a set of l labels. In multi-label learning, each instance $\mathbf{x}_i \in \mathcal{X}$ is associated with multiple labels in $\mathcal{L}$, which is represented as an l-dimensional binary vector $\mathbf{y}_i$ with the k-th element 1 indicating $\mathbf{x}_i$ is associated with the k-th label and -1 otherwise. Given a set of training examples $S = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^m$, the task is to learn a multi-label classifier

$$h : \mathcal{X} \mapsto \mathcal{Y} ,$$

where $\mathcal{Y} \subseteq \{-1, 1\}^l$ is the set of feasible label vectors, such that it can predict labels for unseen instances. In practice, instead of learning h directly, it is often to learn a vector-valued function $f : \mathcal{X} \mapsto \mathbb{R}^l$ which determines the label of $\mathbf{x}$ as

$$\hat{\mathbf{y}} = \operatorname*{argmax}_{\mathbf{y}' \in \mathcal{Y}} \ \mathbf{y}'^{\top} f(\mathbf{x}) \ . \tag{1}$$

We can see that how the argmax can be computed depends on $\mathcal{Y}$; for example, if $\mathcal{Y} = \{-1, 1\}^l$, then $\hat{\mathbf{y}}$ can be obtained as $\operatorname{sign}[f(\mathbf{x})]$.

In multi-label learning, a large variety of multi-label performance measures are *example-based performance measures*, which first quantify the performance on each example and then average them over all the examples as the final result. Generally speaking, these performance measures are in the following two groups:

- **Set based performance measures** which evaluate the performance based on the label set prediction of each example, and its representative examples include Hamming loss, F1-score, etc.;
- **Ranking based performance measures** which are based on the ranking of each label for each example, for example, ranking loss and coverage fall in this group.

Without loss of generality, we denote the concerned performance measure as the risk function $\Delta(\mathbf{y}, f(\mathbf{x}))$, which is the smaller the better. For performance measure which is the larger the better, like F1-score, Δ is simply set to one minus it. Obviously, it will be ideal if the classifier f can minimize the expected risk $\mathbb{E}_{(\mathbf{x},\mathbf{y})}[\Delta(\mathbf{y}, f(\mathbf{x}))]$.

2.1 The Problem

In general, multi-label ensemble methods construct a set of multi-label classifiers $\{h^{(t)} : \mathcal{X} \mapsto \mathcal{Y}\}_{t=1}^k$, and combine them to produce the vector-valued function $f : \mathcal{X} \mapsto \mathbb{R}^l$ as

$$f(\mathbf{x}; \mathbf{w}) = \sum_{t=1}^{k} w_t h^{(t)}(\mathbf{x}) \ , \tag{2}$$

where $\mathbf{w} = [w_1, \ldots, w_k]^{\top}$ is the weighting vector, for example, they are simply set to $1/k$ for voting. It can be found that in (2) the classifier $h^{(t)}$ will be excluded from the ensemble if w_t is zero, and the size of the ensemble is simply $\|\mathbf{w}\|_0$. Then, the task of multi-label selective ensemble becomes to find a weighting vector $\mathbf{w}$, such that the

expected performance in terms of the concerned performance measure is optimized, while the ensemble size $\|\mathbf{w}\|_0$ is small.

Since the expectation is infeasible, empirical risk is often used, and the problem of multi-label selective ensemble can be written as

$$\min_{\mathbf{w}\in\mathcal{W}} \frac{1}{m}\sum_{i=1}^{m} \Delta(\mathbf{y}_i, f(\mathbf{x}_i;\mathbf{w})) \quad \text{s.t. } \|\mathbf{w}\|_0 \leq b\,, \tag{3}$$

where $\mathcal{W}$ is the feasible space of $\mathbf{w}$, and $0 < b \leq k$ is the budget of ensemble size. However, this problem is challenging to solve, mainly because the risk function Δ is among various multi-label performance measures which are generally non-decomposable over labels, non-convex and non-smooth.

2.2 A Convex Formulation

Inspired by the works on structured prediction [23], instead of directly optimizing the empirical risk, in MUSE, we consider to optimize one of its convex upper bounds. Before giving the upper bound, we first make a definition.

Definition 1 *A rank vector $\mathbf{r}$ is a permutation of the integer vector $[1, 2, \ldots, l]$, and the rank vector $\mathbf{r}$ is said to be consistent with the label vector $\mathbf{y}$, if and only if there does not exist an index pair $\langle i, j\rangle$ satisfying $y_i = 1$, $y_j = -1$, and $r_i < r_j$.*

In practice, given a multi-label prediction $\mathbf{p} = f(\mathbf{x})$, the corresponding rank vector can be obtained by sorting $\mathbf{p}$ in ascending order, i.e., if p_t is the smallest in $\mathbf{p}$ then r_t is 1, and the largest corresponds to l. A rank vector $\mathbf{r}$ is consistent with the label vector $\mathbf{y}$, if all the relevant labels indicated by $\mathbf{y}$ have larger rank value than non-relevant ones.

Proposition 1 *Given a multi-label classifier $f : \mathcal{X} \mapsto \mathbb{R}^l$ and a multi-label performance measure Δ ,*

(a) if Δ is a set based performance measure, define the loss function

$$\ell(\mathbf{y}, f(\mathbf{x})) = \max_{\mathbf{y}'\in\mathcal{Y}} \left[(\mathbf{y}' - \mathbf{y})^\top f(\mathbf{x}) + \Delta(\mathbf{y}, \mathbf{y}')\right]\,, \tag{4}$$

where $\mathcal{Y}$ is the set of feasible label vectors,

(b) if Δ is a ranking based performance measure, define the loss function

$$\ell(\mathbf{y}, f(\mathbf{x})) = \max_{\mathbf{r}'\in\Omega} \left[(\mathbf{r}' - \mathbf{r})^\top f(\mathbf{x}) + \Delta(\mathbf{y}, \mathbf{r}')\right]\,, \tag{5}$$

where $\mathbf{r}$ is a rank vector consistent with $\mathbf{y}$ and Ω is the set of possible rank vectors,

then the loss function $\ell(\mathbf{y}, f(\mathbf{x}))$ provides a convex upper bound over $\Delta(\mathbf{y}, f(\mathbf{x}))$.

Proof. It is obvious that the function $\ell(\mathbf{y}, f(\mathbf{x}))$ is convex in f, because it is pointwise maximum of a set of linear functions. For set based performance measure, let $\hat{\mathbf{y}} = \mathrm{sign}[f(\mathbf{x})]$ which is the maximizer of $\mathbf{y}^\top f(\mathbf{x})$ in $\mathcal{Y}$, we can get

$$\ell(\mathbf{y}, f(\mathbf{x})) \geq \hat{\mathbf{y}}^\top f(\mathbf{x}) - \mathbf{y}^\top f(\mathbf{x}) + \Delta(\mathbf{y}, \hat{\mathbf{y}}) \geq \Delta(\mathbf{y}, \hat{\mathbf{y}})\,.$$

For set based performance measures, it holds $\Delta(\mathbf{y}, \hat{\mathbf{y}}) = \Delta(\mathbf{y}, f(\mathbf{x}))$, thus it is an upper bound.

With respect to ranking based performance measure, let $\hat{\mathbf{r}}$ be the rank vector determined by $f(\mathbf{x})$, it is easy to find that $\hat{\mathbf{r}}$ is the maximizer of $\mathbf{r}^\top f(\mathbf{x})$, and then

$$\ell(\mathbf{y}, f(\mathbf{x})) \geq \hat{\mathbf{r}}^\top f(\mathbf{x}) - \mathbf{r}^\top f(\mathbf{x}) + \Delta(\mathbf{y}, \hat{\mathbf{r}}) \geq \Delta(\mathbf{y}, \hat{\mathbf{r}}) \ .$$

Based on above, we can get the conclusion. ■

For the optimization problem in (3), by replacing Δ with its upper bound ℓ, and $\|\mathbf{w}\|_0$ with its continuous relaxation $\|\mathbf{w}\|_1$, we obtain the optimization problem of multi-label selective ensemble as

$$\min_{\mathbf{w}} \ \sum_{i=1}^{m} \ell(\mathbf{y}_i, \mathrm{H}_i\mathbf{w}) + \lambda\|\mathbf{w}\|_1 \ , \tag{6}$$

where

$$\mathrm{H}_i = [h^{(1)}(\mathbf{x}_i), \cdots, h^{(k)}(\mathbf{x}_i)] \in \mathbb{R}^{l\times k} \tag{7}$$

is the matrix collecting the predictions of $\{h^{(t)}\}_{t=1}^k$ on instance $\mathbf{x}_i$, and λ is the regularization parameter trading off the empirical risk and the sparsity of $\mathbf{w}$. Obviously, this is an ℓ_1-regularized convex optimization problem, and we solve it via stochastic optimization subsequently.

2.3 Stochastic Optimization

To solve the ℓ_1-regularized convex problem (6), we employ the state-of-the-art stochastic optimization algorithm presented in [21], and the key is how to compute the subgradient of the loss function $\ell(\mathbf{y}_i, \mathrm{H}_i\mathbf{w})$.

Proposition 2 *Given an example $(\mathbf{x}_i, \mathbf{y}_i)$, a set of multi-label classifiers $\{h^{(t)}\}_{t=1}^k$, a weighing vector $\mathbf{w}_0 \in \mathbb{R}^k$ and a multi-label performance measure Δ, denote*

$$\mathbf{p}_i = \mathrm{H}_i\mathbf{w}_0$$

be the ensemble's prediction on example $(\mathbf{x}_i, \mathbf{y}_i)$ with H_i defined in (7)

(a) if Δ is a set based performance measure, let $\mathbf{g} = (\tilde{\mathbf{y}} - \mathbf{y}_i)^\top \mathrm{H}_i$, where

$$\tilde{\mathbf{y}} = \operatorname*{argmax}_{\mathbf{y}'\in\mathcal{Y}} \left[\mathbf{y}'^\top \mathbf{p}_i + \Delta(\mathbf{y}_i, \mathbf{y}')\right] \ , \tag{8}$$

(b) if Δ is a ranking based performance measure, let $\mathbf{g} = (\tilde{\mathbf{r}} - \mathbf{r}_i)^\top \mathrm{H}_i$, where $\mathbf{r}$ is a rank vector consistent with $\mathbf{y}_i$ and

$$\tilde{\mathbf{r}} = \operatorname*{argmax}_{\mathbf{r}'\in\Omega} \left[\mathbf{r}'^\top \mathbf{p}_i + \Delta(\mathbf{y}, \mathbf{r}')\right] \ , \tag{9}$$

then the vector $\mathbf{g}$ is a subgradient of $\ell(\mathbf{y}_i, \mathrm{H}_i\mathbf{w})$ at $\mathbf{w}_0$.

Algorithm 1 Stochastic optimization algorithm for MUSE

Input: training data $S = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^m$
component classifiers $\{h^{(t)}\}_{t=1}^k$
performance measure $\Delta(\cdot, \cdot)$
regularization parameter λ, step size η

Procedure:

1: let $\mathbf{w} = 0$, $\boldsymbol{\varrho} = 0$ and $p = 2 \ln k$
2: **repeat**
3: select $(\mathbf{x}_i, \mathbf{y}_i)$ uniformly at random from S
4: let $\mathrm{H}_i = [h^{(1)}(\mathbf{x}_i), \cdots, h^{(k)}(\mathbf{x}_i)]$ and $\mathbf{p}_i = \mathrm{H}_i \mathbf{w}$
5: solve the argmax problem, *i.e.*,
$\tilde{\mathbf{y}} \leftarrow$ solve (8) for set based performance measure Δ, or
$\tilde{\mathbf{r}} \leftarrow$ solve (9) for ranking based performance measure Δ
6: compute the sub-gradient, *i.e.*,
$\mathbf{g} = (\tilde{\mathbf{y}} - \mathbf{y}_i)^\top \mathrm{H}_i$ for set based performance measure Δ, or
$\mathbf{g} = (\tilde{\mathbf{r}} - \mathbf{r}_i)^\top \mathrm{H}_i$ for ranking based performance measure Δ
7: let $\widetilde{\boldsymbol{\varrho}} = \boldsymbol{\varrho} - \eta \mathbf{g}$
8: let $\forall t$, $\varrho_t = \mathrm{sign}(\tilde{\varrho}_t) \max(0, |\tilde{\varrho}_t| - \eta\lambda)$
9: let $\forall t$, $w_t = \mathrm{sign}(\varrho_t) |\varrho_t|^{p-1} / \|\boldsymbol{\varrho}\|_p^{p-2}$
10: **until** convergence

Output: weighting vector $\mathbf{w}$

Proof. Since $\ell(\mathbf{y}_i, \mathrm{H}_i\mathbf{w})$ is a pointwise maximum of linear functions in $\mathbf{w}$, it is straightforward to obtain its subgradient if the maximizer of (4) or (5) at $\mathbf{w}_0$ can be obtained. Obviously, the argmax (8) and (9) solve the maximizers for set and ranking based performance measures respectively, which completes the proof. ■

This proposition provides a method to compute the subgradient of $\ell(\mathbf{y}_i, \mathrm{H}_i\mathbf{w})$. Based on this, we can present the stochastic optimization method for solving the optimization problem (6), which is summarized in Algorithm 1. At each iteration, this algorithm first samples an example $(\mathbf{x}_i, \mathbf{y}_i)$ uniformly at random from data S, and then compute the subgradient of $\ell(\mathbf{y}_i, \mathrm{H}_i\mathbf{w})$ (lines 4-6). Since the example $(\mathbf{x}_i, \mathbf{y}_i)$ is chosen at random, the vector $\mathbf{g}$ is an unbiased estimate of the gradient of the empirical risk $\sum_{i=1}^m \ell(\mathbf{y}_i, \mathrm{H}_i\mathbf{w})$. Next, the dual vector $\boldsymbol{\varrho}$ is updated with step size η (line 7) so that the empirical risk is decreased; and also it is truncated to decrease the regularizer $\lambda\|\mathbf{w}\|_1$ (line 8). Finally, the updates of $\boldsymbol{\varrho}$ is translated to the variable $\mathbf{w}$ via a link function in line 9. This procedure iterates until convergence.

Solving the argmax In order to make Algorithm 1 practical, the argmax in (8) and (9) need to be solved for the concerned performance measure. Fortunately, there have been proposed efficient procedures for many commonly used performance measures. For examples, Joachims [12] solved the argmax problem in $O(l^2)$ time for a large class of set based measures including Hamming loss and F1-score, in $O(l \log l)$ time for ranking loss; Yue *et al.* [29] solved it for average precision in $O(l \log l)$ time; Le *et al.* [13]

Algorithm 2 Solve the argmax in (9) for coverage

Input: true label vector $\mathbf{y}$, current prediction $\mathbf{p}$
Procedure:
1: let $max\_val = -\inf$
2: **for** $t \in \{t \mid y_t = 1\}$ **do**
3: let $\mathbf{v} = \mathbf{p} - \mathbf{e}_t$
4: $\mathbf{r} \leftarrow$ obtain rank vector by sorting $\mathbf{v}$ ascendingly
5: **if** $\mathbf{r}^\top \mathbf{v} > max\_val$ **then**
6: let $\tilde{\mathbf{r}} = \mathbf{r}$ and $max\_val = \mathbf{r}^\top \mathbf{v}$
7: **end if**
8: **end for**
Output: rank vector $\tilde{\mathbf{r}}$

solved (9) by a linear assignment problem for a group of ranking based performance measures including precision@k. Of course, if our concerned performance measure is among them, these procedures can be directly employed by Algorithm 1. Here, we omit the detailed procedures, which can be found in [12, 13, 29].

To our best knowledge, there is still no proposal to solve the argmax (9) for *coverage*, which is ranking based performance measure evaluating how far one needs to go along the list of labels to cover all the true labels. Formally, given a true label vector $\mathbf{y}$ and a rank vector $\mathbf{r}$, it is defined as

$$\Delta_c(\mathbf{y}, \mathbf{r}) = \max_{\{t|y_t=1\}} (l - r_t) = \max_{t \in \{t|y_t=1\}} (l - \mathbf{r}^\top \mathbf{e}_t) \,, \tag{10}$$

where $\mathbf{e}_t$ is a vector with t-th element as 1 and others 0. Substituting (10) into the argmax problem (9), we can get an equivalent problem as

$$\tilde{\mathbf{r}} = \operatorname*{argmax}_{\mathbf{r}' \in \Omega} \max_{t \in \{t|y_t=1\}} \mathbf{r}'^\top (\mathbf{p}_i - \mathbf{e}_t) \,. \tag{11}$$

It is not difficult to see that for one single t, the problem

$$\operatorname*{argmax}_{\mathbf{r}' \in \Omega} \ \mathbf{r}'^\top (\mathbf{p}_i - \mathbf{e}_t) \tag{12}$$

can be efficiently solved by sorting the vector $(\mathbf{p}_i - \mathbf{e}_t)$ ascendingly, and obtaining the corresponding rank vector. As a consequence, by enumerating all t's and solving the corresponding problems as (12), the solution to (11) can be obtained. The pseudocode of this procedure is given in Algorithm 2. We can find that in each iteration, the complexity is dominated by the sorting in line 4, thus the total complexity of Algorithm 2 is $O(sl \log l)$, where s is the number of true labels of current example.

Convergence and computational complexity Based on Theorem 3 in [21], we can find that the number of iterations of Algorithm 1 to achieve ϵ-accuracy is bounded by $O(\log k/\epsilon^2)$ with k as the number of component classifiers. It can be found that this

number is independent of the data size. Moreover, in each iteration, all the operations are performed on one single example, and the complexity is dominated by the argmax which as shown above can be solved in polynomial time for various performance measures, also independent of data size. This constitutes one of the appealing properties of MUSE, i.e., at each iteration of Algorithm 1, we neither need to compute the predictions on all examples nor need to solve the argmax on all examples.

3 Experiments

In this section, we perform a set of experiments to evaluate the effectiveness of our proposed MUSE approach.

3.1 Configuration

The experiments are performed on image and music annotation tasks. Specifically, two image annotation tasks are used, including *corel5k* which has 5000 images and 374 possible labels, and *scene* which has 2407 images and 6 possible labels; two music annotation tasks are used, including *cal500* which has 502 songs and 174 possible labels, and *emotion* which has 593 songs and 6 possible labels. Five representative multi-label performance measures are considered for each task, including Hamming loss, precision@k, F1-score, coverage, ranking loss. The formal definition of these performance measures can be found in [20], and the k of precision@k is set to the average number of relevant labels. In total, there are 20 tasks.

In experiments, MUSE is implemented based on ECC [19], that is, by using ECC, we first obtain 100 classifier chains, then use MUSE to obtain the selective ensemble out of them. We compare MUSE with BSVM [1] which trains one SVM for each label, and state-of-the-art methods including the lazy method ML-kNN [31], label ranking method CLR [6] and the full ECC combining all classifiers. It is also compared with the *random strategy* which selects classifiers randomly. Specifically, LibLinear [5] is used to implement the base classifier in BSVM and ECC; the default implementation of ML-kNN and CLR in Mulan [24] are used; the random strategy generates ensembles of the same size of MUSE.

For each task, these comparative methods are evaluated by using 30 times random holdout test, i.e., 2/3 for training and 1/3 for testing in each time; finally, averaged performance and standard derivation are reported; the sizes of the generated selective ensembles are reported. For MUSE, the regularization parameter λ is chosen by 5-fold cross validation on training set.

3.2 Results

Optimizing the upper bound Instead of optimizing the concerned performance measures directly, the proposed MUSE approach tries to optimize its convex upper bound. Thus, a natural question is whether this is effective, or in other words, whether optimizing the upper bound will improve the performance. To answer this question, we record the training and test performance on *cal500*, and the results are shown in Fig.2. It can

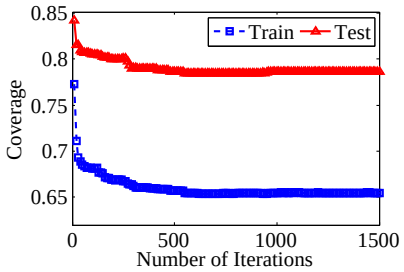

Fig. 2. Both the training and test performance improve during the optimization procedure, where coverage is evaluated on the cal500 data set.

be found that both the training and test performance improve when the optimization procedure goes on, which give a positive answer to above question, that is, optimizing the upper bound is effective in improving the performance. Moreover, we can see from Fig.2 that the performance converges after some iterations. For example, they converge after about 600 iterations for coverage. Noting there are 502 examples in total, and in each iteration, MUSE operates on only one example, this means that we need to scan the data set for only once.

Performance comparison The performance of all the comparative methods are shown in Table 1. For better comparison, we perform paired t-tests at 95% significance level to compare MUSE with other methods, and the results are also shown in Table 1.

It can be seen that the performance of MUSE is quite promising. Compared with the full ensemble ECC_{100}, it achieves 3 wins and 16 ties and loses only 1 time out of all 20 tasks, while the ensemble size is reduced. For example, on *cal500* the F1-score is improved from 0.323 to 0.384, but the ensemble size reduced from 100 to less than 20. Also, when compared with the random strategy, MUSE achieves 9 wins but 0 loss, which shows its effectiveness. Comparing MUSE with other methods, we can see that it achieves significantly better performance (15 wins and 0 loss) against BSVM, also comparable performance against the state-of-the-art methods ML-kNN and CLR.

4 Related Work

In ensemble learning, selective ensemble (*a.k.a.* ensemble pruning or ensemble selection) is an active research topic [34, chapter 6]. In traditional supervised learning, a number of methods have been developed based on different techniques, such as genetic algorithm [35], semi-definite programming [33], clustering [9], ℓ_1-norm regularized sparse optimization [14]. In [15], in order to reduce the size of ECC, Li and Zhou proposed SECC (i.e., selective ensemble of classifier chains), which to our best knowledge is the first work on selective ensemble in the multi-label setting.

In this paper, we address the problem of multi-label selective ensemble by proposing that it is needed to take the concerned performance measure fully into account, and propose the MUSE approach to build the selective ensemble via sparse convex optimization. This is encouraged and inspired by recent works on optimizing complicated

Table 1. Experimental results (mean±std.), where •(○) indicates that MUSE is significantly better (worse) than the corresponding method based on paired t-tests at 95% significance level, the sizes of selective ensembles generated by MUSE are reported (after '/'), and the win/tie/loss counts based on paired t-tests are summarized in the last row. Note that the ensemble size of ECC_{100} is 100, and the Random strategy generates ensembles of the same size of MUSE.

data	BSVM	ML-kNN	CLR	ECC_{100}	Random	MUSE
Hamming loss (the smaller, the better)						
corel5k	.014±.001•	.009±.001	.498±.003•	.010±.001	.010±.002	.010±.001/43.9±12.2
scene	.120±.003•	.089±.004○	.174±.004•	.102±.003	.180±.007•	.101±.003/66.2±18.4
cal500	.212±.012•	.139±.002	.378±.002•	.141±.002	.145±.005•	.137±.002/62.9± 7.4
emotion	.305±.011	.201±.012○	.190±.012○	.302±.012	.312±.012•	.297±.012/36.3±15.9
Precision@k (the larger, the better)						
corel5k	.177±.005•	.213±.005•	.232±.005	.234±.006	.227±.009	.231±.006/63.3±12.6
scene	.451±.006•	.472±.006	.453±.007•	.465±.005	.460±.013•	.469±.001/83.0± 7.8
cal500	.315±.025•	.447±.006	.453±.006	.452±.007	.438±.010	.446±.007/74.3± 7.2
emotion	.605±.017	.618±.020	.582±.016•	.609±.022	.603±.022	.607±.021/53.2±20.1
F1-score (the larger, the better)						
corel5k	.135±.005•	.016±.003•	.034±.001•	.134±.006•	.127±.008•	.149±.006/15.3± 5.6
scene	.595±.014•	.675±.018	.629±.009•	.668±.012	.567±.014•	.672±.002/22.0±10.8
cal500	.314±.029•	.322±.011•	.405±.003○	.323±.010•	.333±.023•	.384±.013/19.6± 4.7
emotion	.614±.014	.602±.029•	.622±.016	.618±.015	.612±.016	.619±.017/63.7±21.2
Coverage (the smaller, the better)						
corel5k	.560±.008•	.309±.004○	.285±.005○	.362±.007	.367±.011	.363±.011/74.1±13.7
scene	.102±.004	.090±.004	.102±.006	.096±.004	.097±.008	.097±.005/76.3±13.2
cal500	.913±.015•	.750±.010○	.756±.008○	.768±.013○	.796±.002•	.790±.002/41.8± 9.7
emotion	.322±.015	.319±.013	.323±.014	.328±.015	.334±.019	.326±.017/73.6±12.5
Ranking loss (the smaller, the better)						
corel5k	.270±.005•	.135±.002•	.119±.002•	.042±.001•	.039±.005•	.032±.003/71.3±11.7
scene	.106±.005•	.082±.005	.079±.005	.087±.004	.087±.007	.086±.004/52.4± 9.7
cal500	.309±.019•	.184±.003	.182±.003	.184±.005	.179±.005	.188±.006/54.9±12.4
emotion	.194±.012•	.169±.015○	.146±.013○	.187±.013	.187±.016	.188±.011/74.1±11.7
Win/Tie/Loss counts (MUSE vs alternatives)						
counts	15/5/0	5/11/4	8/9/3	3/16/1	9/11/0	–

performance measures [12, 17, 15]. Moreover, compared with previous work [15], the MUSE approach is more general, for instance, it can optimize a large variety of performance measures while SECC considers only F1-score. In other words, SECC is a special case of MUSE when it considers only F1-score.

5 Conclusion

In this paper, we study the problem of multi-label selective ensemble, which tries to select a subset of component classifiers whilst keeping or improving the performance. The main motivation is that we need to take the concerned performance measure into account during the selection process, and the MUSE approach is proposed to handle this problem. Specifically, by taking an upper bound over empirical risk, MUSE tries to optimize the concerned performance measure via an ℓ_1-norm regularized convex opti-

mization problem. And this problem can be efficiently solved by stochastic subgradient descend for a large variety of performance measures. Experiments on image and music annotation tasks show the effectiveness of the proposed MUSE approach.

In current work, we consider the component classifier in multi-label ensemble as a general multi-label classifier. Often, the component classifier itself is a group of single-label classifiers, like classifier chain in ECC. Therefore, it will be interesting to consider the multi-label selective ensemble problem at the level of such single-label classifiers.

Acknowledgements: We want to thank anonymous reviewers for helpful comments. This research was supported by the National Science Foundation of China (61273301).

References

1. M. Boutell, J. Luo, X. Shen, and C. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004.
2. S. S. Bucak, R. Jin, and A. Jain. Multi-label learning with incomplete class assignments. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2801–2808, Colorado Springs, CO, 2011.
3. K. Dembczynski, W. Cheng, and E. Hüllermeier. Bayes optimal multilabel classification via probabilistic classifier chains. In *Proceedings of the 27th International Conference on Machine Learning*, pages 279–286, Haifa, Israel, 2010.
4. K. Dembczynski, W. Waegeman, W. Cheng, and E. Hüllermeier. Regret analysis for performance metrics in multi-label classification. In *Proceedings of the 21st European Conference on Machine Learning*, pages 280–295, Barcelona, Spain, 2010.
5. R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.
6. J. Fürnkranz, E. Hüllermeier, E. Loza Mencía, and K. Brinker. Multilabel classification via calibrated label ranking. *Machine Learning*, 73(2):133–153, 2008.
7. W. Gao and Z.-H. Zhou. On the consistency of multi-label learning. *Artificial Intelligence*, 199-200:22–44, 2013.
8. N. Ghamrawi and A. McCallum. Collective multi-label classification. In *Proceedings of the 14th ACM international conference on Information and Knowledge Management*, pages 195–200, Bremen, Germany, 2005.
9. G. Giacinto, F. Roli, and G. Fumera. Design of effective multiple classifier systems by clustering of classifiers. In *Proceedings of the 15th International Conference on Pattern Recognition*, pages 160–163, Barcelona,Spain, 2000.
10. B. Hariharan, L. Zelnik-Manor, S. Vishwanathan, and M. Varma. Large scale max-margin multi-label classification with priors. In *Proceedings of the 27th International Conference on Machine Learning*, pages 423–430, Haifa, Israel, 2010.
11. D. Hsu, S. Kakade, J. Langford, and T. Zhang. Multilabel prediction via compressed sensing. In *Advances in Neural Information Processing Systems 22*, pages 772–780. MIT Press, Cambridge, MA, 2009.
12. T. Joachims. A support vector method for multivariate performance measures. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 377–384, Bonn, Germany, 2005.
13. Q. Le and A. Smola. Direct optimization of ranking measures. *CoRR*, abs/0704.3359, 2007.
14. N. Li and Z.-H. Zhou. Selective ensemble under regularization framework. In *Proceedings of the 8th International Workshop on Multiple Classifier Systems*, pages 293–303, Reykjavik, Iceland, 2009.

15. N. Li and Z.-H. Zhou. Selective ensemble of classifier chains. In *Proceedings of the 11th International Workshop on Multiple Classifier Systems*, pages 146–156, Nanjing, China, 2013.
16. A. McCallum. Multi-label text classification with a mixture model trained by EM. In *Working Notes of AAAI99 Workshop on Text Learning*, 1999.
17. Y. Nan, K. M. Chai, W. Lee, and H. Chieu. Optimizing F-measure: A tale of two approaches. In *Proceedings of the 29th International Conference on Machine Learning*, pages 289–296, Edinburgh, UK, 2012.
18. J. Read, B. Pfahringer, and G. Holmes. Multi-label classification using ensembles of pruned sets. In *Proceedings of the 8th IEEE International Conference on Data Mining*, pages 995–1000, Pisa, Italy, 2008.
19. J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. *Machine Learning*, 85(3):333–359, 2011.
20. R. Schapire and Y. Singer. BoosTexter: A boosting-based system for text categorization. *Machine Learning*, 39(2-3):135–168, 2000.
21. S. Shalev-Shwartz and A. Tewari. Stochastic methods for ℓ_1-regularized loss minimization. *Journal of Machine Learning Research*, 12:1865–1892, 2011.
22. C. Shi, X. Kong, P. Yu, and B. Wang. Multi-label ensemble learning. In *Proceedings of the 22nd European Conference on Machine learning*, pages 223–239, Athens, Greece, 2011.
23. I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.
24. G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, and I. Vlahavas. MULAN: A Java library for multi-label learning. *Journal of Machine Learning Research*, 12:2411–2414, 2011.
25. G. Tsoumakas and I. Vlahavas. Random k-labelsets: An ensemble method for multilabel classification. In *Proceedings of the 18th European Conference on Machine Learning*, pages 406–417, Warsaw, Poland, 2007.
26. D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Semantic annotation and retrieval of music and sound effects. *IEEE Transactions on Audio, Speech and Language Processing*, 16(2):467–476, 2008.
27. N. Ueda and K. Saito. Parametric mixture models for multi-labeled text. In *Advances in Neural Information Processing Systems 15*, pages 721–728. MIT Press, Cambridge, MA, 2003.
28. M. Xu, Y.-F. Li, and Z.-H. Zhou. Multi-label learning with pro loss. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence*, pages 998–1004, Bellevue, WA, 2013.
29. Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 271–278, Amsterdam, Netherlands, 2007.
30. M.-L. Zhang and K. Zhang. Multi-label learning by exploiting label dependency. In *Proceedings of the 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 999–1007, Washington, DC, 2010.
31. M.-L. Zhang and Z.-H. Zhou. ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40(7):2038–2048, 2007.
32. M.-L. Zhang and Z.-H. Zhou. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1819–1837, 2014.
33. Y. Zhang, S. Burer, and W. Street. Ensemble pruning via semi-definite programming. *Journal of Machine Learning Research*, 7:1315–1338, 2006.
34. Z.-H. Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC, Boca Raton, FL, 2012.
35. Z.-H. Zhou, J. Wu, and W. Tang. Ensembling neural networks: Many could be better than all. *Artificial Intelligence*, 137(1-2):239–263, 2002.