Evolving Systems (2019) 10:51–61
DOI 10.1007/s12530-017-9203-y

ORIGINAL PAPER

Hybrid local boosting utilizing unlabeled data in classification tasks

Christos K. Aridas[1] · **Sotiris B. Kotsiantis**[1] · **Michael N. Vrahatis**[1]

Received: 25 January 2017 / Accepted: 24 September 2017 / Published online: 7 October 2017
© Springer-Verlag GmbH Germany 2017

Abstract In many real life applications, a complete labeled data set is not always available. Therefore, an ideal learning algorithm should be able to learn from both labeled and unlabeled data. In this work a two stage local boosting algorithm for handling semi-supervised classification tasks is proposed. The proposed method can be simply described as: (a) a two stage local boosting method, (b) which adds self-labeled examples of unlabeled data and (c) employ them on semi-supervised classification tasks. Grounded on the local application of the boosting-by-reweighting version of AdaBoost, the proposed method utilizes unlabeled data to enhance it's classification performance. Simulations on thirty synthetic and real-world benchmark data sets show that the proposed method significantly outperforms nine other well-known semi-supervised classification methods in terms of classification accuracy.

1 Introduction

Supervised machine learning methods usually require a large number of labeled training data in order to build models with high predictive power and generalization ability. However, since the process of data labelling often requires a human expert, this can be time consuming and expensive. On the other hand, unlabeled data sets are fairly easy to obtain and have been used in a number of research areas including web mining, image classification and text mining, among others.

Semi-supervised classification methods (Zhu and Goldberg 2009) exploit both labeled and unlabeled data during their training process. The ultimate goal of any semi-supervised method is to take advantage of the unlabeled data in order to improve its generalization ability. A variety of semi-supervised learning approaches have been proposed in the literature. They can be categorized as follows: (a) self-labeled methods (Triguero et al. 2015), (b) low-density separation (Chapelle et al. 2010), (c) graph-based methods (Blum and Chawla 2001) and (d) generative models (Nigam et al. 2000).

In the work at hand, a semi-supervised method that extends the local application of AdaBoost (Kotsiantis et al. 2006) by adding self-labeled examples in the training set is presented. The aim of this research work is to propose a method whose performance is significantly increased by using unlabeled data while outperforming other well-known semi-supervised classification methods. The proposed method has been tested on various standard benchmark data sets. From the obtained results, it has been observed that the proposed method tends to outperform other well-known and widely used semi-supervised classification methods in terms of classification accuracy.

The rest of the paper is organized as follows: In Sect. 2 some of the most well-known semi-supervised classification methods are briefly reviewed. In Sect. 3 the proposed algorithm is presented and analyzed. In Sect. 4 experimental results obtained using thirty benchmark data sets are exhibited. The paper ends in Sect. 5 with a synopsis and concluding remarks.

2 Related work

For completeness purposes let us briefly describe some related work. As mentioned in Sect. 1, one main category of

✉ Christos K. Aridas
char@upatras.gr

Sotiris B. Kotsiantis
sotos@math.upatras.gr

Michael N. Vrahatis
vrahatis@math.upatras.gr

[1] Computational Intelligence Laboratory (CILab), Department of Mathematics, University of Patras, Patras 26110, Greece

semi-supervised learning classification methods are the self-labeled techniques (Triguero et al. 2015). In these methods, a small set of labeled instances is used in order to train a single classifier or an ensemble of classifiers. Then, the unlabeled instances are classified using the trained model. Iteratively, the trained learner picks some unlabeled instances based on the confidence of its predictions, e.g. the first instances after the ranking of class probability values. The classifier adds these instances to the training set along with their predicted class labels, in order to enlarge it. The new enlarged training set is used for re-training the classifier and this process is repeated until one or more stopping criteria are fulfilled.

One of the most simple algorithms is the Self-Training method (Yarowsky 1995; Riloff et al. 2003), which enlarges its own training set based on its predictions of unlabeled data without any other restrictions. The performance of the Self-Training algorithm strongly depends on the selected newly-labeled data at each iteration of the training procedure. Tanha et al. (2015) have observed that the most important aspect of the Self-Training procedure, in order to be successful, is to correctly estimate the confidence of the predictions. They have conducted experiments using decisions trees as base learners and they have revealed that Self-Training does not benefit from the available unlabeled data. Furthermore, they have performed several modifications to the basic decision tree learner in order to produce better probability estimates and they have concluded that their modifications, as well as the ensembling of decision trees, benefit the Self-Training procedure. Nonetheless, classification of noisy instances as confident instances and their insertion into the training set can lead to wrong predictions. Furthermore, if the number of the labeled data is too small, it may not represent the underlying structure of the hypothesis space at all. Therefore, Self-Training will also fail since the initial trained learner will produce incorrect predictions for the unlabeled data. It must be noted that Self-Training cannot straightforwardly be applied to Support Vector Machines because the confident instances might not be too informative, as most of them would have a significant distance from the decision boundary. Despite the above, Self-Training processes have been successfully applied to several classification tasks including natural language processing (Riloff et al. 2003), human action recognition (Liu and Yuen 2011), object detection from images (Rosenberg et al. 2005), motion estimation in dynamical systems (Kim 2011), pixel classification of remote sensing imagery (Maulik and Chakraborty 2011), intrusion detection (Li et al. 2010), brain-computer interface system (Li et al. 2008) and face recognition (El Gayar et al. 2006), among others.

SETRED (Li and Zhou 2005) is a Self-Training method that incorporates data editing in order to learn actively from the self-labeled examples. In each Self-Training iteration, SETRED does not accept all the self-labeled examples that may be highly noisy. Instead, it identifies the possibly mislabeled objects from the self-labeled objects by testing a predefined null hypothesis with the local cut edge weight statistic associated with each self-labeled object. If the test indicates a left rejection, the object is regarded as a good object, otherwise it is possibly associated with a wrong label, which should be kept from being added to the learner's training set.

Co-Training (Blum and Mitchell 1998; Nigam and Ghani 2000) is based on the strong assumption that the feature space can be divided into two conditionally independent subsets and each subset is sufficient to train a good classifier. Thus, each classifier is trained using each subset, respectively. Initially, only labeled instances are used for training. Iteratively, each classifier makes predictions for a few unlabeled instances of its sub-dataset and the most confident predictions of each classifier are inserted into the training set of the other classifier. This process continues for a predefined number of iterations until stopping criteria are fulfilled. Didaci et al. (2012) have estimated that the Co-Training performance depends on the size of the labeled training set. Results on real data sets have shown that co-training's performance does not seem to be affected greatly by the training set size. In other words, co-training can work even with very few labeled examples per class (Blum and Mitchell 1998).

While Co-Training algorithms do not examine the reliability of labels provided by each classifier, unsuccessful labelling in a small number of instances can lead to decreased accuracy of subsequent learners. To overcome this drawback, Sun and Jin (2011) have proposed the robust co-training. In this work, canonical correlation analysis (CCA) has been used to examine predictions on the unlabeled data and only predicted labels consistent with the result of CCA have been included into the training set.

Additionally, the selection of the most confident predictions in each iteration relies on their class probability estimates. In some cases, when unlabeled instances have the same class probability values, they will be chosen at random. In order to choose these instances, Wang et al. (2008) have used the class membership probabilities of each instance with a distance metric between unlabeled and labeled instances. Between two instances with the same class membership probability, the one with the smallest distance is chosen to be selected with greater chance.

To overcome the strong assumptions of co-training Tanha et al. (2011) have proposed a method, named ensemble-co-training, that uses an ensemble of classifiers for co-training rather than feature subsets. The ensemble is used to estimate the probability of incorrect labeling and this is used with a theorem by Angluin and Laird (1988) to obtain a degree for deciding if adding a set of unlabeled isntances will reduce the error of an underlying classifier or not.

Xu et al. (2012) have proposed the DCPE co-training, where labeled instances are used for training two different

base learning algorithms. Then, each generated classifier independently predicts the unlabeled instances as well as their class membership probabilities. Unlabeled instances with the same prediction label but with different, most probable, class probabilities by each classifier are inserted into the training set of the classifier which exhibits the lower probability value. Next, each classifier is retrained and this process is repeated until stopping criteria are fulfilled.

CoForest (Li and Zhou 2007) uses unlabeled instances to boost the performance of the model that is trained from the labeled instances. By extending the co-training procedure, it exploits the power of Random Forest to tackle the problem of selecting confident unlabeled instances to label and generate the final hypothesis. Deng and Guo (2011) have proposed a variant of Co-Forest algorithm named ADE-coforest, which is based on a data editing process which detects and removes probable mislabeled instances during iterations.

Hady and Schwenker (2008) have proposed a co-training by committee framework. In their work, an initial committee was built with the available labeled instances. Three ensemble methods have been used: (a) Random Subspace, (b) Bagging and (c) AdaBoost, and these semi-supervised learning methods were named CoBagging, CoAdaBoost and CoRSM, respectively. In this framework, each ensemble predicts the label of a subset of the unlabeled data set and computes their class membership probabilities. Subsequently, a few of the most confident instances are inserted into the labeled training set and the ensemble is retrained. The process is repeated until a stopping criterion is fulfilled.

TriTraining algorithm (Zhou and Li 2005) extends the Co-Training method using three base classifiers which iteratively assign labels to unlabeled instances. In each round, an unlabeled instance is labeled for a base classifier, when the other two classifiers agree on the label prediction of an unlabeled instance. TriTraining does not put any constraint on which supervised learning algorithm is chosen as the base learner, neither assumes that a feature split exists. Therefore, its applicability is quite general. Guo and Li (2012) have proposed an improved TriTraining algorithm (im-tri-training) that addresses some issues which exist in TriTraining, such as unsuitable error estimation.

RASCO (Wang et al. 2008) uses random feature subsets in order to train different classifiers. An unlabeled instance is assigned to a label, based on the combination of the predictions of the classifiers that are trained on the different feature subsets. Rel-RASCO algorithm (Yaslan and Cataltepe 2010) generates relevant random subspaces by using relevance scores of features which are collected using the mutual information between features and class labels, instead of using random feature subspaces.

Aridas and Kotsiantis (2015) have proposed a hybrid technique that uses two models that are trained using Random Forests (Breiman 2001) and Support Vector Machines (Cristianini and Shawe-Taylor 2000), respectively. Initially, both models are trained with the available labeled instances. Afterwards, iteratively, both models are evaluated using cross-validation in the labeled data and the best performer is selected to label the most confident instances from the remaining unlabeled set. This process is repeated until all instances from the unlabeled set have a label. In their implementation, a fixed number of iterations has been used and thus, the number of instances that have been labeled in each iteration have been dynamic.

Most of the proposed self-labeled methods (Triguero et al. 2015) implement an incremental addition mechanism in order to enlarge the initially labeled data set. The main disadvantage of this approach is that in the case where mislabeled instances are added, the latter cannot be removed from the enlarged data set. So, in the next iterations the underlying classifier(s) would be trained using noisy instances and that could lead to a decrement of the performance. The proposed approach belongs to the category of the amending methods that appear to be a solution to the main weakness of the strictly incremental methods (Triguero et al. 2015).

3 Proposed method

Let us recall that Boosting constructs an ensemble of classifiers by subsequently tweaking the distribution of the training set based on the accuracy of the previously created classifiers. There are several boosting variants. These methods assign a weight to each training instance. Initially, all instances are equally weighted. In each iteration a new classification model, named base classifier, is created using the base learning algorithm. The creation of the base classifier has to consider the weight distribution. Then, the weight of each instance is adjusted, depending on the accuracy of the prediction of the base classifier for that instance. Thus, Boosting attempts to construct new classifiers that are able to better classify the "hard" instances for the previous ensemble members. The final classification is obtained from a weighted vote of the base classifiers. AdaBoost (Freund and Schapire 1996) is the most well-known boosting method.

Kotsiantis and Pintelas (2004) have applied local boosting with AdaBoost using Decision Stumps (Iba and Langley 1992) and OneRs (Holte 1993) as base classifiers and they have shown that their approach outperforms other ensembles with the same learners as base classifiers, in most of the cases. Their approach is based on the idea of local learning algorithms (Bottou and Vapnik 1992). For each testing pattern their approach performs the following steps: (a) select few training patterns located in the near of the testing pattern, (b) apply boosting to a base classifier only with these few examples, and (c) apply the resulting ensemble to the testing pattern itself. On the other hand, Zhang and Zhang (2008) have shown empirically that their local boosting-by-resampling technique seems

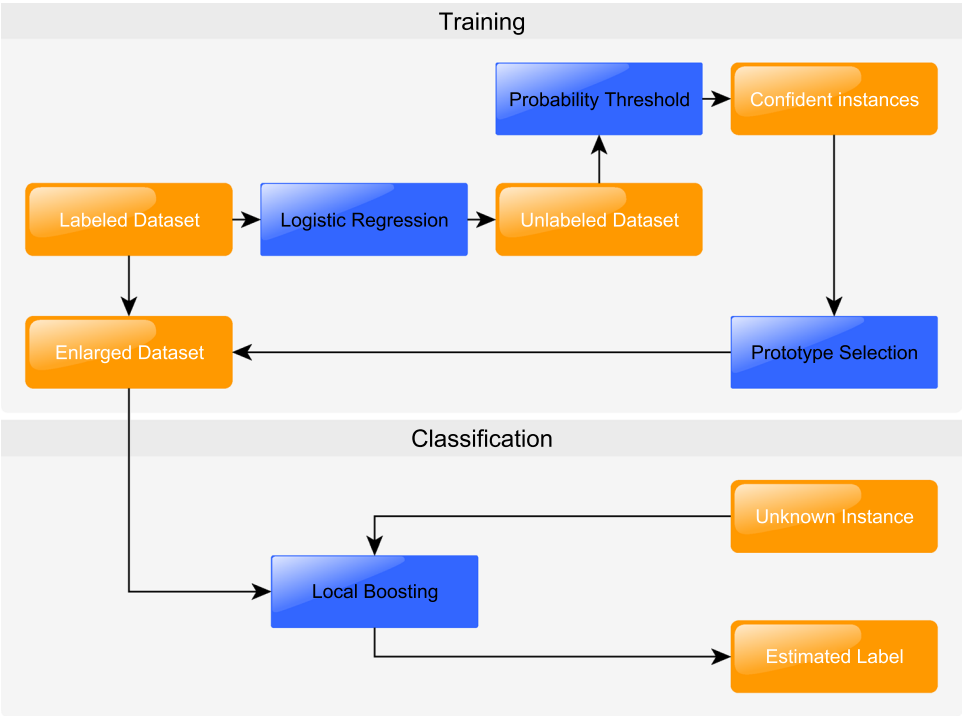

Fig. 1 The proposed hybrid local boosting (HLB) process diagram

to be more robust to noise than the standard AdaBoost. The success of the local boosting, as any other "lazy" algorithm, depends on the available training data set. In Aridas et al. (2016), the authors have achieved to enhance the performance of local boosting by using a data editing technique (Garcia et al. 2012) in the whole available training data set.

By assuming that the limited training labeled set is noise free and that if two points x_1, x_2 are close together, then their corresponding outputs y_1, y_2 are also close together, we propose a modification of the algorithm that has been proposed by Aridas et al. (2016) in order to solve semi-supervised classification tasks.

In the supervised case the local boosting needs a high quality training set in order to locate the neighbors for each testing pattern. The intention of the proposed method is to add high quality patterns from unlabeled data by using a Logistic Regression (LR) (Freedman 2009) model. Specifically, given a labeled set L and an unlabeled set U, a LR model is trained using the L, in one-vs-rest fashion in order to handle the multiclass case. Then the LR model is used to generate class probabilities for each instance x in the set U. All instances that have max class membership probability higher than a predefined acceptance threshold T take the label of the class with the highest probability and then they construct a new set $\bar{L}$. Since the set $\bar{L}$ may include mislabeled examples, a prototype reduction is performed, by removing the newly labeled examples that do not agree with the majority of their K_1 neighbors. So, the main difference with the method that has been proposed by Aridas et al. (2016), is that the initial labeled data will never be removed. This complies with the assumption of a noise free initial data set. After the reduction of the newly labeled examples in $\bar{L}$, L and $\bar{L}$ are concatenated and the new set is stored in order to be used for the predictions. In the classification phase, for each test instance, the K_2 nearest neighbors, from L and $\bar{L}$, are located and a boosting ensemble is yielded using only the K_2 instances. This local ensemble is used to predict the class of the test instance.

The proposed Hybrid Local Boosting (HLB) method is illustrated in the process diagram of Fig. 1, while the entire algorithmic procedure is presented in Algorithm 1.

Algorithm 1 The proposed Hybrid Local Boosting (HLB) Algorithm.

parameters
  Boosting Iterations N
  Learning Rate R
  Distance Metric M
  Neighbors K_1
  Neighbors K_2
  Threshold T
procedure TRAINING(Labeled Set L, Unlabeled Set U)
  Train a Logistic Regression model using L
  Generate class probabilities for the U using the trained model
  Discard the objects that have max class probability below T
  Construct a new set $\bar{L}$ with the remained objects
  for $\forall\, x \in \bar{L}$ **do**
    Find the K_1 nearest neighbors using M
    if the label of x does not agree with the majority of the K_1 **then**
      $\bar{L} \leftarrow \bar{L} - \{x\}$
    end if
  end for
  $L_{\text{ext}} \leftarrow L \cup \bar{L}$
end procedure
procedure CLASSIFICATION(Test Set T)
  for $\forall\, x \in T$ **do**
    Find the K_2 nearest neighbors of x in L_{ext} using M
    Apply AdaBoost for N iterations using the K_2 of L_{ext}
    Predict the class of x using the trained ensemble
  end for
end procedure

Table 1 Features of the data sets used in the experiments

Data set	#Attributes	#Instances	#Classes
Australian	14	690	2
Banana	2	5300	2
Breast	9	277	2
Bupa	6	345	2
Chess	36	3196	2
Contraceptive	9	1473	3
Dermatology	34	358	6
Flare	11	1066	6
German	20	1000	2
Heart	13	270	2
Iris	4	150	3
Marketing	13	6876	9
Mushroom	22	5644	2
Nursery	8	12960	5
Page-blocks	10	5472	5
Penbased	16	10992	10
Phoneme	5	5404	2
Pima	8	768	2
Ring	20	7400	2
Satimage	36	6435	7
Sonar	60	208	2
Spambase	57	4597	2
Splice	60	3190	3
Texture	40	5500	11
Tic-tac-toe	9	958	2
Titanic	3	2201	2
Twonorm	20	7400	2
Vehicle	18	846	4
Wine	13	178	3
Wisconsin	9	683	2

4 Numerical experiments

In this section, the design of the experiments as well as the results and the statistical analysis of the experiments are presented. Specifically, in Sect. 4.1 the evaluation protocol is described as well as the data sets that are used and the other semi-supervised techniques that are included in the comparisons are presented. In Sect. 4.2 the effect of the unlabeled data is discussed and analysed. While, in Sect. 4.3 the performance of the proposed method is compared with other well-known semi-supervised methods using different labeled ratios.

4.1 Design of experiments

In order to evaluate the performance of the proposed method, a number of experiments have been conducted, using several data sets from different domains. Specifically, thirty data sets have been chosen from the KEEL-dataset repository (Alcalá-Fdez et al. 2011). In Table 1 the name, the number of instances, the attributes as well as the number of different classes for each data set are exhibited.

All data sets have been partitioned using a tenfold cross-validation. This procedure divides the instances in 10 equal folds. Each tested algorithm has been trained using ninefolds (training partition) and the fold left out (testing partition) has been used for evaluation of the algorithm in terms of classification accuracy. This has been repeated ten times. Then the average accuracy across all trials has been computed. Each training partition has been divided into two parts: labeled and unlabeled sets. The impact of the amount of labeled data has been examined under the labeled ratio of 10, 20 and 30%, respectively, when the training set is partitioned.

Also, the proposed method is compared with nine other state-of-the-art algorithms as they are implemented with the KEEL (Alcalá-Fdez et al. 2008) software package. Specifically, the proposed method is compared with Self-Training (ST), Co-Training (CT), SETRED, TriTraining(TT), CoForest (CF), ADE-CoForest (ADECF), CoBagging (CB), RASCO (R) and Rel-RASCO(RR) with a Decision Tree (C4.5) (Salzberg 1994) as a base learner. In Table 2 the parameters for each

Table 2 Parameters used in the compared semi-supervised methods

Method	Parameters
Self-Training (ST)	MAX_ITER = 40
Co-Training (CT)	MAX_ITER = 40, Initial unlabelled pool = 75
SETRED	MAX_ITER = 40, Threshold = 0.1
TriTraining (TT)	C4.5
CoForest (CF)	Number of classifiers = 6, Threshold = 0.75
Rasco (R)	MAX_ITER = 40, Number of views/classifiers = 30
Co-Bagging (CB)	MAX_ITER = 40, Committee members = 3, Pool U = 100
Rel-Rasco (RR)	MAX_ITER = 40, Number of views/classifiers = 30
ADE-CoForest (ADECF)	Number of classifiers = 6, Threshold = 0.75, Neighbors k = 3, Minimum number of neighbors=2
C4.5	Pruned tree, confidence = 0.25, 2 examples per leaf

Table 3 Performance percentage change between a model, based on the proposedapproach, that exploits unlabeled data and a model that uses only the labeled data under different labeled ratios

Labeled ratio	10%			20%			30%		
Data set	$L+U$	L	Impr.	$L+U$	L	Impr.	$L+U$	L	Impr.
Australian	84.64	78.84	**7.35**	84.49	78.70	**7.37**	84.35	82.75	**1.93**
Banana	85.91	86.19	−0.33	87.70	87.60	**0.11**	87.98	88.02	−0.04
Breast	69.56	70.87	−1.84	73.31	67.45	**8.69**	72.64	63.88	**13.72**
Bupa	58.22	59.23	−1.69	60.07	57.97	**3.62**	61.33	64.02	−4.21
Chess	95.37	93.12	**2.42**	96.46	96.09	**0.39**	97.40	97.62	−0.23
Contraceptive	47.92	45.35	**5.68**	48.26	46.84	**3.04**	49.35	47.59	**3.72**
Dermatology	80.95	75.16	**7.7**	91.10	86.00	**5.93**	92.13	91.30	**0.91**
Flare	69.98	71.20	−1.71	73.17	73.55	−0.52	72.98	72.24	**1.03**
German	71.10	68.40	**3.95**	70.80	68.10	**3.96**	70.80	67.00	**5.67**
Heart	78.89	72.59	**8.67**	78.15	75.93	**2.93**	79.26	69.26	**14.44**
Iris	91.33	90.67	**0.74**	94.00	90.67	**3.68**	93.33	92.67	**0.72**
Marketing	27.66	27.95	−1.07	28.45	28.66	−0.76	30.77	29.07	**5.85**
Mushroom	99.77	99.80	−0.04	99.95	99.95	0	99.96	99.96	0
Nursery	90.65	89.21	**1.61**	91.57	91.79	−0.24	92.75	93.29	−0.59
Page-blocks	95.19	95.01	**0.19**	95.47	95.58	−0.11	95.72	95.91	−0.19
Penbased	95.46	95.10	**0.38**	97.08	97.15	−0.07	97.60	97.82	−0.22
Phoneme	80.79	81.35	−0.68	83.62	83.36	**0.31**	84.10	84.20	−0.11
Pima	68.35	62.36	**9.62**	72.78	66.80	**8.96**	72.40	73.06	−0.9
Ring	81.58	78.96	**3.32**	82.53	80.72	**2.24**	83.35	81.54	**2.22**
Satimage	85.41	84.65	**0.9**	86.26	86.26	0	87.44	87.23	**0.25**
Sonar	71.10	67.21	**5.77**	77.33	70.12	**10.29**	77.88	73.95	**5.31**
Spambase	89.97	85.73	**4.95**	90.91	87.27	**4.16**	91.04	88.86	**2.45**
Splice	82.23	80.66	**1.94**	84.51	83.70	**0.97**	85.02	83.79	**1.46**
Texture	92.36	88.47	**4.4**	95.20	92.09	**3.38**	96.11	94.47	**1.73**
Tic-tac-toe	79.64	72.97	**9.15**	87.99	75.15	**17.08**	92.48	76.93	**20.21**
Titanic	77.56	77.69	−0.18	78.69	78.33	**0.46**	78.83	78.87	−0.06
Twonorm	95.07	92.22	**3.09**	95.19	93.58	**1.72**	95.14	93.70	**1.53**
Vehicle	59.13	58.03	**1.89**	64.67	63.02	**2.62**	65.49	65.50	−0.01
Wine	88.76	85.29	**4.06**	93.79	82.58	**13.57**	97.75	89.90	**8.72**
Wisconsin	95.92	94.32	**1.7**	96.06	94.29	**1.88**	96.65	95.63	**1.07**

Bold values indicate positive improvement

compared method are presented. For the proposed method the parameters have been $N = 25, R = 1.0$ $M = Euclidean$, $K_1 = 5$, $K_2 = 25$, $T = 0.8$ and for the AdaBoost a Decision Stump (Iba and Langley 1992) has been used as base classifier.

4.2 Using only labeled data

The aim of our contribution is to propose a method that successfully utilizes unlabeled data in order to produce a model that performs better than a model that uses only the initially labeled data. In the case where there aren't any unlabeled data available during the training, the proposed method resorts to the local boosting approach as is described in Kotsiantis and Pintelas (2004). In Table 3 the performance of the proposed method under different ratios of labeled data is presented.

Table 4 Results obtained by the Wilcoxon tests

VS (%)	R^+	R^-	Z	Asymptotic p value
10	411.0	54.0	−3.673	0.000223
20	431.5	33.5	−4.122	0.000041
30	362.5	72.5	−3.136	0.001514

In column L, the classification accuracy of the method using only the initial labeled data is reported, while in column $L+U$, the performance of the proposed method using labeled and unlabeled data at the same time is reported. Column *Impr.* show the percentage change of the performance by using unlabeled data. It is clear that under the labeled ratio of 10% the proposed method successfully utilizes unlabeled data since it achieves a better performance in 22 out of the 30 data sets. This number increases

Table 5 Classification accuracy under the labeled ratio of 10%

Data set	Proposed	ST	CT	SETRED	TT	CF	R	CB	RR	ADECF
Australian	**84.64**	82.75	83.48	80.43	84.49	84.06	76.52	82.75	77.39	83.33
Banana	85.91	84.79	84.81	**86.38**	84.81	52.7	84.94	85.53	84.26	55.21
Breast	69.56	72.16	67.74	68.35	72.16	**73.39**	70.82	72.52	69.39	71.96
Bupa	58.22	53.92	57.39	53.39	57.42	58.51	58.27	**61.19**	57.19	55.38
Chess	95.37	95.43	95.15	81.04	**95.78**	94.4	95.18	95.43	94.74	83.54
Contraceptive	47.92	**48.86**	44.6	41.48	48.13	48.53	46.57	48.27	43.85	43.86
Dermatology	80.95	85.62	84.29	**91.82**	88.16	90.47	37.59	87.6	37.94	85.65
Flare	69.98	**72.14**	57.42	64.45	71.58	40.24	58.63	71.4	60.89	36.49
German	71.1	70.6	69	66.6	**71.7**	68.6	70.50	71.1	69.10	68.8
Heart	**78.89**	67.78	70	74.44	71.48	69.26	69.26	70.37	65.19	78.52
Iris	91.33	84	84.67	91.33	72.67	**93.33**	52.00	80	60.67	92
Marketing	27.66	28.45	28	26	26.94	29.23	27.07	27.06	26.49	**29.75**
Mushroom	**99.77**	99.66	99.68	99.45	99.55	90.84	99.41	99.54	99.38	90.78
Nursery	**90.65**	90.64	90.34	81.01	90.39	38.09	62.58	90.06	55.78	38.09
Page-blocks	95.19	95.23	94.92	93.59	95.61	**95.85**	90.97	95.67	91.26	94.02
Penbased	95.46	89.16	89.57	**97.78**	90.27	95.51	86.69	90.49	87.44	95.81
Phoneme	**80.79**	77.7	76.52	80.46	77.7	80.07	76.94	78.89	76.72	78.96
Pima	**68.35**	66.43	67.04	65.65	65.64	66.27	66.28	63.42	59.65	67.87
Ring	81.58	83.96	83.66	66.91	85.42	**88.23**	82.14	85.82	81.36	61.04
Satimage	85.41	80.45	80.56	85.7	82.24	**86.0**	77.84	82.05	77.51	85.8
Sonar	71.1	64.33	58.19	66.33	70.19	**75.5**	58.12	70.14	60.07	62.45
Spambase	89.97	86.69	88.84	82.81	88.1	**91.86**	87.14	89.51	87.16	85.86
splice	82.23	82.66	**83.1**	69.97	82.54	50.66	79.84	82.48	79.62	48.43
Texture	92.36	83.05	82.89	**95.13**	85.24	90.65	77.75	85	78.31	91.22
Tic-tac-toe	**79.64**	71.08	69.31	72.55	70.88	59.71	70.98	70.35	68.38	62.53
Titanic	77.56	77.51	77.83	64.02	77.65	70.65	77.24	**78.37**	77.69	61.88
Twonorm	**95.07**	81.36	80.85	93.58	86.16	89.89	79.86	85.97	81.08	91.64
Vehicle	59.13	57.92	57.47	58.28	**61.94**	61.24	48.23	60.3	47.87	57.09
Wine	88.76	74.05	80.75	**94.38**	82.03	85.88	55.98	78.66	58.46	88.73
Wisconsin	**95.92**	90.93	90.64	94.78	93.12	93.58	86.15	92.84	86.87	95.63

Bold values indicate superior performance

when the labeled ratio is 20% since the proposed approach performs equally or better in 25 out of 30 data sets. This performance is decreased when the labeled ratio reaches the level of 30%, but even so, the unlabeled data helps the local boosting to achieve a higher performance in most of the cases.

However, the results, of the pairwise comparisons using the Wilcoxon (1945) test in Table 4 indicate that the proposed approach performs significantly better[1] if unlabeled data are present.

4.3 Comparisons with other semi-supervised methods

In this Subsection, the performance of the proposed method is reported and analyzed against other state-of-the-art self-labeled techniques.

Tables 5, 6 and 7 show that the hypotheses generated by the proposed method are apparently better for all the labeled ratios since the proposed method algorithm has the best accuracy score in most of the cases. According to Demšar (2006) and Trawiński et al. (2012), non-parametric tests should be preferred instead of parametric ones in the context of machine learning problems, since they do not assume normal distributions or homogeneity of variance, especially when the number of the test cases is less than 30. Thus, in order to validate the significance of the results, the Friedman test (Demšar 2006), which is a rank-based non-parametric test for comparing several machine learning algorithms on multiple data sets, has been used, having as control method the proposed algorithm. The null hypothesis of the test states that all the algorithms perform equivalently and therefore their ranks should be equal. The average rankings, according to the Friedman tests for the different labeled ratios, are presented in Table 8.

[1] Based on negative ranks

Table 6 Classification accuracy under the labeled ratio of 20%

Data set	Proposed	ST	CT	SETRED	TT	CF	R	CB	RR	ADECF
Australian	84.49	85.8	83.77	83.04	85.22	84.2	76.52	83.62	77.39	**85.94**
Banana	87.7	**88.04**	87.43	86.91	87.36	54.11	84.94	88.04	84.26	55.17
Breast	**73.31**	71.56	72.27	67.88	72.25	72.65	70.82	72.24	69.39	68.86
Bupa	60.07	60.55	60.55	56.55	**61.99**	60.23	58.27	60.26	57.19	57.73
Chess	96.46	97.78	97.59	84.89	**97.81**	95.06	95.18	97.78	94.74	88.33
Contraceptive	48.26	47.79	**50.44**	43.79	48.81	49.42	46.57	48.06	43.85	46.1
Dermatology	91.1	91	87.9	**94.67**	92.14	92.44	37.59	91.86	37.94	93.85
Flare	**73.17**	72.8	68.97	66.6	72.7	41.56	58.63	71.76	60.89	39.97
German	70.8	69.7	69.9	66.2	68.4	68.4	70.50	**71.2**	69.10	69.3
Heart	78.15	75.19	75.56	77.41	**79.26**	72.96	69.26	74.81	65.19	75.19
Iris	**94.0**	89.33	89.33	92	88	94	52.00	86.67	60.67	91.33
Marketing	28.45	28.91	28.98	26.06	28.44	29.15	27.07	28.7	26.49	**30.64**
Mushroom	**99.95**	99.91	99.91	99.84	99.89	90.96	99.41	99.89	99.38	90.86
Nursery	91.57	92.35	**92.6**	83.27	92.58	38.42	63.25	92.35	59.42	38.42
Page-blocks	95.47	96.02	96.09	94.48	96.11	95.85	90.64	**96.13**	91.14	94.44
Penbased	97.08	92.41	92.47	**98.71**	92.87	96.56	86.69	93.7	87.44	97.15
Phoneme	**83.62**	78.39	80.24	83.44	79.77	83.05	76.94	82.44	76.72	81.57
Pima	**72.78**	68.1	68.74	63.69	69.39	71.08	66.28	70.84	59.65	68.62
Ring	82.53	86.58	86.3	70.07	87.95	**89.26**	82.14	89.18	81.36	67.55
Satimage	86.26	82.38	82.61	**87.4**	83.53	87.24	77.84	84.1	77.51	86.87
Sonar	**77.33**	66.36	63.36	70.12	66.31	74.48	58.12	65.86	60.07	67.24
Spambase	90.91	89.12	89.08	85.29	89.41	**93.08**	87.14	89.6	87.16	88.38
Splice	84.51	88.34	87.93	69.78	88.43	52.13	79.84	**88.75**	79.62	49.66
Texture	95.2	86.69	86.31	**97.13**	89.29	93.53	77.75	88.4	77.40	93.47
Tic-tac-toe	**87.99**	75.68	72.13	76.21	75.05	60.65	70.98	72.86	68.38	65.24
Titanic	**78.69**	78.24	78.24	64.07	78.15	72.29	77.24	78.28	77.69	64.38
Twonorm	**95.19**	81.65	82.76	94.11	86.74	90.32	79.86	87.41	81.08	91.49
Vehicle	64.67	64.89	64.89	62.53	**66.2**	65.49	48.23	65.49	47.87	63.84
Wine	**93.79**	83.69	78.63	91.54	84.22	84.22	55.98	82.55	58.46	91.6
Wisconsin	**96.06**	93.43	93.43	94.63	92.53	93.59	86.15	93.6	86.87	95.06

Bold values indicate superior performance

The p-value ($p < 10^{-6}$) of the Friedman tests indicates that the null hypothesis has to be rejected. So, there is at least one method that performs statistically different from the proposed method. With the intention of investigating the aforementioned, the post hoc procedure that is proposed by Li (2008) is used. Tables 9, 10 and 11 agree that the proposed algorithm performs significantly better almost than any compared algorithm, in most of the cases.

5 Conclusion

In this research work, a two-stage local boosting algorithm for handling semi-supervised classification tasks has been presented. Experiments on several standard benchmark data sets under different labeled ratios (10, 20 and 30%) show that the proposed method can take advantage of the presence of unlabeled data and significantly improve it's performance, especially when the available labeled data are very few. Also, it has been performed an in depth comparison with other well-known semi-supervised classification techniques and the results show that the proposed method exhibits the best performance in most of the cases.

A drawback of the proposed method is that for each test instance a local model is built and that it can be time consuming. A possible improvement could be to introduce a feature selection step in order to compact further the training data. The effect of the feature selection and the application of the proposed method in the multi-label classification are issues that deserve further study.

Table 7 Classification accuracy under the labeled ratio of 30%

Data set	Proposed	ST	CT	SETRED	TT	CF	R	CB	RR	ADECF
Australian	84.35	84.64	84.78	81.01	84.2	84.64	83.04	85.07	84.20	**85.36**
Banana	87.98	**88.19**	87.85	87	88.06	53.79	87.68	88.11	87.77	54.68
Breast	**72.64**	71.79	70.79	64.95	70	70.71	70.83	71.11	72.61	72.21
Bupa	61.33	58.42	61.37	58.48	58.5	**64.33**	62.00	61.59	61.49	57.56
Chess	97.4	**98.56**	98.40	86.48	98.43	96.53	98.22	98.19	98.25	89.71
Contraceptive	49.35	49.22	49.76	44.26	**51.05**	51.04	48.88	50.92	49.22	47.05
Dermatology	92.13	92.09	90.98	93.8	92.4	**96.33**	64.14	91.84	65.05	95.81
Flare	72.98	72.99	72.24	65.94	72.98	41.65	72.89	**73.08**	71.77	42.12
German	70.8	**71.0**	68.40	69.6	70.3	69.4	70.20	69.2	69.70	68.7
Heart	**79.26**	75.56	77.04	78.52	75.56	76.67	73.70	77.04	70.37	75.56
Iris	**93.33**	92	91.33	92.67	91.33	93.33	80.00	90.67	76.00	89.33
Marketing	**30.77**	28.72	29.44	26.83	28.93	30.1	28.55	29.42	28.71	30.39
Mushroom	**99.96**	99.96	99.96	99.91	99.95	91.07	99.86	99.91	99.82	91.07
Nursery	92.75	93.77	93.63	83.57	93.62	39.55	87.52	**93.82**	89.50	39.55
Page-blocks	95.72	96.35	96.18	94.61	96.35	**96.49**	92.89	96.22	93.28	94.85
Penbased	97.6	94.09	93.91	**99.01**	94.31	97.5	93.52	94.85	93.49	97.45
Phoneme	84.1	81.37	82.18	**84.7**	82.49	83.9	81.66	82.75	80.87	83.29
Pima	72.4	72.52	71.23	66.94	70.45	72.13	71.09	70.85	68.47	**73.18**
Ring	83.35	87.54	87.84	71.04	88.81	**90.27**	87.12	89.43	87.45	70.86
Satimage	87.44	82.7	84.07	**88.62**	84.38	88.3	83.11	85.28	83.37	88.08
Sonar	**77.88**	67.62	63.33	76.45	69.12	70.19	64.38	71.55	62.93	69.14
Spambase	91.04	89.99	90.52	86.95	90.73	**93.1**	90.04	91.17	89.99	89.82
Splice	85.02	91.66	91.76	71.54	91.57	51.54	91.47	**92.01**	91.16	49.56
Texture	96.11	88.91	89.71	**98.05**	90.55	94.96	88.13	91.15	87.67	94.64
Tic-tac-toe	**92.48**	76.1	74.95	79.23	76.41	63.26	76.82	77.45	75.78	65.04
Titanic	**78.83**	77.87	77.83	64.07	77.83	72.29	78.37	77.97	78.06	65.7
Twonorm	**95.14**	82.55	83.23	94.39	87.43	91.42	82.97	87.78	83.04	91.19
Vehicle	65.49	65.84	66.56	65.83	**67.02**	67.01	62.29	65.61	64.64	66.32
Wine	**97.75**	84.15	82.55	92.75	90.39	93.82	68.95	88.17	67.88	94.38
Wisconsin	**96.65**	94.43	94.74	95.35	95.02	94.76	94.02	94.31	92.97	95.47

Bold values indicate superior performance

Table 8 Friedman test rankings under the different labeled ratios

Algorithm	Labeled 10%	Algorithm	Labeled 20%	Algorithm	Labeled 30%
Proposed	3.1833	Proposed	2.9833	Proposed	3.3
TT	4.2167	CB	4.1833	CB	4.3667
CB	4.3833	TT	4.25	CF	4.75
CF	4.5667	ST	4.8667	TT	4.8667
ST	5.2667	CF	4.9167	ST	5.2833
SETRED	5.6333	CT	4.9333	CT	5.5667
CT	6.0167	SETRED	5.9333	SETRED	5.9833
ADECF	6.0833	ADECF	6.1	ADECF	6.1333
R	7.4833	R	8.2	R	7.2333
RR	8.1667	RR	8.6333	RR	7.5167
Statistic	68.256364	Statistic	92.774545	Statistic	48.585455
p value	$<10^{-6}$	p value	$<10^{-6}$	p value	$<10^{-6}$

Table 9 Adjusted p values obtained through the application of Li's post hoc procedure under the labeled ratio of 10%

i	Method	$p_{unadjusted}$	p_{Li}
1	RR	$<10^{-6}$	$<10^{-6}$
2	R	$<10^{-6}$	$<10^{-6}$
3	ADECF	0.000208	0.000255
4	CT	0.000290	0.000356
5	SETRED	0.001724	0.002114
6	ST	0.007699	0.009372
7	CF	0.076799	0.086235
8	CB	0.124773	0.132942
9	TT	0.186220	0.186220

Table 10 Adjusted p values obtained through the application of Li's post hoc procedure under the labeled ratio of 20%

i	Method	$p_{unadjusted}$	p_{Li}
1	RR	$<10^{-6}$	$<10^{-6}$
2	R	$<10^{-6}$	$<10^{-6}$
3	ADECF	0.000067	0.000076
4	SETRED	0.000161	0.000184
5	CT	0.012615	0.014209
6	CF	0.013394	0.015072
7	ST	0.015989	0.017941
8	TT	0.105162	0.107266
9	CB	0.124773	0.124773

Table 11 Adjusted p values obtained through the application of Li's post hoc procedure under the labeled ratio of 30%

i	Method	$p_{unadjusted}$	p_{Li}
1	RR	$<10^{-6}$	$<10^{-6}$
2	R	$<10^{-6}$	0.000001
3	ADECF	0.000290	0.000350
4	SETRED	0.000598	0.000722
5	CT	0.003737	0.004496
6	ST	0.011178	0.013327
7	TT	0.045061	0.051637
8	CF	0.063618	0.071385
9	CB	0.172415	0.172415

Acknowledgements We wish to thank all three anonymous referees for their valuable comments which helped to improve the manuscript.

References

Alcalá-Fdez J, Sánchez L, García S, del Jesus MJ, Ventura S, Garrell JM, Otero J, Romero C, Bacardit J, Rivas VM, Fernández JC, Herrera F (2008) KEEL: a software tool to assess evolutionary algorithms for data mining problems. Soft Comput 13(3):307–318. doi:10.1007/s00500-008-0323-y

Alcalá-Fdez J, Fernandez A, Luengo J, Derrac J, Garcia S (2011) KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. Multiple Value Logic Soft Comput 17(2–3):255–287

Angluin D, Laird P (1988) Learning from noisy examples. Mach Learn 2(4):343–370. doi:10.1023/A:1022873112823

Aridas C, Kotsiantis S (2015) Combining random forest and support vector machines for semi-supervised learning. In: Proceedings of the 19th Panhellenic Conference on Informatics, ACM, pp 123–128

Aridas CK, Kotsiantis SB, Vrahatis MN (2016) Combining prototype selection with local boosting, IFIP advances in information and communication technology, vol 475. Springer International Publishing, Switzerland, pp 94–105

Blum A, Chawla S (2001) Learning from Labeled and Unlabeled Data Using Graph Mincuts. In: Proceedings of the eighteenth international conference on machine learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, ICML '01, pp 19–26

Blum A, Mitchell T (1998) Combining Labeled and Unlabeled Data with Co-training. In: Proceedings of the eleventh annual conference on computational learning theory, ACM, Madison, Wisconsin, USA, COLT' vol 98, pp 92–100. doi:10.1145/279943.279962

Bottou L, Vapnik V (1992) Local learning algorithms. Neural Comput 4(6):888–900. doi:10.1162/neco.1992.4.6.888

Breiman L (2001) Random forests. Mach Learn 45(1):5–32. doi:10.1023/A:1010933404324

Chapelle O, Schlkopf B, Zien A (2010) Semi-supervised learning, 1st edn. MIT, Cambridge, MA

Cristianini N, Shawe-Taylor J (2000) An introduction to support vector machines: and other kernel-based learning methods. Cambridge University Press, New York

Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30

Deng C, Guo M (2011) A new co-training-style random forest for computer aided diagnosis. J Intell Inf Syst 36(3):253–281. doi:10.1007/s10844-009-0105-8

Didaci L, Fumera G, Roli F (2012) Analysis of co-training algorithm with very small training sets. In: Gimelfarb G, Hancock E, Imiya A, Kuijper A, Kudo M, Omachi S, Windeatt T, Yamada K (eds) Structural, syntactic, and statistical pattern recognition, vol 7626. Lecture notes in computer science. Springer, Berlin Heidelberg, pp 719–726

El Gayar N, Shaban SA, Hamdy S (2006) Face Recognition with Semi-supervised Learning and Multiple Classifiers. In: Proceedings of the 5th WSEAS international conference on computational intelligence, man-machine systems and cybernetics, world scientific and engineering academy and society (WSEAS), Venice, Italy, CIMMACS'06, pp 296–301

Freedman DA (2009) Statistical models: theory and practice. Cambridge University Press, Cambridge

Freund Y, Schapire RE (1996) Experiments with a new boosting algorithm. In: ICML, vol 96, pp 148–156. http://www.public.asu.edu/~jye02/CLASSES/Fall-2005/PAPERS/boosting-icml.pdf

Garcia S, Derrac J, Cano JR, Herrera F (2012) Prototype selection for nearest neighbor classification: taxonomy and empirical study. IEEE Trans Pattern Anal Mach Intell 34(3):417–435. doi:10.1109/TPAMI.2011.142

Guo T, Li G (2012) Improved tri-training with unlabeled data. In: Wu Y (ed) Software engineering and knowledge engineering: theory and practice, advances in intelligent and soft computing, vol 115. Springer, Berlin Heidelberg, pp 139–147

Hady M, Schwenker F (2008) Co-training by Committee: A New Semi-supervised Learning Framework. In: Data Mining Workshops,

2008. ICDMW '08. IEEE International Conference on, pp 563–572. doi:10.1109/ICDMW.2008.27
Holte RC (1993) Very simple classification rules perform well on most commonly used datasets. Mach Learn 11(1):63–90. doi:10.1023/a:1022631118932
Iba W, Langley P (1992) Induction of One-Level Decision Trees. In: Proceedings of the Ninth International Workshop on Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, ML '92, pp 233–240
Kim M (2011) Discriminative semi-supervised learning of dynamical systems for motion estimation. Pattern Recogn 44(1011):2325–2333. doi:10.1016/j.patcog.2010.12.002
Kotsiantis S, Pintelas P (2004) Local boosting of weak classifiers. In: Proceedings of intelligent systems design and applications (ISDA 2004), pp 26–28
Kotsiantis SB, Kanellopoulos D, Pintelas PE (2006) Local boosting of decision stumps for regression and classification problems. J Comput. doi:10.4304/jcp.1.4.30-37
Li J (2008) A two-step rejection procedure for testing multiple hypotheses. J Stat Plan Inference 138(6):1521–1527. doi:10.1016/j.jspi.2007.04.032
Li J, Zhang W, Li K (2010) A novel semi-supervised SVM based on tri-training for intrusition detection. JCP. doi:10.4304/jcp.5.4.638-645
Li M, Zhou ZH (2005) SETRED: self-training with editing. In: Ho T, Cheung D, Liu H (eds) Advances in knowledge discovery and data mining, vol 3518. Lecture notes in computer science. Springer, Berlin Heidelberg, pp 611–621
Li M, Zhou ZH (2007) Improve computer-aided diagnosis with machine learning techniques using undiagnosed samples. Syst Man Cybern Part A Syst Hum IEEE Trans 37(6):1088–1098. doi:10.1109/TSMCA.2007.904745
Li Y, Guan C, Li H, Chin Z (2008) A self-training semi-supervised SVM algorithm and its application in an EEG-based brain computer interface speller system. Pattern Recogn Lett 29(9):1285–1294. doi:10.1016/j.patrec.2008.01.030
Liu C, Yuen PC (2011) A boosted co-training algorithm for human action recognition. IEEE Trans Circ Syst Video Technol 21(9):1203–1213. doi:10.1109/tcsvt.2011.2130270
Maulik U, Chakraborty D (2011) A self-trained ensemble with semi-supervised SVM: an application to pixel classification of remote sensing imagery. Pattern Recogn 44(3):615–623. doi:10.1016/j.patcog.2010.09.021
Nigam K, Ghani R (2000) Analyzing the effectiveness and applicability of co-training. In: Proceedings of the ninth international conference on information and knowledge management, ACM, New York, NY, USA, CIKM '00, pp 86–93. doi:10.1145/354756.354805
Nigam K, McCallum AK, Thrun S, Mitchell T (2000) Text classification from labeled and unlabeled documents using EM. Mach Learn 39(2–3):103–134. doi:10.1023/A:1007692713085
Riloff E, Wiebe J, Wilson T (2003) Learning Subjective Nouns Using Extraction Pattern Bootstrapping. In: Proceedings of the seventh conference on natural language learning at HLT-NAACL 2003—Volume 4, Association for Computational Linguistics, Edmonton, Canada, CONLL '03, pp 25–32. doi:10.3115/1119176.1119180
Rosenberg C, Hebert M, Schneiderman H (2005) Semi-Supervised Self-Training of Object Detection Models. In: Application of computer vision, 2005. WACV/MOTIONS '05 volume 1. Seventh IEEE Workshops on, vol 1, pp 29–36. doi:10.1109/ACVMOT.2005.107
Salzberg SL (1994) C4.5: Programs for machine learning by J. Ross Quinlan. Morgan Kaufmann Publishers Inc, 1993. Mach Learn 16(3):235–240. doi:10.1007/BF00993309
Sun S, Jin F (2011) Robust co-training. Int J Pattern Recogn Artif Intell 25(07):1113–1126. doi:10.1142/S0218001411008981
Tanha J, v Someren M, Afsarmanesh H (2011) Disagreement-based co-training. In: 2011 IEEE 23rd International Conference on Tools with Artificial Intelligence, pp 803–810, doi:10.1109/ICTAI.2011.126
Tanha J, van Someren M, Afsarmanesh H (2015) Semi-supervised self-training for decision tree classifiers. Int J Mach Learn Cybern. doi:10.1007/s13042-015-0328-7
Trawiński B, Smetek M, Telec Z, Lasota T (2012) Nonparametric statistical analysis for multiple comparison of machine learning regression algorithms. Int J Appl Math Comput Sci. doi:10.2478/v10006-012-0064-z
Triguero I, Garcia S, Herrera F (2015) Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study. Knowl Inf Syst 42(2):245–284. doi:10.1007/s10115-013-0706-y
Wang J, Luo Sw, Zeng Xh (2008) A random subspace method for co-training. In: Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on, pp 195–200. doi:10.1109/IJCNN.2008.4633789
Wilcoxon F (1945) Individual comparisons by ranking methods. Biom-etr Bull 1(6):80. doi:10.2307/3001968
Xu J, He H, Man H (2012) DCPE co-training for classification. Neurocomputing 86:75–85. doi:10.1016/j.neucom.2012.01.006
Yarowsky D (1995) Unsupervised word sense disambiguation rivaling supervised methods. In: Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '95, pp 189–196. doi:10.3115/981658.981684
Yaslan Y, Cataltepe Z (2010) Co-training with relevant random subspaces. Neurocomputing 73(1012):1652–1661. doi:10.1016/j.neucom.2010.01.018
Zhang CX, Zhang JS (2008) A local boosting algorithm for solving classification problems. Comput Stat Data Anal 52(4):1928–1941. doi:10.1016/j.csda.2007.06.015
Zhou ZH, Li M (2005) Tri-training: exploiting unlabeled data using three classifiers. Knowl Data Eng IEEE Trans 17(11):1529–1541. doi:10.1109/TKDE.2005.186
Zhu X, Goldberg AB (2009) Introduction to semi-supervised learning. Synth Lect Artif Intell Mach Learn 3(1):1–130. doi:10.2200/s00196ed1v01y200906aim006