Classifier Ensemble by Exploring Supplementary Ordering Information

Ou Wu

Abstract—Supplementary information has been proven to be particularly useful in many machine learning tasks. In ensemble learning for a set of trained base classifiers, there also exists abundant implicit supplementary information about the performance orderings for the trained base classifiers in previous literature. However, few classifier ensemble studies consider exploring and utilizing supplementary information. The current study proposes a new learning method for stack classifier ensembles by considering the implicit supplementary ordering information regarding a set of trained classifiers. First, a new metric learning algorithm for measuring the similarities between two arbitrary learning tasks is introduced. Second, supplementary ordering information for the trained classifiers of a given learning task is inferred based on the learned similarities and related performance results reported in the previous literature. Third, a set of ordered soft constraints is generated based on the supplementary ordering information, and achieving the optimal combination weights of the trained classifiers is formalized into a goal programming problem. The optimal combination weights are then obtained. Finally, the experimental results verify the effectiveness of the proposed new classifier ensemble method.

Index Terms—Classifier ensemble, supplementary information, performance ordering, soft constraints.

1 Introduction

In general, an ensemble of different base classifiers is more accurate than a single base classifier [1] [2] [3]. This work focuses on stacking classifier ensembles. When a set of base classifiers is given, stacking classifier ensemble learning weightily combines the base classifiers in which the ensemble weights are learnt based on a new training set. This scenario has been investigated and applied to numerous applications [4] [5] [6]. One of the key issues in stacking classifier combination is producing the optimal ensemble weights of the involved base classifiers.

This study investigates a new approach to infer optimal combination weights for base classifiers in stacking classifier ensemble learning, that is, an ensemble that explores the supplementary information regarding performance orderings among base classifiers (hereafter referred to as supplementary ordering information (SOI)). Supplementary information is attainable and useful in numerous learning tasks, such as classification [7] [8], clustering [9] [10], and metric learning [11] [12]. In a standard classification problem, supplementary information can be the reliability of labels [7]; while it can be must-link or must-not-link constraints[1] for training instance pairs in clustering and metric learning [11][2]. Given that supplementary information is auxiliary to standard supervised information (e.g, labels), learning performances can be substantially improved when the former is effectively utilized [13]. In many ensemble learning problems, there is also substantial implicit supplementary information such as the performance orderings among involved trained classifiers. Nevertheless, this type of implicit supplementary information has received minimal attention from previous ensemble learning studies. Small et al. [2011] utilized a similar type of supplementary information to aid the learning weights of features in linear classification. The experimental results verified the usefulness of the utilized supplementary information.

Without loss of generality, the present research focuses on the ensemble of a set of trained binary classifiers. With a set of trained (binary) base classifiers and a new training set[3] of features and labels, stacking ensemble learning can assign optimal combination weights to the trained classifiers [14]. Optimal combination weights indicate the relative importance or performance orderings of trained classifiers. If the auxiliary information about the performance orderings of the involved trained classifiers in ensemble is available before ensemble, then this information is assumed to facilitate the pursuance of the optimal combination weights for the trained base classifiers. This type of SOI does exist in practice. The performance of a trained classifier depends on its corresponding learning algorithm and the characteristics of the learning (classification) task to be processed. For most common learning algorithms, the previous literature has tested and compared the performances of their trained classifiers in many existing learning (classification) tasks. Therefore, the approximate performance orderings for the trained classifiers on a new learning task can be inferred based on the reported performance results or conclusions of these existing studies. For example, assuming that three classifiers, that is, L_A, L_B, and L_C, are respectively trained based on three popular learning algorithms support vector machine (SVM) [15], logistic regression (LR) [16], and random forest (RF) [17], a performance ordering prior among L_A, L_B, and L_C can be approximately attained based on the classification performances of the classifiers trained using

Center for Applied Mathematics, Tianjin University, wuou@tju.edu.cn

1. The term “must-link” indicates two instances must be in the same cluster, while the term “must-not-link” indicates two instances must not be in the same cluster.

2. In clustering and metric learning, the aforementioned supplementary information is called side information.

3. To distinguish this training set with the training data for base classifiers, this set is called validation set in this paper.

SVM, LR, and RF on the existing learning tasks reported in previous literature. When all the three classifiers L_A, L_B, and L_C are required to ensemble into a combined classifier on a new learning task, the ordering prior among L_A, L_B, and L_C can be used as supplementary information to improve the ensemble.

In the current study, a new classifier ensemble method is proposed by exploring the SOI regarding a set of trained classifiers based on the available performance comparison data presented in the previous literature. However, two challenges can be encountered in applying this new method:

(1) Supplementary ordering information is inferred based on the performance comparison data presented in previous studies for existing learning tasks; hence, whether the ordering information is still applicable to the processing learning task should be further justified. Intuitively, a strong similarity between existing learning tasks and a given processing learning task is equal to increase the enhanced usefulness of the performance data on existing learning tasks. Therefore, the similarity between the processing task and existing learning tasks must be measured.

(2) How can the inferred SOI be embedded into the mathematical optimization problem for pursuing optimal classifier combination weights? The inferred SOI must intuitively be transformed into additional constraints and subsequently be added into the optimization problem. Nevertheless, constraints from the inferred SOI cannot be simply attributed to common hard constraints because the former may be unreliable or even incorrect in the worst cases.

For the first challenge, the inference of the similarity between two different learning tasks is simply reduced to the measurement of the similarity between the meta-features of the data sets for these learning tasks. The meta-features of a data set are extracted based on a recent progress in machine learning algorithm recommendation [18] [19]. Metric learning is introduced to learn effective similarity measures between the meta-features of two data sets in this study. However, compiling similar pairwise data yet is relatively easy, whereas compiling dissimilar pairwise data is difficult when metric learning is applied in this study. Therefore, motivated by the problem of learning using positive and unlabeled data (LPU) [20] [21] [22], a new learning problem, that is, metric learning with positive training data (pairwise data with similar labels) and unlabeled training data (pairwise data without similar and dissimilar labels), is firstly introduced and investigated. This new problem is called metric LPU (MLPU) for brevity. With the learned similar metrics in this study, more reliable SOI can be obtained.

For the second challenge, we note that soft constraints are typically used when the underlying information is insufficiently reliable in goal programming [23]. Therefore, the inferred SOI regarding the trained classifiers is used as a set of soft constraints in a classical ensemble learning framework, i.e., LPBoost[4] [26] [27]. Moreover, soft constraints are ordered according to the reliability degrees of the SOI regarding each pair of input trained classifiers.

4. LPBoost is a learning method for classification. It is also used as a staking classifier ensemble method in various ensemble applications [24] [25].

The main contributions of the present research can be summarized as follows:

- Existing studies on stacking ensemble learning do not consider supplementary information. Given that supplementary information is quite helpful in many learning tasks and implicitly exists in classifier ensembles, the current study investigates a new classifier ensemble learning method to utilize SOI for the involved trained classifiers. To our knowledge, this study is the first to consider and utilize supplementary information in classifier ensembles.
- This study explores a new metric learning problem, that is, MLPU, to obtain more reliable SOI for the input trained classifiers in ensemble based on the available performance comparison data presented in previous literature. A new learning algorithm is then proposed for MLPU.
- The inferred SOI is transformed into a set of soft constraints and the soft constraints are ordered based on their reliability degrees. The pursuance of optimal ensemble weights is reduced to a goal optimization problem on a validation data set with the ordered soft constraint set.

The rest of the paper is organized into five sections. Following the introduction, Section 2 briefly presents the background of the current study. Section 3 introduces the research methodologies in this study including the SOI inference and how to apply it into classifier ensemble learning. Section 4 reports the experimental results. Finally, conclusions are given in Section 5.

2 Backgrounds

This section introduces the background studies including those on classifier ensemble, learning with supplementary information, optimization with soft constraints, and automatic learning algorithm selection.

2.1 Classifier ensemble

Ensemble methods either train multiple learners and then combine them for use or directly combine trained multiple learners [14] [28] [29]. Gao et al. [30] [31] provided an excellent summary of various learning problems according two dimensions, namely, supervision and ensemble level. The current study belongs to the interaction of supervised learning and ensemble at output level. In particular, it focuses on the stacking classifier ensemble learning [32], which aims to train a new classifier by weighted combining the trained individual classifiers. With a set of trained multiple classifiers based on the original training data, a stacking classifier ensemble algorithm generates a validation set and considers the outputs of the trained classifiers on this validation set as new input features. Finally, the optimal classifier combination weights are achieved based on learning with both the new features and the original labels. The trained classifiers can then be combined based on the obtained optimal combination weights to classify new data. The two-class LPBoost [33] algorithm is typically applied to deriving the combination weights of trained classifiers. LPBoost determines an optimal combination weight vector by solving a linear programming problem.

Some classical methods (i.e., random forest [17], AdaBoost [34], and extreme learning machine [35]) can also be seen as ensemble algorithms. Nevertheless, these methods train base classifiers and infer combination weights simultaneously, which does not belong to "stacking classifier ensemble". Therefore, these methods are not directly compared with the proposed method in the experiments

2.2 Learning with supplementary information

Supplementary information (e.g., label reliability, must-link constraints for pairwise data, or labeled features[5]) is extremely useful as a supplement to standard supervised information such as labels. Supplementary information has been explored in many learning paradigms in previous literature. Xing et al. [11] investigated metric learning with user-provided supplementary information (i.e., similar and dissimilar pairs of data points). The learned metrics were demonstrated to be able to improve clustering performance significantly. Wu et al. [12] utilized implicit and uncertain supplementary information from tags and contents generated by users for automated photo tagging. Zhang et al. [37] developed a new semi-supervised boosting framework by utilizing the pairwise constraints between instances. The supplementary information on pairwise constraints describes whether two instances should belong to the same or different classes. Hu and Kwok [38] studied kernel learning when supplementary information was presented. Xu et al. [8] explored supplementary information, such as demographical information and attributes of users in web mining to expedite matrix completion and proposed a novel multi-label learning algorithm. Zhu et al. [39] incorporated supplementary prior information for posterior distributions into Bayesian inference and proposed a new regularized Bayesian inference framework. In text mining, it is feasible for domain experts to encode knowledge among labels and features which is referred to as labeled features [40]. Small et al. [41] investigated the learning problems when the importance ordering for some features has been given by domain experts. They developed a constrained weight space SVM to learn classifiers in the presence of ranked features.

The present study is partially inspired by the research of Small et al. [41]. We aim to integrate supplementary information for performance orderings (rankings) for classifiers into the combination weight learning. However, distinct differences exist between our study and that of Small et al. [41]. First, our study focuses on classifier ensemble instead of individual classifier training. Second, the supplementary information for ranked classifiers is not explicitly provided by users and is required to be inferred from related performance results in previous literature, whereas the ranked features are explicitly provided in the work of Small et al. Third, because the SOI for classifiers is obtained by inference, the corresponding constraints of these classifiers transformed from the inferred SOI may be unreliable. Therefore, the reliability of constraints must be considered in our work. By contrast, the constraints in the work of Small et al. are regarded as absolutely reliable[6].

2.3 Optimization with soft constraints

The majority of machine learning problems are finally reduced to a mathematical optimization problem with constraints similar to those in SVM and LPBoost. Constraints are typically derived from supervised information. For example, in SVM, most constraints come from the ground-truth labels in training data. Because it is difficult to satisfy all the constraints simultaneously, slack variables are usually added. In such situation, the original (hard) constraints become soft, thereby implying that the former can be violated with the cost of the slack variables. Moreover, supervised or prior information in numerous practical learning problems is not absolutely reliable or correct. In this case, soft constraints must be used. For example, the labels provided by users may be incorrect, and the supplementary information such as ranking among features may be unreliable. In this instance, the corresponding constraints should be soft. Meanwhile, some supervised or supplementary information may be more correct or more reliable than others. In this event, the ordering among soft constraints should be considered. Notably, the optimization problem in LPBoost implicitly uses soft constraints. Nevertheless, ordering among soft constraints is not referred to in LPBoost.

Optimization with ordered soft constraints is primarily investigated in goal programming [42] and constraint satisfaction problems [23]. Our study introduces concepts and methods in goal programming to describe our optimization problem to effectively embed the inferred SOI for classifiers.

2.4 Automatic learning algorithm selection

Learning algorithm selection refers to the problem of which learning algorithm is likely to perform best for the current learning tasks [43]. The No Free Lunch theorems [44] suggest that we need to move away from a black-box approach to algorithm selection and should understand more about the characteristics of learning tasks in order to identify the most appropriate algorithm. Researchers have explored meta-learning techniques [45], learning about learning, to predict the related performances of learning algorithms according to features of classification data sets. Wang et al. [19] proposed a novel multi-label learning-based method to recommend a classification algorithm for new learning tasks. Experiments on over 1,000 benchmark learning tasks verified the effectiveness of their proposed method.

Studies on learning algorithm selection indicate that the performances of learning algorithms are highly related to the characteristics of learning tasks [19]. Intuitively, if the characteristics of two learning tasks are highly similar, then the relative performances for the same learning algorithm on both tasks are assumed to be also quite similar. Therefore, the current study selects the performance ordering information presented in existing learning tasks to infer SOI, which can aid the ensemble learning for new learning tasks.

5. A labeled feature denotes that a particular feature, for example the word "call", is highly indicative of a particular label, such as "contact" [36].

6. In fact, in our point of view, the ordering for features provided by experts may also be unreliable, particularly when the feature dimension is high.

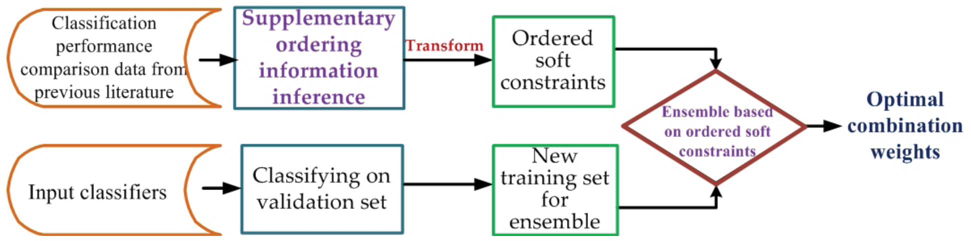
Classification performance comparison data from previous literature
Supplementary ordering information inference
Transform
Ordered soft constraints
Ensemble based on ordered soft constraints
Optimal combination weights
Input classifiers
Classifying on validation set
New training set for ensemble

Fig. 1. Overview of the proposed method for classifier ensemble learning with SOI exploration.

3 Methodologies

This section introduces the proposed new ensemble learning method based on the considering of implicit SOI on input trained base classifiers. We assume that H trained binary classifiers, denoted as $L_1, \cdots, L_H$, are existing and ready to use for ensemble. For each training instance on the validation training set, a new feature vector x_i is obtained by running the H classifiers on the corresponding raw feature vector. Each instance is associated with a label y_i. The task is to learn a weight vector for the H classifiers based on the new feature vectors, the raw labels, and SOI on the H classifiers. In this work, LPBoost is used as the reference ensemble learning algorithm.

3.1 Overview of the proposed method

This study aims to infer the SOI regarding the input trained classifiers and then utilizes it to combine the input classifiers effectively. Consequently, we encounter two challenges: inferring the SOI for classifiers, and building an ensemble based on the inferred SOI. These two challenges are addressed by proposing a new learning method as shown in Fig. 1. In the proposed method, the step of SOI inference aims to generate reliable SOI based on the performance comparison (ordering) data for input classifiers on existing learning tasks investigated in previous studies. Meanwhile, the step of ensemble learning intends to generate the optimal combination weights for the input trained classifiers based on the ordered soft constraints (generated from SOI) and input new training (validation) data. The following subsections describe the details of these key steps.

3.2 Supplementary ordering information inference

There are a large number of studies in previous literature that refer to the empirical evaluation and comparison of the performances of different classifiers trained from various learning algorithms. In general, the performance comparison results summarize the performance orderings of different classifiers and those of their corresponding learning algorithms on the specific tasks. These performance ordering data provide researchers or engineers with intuitive hints and evidences for selecting the corresponding learning algorithms when they are confronted with new learning (classification) tasks. For example, if the performance comparison results presented in most previous studies indicate that SVM is superior to decision tree in nearly all existing learning tasks, then engineers may use SVM instead of the decision tree to address a new classification task.

In the present work, performance ordering among classifiers based on the previous literature is adopted in an alternative manner. In particular, the performance comparison data presented in the previous literature are applied to explore the SOI for classifiers. When two learning tasks are highly similar to each other, the performance orderings for different classifiers on these two tasks should intuitively be relatively consistent. Therefore, a similarity metric for two different learning tasks should be established at the outset.

3.2.1 Metric learning for similarity measurement between two learning tasks

The similarity between two learning (classification) tasks can be measured based on the characteristics of each learning task. In recommendation of a learning algorithm [19], the meta-features for the data set of a learning task are used to describe the characteristics of the learning task. Meta-features described in [19] are also applied in this research and can be divided into the following categories[7]:

(1) *Statistical and information theory-based features.* This category includes the number of features, the number of training instances, the ratio of instances to features, the ratio of binary features, the correlation coefficient between features, and so on.
(2) *Model structure-based features*. This category uses the included decision tree to model the data structure of the learning task. The measures for the constructed decision tree, including the ratio of the number of nodes to the number of features, the ratio of the number of nodes to the number of instances, are used as the meta-features.
(3) *Landmarking-based features*. This category of features is considered based on the assumption that the performance of the candidate algorithms could be predicted through the performance of a set of simple learning algorithms (also called landmarkers). The landmarkers used in this work are Naïve Bayes, 1-NN, and a decision node learner.
(4) *Problem complexity-based features*. This category of features describes the geometrical characteristics of the training data. Typical features include Fisher's discriminant ratio, and the percentage of instances in the problem that lie next to the class boundary.
(5) *Structural information-based features*. This category of features calculates the frequencies of the 1, 2-itemsets of the training data, and then extracts the quantiles of these frequency sequences as features.

7. The effectiveness of these categories of meta-features has been verified by Wang et al. [19] on over 1,000 benchmark learning tasks.

On the basis of the aforementioned meta-features, the similarity between two learning tasks can be directly calculated using conventional similarity measurements, such as cosine similarity. Nevertheless, these meta-features are heterogeneous, and dealing with them via a simple procedure may be inappropriate. A more suitable means is to utilize metric learning to learn a similarity metric which can assign certain weights to the features defined above.

The similarity between two learning tasks (denoted as tasks T_i and T_j without loss of generality) is calculated based on the meta-features of the training data sets of these two learning tasks. Therefore, let $\bar{\boldsymbol{v}}_i$ and $\bar{\boldsymbol{v}}_j$ denote the meta-features of the two labeled data sets of the tasks i and j, respectively. Let $y_{ij} \in \{-1, 1\}$ be the similarity label of the pair T_i, T_j. If the pair is similar, $y_{ij} = 1$, if the pair is dissimilar, $y_{ij} = -1$. Accordingly, the measurement for similar degree for T_i and T_j is defined as the probability of that y_{ij} is equal to 1 given $\boldsymbol{v}_i$ and $\bar{\boldsymbol{v}}_j$. That is,

$$Sim(T_i, T_j) = p(y_{ij} = 1 | \bar{\boldsymbol{v}}_i, \bar{\boldsymbol{v}}_j), \tag{1}$$

The value of $Sim(T_i, T_j)$ ranges within $[0, 1]$ where a high value indicates a high similarity degree.

The similarity measurement described in (1) can be learned by first constructing a training set, which consists of similar and dissimilar pairs of learning tasks. A similar set can be constructed by simply collecting classification tasks from the same or similar application domains. Nevertheless, judging whether two tasks are dissimilar is difficult even if they come from different application domains. Therefore, formulating an effective training set is difficult. Inspired by previous studies on learning with positive and unlabeled data (LPU), this study introduces a new learning problem, called metric learning with positive (similar) and unlabeled data (MLPU). The learning problem is formalized as follows.

MLPU: By considering a set of M-dimensional meta-feature vectors $\{\bar{\boldsymbol{v}}_i\}_{i=1}^N \in R^M$ for N classification tasks $\{T_i\}_{i=1}^N$, and by noting that certain pairs are "similar", we can construct the following similar set S:

$$S : (\bar{\boldsymbol{v}}_i, \bar{\boldsymbol{v}}_j) \in S \quad \text{if} \quad T_i \quad \text{and} \quad T_j \quad \text{are} \quad \text{similar}$$

How can we learn a similarity measurement function defined in Eq. (1)?

Assuming η is a random binary variable. Let $\eta = 1$ if a pair of learning tasks is labeled, and let $\eta = 0$ if a pair is unlabeled. Only positive pairs are labeled; hence, $y_{ij} = 1$ is certain when $\eta = 1$. However, when $y_{ij} = -1$, either $\eta = 1$ or $\eta = 0$ may be true.

A fixed unknown distribution $p(\bar{\boldsymbol{v}}_i, \bar{\boldsymbol{v}}_j, y_{ij}, \eta)$ can be observed over the involved variables. The training set of MLPU is a sample drawn from this distribution that consists of labeled samples $< \bar{\boldsymbol{v}}_i, \bar{\boldsymbol{v}}_j, \eta = 1 >$ and unlabeled samples $< \bar{\boldsymbol{v}}_i, \bar{\boldsymbol{v}}_j, \eta = 0 >$. The fact that only positive samples are labeled can be precisely described by the following equation

$$p(\eta = 1 | \bar{\boldsymbol{v}}_i, \bar{\boldsymbol{v}}_j, y_{ij} = -1) = 0. \tag{2}$$

By following the work of previous studies on LPU [22], we primarily assume that the labeled positive examples are completely selected in a random manner from all positive examples. Such an assumption is expressed as follows:

$$p(\eta = 1 | \bar{\boldsymbol{v}}_i, \bar{\boldsymbol{v}}_j, y_{ij} = 1) = p(\eta = 1 | y_{ij} = 1). \tag{3}$$

Let $\pi = p(\eta = 1 | y_{ij} = 1)$ be the probability that a positive example pair is labeled. The value of π is constant according to Eq. (3). With the above assumption, the following lemma is established in LPU.

Lemma 1 [22]: Suppose that the "selected completely at random assumption" holds. Then

$$p(y = 1 | \bar{\boldsymbol{v}}_i, \bar{\boldsymbol{v}}_j) = p(\eta = 1 | \bar{\boldsymbol{v}}_i, \bar{\boldsymbol{v}}_j) / \pi. \tag{4}$$

Therefore, the calculation of $Sim(T_i, T_j)$ relies on the values of $p(\eta = 1 | \bar{\boldsymbol{v}}_i, \bar{\boldsymbol{v}}_j)$ and π. To estimate $p(\eta = 1 | \bar{\boldsymbol{v}}_i, \bar{\boldsymbol{v}}_j)$, we can construct a new training set based on the MLPU training set. Then for the example pairs in S, the label $\eta = 1$; otherwise, the label $\eta = 0$. According to [46], $p(\eta = 1 | \bar{\boldsymbol{v}}_i, \bar{\boldsymbol{v}}_j)$ is expressed based on the logistic function as follows:

$$p(\eta = 1 | \bar{\boldsymbol{v}}_i, \bar{\boldsymbol{v}}_j) = \frac{1}{1 + \exp(-||\bar{\boldsymbol{v}}_i - \bar{\boldsymbol{v}}_j||_{\mathbf{A}}^2 + \mu)},$$

where μ is the threshold, $\mathbf{A}$ is a diagonal and nonnegative matrix, and

$$||\bar{\boldsymbol{v}}_i - \bar{\boldsymbol{v}}_j||_{\mathbf{A}}^2 = (\bar{\boldsymbol{v}}_i - \bar{\boldsymbol{v}}_j)^T \mathbf{A} (\bar{\boldsymbol{v}}_i - \bar{\boldsymbol{v}}_j), \tag{5}$$

Based on the new MLPU training set and on the above definitions, the overall log likelihood for all similar/unlabeled pairs is as follows:

$$\begin{aligned} L(\mathbf{A}, \mu) = & - \sum_{(\bar{\boldsymbol{v}}_i, \bar{\boldsymbol{v}}_j) \in S} \log(1 + \exp(-||\bar{\boldsymbol{v}}_i - \bar{\boldsymbol{v}}_j||_{\mathbf{A}}^2 + \mu)) \\ & - \sum_{(\bar{\boldsymbol{v}}_k, \bar{\boldsymbol{v}}_l) \notin S} \log(1 + \exp(||\bar{\boldsymbol{v}}_k - \bar{\boldsymbol{v}}_l||_{\mathbf{A}}^2 - \mu)). \end{aligned} \tag{6}$$

Meanwhile, based on the maximum likelihood estimation, the pursuance of variables $\mathbf{A}$ and μ is casted into the following optimization problem:

$$\begin{aligned} & \min L(\mathbf{A}, \mu) \\ & s.t. \quad \mathbf{A} \succeq 0, \mu \geq 0 \end{aligned}, \tag{7}$$

where the constraint $\mathbf{A} \succeq 0$ indicates that $\mathbf{A}$ is a positive semi-definitive matrix.

To solve the above problem, we define $\mathbf{M} = \frac{1}{N} \sum_{i=1}^{N} \bar{\boldsymbol{v}}_i \bar{\boldsymbol{v}}_i^T$. Let $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_K$ be the top K eigenvectors of matrix $\mathbf{M}$. We then assume $\mathbf{A}$ is a linear combination of the top K eigenvectors

$$\mathbf{A} = \sum_{i=1}^{K} \gamma_i \boldsymbol{v}_i \boldsymbol{v}_i^T, \gamma_i \geq 0, i = 1, ..., K. \tag{8}$$

The above optimization problem is simplified into the following form according to the procedures in [47]:

$$\begin{aligned} & \min_{\gamma_i, \mu} L(\mathbf{A}, \mu) \\ & s.t. \quad \mathbf{A} = \sum_{i=1}^{K} \gamma_i \boldsymbol{v}_i \boldsymbol{v}_i^T, \\ & \qquad \mu \geq 0, \\ & \qquad \gamma_i \geq 0 \\ & \qquad i = 1, \cdots, K \end{aligned} . \tag{9}$$

Such an optimization problem is a convex programming problem; thus, it can be solved by directly using Newton's method. If $\mathbf{A}$ is obtained, $Sim(T_i, T_j)$ defined in Eq. (1) is calculated as follows:

$$\begin{aligned} Sim(T_i, T_j) &= \frac{1}{[1+\exp(-||\bar{\boldsymbol{v}}_i - \bar{\boldsymbol{v}}_j||^2_{\mathbf{A}} + \mu)] \times \pi} \\ &\sim \frac{1}{[1+\exp(-||\bar{\boldsymbol{v}}_i - \bar{\boldsymbol{v}}_j||^2_{\mathbf{A}} + \mu)]} \end{aligned}, \quad (10)$$

where π can be omitted as it is constant.

3.2.2 *Supplementary ordering information inference based on learned similarity measurement*

Given two learning algorithms for classification, we generally have historical comparison data regarding the performances of their trained classifiers on various existing learning tasks from different application domains which can be found in the previous literature. In particular, we assume that two classifiers that denote L_1 and L_2, which are trained by two different learning algorithms, are available. The performance comparison data for the two classifiers on the G existing learning tasks $T_1, T_2, \cdots, T_G$ should exhibit the form listed in Table 1:

TABLE 1
The performance comparison data for classifiers L_1 and L_2

Classification task T_1: $L_1 \succ L_2$ (L_1 outperforms L_2)
Classification task T_2: $L_1 \prec L_2$ (L_2 outperforms L_1)
$\cdots$
Classification task T_G: $L_1 \succ L_2$ (L_1 outperforms L_2)

The above historical comparison data can be used to estimate the reliability of ordering between L_1 and L_2 on a new classification task T_u. Given the performance comparison data in Table 1 and the learned similarity metric $Sim(,)$, the reliability degree ($\mathcal{Q}$) of that L_1 outperforms L_2 on the task T_u is defined as follows:

$$\begin{aligned} &\mathcal{Q}(L_1 \succ L_2 : T_u) \\ &= \frac{1}{\sum_{g=1}^{G} Sim(T_u, T_g)} \times \quad [\sum_{T_i : L_1 \succ L_2} Sim(T_u, T_i) \\ &- \sum_{T_j : L_1 \prec L_2} Sim(T_u, T_j))] \end{aligned}, \quad (11)$$

where $Sim(,)$ is calculated according to Eq. (10).

The value of $\mathcal{Q}(L_1 \succ L_2 : T_u)$ ranges within [0, 1]. When the value is high, L_1 is inclined to outperform L_2 on the new classification task based on performance ordering data for the existing learning tasks reported in the previous literature. According to Eq. (11), if L_1 outperforms L_2 in all previous classification tasks[8], then $\mathcal{Q}(L_1 \succ L_2 : T_u) = 1$, and vice versa. In addition, assume that $\mathcal{Q}(L_1 \succ L_2 : T_u) = 0.6$ and $\mathcal{Q}(L_3 \succ L_4 : T_u) = 0.9$, then the inferred ordering $L_3 \succ L_4$ is more reliable than $L_1 \succ L_2$.

For each pair of classifiers L_i and L_j, we can estimate the ordering relationships by calculating $\mathcal{Q}(L_i \succ L_j : T_u)$ on the new classification task T_u. In the next section, these estimated ordering relationships are used in the concrete ensemble learning for the optimal combination weights of input trained classifiers.

To learn a similarity metric between two data sets, a training corpus that consists of similar pairs of data sets and unlabeled data sets should be constructed. The similar pairs of data sets and unlabeled data sets used in this study are constructed as follows. The 121 data sets compiled based on the UCI machine learning repository[9] used in [48] are introduced as the basic data sets. Based on these basic data sets, a set of new training data is established for MLPU. For the similar data set, the data pairs are selected based on the following procedures. A total of 36 pairs of data sets, each of which come from exactly the same application domain, are considered similar. Each of the 121 data sets is processed with the feature reduction technique principal component analysis [49] and a new data set is obtained. The original and new corresponding data sets are considered as a pair of similar data sets. A total of 157 similar pairs of data sets are then obtained. For the unlabeled data pairs, the six adjacent data sets for each of the 121 data sets, which are from different application domains, listed in [48], are selected with the data set to generate six unlabeled pairs of data sets. In total, there are 726 unlabeled pairs of data sets. All the aforementioned data are used to learn a new similarity measurement based on the proposed algorithm.

3.3 Ensemble with inferred SOI

Given an ensemble training (or validation) set and a set of trained classifiers for ensemble, the optimal combination weights of the trained classifiers can be pursued with existing learning algorithms such as LPBoost [33]. Generally, the original LPBoost without introducing slack variables can be formalized as follows:

$$\begin{aligned} &\min_{\rho, \boldsymbol{\beta}} \; -\rho \\ &s.t. \quad y_n \boldsymbol{\beta}^T x_n \geq \rho \quad n = 1, \cdots, N \\ &\qquad \sum_{h=1}^{H} \boldsymbol{\beta}(h) = 1, \boldsymbol{\beta}(h) \geq 0 \end{aligned}, \quad (12)$$

where $\boldsymbol{\beta}$ indicates the combination weights for the H input trained classifiers.

The first family of constraints in Eq. (12) is hard. When all of these constraints are satisfied, all the training data are correctly classified by the corresponding solution. As described in the previous subsection, we are able to obtain some orderings between the input classifiers for ensemble. If $\mathcal{Q}(L_i \succ L_j : T_u) > 0$, then the historical comparison data indicate that L_i outperforms L_j in the current classification task, and vise versa. Intuitively, this SOI ($\mathcal{Q}$) for two classifiers can be transformed into the ordering for their combination weights according to the following form:

$$if \quad \mathcal{Q}(L_i \succ L_j : T_u) > 0 \quad \text{then} \quad \boldsymbol{\beta}(i) > \boldsymbol{\beta}(j). \quad (13)$$

where $\boldsymbol{\beta}(i)$ and $\boldsymbol{\beta}(j)$ are the combination weights of the classifiers L_i and L_j, respectively. Eq. (13) signifies that if the classifier L_i is expected to be superior to L_j, its combination weight should be larger than that of L_j. This intuition is

8. If the comparison results on some classification tasks are not given, the calculation can still be performed with a slight modification that the denominator only sums the involved tasks.

9. http://mlr.cs.umass.edu/ml/index.html

similar in spirit with the assumption in [41], that is, if one feature is judged by a human expert to be better than another, its weight should be larger. However, this intuitive consideration encounters the following issues:

(1) In this study, the reliability degree of the estimated ordering $L_i \succ L_j$ in Eq. (13) has no theoretical guarantees, and may be inaccurate or even wrong[10]. In this case, the concluded ordering $\boldsymbol{\beta}(i) > \boldsymbol{\beta}(j)$ in Eq. (13) is not highly significant. Therefore, the reliability issue must not be ignored.

(2) Even if the estimated ordering (i.e., $L_i \succ L_j$) in Eq. (13) is correct, the concluded ordering (i.e., $\boldsymbol{\beta}(i) > \boldsymbol{\beta}(j)$) may also be insignificant because the SOI only considers the relationships between two classifiers. Another classifier such as L_3 may indicate that the combination of L_2 and L_3 can achieve good results. In this case, $\boldsymbol{\beta}(1)$ becomes zero and the conclusion that $\boldsymbol{\beta}(1) > \boldsymbol{\beta}(2)$ becomes insignificant.

(3) Even if the inference in Eq. (13) is correct, the absolute value of $\mathcal{Q}(L_i \succ L_j : Tu)$ indicates the reliability of the conclusion. For example, $\mathcal{Q}(L_i \succ L_j : T_u) = 0.51$ and $\mathcal{Q}(L_i \succ L_j : T_u) = 0.99$ are different. The concluded ordering ($\boldsymbol{\beta}(i) > \boldsymbol{\beta}(j)$) obtained by the latter value is more reliable.

The utilization of the SOI obtained by using Eq. (13) should not ignore the above three problems. To this end, the obtained supplementary orderings among combination weights in this study are treated as soft constraints which can be violated. That is, the optimal combination weights should first satisfy the hard constraints in (13). After all the hard constraints are satisfied, the optimal combination weights that satisfy the soft constraints are selected as the final weights. Moreover, the soft constraints are ordered such that the satisfaction of the constraints corresponding to larger $\mathcal{Q}$ values should be prioritized. The utilization of the ordered soft constraints can address the three aforementioned issues. First, if the estimated ordering is inaccurate or insignificant, the optimal combination weights can violate the corresponding inaccurate soft constraint. Second, the soft constraints that correspond to different $\mathcal{Q}$ values are dealt with in orders according to the $\mathcal{Q}$ values. In summary, the optimization problem can be summarized as follows:

$$
\begin{aligned}
&\min_{\rho,\boldsymbol{\beta}} \; -\rho \\
s.t.\quad & y_n \boldsymbol{\beta}^T x_n \geq \rho \qquad n = 1, \cdots, N \quad \text{(hard constraints)} \\
& \left.\begin{array}{ll}
\boldsymbol{\beta}(i) > \boldsymbol{\beta}(j), & \text{if} \quad \mathcal{Q}(L_i \succ L_j : T_u) \\
\cdots & \\
\boldsymbol{\beta}(k) > \boldsymbol{\beta}(l), & \text{if} \quad \mathcal{Q}(L_k \succ L_l : T_u)
\end{array}\right\} \text{(ordered soft constraints according to } \mathcal{Q}) \\
& i, j, k, l \in [1, H] \\
& \sum_{h=1}^{H} \boldsymbol{\beta}(h) = 1, \boldsymbol{\beta}(h) \geq 0
\end{aligned} \tag{14}
$$

where H is the number of classifiers in the ensemble. In the above optimization problem, the soft constraints are ordered according to their corresponding supplementary ordering values (i.e., $\mathcal{Q}$). Therefore, assuming that $\mathcal{Q}(L_i \succ L_j : T_u) > \mathcal{Q}(L_k \succ L_l : T_u)$ in (14), the soft constraint that $\boldsymbol{\beta}(i) > \boldsymbol{\beta}(j)$ is ordered higher than that $\boldsymbol{\beta}(k) > \boldsymbol{\beta}(l)$.

In practice, the hard constraints in (14) are also difficult to be satisfied[11]. These constraints can also be relaxed as soft constraints[12] if the ordering degrees indicate that the relaxed soft constraints are significantly higher than the current soft constraints. We solve this problem by adopting the goal programming technique. The solving details are introduced in Appendix A. Comparing (12) with (14), the main difference between LPBoost and our method lies in that the optimization problem in our method considers new soft constraints which are inferred based on the SOI for the base classifiers to ensemble.

ALGORITHM 1: EnsemSP

Input: Validation data for the classification task T_u; a set of trained classifiers $\{L_h\}$, $h = 1, \cdots, H$; multiple labeled data sets from different learning tasks $\{T_g\}$, $g = 1, \cdots, G$ and their meta-features; the performance data of different learning algorithms reported in the previous literature; learned similarity measurement function f_{sim}; parameters λ_1 and λ_2.

Output: Optimal combination weights $\boldsymbol{\beta}$.

Steps:

```
1. Extract the meta-feature of $T_u$ described in Section 3.2.1;

2. Calculate the similarities (i.e., $Sim(,)$) using Eq. (10)
   between the validation set $T_u$ and the historical data sets
   from the learning tasks $\{T_g\}$, $g = 1, \cdots, G$ based on their
   meta-features;

3. Infer the supplementary ordering relationships and the
   reliability degree between each pair of the trained classifiers
   in $\{L_h\}$, $h = 1, \cdots, H$ using Eq. (11) based on the calculated
   similarities and historical comparison results on $\{T_g\}$,
   $g = 1, \cdots, G$;

4. Run the trained classifiers $\{L_h\}$, $h = 1, \cdots, H$ on the
   validation data for $T_u$. The scores of all the trained classifiers
   on each validation instance are consisting of a new feature
   vector for the validation instance. A new training set for the
   successive classifier combination weights can then be
   obtained;

5. Construct the optimization problem (14) for classifier
   combination weights based on the new training set and the
   inferred the supplementary ordering relationships and the
   , reliability degrees;

6. Solve the optimization problem (14) to produce the optimal
   ensemble weights $\boldsymbol{\beta}$.
```

10. In fact, the estimated ordering for features in [41] may also be inaccurate. The estimation accuracy depends on the expertise degree of human expertise and the complexity of the relationships among features.

11. Furthermore, if all the hard constraints are satisfied, the generalization ability of the obtained ensemble weights may be low.

12. In fact, we will show that the introduction of slack variables is essentially to take the hard constraints to soft in the Appendix A.

3.4 Steps

The entire process of classifier **ensem**ble learning with **s**u**p**plementary ordering information is called EnsemSP for brevity. The primary steps for applying EnsemSP are summarized in Algorithm 1.

3.5 Computational complexity

The computational complexity of the proposed algorithm depends on two parts. The first part is the meta-feature extraction for training data, and the second part is solving the optimization problem, that is, Eq. (14). Meta-feature extraction follows the features used in the work of Wang et al. [19]. In this work, the meta-feature extraction process for each experimental data set (including both T_g and T_u in Algorithm 1) does not exceed 10 seconds on a computer with Intel Core i7-2600 CPU 3.4GHz and 8G RAM. For linear programming, computational complexity is subject to the number of variables (i.e., H+1). The value of H is generally small (5 in this study) for classifier ensemble, and thus, the time consumption is low in our experiments. Overall, the computational complexity of EnsemSP is larger than that of LPBoost. Nevertheless, the computational complexity of EnsemSP is acceptable because meta-feature extraction is performed only once for each data set in $\{T_g\}$ for different tasks T_us in Algorithm 1.

4 Experiments

In this section, experiments are performed to verify the effectiveness of the proposed method, i.e., EnsemSP. Given that there is little work that investigates ensemble learning with supplementary information in previous literature, existing classical ensemble methods[13] will be used as the competing methods.

4.1 Experimental data

Running the proposed method requires two data corpora. The first data corpus consists of three parts. The first part (called training set directly) is used to train a set of base classifiers for the ensemble. The second part (called validation set[14]) is used for ensemble learning to achieve the optimal combination weights for the trained base classifiers. The third part (called test set) is used to evaluate the performances of the competing ensemble methods. Meanwhile, the second data corpus comprises numerous data sets and associated historical performance comparison data for the classifiers' corresponding learning algorithms on these data sets. Fernńdez-Delgado et al. [48] conducted extensive investigations to compare various classical learning algorithms (e.g., random forest, support vector machine, AdaBoost, etc.) across 121 UCI data sets. Accordingly, in the experiments, the involved 121 UCI data sets and reported performance data from [48] are used as the $\{T_g\}$ set and the historical performance ordering data in Algorithm 1, respectively.

In the present research, 17 data sets are used to construct the first data corpus. Only binary classification is performed in the experiments. Therefore, new data sets are compiled based on the raw data sets for those with more than two categories. For example, two new sets (i.e., yeast01 and yeast23) are constructed by choosing data from the "0" and "1" categories and data from the "2" and "3" categories, respectively. Table 2 shows the details of the involved data sets. The former 13 data sets are obtained from the UCI Machine Learning Repository[15]. The ImageAes data set consists of photos and their 'high'/'low' aesthetic labels[16] which can be obtained from http://ritendra.weebly.com/aesthetics-datasets.html (the good ol' data set). The rest three sets are obtained from LibSVM data[17]. Considering that both the evaluation data and the historical comparison data are mainly obtained from the UCI repository, when a UCI data set is used for evaluation, its associated historical performance comparison data are excluded from the historical comparison data corpus.

TABLE 2
The details of the involved seventeen data sets.

Data set	No. of instances	No. of features
heat	270	13
mfeat1	400	573
mfeat2	400	573
mfeat3	400	573
mfeat4	400	573
biodeg	1055	41
germannoise	1000	24
hillvalleynoise	1212	100
hillvalleynonoise	1212	100
thyroidnoise	215	5
winered	1599	11
Yeast01	892	8
Yeast23	407	8
ImageAes	3581	58
splice	3175	60
gisette	7000	5000
svmguide3	1284	21

4.2 Basic learning algorithms and competing ensemble methods

In our experiments, five basic learning algorithms (or base learners) are used for training classifiers: support vector machines (SVM), random forest (RF) [17], AdaBoost [34], C4.5 decision tree (C4.5) [51], and logistic regression (LR) [16]. Four competing ensemble methods are considered, i.e., majority voting (Voting), weigted voting (wVoting), LPBoost, and EnsemSP. The parameter settings of all the involved algorithms and methods are introduced in the next subsection.

13. Some classical methods (e.g., UDEED [50] and consensus maximization [30]) are not included because these methods run in different ensemble settings.

14. To avoid overfitting, the validation data used for ensemble learning should be excluded from the training data for base learners [14]. This strategy is adopted in our experiments. The data used in the first and second parts are different.

15. http://archive.ics.uci.edu/ml/datasets.html

16. The 'high'/'low' aesthetic label of a photo is generated according to its associated average user score. In the experiments, an average score larger than five is transformed into the 'high' aesthetic label.

17. https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html

The relationships among the involved experimental training data, base learners, and ensemble methods are shown in Table 3.

4.3 Ensemble results

In the first experiments, the parameter settings of the five basic learning algorithms are as follows. For SVM, the radial basis kernel is chosen. The parameters C and g are fixed with 1 and 0.1, respectively. For RF, the number of trees is fixed as 50, and other parameters are default. For AdaBoost, the number of basic learners is fixed as 50, and other parameters are default. For C4.5 decision tree, the pruning confidence threshold is set as 0.25. For LR, the codes in MATLAB are used, and all the parameters are default. The parameter settings of LPBoost and EnsemSP[18] are defined as follows. The parameter λ_1 (please see (16) in Appendix A) is searched via 5-fold cross validation in {0.1, 1, 10, 50, 100} and for each value of λ_1, the parameter λ_2 is searched in {0, $\lambda_1/10$, $\lambda_1/5$, $\lambda_1/2$, λ_1}. When λ_2 equals 0, EnsemSP is reduced to LPBoost. The weighted setting for wVoting follows the strategy proposed by Wang et al. [52].

Given a data set, to perform EnsemSP, the similarity between the data set and each of 121 UCI data sets is calculated according to Step 2 in Algorithm 1. Then the supplementary ordering relationships as well as reliability degrees between each pair of the five basic learning algorithms are inferred according to Step 3 in Algorithm 1. The inferred supplementary ordering relationships as well as reliability degrees are transformed into soft constrains as shown in Eq. (14).

In the experiment, the parameters of basic classifiers are tuned on training data. For SVM, the parameters C and g are searched via five-fold cross validation from {0.1, 1, 5, 10, 100} and {0.01, 0.1, 1, 5, 10}, respectively. For RF, the number of trees is searched via five-fold cross validation from {10, 20, 50, 100, 200, 500}. For AdaBoost, the number of basic learners is searched via five-fold cross validation from {10, 20, 50, 100, 200}. For C4.5, the pruning confidence threshold is searched via five-fold cross validation from {0.15, 0.2, 0.25, 0.3, 0.35}. Table 4 lists the classification accuracies of the three competing ensemble methods on 17 data sets. On most data sets, Voting achieves the lowest classification accuracies. Meanwhile, LPBoost also produces inferior results than EnsemSP.

In Section 3.1, we learn the similarity measurement function for the meta-features[19] of the data sets between two learning tasks. We further investigate whether the learned similarity measurement function is useful by comparing three ensemble methods, namely, EnsemSP-D (EnsemSP that does not consider the similarities between the historical and the current data sets), EnsemSP-C (EnsemSP that uses cosine similarities), and EnsemSP (uses the learned similarity metrics). In EnsemSP-D, the similarities between historical data sets and the current data set are not used; thus the ordering relationships among the input classifiers are nearly the same. In other words, in EnsemSP-D, the meta-features are not used; in EnsemSP-C, the meta-features are used but MLPU is not used; in EnsemSP, both the meta-features and MLPU are used. Table 5 presents the classification accuracies of the three competing ensemble methods on the used seventeen data sets. In most data sets, EnsemSP-D achieves the lowest classification accuracies. EnsemSP, which utilizes the learned similarity metric, outperforms both EnsemSP-D and EnsemSP-C in most data sets. The results verify that both the used meta-features and learned similarity metric are useful.

TABLE 5
The classification accuracies of three versions of EnsemSP.

Data set	EnsemSP-D	EnsemSP-C	EnsemSP
heat	0.808	0.816	**0.828**
mfeat1	0.968	0.989	**0.996**
mfeat2	0.957	0.972	**0.979**
mfeat3	0.983	**0.999**	**0.999**
mfeat4	0.989	0.994	**0.997**
biodeg	0.8502	0.8563	**0.8740**
germannoise	0.7497	0.7475	**0.7699**
hillvalleynoise	0.5393	0.5526	**0.5688**
hillvalleynonoise	0.5241	0.5076	**0.5394**
thyroidnoise	0.965	**0.971**	0.968
winered	**0.7794**	0.7628	0.7771
Yeast01	0.675	0.634	**0.686**
Yeast23	0.911	0.917	**0.923**
ImageAes	0.6883	**0.6931**	0.6912
splice	0.9087	0.9203	**0.9285**
gisette	0.9786	0.9753	**0.9806**
svmguide3	0.9108	0.9217	**0.9320**

The robustness of the EnsemSP method is further investigated by adding noisy labels into the involved validation data sets and by performing new comparisons. For each validation data[20], that labels of $n\%$ $(n = 10, 20, 30, 40, 50)$ samples are flipped. The voting algorithm is not referred to in this comparison because it does not require validation data. Figure 2[21] shows the variations of classification accuracies of the ensemble learning methods in terms of $n\%$ noisy labels. The performance of LPBoost decreases dramatically with the increasing proportion of noisy labels in Figs. 2(b), 2(c), 2(d), 2(e), 2(h), and 2(j). By contrast, the performance of EnsemSP slightly decreases on most data sets. We also investigate the performances of the competing methods when the label-flapping proportions of both the training and validation sets are set as 30%. The results are shown in Table 6. The performances of all the competing methods are reduced. Nevertheless, EnsemSP still achieves the highest accuracy in most data sets.

To further verify the effectiveness of EnsemSP when the inferred SOI is useless or even wrong, all the inferred orderings are reversed when applying EnsemSP, which is called EnsemSP-reverse. The experimental results show that

18. In the performing of EnsemSP, the similarity measurement is used to infer supplementary ordering which is transformed into soft constraints in Eq. (14). The detailed steps are described in Algorithm 1.

19. We appreciate Dr. Guangtao Wang for providing us the source codes (http://gr.xjtu.edu.cn/web/qbsong/8) of meta-feature extraction described in [19].

20. The competing ensemble methods are performed on validation data, while base classifiers are trained on training data.

21. The performances of the competing methods on both the hillvalleynoise and hillvalleynonoise data are poor and only slightly better than 0.5, so these two data sets are unexplored in this experiment.

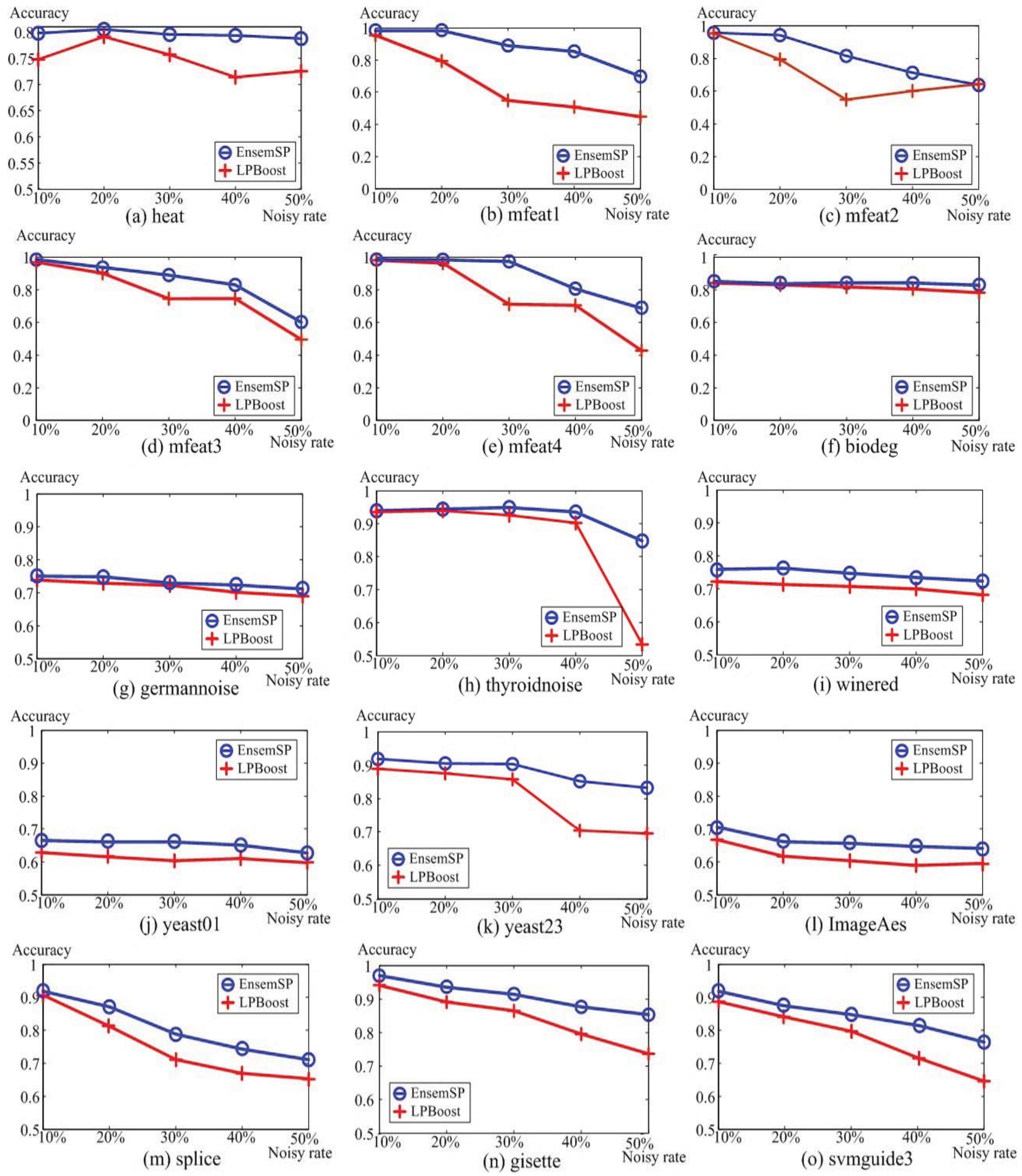

Fig. 2. The variation of classification accuracies in terms of the proportions of noisy labels on validation data.

TABLE 3
The used experimental training data for the involved methods.

Data parts	Base learners (SVM, RF, AdaBoost, C4.5, and LR)	Voting	LPBoost	EnsemSP
The first part (training data)	Used	Not used	Not used	Not used
The second part (validation data)	Not used	Not used	Used	Used

TABLE 4
The classification accuracies of the competing ensemble methods when the parameters of the five basic learning algorithms are searched via cross validation.

Data set	RF	Voting	wVoting	LPBoost	EnsemSP
Heat	0.787	0.784	0.773	0.763	**0.828**
mfeat1	0.879	0.737	0.976	0.985	**0.996**
mfeat2	0.947	0.953	0.963	**0.979**	**0.979**
mfeat3	0.982	0.985	0.968	**0.999**	**0.999**
mfeat4	0.735	0.583	0.954	0.990	**0.997**
biodeg	0.8770	0.8513	**0.8926**	0.8511	0.8740
germannoise	0.7198	0.7551	0.7554	0.7660	**0.7699**
hillvalleynoise	0.5201	0.5660	0.5648	0.5660	**0.5688**
hillvalleynonoise	0.5131	0.4991	0.5185	0.5017	**0.5394**
thyroidnoise	0.946	0.940	0.959	**0.968**	**0.968**
winered	0.7453	0.7693	0.7652	0.7553	**0.7771**
Yeast01	0.631	0.662	0.662	0.677	**0.686**
Yeast23	0. 882	0.916	0.918	0.917	**0.923**
ImageAes	0.6748	0.6693	0.6765	0.6858	**0.6912**
splice	0.8514	0.9075	**0.9328**	0.9272	0.9285
gisette	0.9671	0.9707	0.9768	0.9685	**0.9806**
svmguide3	0.8892	0.8943	0.9007	0.9204	**0.9320**

TABLE 6
The classification accuracies of the competing ensemble methods when the label-flapping proportions of both the training and validation sets are set as 30%.

Data set	RF	Voting	wVoting	LPBoost	EnsemSP
heat	0.603	0.634	0.626	0.707	**0.762**
mfeat1	0.577	0.618	0.651	0.572	**0.843**
mfeat2	0.845	0.909	**0.918**	0.584	0.799
mfeat3	0.876	**0.902**	0.896	0.761	0.851
mfeat4	0.699	0.493	0.572	0.650	**0.827**
biodeg	0.764	0.7526	0.7675	0.7825	**0.7910**
germannoise	0.6613	0.6571	0.6624	0.6623	**0.6852**
thyroidnoise	0.861	0.876	0.906	**0.917**	**0.917**
winered	0.6963	0.7016	0.7036	0.6846	**0.7449**
Yeast01	0.591	0.602	0.625	0.585	**0.641**
Yeast23	0.842	0.831	0.833	0.812	**0.896**
ImageAes	0.6042	0.6121	0.6118	0.6011	**0.6570**
splice	0.6934	0.7583	0.7680	0.7144	**0.7856**
gisette	0.9255	0.9418	**0.9541**	0.8528	0.9135
svmguide3	0.8287	0.8253	0.8314	0.7760	**0.8337**

the performance of EnsemSP-reverse is equal to that of LPBoost because when the searched optimal value of the parameter λ_2 in EnsemSP-reverse is zero in the experiments, EnsemSP-reverse (and EnsemSP) is reduced to LPBoost.

4.4 Discussions

The above experimental results validate the effectiveness of the proposed EnsemSP. On most experimental data sets, EnsemSP outperforms the classical ensemble methods (majority voting and LPBoost). The learned similarity metric is useful in EnsemSP. Based on the learned similarity measurement function, the classification accuracies of EnsemSP are higher than those of the method without similarity measurement and the method by using a simple cosine similarity measurement on most data sets. Moreover, EnsemSP is determined to be robust even under varying parameter settings. EnsemSP is superior to LPBoost even when the labels are noisy.

Ensemble learning with SOI has been initially explored; hence, there are certain limitations of the proposed method. For example, the SOI is inferred only for pairwise classifiers. As previous studies may simultaneously record the performances of triple classifiers, SOI may be inferred for some triple classifiers. In addition, only the performance ordering information in previous literature is used. However, the extent that one classifier is better than another classifier is not used.

5 Conclusions

In various machine learning techniques, supplementary information is achievable and has been utilized to improve learning performances. This study explores the implicit SOI for a set of trained classifiers to improve learning for the ensemble of the trained classifiers. Our study confronts two challenges, namely, the similarity measurement between two learning (classification) tasks and the pursuance of the optimal combination weights with the estimated SOI for classifiers to ensemble. To address these two challenges, a new learning method called EnsemSP is proposed. In EnsemSP, a new metric learning problem, namely, MLPU, is presented and solved. The learned measurement function can measure the similarity between two learning tasks according to the meta-features of their corresponding data sets. For the second challenge, the SOI is transformed into a set of ordered soft constraints for the classifiers to ensemble. Inspired by goal programming, this study establishes a new optimization problem. Solving this problem can yield the optimal ensemble weights. The experimental results verify the effectiveness of the proposed method.

In this study, SOI is only for pairwise classifiers. In the future, SOI for triple classifiers will be explored. In addition, further investigation will be conducted on mining the historical performance data for existing learning algorithms to benefit for the new learning algorithm design.

Appendix A: The solving details of (14)

Given a soft constraint (e.g., $f(x) \geq t$), goal programming introduces two additional variables, namely, negative deviation (ϕ) and positive deviation (ω). A soft constraint is then transformed into the following form:

$$f(x) + \phi - \omega = t. \tag{15}$$

The variable ϕ quantifies the under-satisfaction of the constraint and ω quantifies the over-satisfaction of the constraint. If $\phi > 0$, then $\omega = 0$ and the constraint is not satisfied; if $\omega > 0$, then $\phi = 0$ and the constraint is satisfied. In general, the constraint is particularly satisfied if the value of ϕ is low. By introducing negative and positive derivations into (15), it can be transformed into the following form:

$$\begin{aligned} \min_{\rho,\phi_n,\phi_{r+N},\boldsymbol{\beta}} \; & -\rho + \tfrac{\lambda_1}{N}\sum_n \phi_n + \tfrac{\lambda_2}{R}\sum_r \mathcal{Q}_r \cdot \phi_{r+N} \\ s.t. \quad & y_n \boldsymbol{\beta}^T x_n + \phi_n - \omega_n = \rho \quad n = 1, \cdots, N \\ & \boldsymbol{\beta}(i) - \boldsymbol{\beta}(j) + \phi_{1+N} - \omega_{1+N} = 0 \quad i,j \in [1,H] \\ & \cdots \\ & \boldsymbol{\beta}(k) - \boldsymbol{\beta}(l) + \phi_{R+N} - \omega_{R+N} = 0 \quad k,l \in [1,H] \\ & \sum_{h=1}^{H} \boldsymbol{\beta}(h) = 1, \boldsymbol{\beta}(h) \geq 0 \\ & \phi_i \geq 0 \qquad \omega_i \geq 0 \qquad i = 1, \cdots, N+R \end{aligned}, \tag{16}$$

where λ_1 and λ_2 are the balance parameters that indicate the strength of the corresponding soft constraints; $\mathcal{Q}_r$ indicates the reliability degree of the rth soft constraint; ϕ_{r+N} is the variable that quantifies the under-satisfaction of the rth soft constraint. In this work, we set that the value of λ_2 is smaller than that of λ_1. The above problem can be solved with conventional mathematical optimization methods such as preconditioned conjugate gradients.

Acknowledgments

The author wishes to thank Mr. Lei Ma and Pinglong Zhao for the very useful comments and suggestions on the English writings of the paper. This work is partially supported by NSFC (61673377 and 61732011), and Tianjin AI Funding (17ZXRGGX00150).

References

[1] X. Z. Wang, R. Wang, H. M. Feng, and H. C. Wang, "A new approach to classifier fusion based on upper integral," *IEEE Transactions on Cybernetics*, vol. 44, no. 5, pp. 620–635, 2014.

[2] A. Kumar and B. Raj, "Unsupervised fusion weight learning in multiple classifier systems," *Computer Science*, 2015.

[3] Y. Zhu, W. Chen, and G. Guo, "Fusing multiple features for depth-based action recognition," *Acm Transactions on Intelligent Systems and Technology*, vol. 6, no. 2, pp. 1–20, 2015.

[4] T. T. Nguyen, T. T. T. Nguyen, C. P. Xuan, and W. C. Liew, "A novel combining classifier method based on variational inference," *Pattern Recognition*, vol. 49, no. C, pp. 198–212, 2015.

[5] E. Kannatey-Asibu, J. Yum, and T. H. Kim, "Monitoring tool wear using classifier fusion," *Mechanical Systems and Signal Processing*, vol. 85, pp. 651–661, 2017.

[6] L. Peng, L. Chen, X. Wu, H. Guo, and G. Chen, "Hierarchical complex activity representation and recognition using topic model and classifier level fusion." *IEEE transactions on bio-medical engineering*, vol. in press, 2017.

[7] Q. Chen, Z. Song, Y. Hua, Z. Huang, and S. Yan, "Hierarchical matching with side information for image classification," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2012, pp. 3426–3433.

[8] M. Xu, R. Jin, and Z.-H. Zhou, "Speedup matrix completion with side information: Application to multi-label learning," in *Advances in Neural Information Processing Systems (NIPS), 15*. MIT Press, 2013, pp. 2301–2309.

[9] C. C. Aggarwal, Y. Zhao, and P. S. Yu, "On text clustering with side information," in *IEEE 28th International Conference on Data Engineering*, 2012, pp. 894–904.

[10] Y. Zhao and P. S. Yu., "On graph stream clustering with side information," in *SIAM International Conference on Data Mining (SDM)*, 2013, pp. 139–150.

[11] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell, "Distance metric learning, with application to clustering with side-information," in *Advances in Neural Information Processing Systems (NIPS), 15*. MIT Press, 2003, pp. 505–512.

[12] L. Wu, S. C. Hoi, R. Jin, J. Zhu, and N. Yu, "Distance metric learning from uncertain side information for automated photo tagging," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 2, pp. 13:1–13:28, Feb. 2011.

[13] J. Hoffman, S. Gupta, and T. Darrell, "Learning with side information through modality hallucination," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 826–834.

[14] Z.-H. Zhou, *Ensemble methods : foundations and algorithms*. Boca Raton, FL: Chapman & Hall/CRC, 2012.

[15] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[16] D. A. Freedman, *Statistical Models: Theory and Practice*. Cambridge University Press, 2012.

[17] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–2, 2001.

[18] Q. Song, G. Wang, and C. Wang, "Automatic recommendation of classification algorithms based on data set characteristics," *Pattern Recogn.*, vol. 45, no. 7, pp. 2672–2689, 2012.

[19] G. Wang, Q. Song, X. Zhang, and K. Zhang, "A generic multilabel learning-based classification algorithm recommendation method," *ACM Trans. Knowl. Discov. Data*, vol. 9, no. 1, pp. 7:1–7:30, 2014.

[20] B. Liu, Y. Dai, X. Li, W. S. Lee, and P. Yu, "Building text classifiers using positive and unlabeled examples," in *Third IEEE International Conference on Data Mining (ICDM)*, Nov 2003, pp. 179–186.

[21] B. Liu, Y. Dai, X. Li, and W. S. Lee, "Learning from positive and unlabeled examples with different data distributions," in *European Conference on Machine Learning (ECML)*, Nov 2005, pp. 218–229.

[22] C. Elkan and K. Noto, "Learning classifiers from only positive and unlabeled data," in *Proceedings of the Fourteenth International Conference on Knowledge Discovery and Data Mining (KDD 08)*, 2008, pp. 213–220.

[23] C. Domshlak, F. Rossi, K. B. Venable, and T. Walsh, “Reasoning about soft constraints and conditional orderings: Complexity results and approximation techniques,” in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2003, pp. 215–220.

[24] P. Gehler and S. Nowozin, “On feature combination for multiclass object classification,” in *IEEE International Conference on Computer Vision (ICCV)*, 2009, pp. 221–228.

[25] M. Awais, F. Yan, K. Mikolajczyk, and J. Kittler, “Augmented kernel matrix vs classifier fusion for object recognition,” in *British Machine Vision Conference*, 2011, pp. 60.1–60.11.

[26] J. P. Ignizio, “Generalized goal programming an overview,” *Computers & Operations Research*, vol. 10, no. 4, pp. 277–289, 1983.

[27] A. Ma and P. Yuen, “Linear dependency modeling for feature fusion,” in *IEEE International Conference on Computer Vision (ICCV)*, Nov 2011, pp. 2041–2048.

[28] M.-L. Zhang and Z.-H. Zhou, “Exploiting unlabeled data to enhance ensemble diversity,” in *Proceedings of the 10th IEEE International Conference on Data Mining (ICDM)*, 2010, pp. 609–618.

[29] N. Li, Y. Yu, and Z.-H. Zhou, “Diversity regularized ensemble pruning,” in *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, 2012, pp. 330–345.

[30] J. Gao, F. Liang, W. Fan, Y. Sun, and J. Han, “Graph-based consensus maximization among multiple supervised and unsupervised models,” in *Advances in Neural Information Processing Systems (NIPS)*, 2009, pp. 585–593.

[31] J. Gao, F. Liang, and W. Fan, “A graph-based consensus maximization approach for combining multiple supervised and unsupervised models,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 99, pp. 15–28, 2013.

[32] D. H. Wolpert, “Stacked generalization,” *Neural Networks*, vol. 5, pp. 241–259, 1992.

[33] G. Rätsch, B. Schölkopf, A. Smola, S. Mika, T. Onoda, and K.-R. Müller, “Robust ensemble learning for data mining,” in *The Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, 2000, pp. 341–344.

[34] Y. Freund and R. E. Schapire, “Experiments with a new boosting algorithm,” in *In Proceedings of International Conference on Machine Learning (ICML)*, 1996, pp. 148–156.

[35] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, “Extreme learning machine for regression and multiclass classification,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 2, pp. 513–529, 2012.

[36] G. Druck, B. Settles, and A. Mccallum, “Active learning by labeling features.” in *Conference on Empirical Methods in Natural Language Processing, EMNLP*, 2009, pp. 81–90.

[37] C. Zhang, Q. Cai, and Y. Song, “Boosting with pairwise constraints,” *Neurocomputing*, vol. 73, no. 4, pp. 908–919, 2010.

[38] E.-L. Hu and J. T. Kwok, “Efficient kernel learning from side information using admm,” in *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, ser. IJCAI ’13, 2013, pp. 1408–1414.

[39] J. Zhu, N. Chen, and E. P. Xing, “Bayesian inference with posterior regularization and applications to infinite latent svms,” *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1799–1847, Jan. 2014.

[40] B. Liu, X. Li, W. S. Lee, and P. S. Yu, “Text classification by labeling words,” in *Proceedings of the 19th National Conference on Artifical Intelligence*, ser. AAAI’04, 2004, pp. 425–430.

[41] K. Small, B. C. Wallace, C. E. Brodley, and T. A. Trikalinos, “The constrained weight space svm: Learning with ranked features,” in *International Conference on Machine Learning (ICML)*, 2011, pp. 754–763.

[42] J. P. Ignizio, “A note on computational methods in lexicographic linear goal programming,” in *The Journal of the Operational Research Society*, 1983, pp. 539–542.

[43] K. A. Smith-Miles, “Cross-disciplinary perspectives on meta-learning for algorithm selection,” *ACM Computing Surveys*, vol. 41, no. 1, pp. 6:1–6:25, 2009.

[44] D. Wolpert and W. Macready, “No free lunch theorems for optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[45] P. Brazdil, C. Giraud-Carrier, C. Soares, and R. Vilalta, *Meta-Learning: Applications to Data Mining*. Springer, 2009.

[46] L. Yang, R. Jin, and R. Sukthanka, “Experiments with a new boosting algorithm,” in *Proceedings of Conference on Uncertainty in Artificial Intelligence (UAI)*, 2007, pp. 442–449.

[47] L. Yang, “Distance metric learning: A comprehensive survey,” in *Michigan State University, Technical Report*, 2006.

[48] M. Fernández-Delgado, E. Cernadas, S. Barro, and D. Amorim, “Do we need hundreds of classifiers to solve real world classification problems?” *Journal of Machine Learning Research (JMLR)*, vol. 15, pp. 3133–3181, 2014.

[49] C. Ding and X. He, “K-means clustering via principal component analysis,” in *In Proceedings of International Conference on Machine Learning (ICML)*, 2004, pp. 225–232.

[50] M. L. Zhang and Z. H. Zhou, “Exploiting unlabeled data to enhance ensemble diversity,” *Data Mining and Knowledge Discovery*, vol. 26, no. 1, pp. 98–129, 2013.

[51] R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, 1993.

[52] H. Wang, Q. Xu, and Z. Lifeng, “A graph-based consensus maximization approach for combining multiple supervised and unsupervised models,” *PLoS ONE*, vol. 10, no. 2, p. e0117844, 2015.

Ou Wu received the BSc degree in Electrical Engineering from Xi’an Jiaotong University, China, in 2003, and the MSc degree and the PhD degree in Computer Science from the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences, China, in 2006 and 2012. From April 2007, he joined NLPR as an Assistant Professor. From February 2017, he joined Center for Applied Mathematics, Tianjin University as a full Professor. His research interests include data mining and machine learning.