Selective Ensemble Under Regularization Framework

Nan Li and Zhi-Hua Zhou

National Key Laboratory for Novel Software Technology,
Nanjing University, Nanjing 210093, China
{lin,zhouzh}@lamda.nju.edu.cn

Abstract. An ensemble is generated by training multiple component learners for a same task and then combining them for predictions. It is known that when lots of trained learners are available, it is better to ensemble some instead of all of them. The selection, however, is generally difficult and heuristics are often used. In this paper, we investigate the problem under the regularization framework, and propose a regularized selective ensemble algorithm RSE. In RSE, the selection is reduced to a quadratic programming problem, which has a sparse solution and can be solved efficiently. Since it naturally fits the semi-supervised learning setting, RSE can also exploit unlabeled data to improve the performance. Experimental results show that RSE can generate ensembles with small size but strong generalization ability.

1 Introduction

Ensemble learning [11] is a learning paradigm where multiple component learners are trained to solve a problem. Since an ensemble often has better performance than a single learner, it has achieved successes in many domains.

In general, an ensemble is built in two steps, i.e., training multiple component learners and then combining them. According to the styles of training component learners, popular ensemble algorithms can be roughly categorized into two classes, that is, approaches where component learners are trained in parallel, and approaches where component learners must be trained sequentially. Representatives of the former include Bagging [4], Random Subspace [14], Random Forest [6], GASEN [24], etc. Representatives of the latter include AdaBoost [13], Arc-x4 [5], LPBoot [10], etc.

In most ensemble algorithms, all of the obtained component learners are employed to build an ensemble. However, some researchers [24, 23, 17] show that through *selective ensemble*, i.e., ensembling some instead of all the available component learners, a better ensemble can be generated. The selection, however, is not easy and thus many heuristics have been used. For example, for the selection, Zhou et al. [24, 23] used a genetic algorithm; Castro et al. [7] employed artificial immune algorithm; Coyle and Smith [9] utilized case similarity; Martínez-Muñoz and Suárez [17] proposed to order the component learners and then select the first ones to use.

As a generic learning framework, regularized approaches [18] work by minimizing the regularized risk function, and have been found useful in ensemble learning. For example, LPBoost [10] takes margin as regularizer; RegBoost [15] utilizes the graph Laplacian regularizer [2] to make base classifiers cut through sparse regions; in [8], the regularizer is used to take local smoothness constraints, yielding a more general regularized Boosting.

In contrast to previous heuristic methods, in this paper we study the selective ensemble problem under the regularization framework. We utilize the hinge loss and the graph Laplacian regularizer, and present a regularized selective ensemble approach RSE. In RSE, the selection problem is reduced to a quadratic programming (QP) problem, which has a sparse solution and meanwhile can be solved efficiently. Empirical study shows that RSE is able to generate ensembles with small size while have strong generalization ability, which is superior to many existed ensemble methods.

In practical applications, unlabeled examples are much easier to obtain. Therefore, semi-supervised learning [25], which attempts to exploit unlabeled examples to help improve learning performance, has attracted much attention. A prominent advantage of RSE is that it naturally fits the semi-supervised setting. Experiments show that RSE can exploit unlabeled data effectively.

In the following of the paper we will start with a brief review on related work. Then, we propose RSE and its semi-supervised extension, followed by reports on experiments. Finally, we conclude the paper.

2 Related Work

After obtaining the component learners, most ensemble algorithms combine all of them to build an ensemble, however, it has been shown that it is better to ensemble some instead of all of them [24, 23, 17].

Zhou et al. [24] analyzed the relationship between ensemble and its component learners from the context of both regression and classification, and proved that it may be better to combine many instead of all of the learners. Since it is difficult to select the component learners, they used genetic algorithm to select a part of learners to build the ensemble. Empirical studies show that, comparing with some popular ensemble approaches such as Bagging and Boosting, the proposed GASEN algorithm can generate ensembles with smaller sizes but stronger generalization ability.

Martínez-Muñoz and Suárez [17] proposed a heuristic method, where the component learners obtained from bagging are reordered, and a part of the top-ranked ones are included in the final ensemble. The experimental result also shows that selective ensemble may improve ensemble's performance while reducing its size.

Selective ensemble is a special paradigm of ensemble learning, which aims at building strong ensembles with small sizes. In some sense it is related to ensemble pruning [16, 22]. However, it is noteworthy that in earlier researches of ensemble prunning [16], the goal was to use a small size of ensemble to achieve an equivalent performance of a boosted ensemble. This has been proved to be NP-hard and is even hard to approximate [19], and the pruning may sacrifice the generalization ability of the final ensemble. Since Zhou et al.'s work [24], it is known that by using selective ensemble it is possible to get a small yet strong ensemble. What makes this difference is that earlier ensemble pruning works on ensembles where the component learners must be trained sequentially, while selective ensemble works on ensembles where the component learners can be generated in parallel, and the sequential generation of the former makes the problem much more difficult to tackle.

3 The RSE Approach

Let $\mathcal{X}$ and $\mathcal{Y}$ denote the feature space and the set of class labels, respectively. Here, binary classification is considered for simplicity, assuming $\mathcal{Y} = \{-1, +1\}$. A training set $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$ is given, where $\boldsymbol{x}_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$.

From training set $\mathcal{D}$, ensemble learning algorithms try to train a set of component classifiers $\{h_1, \ldots, h_M\}$ and choose weights $\{w_1, \ldots, w_M\}$ to combine them as $H(\boldsymbol{x}) = \sum_{i=1}^{M} w_i h_i(\boldsymbol{x})$, and it is often assumed that $w_i \geq 0$ and $\sum_{i=1}^{M} w_i = 1$. The classification decision of the ensemble H on instance $\boldsymbol{x}$ is $+1$ if $H(\boldsymbol{x}) \geq 0$ and -1 otherwise. The number of component classifiers is called the *size* of the ensemble. It is obvious that classifiers with zero weights will be excluded from the ensemble.

In this work, we concern on the second step of building an ensemble, i.e., choosing weights to combine the component classifiers. In this section, we study the selective ensemble problem under the regularization framework.

3.1 Regularized Risk Function

Let $\boldsymbol{w} = [w_1, \ldots, w_M]^\top$ denote the weights used to combine the component classifiers $\{h_1, \ldots, h_M\}$. The ensemble's output on instance $\boldsymbol{x}_i$ is

$$H(\boldsymbol{x}_i) = \sum_{k=1}^{M} w_k h_k(\boldsymbol{x}_i) = \boldsymbol{p}_i^\top \boldsymbol{w}, \tag{1}$$

where $\boldsymbol{p}_i = [h_1(\boldsymbol{x}_i), \ldots, h_M(\boldsymbol{x}_i)]^\top$ are the component classifiers' predictions on $\boldsymbol{x}_i$.

Under the regularization framework, the weights $\boldsymbol{w}$ is usually determined by minimizing the regularized risk function

$$R(\boldsymbol{w}) = \lambda\, V(\boldsymbol{w}) + \Omega(\boldsymbol{w}), \tag{2}$$

where $V(\boldsymbol{w})$ is the empirical loss which approximately measures the misclassification loss of classifier on the training examples in $\mathcal{D}$, $\Omega(\boldsymbol{w})$ is the regularization term, and λ is the regularization parameter which specifies the tradeoff between the minimization of $V(\boldsymbol{w})$ and the smoothness or simplicity enforced by minimization of $\Omega(\boldsymbol{w})$.

Empirical Loss The empirical loss considers the loss between each example's class label y_i and its corresponding prediction $H(\boldsymbol{x}_i) = \boldsymbol{p}_i^\top \boldsymbol{w}$.

Here, we use the *hinge loss* function $\ell\big(y_i, H(\boldsymbol{x}_i)\big) = \max\big(0, 1 - y_i H(\boldsymbol{x}_i)\big)$. Typically, this is the empirical loss minimized in support vector machines [18], and it is usually written as a sum of slack variables included in the constraints. Afterwards, we can define the empirical loss function $V(\boldsymbol{w})$ for the ensemble H on training set $\mathcal{D}$ as

$$V(\boldsymbol{w}) = \sum_{i=1}^{N} \ell\big(y_i, H(\boldsymbol{x}_i)\big) = \sum_{i=1}^{N} \max(0, 1 - y_i \boldsymbol{p}_i^\top \boldsymbol{w}), \tag{3}$$

where, as same as previous definition, $\boldsymbol{w}$ are the weights, and $\boldsymbol{p}_i$ are predictions on $\boldsymbol{x}_i$. Obviously, the empirical loss function in Eq. 3 is convex and continuous.

Regularization Term The regularization term $\Omega(\boldsymbol{w})$ is used to smooth or simplify the function. In this work, the *graph Laplacian regularizer* [2] is adopted.

Let $G = (\mathcal{V}; \mathcal{E})$ be the neighborhood graph of the training set $\mathcal{D}$, where the vertex set $\mathcal{V}$ is the set of all the examples in $\mathcal{D}$, and the edge set $\mathcal{E}$ contains pairs of neighboring examples. Based on the idea that the classifier should make similar predictions on neighboring examples, the classifier that cut through dense regions is penalized by

$$P_{\mathcal{L}}(H) = \sum\nolimits_{i=1}^{N} \sum\nolimits_{j=i+1}^{N} W_{ij} \big(H(\boldsymbol{x}_i) - H(\boldsymbol{x}_j)\big)^2, \tag{4}$$

where $\boldsymbol{W}$ is the (weighted) adjacency matrix of G.

Let the diagonal matrix $\boldsymbol{D}$ defined by $D_{ii} = \sum_{j=1}^{N} W_{ij}$, the normalized *graph Laplacian* of G is $\boldsymbol{L} = \boldsymbol{D}^{-1/2}(\boldsymbol{D} - \boldsymbol{W})\boldsymbol{D}^{-1/2}$. Then, the function in Eq. 4 can be rewritten as $P_{\mathcal{L}}(H) = \boldsymbol{h}^\top \boldsymbol{L} \boldsymbol{h}$, where $\boldsymbol{h} = [H(\boldsymbol{x}_1), \ldots, H(\boldsymbol{x}_N)]^\top$ is ensemble H's predictions on the examples. By Eq. 1, it follows that $\boldsymbol{h} = [\boldsymbol{p}_1^\top \boldsymbol{w}, \ldots, \boldsymbol{p}_N^\top \boldsymbol{w}] = \boldsymbol{P}^\top \boldsymbol{w}$, where $\boldsymbol{P} \in \{-1, +1\}^{M \times N}$ is the *prediction matrix* which collects every component classifier's prediction on every example, and $\boldsymbol{P}_{ij} = h_i(\boldsymbol{x}_j)$.

Using the graph Laplacian regularizer, the regularization term is defined as

$$\Omega(\boldsymbol{w}) = \boldsymbol{h}^\top \boldsymbol{L} \boldsymbol{h} = \boldsymbol{w}^\top \boldsymbol{P} \boldsymbol{L} \boldsymbol{P}^\top \boldsymbol{w}, \tag{5}$$

where, as defined above, $\boldsymbol{L}$ is the *graph Laplacian* and $\boldsymbol{P}$ is the *prediction matrix*.

Here, rather than building the neighbor graph on the training set, we assume that the graph is fully connected, and the weight of the edge between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is determined by the distance function $W_{ij} = \exp(-\sigma \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2)$, where σ is a bandwidth parameter.

3.2 Optimization Problem with Sparsity Constraint

Using the empirical loss function defined by Eq. 3 and the graph Laplacian regularization term in Eq. 5, we have the following optimization problem:

$$\begin{aligned} \min_{\boldsymbol{w}} \quad & \boldsymbol{w}^\top \boldsymbol{P} \boldsymbol{L} \boldsymbol{P}^\top \boldsymbol{w} + \lambda \sum\nolimits_{i=1}^{N} \max(0, 1 - y_i \boldsymbol{p}_i^\top \boldsymbol{w}) \\ \text{subject to} \quad & \boldsymbol{1}^\top \boldsymbol{w} = 1, \quad \boldsymbol{w} \geq \boldsymbol{0}. \end{aligned} \tag{6}$$

Since $\max(\cdot, \cdot)$ is non-smooth, Eq. 6 can be equivalently rewritten as the following

$$\begin{aligned} \min_{\boldsymbol{w}} \quad & \boldsymbol{w}^\top \boldsymbol{P} \boldsymbol{L} \boldsymbol{P}^\top \boldsymbol{w} + \lambda\, \boldsymbol{1}^\top \boldsymbol{\xi} \\ \text{subject to} \quad & y_i \boldsymbol{p}_i^\top \boldsymbol{w} + \xi_i \geq 1, \quad i = 1, \ldots, N \\ & \boldsymbol{1}^\top \boldsymbol{w} = 1, \quad \boldsymbol{w} \geq \boldsymbol{0}, \quad \boldsymbol{\xi} \geq \boldsymbol{0} \end{aligned} \tag{7}$$

where $\boldsymbol{\xi} = [\xi_1, \ldots, \xi_N]^\top$ are slack variables, λ is the regularization parameter, $\boldsymbol{1}$ and $\boldsymbol{0}$ are all-1 and all-0 vector, respectively. It is evident that Eq. 7 is a standard QP problem, which can be efficiently solved using many existed optimization packages.

Note that the constraint, $\boldsymbol{1}^\top \boldsymbol{w} = 1, \boldsymbol{w} \geq \boldsymbol{0}$, is ℓ_1-norm constraint on the weights $\boldsymbol{w}$, which is a sparsity constraint that will force some w_i's to be zero. Recall that the ensemble size is equal to the number of non-zero elements in $\boldsymbol{w}$, thus ensemble with small size is encouraged.

3.3 The RSE Algorithm

The RSE algorithm is summarized in Fig. 1.

```
Input:  training set $\mathcal{D}$, component learner $L$, trials $M$
Process:
  1.  for $i = 1$ to $M\{$
  2.      $\mathcal{S}_i$ = bootstrap sample from $\mathcal{D}$ ;
  3.      $h_i = L(\mathcal{S}_i)$ ;   $\}$
  4.  Get the component classifiers' predictions $\boldsymbol{P}$;
  5.  Build the neighbor graph on $\mathcal{D}$, and calculate $\boldsymbol{L}$;
  6.  Solve the QP problem in Eq. 7 to get the weights $\boldsymbol{w}$;
Output:  ensemble $H$
  $H(\boldsymbol{x}) = \sum_{w_i>0} h_i(\boldsymbol{x})$        for selective ensemble (RSE)
  $H(\boldsymbol{x}) = \sum_{w_i>0} w_i h_i(\boldsymbol{x})$     for weighted ensemble (RSE-w)
```

Fig. 1. Pseudo-code of the RSE Algorithm

Note that RSE uses bootstrap sampling to train component classifiers, which is borrowed from Bagging, other methods to train component classifiers can also be used.

3.4 Semi-Supervised Extension

In many real-world applications, it is often the case that abundant unlabeled examples are available. Thus, semi-supervised learning [25], which attempts to exploit unlabeled examples to improve the performance, has attracted much attentions.

Using the unlabeled examples for regularization [2] is one of the important approaches in semi-supervised learning, and *graph Laplacian regularizer* is a representative example. RSE uses the graph Laplacian regularizer, and it naturally fits the semi-supervised learning setting, we call the corresponding semi-supervised version as RSE$_{ss}$, and RSE-w$_{ss}$ corresponds to the semi-supervised version of RSE-w.

Obviously, the graph Laplacian regularizer does not rely on the class labels. So, when given training set $\mathcal{D}$ with L labeled examples and U unlabeled ones, the graph Laplacian regularizer can be derived by Eq. 5, except that the prediction matrix $\boldsymbol{P}$ and the graph Laplacian $\boldsymbol{L}$ should be computed on both labeled and unlabeled examples.

4 Experiments

4.1 Configuration

We use 14 two-classes UCI data sets [3] in experiments. These data sets span a broad range of real domains, and some statistics of the data sets are shown in Table 1.

We compared RSE with Bagging [4], AdaBoost [13], and the first selective ensemble algorithm GASEN [24]. RSE-w is also evaluated. Without loss of generality, C4.5 tree is used as the component classifier. For Bagging and AdaBoost, the ensemble size is 100, while GASEN and RSE select component learners from 100 bagged C4.5 trees. They are all implemented in WEKA [21], and the trees are pruned following the default

Table 1. Experimental data sets.

Date set	#Examples	#Attributes	Date set	#Examples	#Attributes
australian	690	14	*heart-s*	270	13
ballons	76	4	*ionosphere*	351	34
cancer	286	9	*kr-vs-kp*	3196	36
cylinder-b	540	39	*live-dis*	345	6
diabetes	768	8	*spectf*	267	44
germen	1000	20	*spambase*	4601	57
haberman	306	3	*vote*	435	16

settings. The genetic algorithm employed by GASEN is implemented using MATLAB [12], the selection threshold is set to 0.01, and the parameters of genetic algorithm are set to the default values. The bandwidth σ is set to 0.01, and the regularization parameter λ is selected by 5-fold cross validation on training sets for RSE and RSE-w, respectively. The QP problem is solved using MOSEK [1].

For each data set, 50 runs of hold-out tests are executed. In each run, one-third of examples are selected randomly for testing, and the remaining for training. The average predictive error rates and the ensemble sizes are recorded.

In order to study how well RSE$_{ss}$ and RSE-w$_{ss}$ can exploit unlabeled examples, experiments under semi-supervised setting are also carried out, during which the training data set is partitioned into labeled set $\mathcal{L}$ and unlabeled set $\mathcal{U}$ under a *label rate*. To simulate different amount of unlabeled examples, two different label rates, 5% and 10%, are investigated here. Under 5% (10%) label rate, 5% (10%) of the training set is used as labeled training data, while the remaining 95% (90%) as unlabeled data.

4.2 Results

Results under Supervised Setting The predictive errors are shown in Table 2, the last rows present the average error over all data sets, and the summary of pairwise t-tests with 95% significance level, where *W/T/L* means that RSE (RSE-w) wins, ties and loses on *#W*, *#T* and *#L* data sets, respectively. The ensemble sizes are presented in Table 3.

It can be observed from Table 2 that selective ensemble algorithms consistently perform better than single decision trees. Moreover, RSE has the lowest averaged error rate, and t-test results indicate that RSE performs significantly better than all the comparing methods. Compared with the non-selective ensemble algorithms Bagging and AdaBoost, RSE wins on 11 and 9 data sets. respectively. RSE only loses to AdaBoost on *ionosphere* and *spambase*, where AdaBoost performs significantly better and others ensemble methods have similar performance. Compared with GASEN, RSE also performs quite well (wins on 6 data sets and never loses).

It is quite interesting that RSE performs much better than RSE-w which weights the selected classifier by $\boldsymbol{w}$. Similar phenomenon was observed before for GASEN [24]. A possible reason is that the weights of RSE-w were determined by minimizing the hinge loss, which is only an approximation of the 0-1 misclassification loss, while the optimal weights for the hinge loss may not be optimal for the 0-1 loss.

It can be found from Table 3 that the ensemble sizes of RSE and RSE-w are much smaller than that of GASEN. Since different regularization parameters are selected via cross validation, RSE and RSE-w select 19.1 and 19.2 trees, respectively, while GASEN selects 41.2 trees on average.

Table 2. Comparison of the predictive errors (mean±std) under supervised setting, where the best performance on each data set is bolded. The *average* row presents the results averaged over all data sets; *W/T/L* row summarizes the comparison of RSE (RSE-w) against other algorithms according to pairwise t-tests with 95% significance level.

Data set	C4.5	Bagging	AdaBoost	GASEN	RSE-w	RSE
australian	.148±.021	.133±.018	.137±.015	.131±.017	.131±.018	**.127**±.016
balloons	.316±.067	.269±.089	.243±.077	.262±.079	**.223**±.069	.241±.074
cancer	.278±.035	.267±.027	.343±.049	.273±.029	.272±.020	**.266**±.018
cylinder-b	.328±.021	.328±.021	.328±.020	.322±.020	.337±.045	**.319**±.033
diabetes	.267±.024	**.241**±.024	.265±.023	**.241**±.021	.246±.018	.243±.019
germen	.281±.018	.249±.019	.258±.025	.248±.019	.249±.017	**.245**±.017
haberman	.285±.040	.266±.031	.308±.036	.267±.029	.260±.013	**.256**±.020
heart-s	.217±.041	.181±.034	.188±.029	.177±.031	.175±.030	**.168**±.031
ionosphere	.110±.030	.077±.023	**.062**±.023	.075±.022	.074±.021	.070±.021
kr-vs-kp	.008±.003	.007±.003	**.005**±.003	.007±.002	.006±.002	.006±.002
liver-dis	.352±.036	.283±.035	.317±.046	.283±.033	.276±.037	**.269**±.034
spectf	.173±.036	.125±.028	.128±.029	.125±.028	.129±.027	**.122**±.028
spambase	.079±.008	.060±.007	**.050**±.006	.059±.007	.059±.007	.059±.007
vote	.047±.016	.042±.015	.056±.017	.042±.014	.040±.012	**.038**±.012
average	0.206	0.183	0.189	0.179	0.176	**0.171**
W/T/L (RSE)	—	11/3/0	9/3/2	6/8/0	8/5/1	—
W/T/L (RSE-w)	—	3/11/0	9/2/3	3/9/2	—	1/5/8

Table 3. Comparison of the ensemble size (mean±std) under supervised setting, where the smallest size on each data set is bolded. The *average* row presents the size averaged over all data sets.

Date set	GASEN	RSE-w	RSE
australian	45.3±3.0	20.5±5.7	**19.1**±5.7
balloons	38.9±**15.8**	15.8±14.6	15.9±18.9
cancer	41.0±4.2	**11.9**±3.6	13.0±4.2
cylinder-b	15.5±13.7	**9.1**±6.7	11.4±9.5
diabetes	45.0±2.5	**26.5**±7.2	**26.5**±8.2
germen	45.5±2.8	29.9±5.0	**29.3**±4.5
haberman	43.1±9.1	15.1±4.8	**13.3**±4.7
heart-s	45.5±3.1	**19.8**±7.7	21.7±7.2
ionosphere	43.1±8.7	15.0±6.3	**14.9**±5.6
kr-vs-kp	40.4±7.1	10.1±6.3	**8.9**±5.8
liver-dis	44.7±3.4	31.4±8.6	**29.9**±8.3
spectf	45.0±3.3	17.7±4.4	**17.3**±3.4
spambase	44.5±3.3	37.8±5.6	**37.2**±5.8
vote	38.6±16.1	8.8±6.0	**8.5**±5.8
average	41.2	19.2	**19.1**

Based on these result, it is believed that RSE can generate ensembles with small size but strong generalization ability.

Results under Semi-Supervised Setting The predictive error rates under label rates of 5% and 10% are shown in Table 4 and Table 5, respectively, and the ensemble sizes are shown in Table 6 and Table 7, respectively. Here, RSE$_{ss}$ and RSE-w$_{ss}$ use unlabeled data to improve performance, while other methods only use the labeled data.

It can be found from Table 4 and Table 5 that, under 5% and 10% label rates, RSE$_{ss}$ get the best performance on 11 and 13 data sets, respectively. Comparing RSE$_{ss}$ with RSE under 5% label rate, it can be found that the averaged error rate is reduced from 0.237 to 0.210. Especially, it is reduced from 0.336 to 0.245 on *spectf*. On *cylinder-b*, RSE has the worst error rate 0.420, but RSE$_{ss}$ performs best with error rate 0.332. Similar results can be found by comparing RSE-w$_{ss}$ with RSE-w, also under 10% label rate. Therefore, we can see the unlabeled data could be useful in constructing ensemble, even when the component classifiers are learned on the labeled data.

It can be found that under 5% label rate, the average error rate of RSE is 0.267 and that of RSE$_{ss}$ is 0.232, with a reduction of 15.1%; while under 10% label rate,

Table 4. Comparison of the predictive errors (mean±std) under 5% label rate, where the best one on each data set is bolded. The *average* row presents the results averaged over all data sets.

Data set	C4.5	Bagging	AdaBoost	GASEN	RSE-w	RSE	RSE-w$_{ss}$	RSE$_{ss}$
australian	.200±.065	.184±.051	.210±.050	.206±.061	.201±.053	.197±.051	**.151**±.025	**.151**±.023
balloons	**.458**±.001	**.458**±.001	**.458**±.001	.475±.033	**.458**±.001	**.458**±.001	**.458**±.001	**.458**±.001
cancer	.312±.062	.310±.051	.336±.074	.340±.091	.289±.016	.288±.023	.292±.010	**.285**±.022
cylinder-b	.394±.060	.393±.054	.395±.046	.396±.053	.403±.064	.420±.061	.336±.038	**.332**±.037
diabetes	.342±.046	.299±.036	.317±.035	.302±.038	.289±.035	.286±.034	.262±.027	**.259**±.028
germen	.345±.044	.308±.028	.326±.032	.310±.028	.329±.033	.316±.032	.291±.012	**.284**±.015
haberman	.280±.055	.273±.041	.288±.056	.292±.059	.257±.012	.265±.030	.256±.012	**.255**±.015
heart-s	.318±.081	.268±.070	.302±.079	.287±.078	.296±.069	.291±.071	**.220**±.045	.221±.043
ionosphere	.267±.095	.238±.068	.258±.084	.242±.073	.226±.065	.218±.056	**.176**±.030	**.176**±.029
kr-vs-kp	.074±.021	.064±.016	.067±.017	.064±.015	.065±.016	.064±.016	**.047**±.013	.049±.014
liver-dis	.443±.059	.434±.047	.435±.054	.427±.052	.411±.028	.404±.038	.379±.037	**.375**±.040
spectf	.362±.082	.281±.055	.315±.065	.300±.059	.328±.067	.336±.072	.262±.019	**.245**±.030
spambase	.147±.019	.107±.015	.087±.013	.107±.014	.116±.015	.110±.011	.100±.011	**.098**±.011
vote	.097±.068	.094±.062	.103±.072	.087±.058	.096±.061	.084±.058	**.059**±.045	.060±.046
average	0.288	0.265	0.278	0.274	0.269	0.267	0.235	**0.232**

Table 5. Comparison of the predictive errors (mean±std) under 10% label rate, where the best one on each data set is bolded. The *average* row presents the results averaged over all data sets

Data set	C4.5	Bagging	AdaBoost	GASEN	RSE-w	RSE	RSE-w$_{ss}$	RSE$_{ss}$
australian	.182±.041	.167±.032	.190±.037	.173±.029	.191±.035	.177±.036	.152±.016	**.149**±.019
balloons	.423±.114	.403±.099	.425±.100	.405±.113	.374±.079	.390±.107	.368±.070	**.346**±.073
cancer	.297±.029	.293±.034	.333±.047	.299±.033	.289±.015	.283±.022	.280±.025	**.275**±.026
cylinder-b	.369±.055	.349±.048	.377±.057	.347±.044	.370±.055	.380±.052	.318±.029	**.306**±.025
diabetes	.310±.046	.279±.038	.301±.045	.277±.039	.272±.031	.262±.034	.250±.026	**.243**±.025
germen	.335±.036	.300±.024	.309±.029	.299±.024	.305±.028	.289±.019	.286±.017	**.277**±.016
haberman	.273±.060	.267±.033	.288±.054	.283±.052	**.249**±.018	.254±.019	.251±.017	**.249**±.023
heart-s	.290±.058	.251±.063	.266±.066	.255±.073	.267±.058	.246±.052	**.212**±.039	**.212**±.046
ionosphere	.203±.083	.190±.053	.199±.078	.190±.058	.181±.054	.173±.052	.148±.039	**.143**±.035
kr-vs-kp	.057±.015	.051±.010	.042±.012	.049±.010	.044±.013	.044±.012	**.035**±.008	.036±.009
liver-dis	.403±.053	.397±.058	.398±.058	.397±.057	.384±.046	.378±.041	.347±.035	**.346**±.036
spectf	.328±.067	.269±.043	.289±.066	.273±.045	.284±.045	.285±.054	.235±.025	**.226**±.031
spambase	.127±.012	.095±.011	.073±.007	.097±.012	.099±.011	.094±.010	.088±.009	**.087**±.009
vote	.074±.049	.058±.029	.080±.044	.069±.038	.072±.033	.061±.029	**.049**±.023	**.049**±.023
average	0.262	0.241	0.255	0.244	0.241	0.237	0.216	**0.210**

the average error rate of RSE is 0.237 and that of RSE$_{ss}$ is 0.210, with a reduction of 12.9%. This validates that RSE$_{ss}$ can benefit from using unlabeled examples.

Comparing ensemble sizes in Table 6 and Table 7, we can find that RSE$_{ss}$ also generates smaller ensembles than GASEN, and much smaller than that of Bagging and AdaBoost. The sizes of RSE$_{ss}$ (RSE-w$_{ss}$) ensembles are often larger than that of RSE(RSE-w) ensembles; this is not difficult to understand because RSE$_{ss}$(RSE-w$_{ss}$) uses much more data than RSE(RSE-w) since the latter does not use unlabeled data.

5 Conclusion

Given a number of trained component learners, it is better to build an ensemble that contains some instead of all of the component learners. The selection, however, is generally difficult and some smart heuristics were used in previous studies. In this paper, we study the problem of selective ensemble under the regularization framework. In the proposed RSE approach, the selection problem is reduced to a QP problem which has a sparse solution and can be efficiently solved. Moreover, RSE can exploit unlabeled

Table 6. Comparison of the ensemble size (mean±std) under 5% label rate, where the smallest size on each data set is bolded. The *average* row presents the size averaged over all data sets.

Data set	GASEN	RSE-w	RSE	RSE-w$_{ss}$	RSE$_{ss}$
australian	34.9±9.5	**10.1**±3.8	11.6±3.5	25.1±9.1	30.4±8.3
balloons	**13.4**±9.7	75.2±4.5	75.2±4.5	75.2±4.5	75.2±4.5
cancer	**27.7**±9.6	53.1±18.2	53.6±14.4	50.2±15.0	50.7±17.9
cylinder-b	46.9±15.0	**4.0**±12.3	4.1±4.4	8.5±3.8	12.4±4.2
diabetes	46.9±13.6	**7.0**±3.7	7.8±7.2	22.0±9.7	21.0±7.2
germen	49.5±6.1	5.5±2.6	**5.3**±2.7	8.0±4.9	10.5±4.4
haberman	**25.7**±12.3	65.8±13.9	65.2±14.4	71.3±20.8	63.0±21.4
heart-s	**29.5**±14.5	36.1±10.9	37.6±13.1	49.3±13.2	49.1±15.5
ionosphere	34.3±14.1	36.7±14.8	48.2±19.6	**13.3**±8.0	19.2±9.2
kr-vs-kp	41.2±10.3	10.9±6.7	10.7±8.2	**7.1**±8.4	7.9±8.0
liver-dis	47.7±11.7	20.2±16.3	**19.5**±11.7	20.2±7.0	23.4±11.2
spectf	38.8±11.4	27.6±14.7	27.9±14.9	**17.3**±9.4	25.7±11.2
spambase	50.4±4.8	**10.6**±3.0	11.3± 2.5	17.8±5.9	18.5±4.3
vote	**24.6**±9.8	58.7±19.5	64.5±17.7	63.6±19.6	70.8±19.6
average	36.5	30.1	31.6	32.3	34.1

Table 7. Comparison of the ensemble size (mean±std) under 10% label rate, where the smallest size are bolded. The *average* row presents the size averaged over all data sets.

Data set	GASEN	RSE-w	RSE	RSE-w$_{ss}$	RSE$_{ss}$
australian	39.2±5.9	**4.9**±3.9	5.1±4.6	6.8±3.1	8.4±4.8
balloons	**29.7**±13.9	48.0±17.9	61.6±15.8	39.9±11.1	65.6±17.3
cancer	**33.8**±11.3	37.2±13.3	42.3±13.3	25.2±15.6	37.0±12.8
cylinder-b	44.6±7.8	**2.7**±1.6	2.9±1.7	8.3±8.6	10.1±8.8
diabetes	44.8±5.4	**7.6**±3.8	8.4±3.6	15.8±10.5	11.9±11.3
germen	44.9±3.6	6.9±3.7	8.3±3.6	**6.3**±6.3	11.3±8.6
haberman	**27.4**±6.1	52.1±9.6	56.4±7.4	53.2±9.7	55.7±12.9
heart-statlog	42.3±14.3	12.4±17.7	**10.8**±15.0	18.2±9.4	17.5±11.7
ionosphere	35.5±20.5	15.0±9.1	15.8±10.9	7.6±16.3	**11.9**±11.6
kr-vs-kp	41.4±13.9	9.7±7.0	8.9±6.5	8.9±8.1	**6.2**±7.9
liver-dis	42.7±13.5	**7.7**±4.2	8.2±4.6	8.1±6.8	12.0±9.0
spectf	43.6±11.9	7.3±8.2	**6.9**±6.5	7.3±6.8	10.2±9.9
spambase	44.1±9.0	**14.0**±3.4	14.4±2.7	21.2±5.0	21.6±4.4
vote	**15.3**±12.8	57.5±21.1	66.4±16.9	57.3±14.1	64.7±13.7
average	37.8	20.2	22.6	20.3	24.6

examples easily. Experiments show that RSE can generate ensembles with smaller sizes but strong generalization ability, and the use of unlabeled data is helpful.

In this paper, RSE is designed to handle two-class classification problems. A multi-class extension will be studied in future work. Considering that combining class probabilities may lead to better results than combining class labels [20], it may also an interesting future work is to develop selective ensemble algorithms which selectively combines class probabilities.

Acknowledgment

This work was supported by the National Science Foundation of China (60635030, 60721002) and the Jiangsu Science Foundation (BK2008018).

References

1. E. D. Andersen, B. Jensen, R. Sandvik, and U. Worsoe. The improvements in mosek version 5. Technical report, The MOSEK Inc., 2007.

2. M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.
3. C. Blake, E. Keogh, and C. J. Merz. UCI repository of machine learning databases. http://www.ics.uci.edu/∼mlearn/MLRepository.html, 1998.
4. L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.
5. L. Breiman. Arcing classifiers. *Annals of Statistics*, 26(3):801–849, 1998.
6. L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
7. P. D. Castro, G. P. Coelho, M. F. Caetano, and F. J. Von Zuben. Designing ensembles of fuzzy classification systems: An immune-inspired approach. In *Proceedings of the 4th International Conference on Artificial Immune Systems*, pages 469–482, Banff, Canada, 2005.
8. K. Chen and S. Wang. Regularized boost for semi-supervised learning. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 281–288. MIT Press, Cambridge, MA, 2008.
9. M. Coyle and B. Smyth. On the use of selective ensembles for relevance classification in case-based web search. In *Lecture Notes in Computer Science 4106*, pages 370–384. Berlin:Springer, 2006.
10. A. Demiriz, K. P. Bennett, and J. Shawe-Taylor. Linear programming boosting via column generation. *Machine Learning*, 46(1-3):225–254, 2006.
11. T. G. Dietterich. Ensemble methods in machine learning. In *Proceedings of the 1st International Workshop on Multiple Classifier Systems*, pages 1–15, Cagliari, Italy, 2000.
12. D. Doherty, M. A. Freeman, and R. Kumar. Optimization with matlab and the genetic algorithm and direct search toolbox. Technical report, The MathWorks Inc., 2004.
13. Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.
14. T. K. Ho. The random subspace method for constructing decision forests. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.
15. B. Kégl and L. Wang. Boosting on manifolds: Adaptive regularization of base classifiers. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 665–672. MIT Press, Cambridge, MA, 2005.
16. D. Margineantu and T. G. Dietterich. Pruning adaptive boosting. In *Proceedings of the 14th International Conference on Machine Learning*, pages 211–218, Nashville, TN, 1997.
17. G. Martínez-Muñoz and A. Suárez. Pruning in ordered bagging ensembles. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 609–616, Pittsburgh, PA, 2006.
18. B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, 2001.
19. C. Tamon and J. Xiang. On the boosting pruning problem. In *Proceedings of the 11th European Conference on Machine Learning*, pages 404–412, Barcelona, Spain, 2000.
20. K. M. Ting and I. H. Witten. Issues in stacked generalization. *Journal of Artificial Intelligence Research*, 10:271–289, 1999.
21. I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, San Francisco, CA, 2000.
22. Y. Zhang, S. Burer, and W. N. Street. Ensemble pruning via semi-definite programming. *Journal of Machine Learning Research*, 7:1315–1338, 2006.
23. Z.-H. Zhou and W. Tang. Selective ensemble of decision trees. In *Lecture Notes in Artificial Intelligence 2639*, pages 476–483. Berlin: Springer, 2003.
24. 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.
25. X. Zhu. Semi-supervised learning literature survey. Technical report, Department of Computer Sciences, University of Wisconsin Madison, 2007.