Intelligent Data Analysis 17 (2013) 387–398
DOI 10.3233/IDA-130585
IOS Press

Sharpened graph ensemble for semi-supervised learning

Inae Choi, Kanghee Park and Hyunjung Shin*
Department of Industrial Engineering, Ajou University, Suwon, Korea

Abstract. The generalization ability of a machine learning algorithm varies on the specified values to the model parameters and the degree of noise in the learning dataset. If the dataset has an enough amount of labeled data points, the optimal value for the model parameter can be found via validation by using a subset of the given dataset. However, for semi-supervised learning – one of the most recent learning algorithms, this is not as available as in conventional supervised learning. In semi-supervised learning, it is assumed that the dataset is given with only a few labeled data points. Therefore, holding out some of labeled data points for validation is not easy. The lack of labeled data points, furthermore, makes it difficult to estimate the degree of noise in the dataset. To circumvent the addressed difficulties, we propose to employ ensemble learning and graph sharpening. The former replaces the model parameter selection procedure to an ensemble network of the committee members trained with various values of model parameter. The latter, on the other hand, improves the performance of algorithms by removing unhelpful information caused by noise. The experimental results demonstrate the applicability of the proposed method for many real-world problems with no concern for the technical difficulties, by selecting the best parameter values and mitigating the influence of noise.

Keywords: Data mining, machine learning, semi-supervised learning, graph sharpening, ensemble learning, parameter selection, noise reduction

AUTHOR COPY

1. Introduction

In supervised learning, the performance of a model can be improved and stabilized as data with class labels become more available, since learning of the model has more chance to be correctly guided by the class labels [1]. However, it is often difficult, expensive, and time-consuming to collect the data with labels while unlabeled data are readily available or relatively easy to collect [2]. The research domains such as speech recognition, text categorization, parsing, video surveillance, and protein structure prediction, etc., may provide good examples for the case. For speech recognition, speech can easily be recorded from radio broadcasts; for text categorization, text documents can be collected from the Internet; for parsing, sentences are everywhere; for video surveillance, surveillance cameras run continuously; and for protein structure prediction, protein sequences are readily available from gene databases [2]. One may assume that those unlabeled data still give valuable information for learning. In order to utilize information from unlabeled data for learning, a new learning framework called semi-supervised learning (SSL) has been proposed [3–5]. This learning framework mainly deals with situations where labeled data are only a few while unlabeled data are given in a large quantity. Many related researches have shown its validity in a number of application domains such as spam filtering [2], document categorization [6], video

*Corresponding author: Hyunjung (Helen) Shin, Department of Industrial Engineering, Ajou University, San 5 Wonchun-dong, Yeongtong-gu, Suwon 443749, Korea. E-mail: shin@ajou.ac.kr.

1088-467X/13/$27.50 © 2013 – IOS Press and the authors. All rights reserved

surveillance [7], text classification [8], text chunking [9], time-series classification [10], gene expression data classification [11,12], visual classification [13], question-answering task for ranking candidate sentences [14], and webpage classification [15], etc. In those literatures, SSL is often compared with the representative models of supervised learning, and shows its superiority over them thanks to its capability of learning from only a few labeled data utilizing a large amount of unlabeled data.

Although the merit of SSL, one may encounter more difficulties with SSL (than with supervised learning) when facing to the issues of the model parameter selection and the robustness to data noise. The generalization ability of a model, irrespective of supervised learning or SSL, is dependent on the values specified for the model parameters and the degree of noise in the learning dataset [16]. As the parameter selection depends on the degree of noise, the two problems are often approached together as a single problem. Difficulty in model parameter selection is rather related to the complexity of the problem which is unlikely to be identified in advance than to the degree of noise itself since it still remains even after the noisy data are removed from the dataset. If the dataset has a sufficient amount of labeled data, the optimal value for the parameter can be found by trial-and-error by checking the validation performance. In supervised learning, cross-validation is generally used for this. However, for SSL, it is difficult to reserve some of the labeled data from the training set in order to make a separate set for validation because of lack of usable labeled data. Meanwhile, a higher degree of noise also complicates the problem and increases the model complexity. If the degree of noise is known in advance, overfitting to the noise can be prevented by imposing a greater penalty on a more complicated model. In supervised learning, even if the degree of noise is not known in advance, it can be estimated by checking the class impurity with the labeled data. In SSL, however, such noise estimation is not easily available because of the lack of labeled data as aforementioned. Therefore, we propose the application of ensemble learning and graph sharpening to circumvent the addressed difficulties. In ensemble learning, many models trained with different values of model parameters are combined, therefore it does not require parameter selection procedure to pick the best value of the parameter. Ensemble learning also has effect of performance improvement by reducing the bias or variance of the error of model estimation [17–20]. On the other hand, in graph sharpening– one of the most recently proposed methods in SSL, the noisy or corrupt information in the dataset is eliminated or reduced by taking into explicit account the values of relationship between data [21]. To rephrase this, the influence of noise in the dataset is alleviated or removed by graph sharpening.

The paper is organized as follows. In Section 2, the related work adopted for the proposed method is introduced. In Section 3, the basic idea of the proposed method is described. Section 4 provides the experimental results on an artificial problem, five benchmark problems, and a real-world problem on insurance company's marketing campaign. Concluding remarks are presented in Section 5.

2. Related work

2.1. Graph-based semi-supervised learning (SSL)

In many recent real world classification problems, the number of class-labeled data points is small because they are often difficult, expensive, or time-consuming to acquire, requiring qualified human annotators [7,22]. On the other hand, unlabeled data can easily be gathered and can provide valuable information for learning [23]. However, supervised-learning algorithms use only labeled data; therefore, they encounter difficulties when only a few labeled data are given. SSL uses both labeled and unlabeled

AUTHOR COPY

data to improve on the performance of supervised learning [3,23]. Among several types of SSL algorithms, a graph-based SSL is employed in our study. There are many graph-based SSL algorithms, e.g., mincut, Gaussian random fields and harmonic functions, local and global consistency, Tikhonov regularization, manifold regularization, graph kernels from the Laplacian spectrum, and tree-based Bayes [5]. In the graph-based SSL, a data point $x_i (i = 1, \ldots, n)$ is represented as a node i in a graph, and the similarity between data points is represented by an edge (see Fig. 1) [24]. The connection strength from each node j to each other node i is encoded in element w_{ij} of a weight matrix W. Often, a Gaussian function between points is used to specify the connection strength [25]:

$$w_{ij} = \begin{cases} \exp\left(-\dfrac{(x_i - x_j)^T (x_i - x_j)}{\sigma^2}\right) & \text{if } i \sim j, \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where $i \sim j$ stands for nodes i and j having an edge between them that can be established by k nearest neighbors (kNN) and k is a user-specified parameter. The labeled nodes have labels $y_l \in \{-1, 1\}$, whereas the unlabeled nodes have zeros $y_u = 0$. The algorithm will output an n-dimensional real-valued vector $f = [f_l^T \ f_u^T]^T = (f_1, \ldots, f_l, f_{l+1}, \ldots, f_{n=l+u})^T$, which can be thresholded to make label predictions on $f_{l=1}, \ldots, f_n$ after learning. The graph-based SSL assumes label smoothness over the graph. This assumption states that if two data points are coupled by a path of high density (e.g., it is more likely that both belong to same group or cluster), then their outputs are likely to be close, whereas if they are separated by a low-density region then their outputs need not be close [3]. This can be rephrased as f_i of a node should not be greatly different from f_j of its adjacent nodes (smoothness condition). The other assumption is about the loss or error, which states, in labeled nodes, the value of f_i should be similar to the value of the given label y_i (loss condition). Two conditions are included in the following quadratic objective function, and one can obtain the output vector f by minimizing

$$\min_f (f - y)^T (f - y) + \mu f^T L f \tag{2}$$

where $y = (y_1, \ldots, y_l, 0, \ldots, 0)^T$, and the matrix L, called the *graph Laplacian*, is defined as $L = D - W$ where $D = diag(d_i)$, $d_i = \sum_j w_{ij}$ [24,26]. The parameter μ trades off loss versus smoothness. The solution of this problem is obtained as

$$f = (I + \mu L)^{-1} y. \tag{3}$$

To obtain the output vector f, one must calculate the inverse of the matrix which may incur computational difficulties. Generally, the matrix representing the graph is sparse, therefore f is usually obtained by solving a large sparse linear system $y = (I + \mu L) f$. This numerical problem has been intensively studied, and there exist efficient algorithms, whose computational time is nearly linear in the number of non-zero entries in the weight matrix [27]. This implies that the computation gets faster as the graph gets sparser. Moreover, when the linear system solver is parallelized and distributed on a cluster system, the graph-based learning algorithm easily scales to much larger graphs.

2.2. Graph sharpening

The recently proposed graph-sharpening is an efficient method to improve the performance of graph-based SSL by removing noisy or undesirable relationships from the [7,22]. The algorithm operates on

AUTHOR COPY

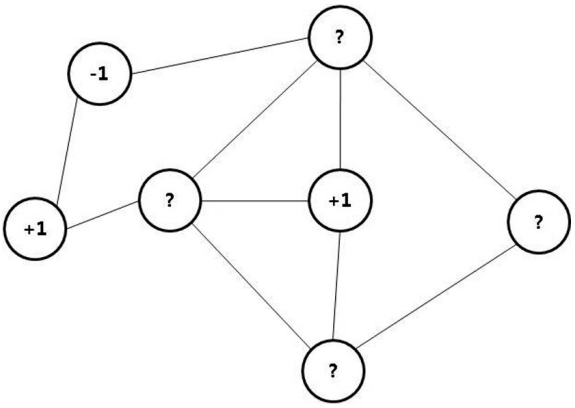

Fig. 1. An ordinary graph by W: The labeled node is denoted as "+1" or "−1", and the unlabeled node as "?". The edge has no directionality.

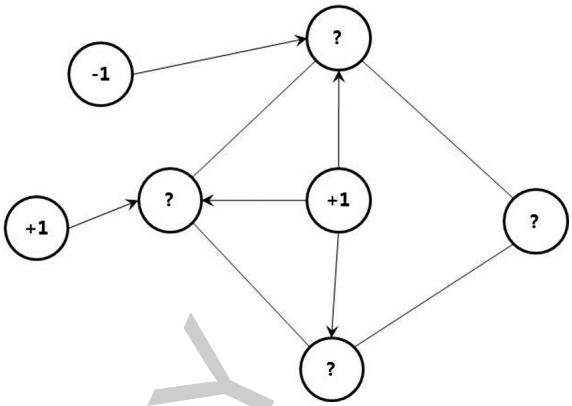

Fig. 2. A graph sharpened by W_s. By graph sharpening some edges have been removed or have assumed directionality according to the importance of the information flow.

the similarity (weight) matrix W, adjusting the edges (connections) between the nodes (data points). As described above, the relationship between the data points is represented by the similarity matrix W which plays a critical role in prediction as a form of graph-Laplacian L. Related to the matrix W, graph sharpening addresses two points. First, noisy data points form unnecessary edges and degrade the algorithm performance. Second, the matrix W is considered to be fixed and symmetric, the edge to be directionless, and the reflected similarity to be an undirected edge. As the weight matrix W, however, describes the relationships between the labeled and unlabeled points, it is not necessarily desirable to regard all such relationships as symmetric. That is, the contribution of all edges to the information flow may be varied by weighing them unequally. Graph sharpening improves the algorithm performance by changing the weight matrix to remove the edge caused by noise and to employ directionality between edges by asymmetrically weighing edge-weights. If an ordinary weight matrix is represented as a block matrix $W = [W_{ll}W_{lu}|W_{ul}W_{uu}]$, graph sharpening changes the matrix as $W_s = [\textit{diagonal}\ 0\,|W_{ul}W_{uu}]$ in the simplest case. W_{lu} should be read as the weight of an edge from an unlabeled node to a labeled one ($u \to l$). The output for unlabeled nodes can be predicted using the following equation:

$$f_u = \mu\left(I + \mu\left(D_{uu} - W_{uu}\right)\right)^{-1} W_{ul} y_l \tag{4}$$

Figure 2 shows the change in the graph after sharpening. Some of the edges have been removed and have assumed directionality. Graph sharpening yields a sparser graph that contains less noise and, in turn, reduces computational expense and improves prediction with no additional parameters. A detailed mathematical foundation and empirical validation of graph sharpening can be found in [21].

2.3. Ensemble learning

The key characteristic of ensemble learning is that it involves combining a set of classifiers each of which essentially accomplishes the same task [17,28,29]. The use of an ensemble can provide an effective alternative to the tradition of generating a population of classifiers, and then choosing the one with the best performance, while discarding the rest. The basic idea underlying the ensemble-based approach is to find ways of exploiting instead of ignoring the information contained in these redundant classifiers. In principle, a set of classifiers could vary in terms of their model parameters. There are a

AUTHOR COPY

number of parameters which can be manipulated in efforts to obtain a set of classifiers which generalize differently. These include the following: varying the model parameters and varying the data. Varying the model parameters: A set of classifiers can be created by varying the model parameters. Varying the data: The methods which seem to be most frequently used for the creation of ensemble are those which involve altering the training data. There are a number of different ways in which this can be done which include: bagging [17], boosting [30] and the Random Subspace Method (RSM) [31]. In bagging, one samples the training set, generating random independent bootstrap replicates [17], constructs the classifier on each of these, and aggregates them by a simple majority vote in the final decision rule. In boosting, classifiers are constructed on weighted versions of the training set, which are dependent on previous classification results. Initially, all data points have equal weights, and the first classifier is constructed on this data set. Then, weights are changed according to the performance of the classifier. Erroneously classified data points get larger weights, and the next classifier is boosted on the reweighted training set. In this way, a sequence of training sets and classifiers is obtained, which is then combined by simple majority voting or by weighted majority voting in the final decision. In the random subspace method, classifiers are constructed in random subspaces of the data feature space. These classifiers are usually combined by simple majority voting in the final decision rule.

In general, the ensemble learning is effective when the size of dataset used for model construction (training set or validation set) is small, it may be difficult to construct a good single classifier. Usually, a classifier constructed on small training sets is biased and has a large variance as the model parameters are poorly estimated. Consequently, such a classifier may be weak, having a poor performance. Moreover, often it will be unstable: small changes in the training set cause large changes in the classifier. The ensemble learning is known to reduce the bias or variance of weak classifiers which includes badly performing classifiers, unstable classifiers, classifiers of a low complexity, or classifiers depending upon certain assumed models that are not always true [17–20,32].

3. Proposed method

The procedure of the proposed method begins with generation of multiple graphs set to various values of model parameter. For each of the graphs, then, the noisy or redundant edges are removed by employing graph sharpening. These multiple "sharpened" graphs are combined into an ensemble network. And the final output is calculated by taking the simple mean of the output values of the member graphs of the network. The followings explains the aforementioned steps in due order.

The ensemble learning can be an alternative to the optimal parameter selection procedure. Instead of choosing the best single one from multiple classifiers each of which is trained with different parameter values, one can employ all of them without discarding any [17,28,29]. This is of great benefit to SSL, particularly concerning the difficulties in parameter selection. Using a validation set for parameter selection, which is the most representative approach and is originally designed for classifiers in supervised learning, does not fit well into SSL because the amount of labeled data is insufficient for learning, i.e., too deficient even for making a training dataset. Therefore, further splitting this dataset to make a validation dataset is undesirable. In the proposed method, multiple graphs are trained with various parameter values, without using a validation set, and then combined into an ensemble network. We construct a set of graphs each of which is trained on a pair of the two model parameters, the number of neighbors $\kappa\,(\kappa = 1, \ldots, K)$ in Eq. (1) and the loss-smoothness tradeoff $\mu\,(\mu = 1, \ldots, M)$ in Eq. (2). The size of all the possible combinations of two parameters becomes the number of the graphs of the ensemble network. In the next, each of the $|K| \times |M|$ graphs is refined using the graph sharpening. First, the graph

AUTHOR COPY

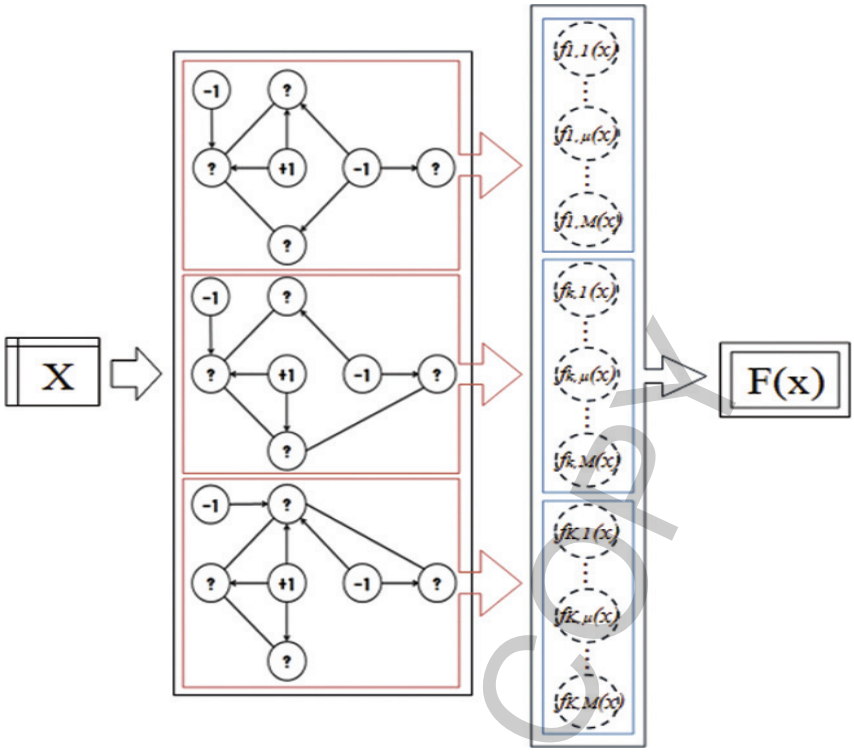

Fig. 3. Ensemble network of multiple graphs. (Colours are visible in the online version of the article; http://dx.doi.org/10.3233/IDA-130585)

sharpening penalizes the information flow from unlabeled to labeled nodes because this may deliver uncertain information to the graph. Second, the graph sharpening disconnects the edges directly linked to oppositely labeled nodes because they may introduce conflict of information to the graph. The sharpening procedure removes noisy or undesirable connections, and this leads to performance improvement. Then, the final output of the ensemble network, $F(x)$ in Eq. (5), is calculated by taking the simple mean of the individual outputs, $f(x)$ in Eqs (3) or (4), of the $|K| \times |M|$ graphs. This replaces the selection of a single best graph via a validation set, which is becomes a more practical approach for SSL. Figure 3 illustrates the procedure.

$$F(x) = \frac{\sum_{\kappa=1}^{K} \sum_{\mu=1}^{M} f_{\kappa,\mu}(x)}{|K| \times |M|} \quad \kappa\,(\kappa = 1, \ldots, K)\,, \quad \mu\,(\mu = 1, \ldots, M) \tag{5}$$

The proposed method has a complexity of $O(q \times |K| \times |M|)$. The q indicates the number of non-zero entries of the similarity matrix W which should be a sparse matrix when representing connections in a graph. As mentioned before, solving an individual graph takes nearly linear time proportional to q. And the graph sharpening procedure cuts many of edges, which will further lead to decrease in q. Additional time complexity by adopting ensemble approach in the proposed method is $|K| \times |M|$. However, it should be noted that conventional parameter searching to select the best single graph also requires the same number of repetitions. Consequently, there is no undesirable gain in terms of computational complexity.

AUTHOR COPY

Table 1
Summary of the five benchmark data sets

Data set	Classes	Dimension	Points	Comment
Digit1	2	241	1,500	Artificial
USPS	2	241	1,500	Imbalanced
BCI	2	117	400	Small, noisy
g241c	2	241	1,500	Artificial
g241n	2	241	1,500	–

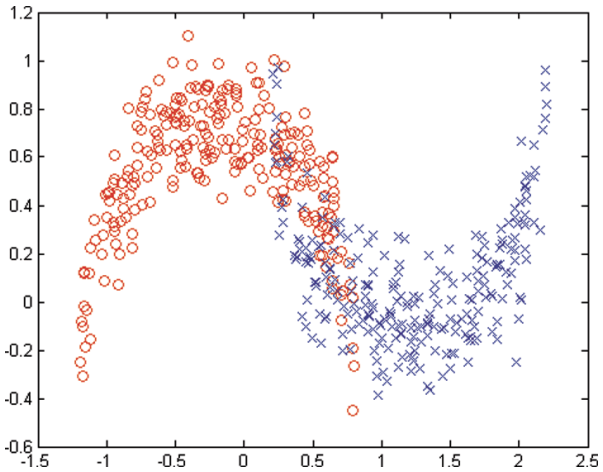

Fig. 4. Two-moon toy problem. (Colours are visible in the online version of the article; http://dx.doi.org/10.3233/IDA-130585)

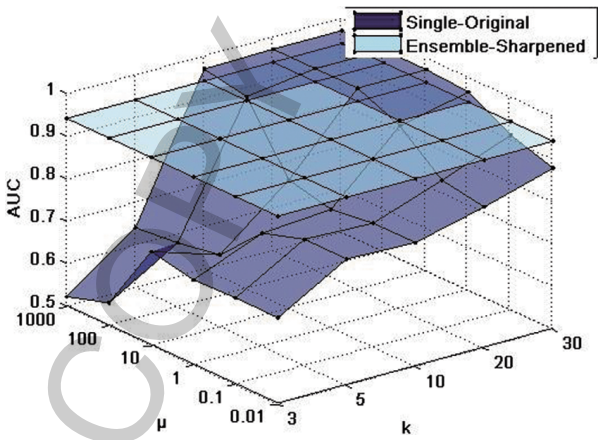

Fig. 5. AUC comparison over the parameter variation (k and μ). (Colours are visible in the online version of the article; http://dx.doi.org/10.3233/IDA-130585)

4. Experiments

The proposed method was evaluated on both artificial and benchmark data, and then applied to the real-world response modeling for insurance promotion provided by the CoIL Challenge 2000. We compared the proposed method, the ensemble of the sharpened graphs (*Ensemble-Sharpened*), with the original graph (*Single-Original*) in terms of the area under the ROC curve (AUC) [33–35].

4.1. Artificial data

The first dataset is artificial data sourced from the two-moon toy problem in Fig. 4. Five hundred input data were generated from two classes, each with 245 unlabeled and 5 labeled data. The AUC was measured under various combinations of parameters such as $(k, \mu) \in \{3, 5, 10, 20, 30\} \times \{0.01, 0.1, 1.0, 10, 100, 1000\}$, where k and μ indicate the number of *kNN* in Eq. (1) and the loss-smoothness tradeoff parameter in Eq. (2), respectively.

Figure 5 shows the change in the AUC over the parameter variation. Compared with the *Single-Original*, the *Ensemble-Sharpened* exhibits more stability and higher accuracy.

4.2. Benchmark data

Table 1 shows the AUC comparison results between *Single-Original* and *Ensemble-Sharpened*for five benchmarking datasets. Each dataset has two sets of 12 predetermined splits, one is for 10 labeled

AUTHOR COPY

Table 2
AUC comparison for the five benchmark data sets (mean ± std)

Dataset (dimension, number of points)		Single-Original	Ensemble-Sharpened	p-value
(1) Digit1(241,1500)	10 label	0.89 ± 0.04	0.93 ± 0.05	0.00
	100 label	0.97 ± 0.02	0.99 ± 0.01	0.00
(2) USPS(241,1500)	10 label	0.65 ± 0.09	0.68 ± 0.10	0.00
	100 label	0.87 ± 0.08	0.97 ± 0.01	0.00
(3) BCI(117,400)	10 label	0.50 ± 0.01	0.50 ± 0.03	0.93
	100 label	0.53 ± 0.03	0.56 ± 0.02	0.00
(4) g241c(241,1500)	10 label	0.55 ± 0.03	0.56 ± 0.05	0.00
	100 label	0.63 ± 0.05	0.65 ± 0.04	0.00
(5) g241n(241,1500)	10 label	0.55 ± 0.03	0.56 ± 0.04	0.00
	100 label	0.63 ± 0.04	0.65 ± 0.04	0.00

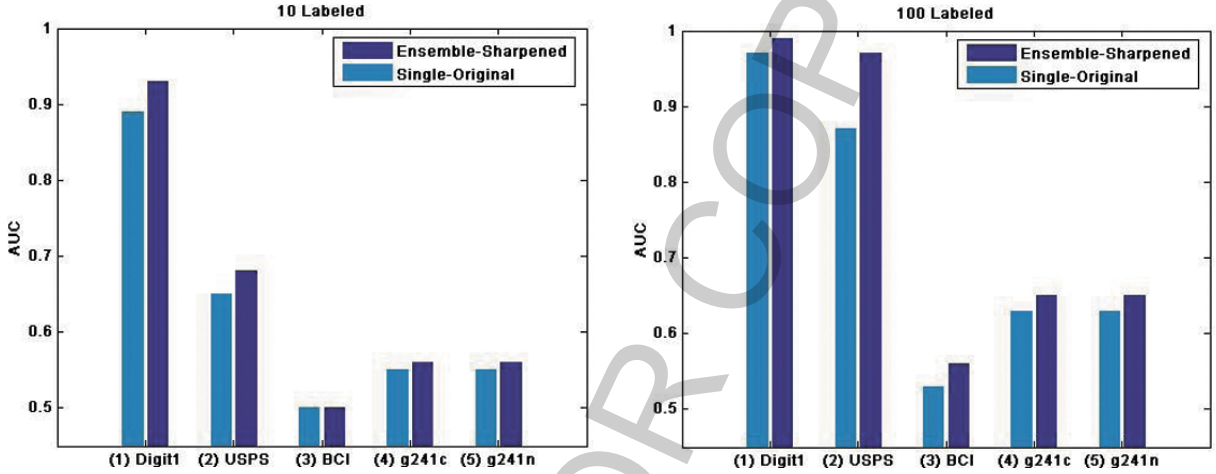

Fig. 6. AUC improvement by Ensemble-Sharpened. (Colours are visible in the online version of the article; http://dx.doi.org/10.3233/IDA-130585)

data and the other is for 100 labeled data. The AUC was measured at every combination of parameters $(k, \mu) \in \{3, 5, 10, 20, 30\} \times \{0.01, 0.1, 1.0, 10, 100, 1000\}$. For *Single-Original*, the values of model parameter (k, μ) were obtained at the best performance. The Wilcoxon signed-rank test was used to verify the performances of both methods, where a smaller value of p indicates a more significant difference between them [36]. The values listed in the table are the mean and standard deviation of AUC values across the 12 splits in the datasets. In most of cases, the proposed method increased the AUCs and the effect was statistically significant.

The five pairs of bar graphs in Fig. 6 depict the average AUCs of *Ensemble-Sharpened* and of *Single-Original*. The AUCs of *Ensemble-Sharpened* are consistently higher than those of *Single-Original* in both the 10 labeled and the 100 labeled data cases. The improvement is much greater in the latter cases, thereby confirming the greater effectiveness of the ensemble-sharpening method with increasing number of labeled data.

To summarize our experiment results, the maximum average increase in AUC of 0.10 was obtained from the USPS-100 labeled dataset and the minimum of 0 from the BCI-10 labeled dataset, thereby guaranteeing 'no loss' even in the worst case.

AUTHOR COPY

Table 3
The CoIL challenge 2000 data sets

Data set	Classes	Dimension	Points
All	2	134	4,000
Demographic	2	92	4,000
Monetary	2	21	4,000
Frequency	2	21	4,000

Table 4
AUC comparison of different classification models (*: best performance of the column)

Classification model	Size of training data			
	800	1200	1600	2000
Logistic regression	0.55	0.54	0.54	0.53
K-nearest neighbor	0.67	0.67	0.67	0.68
Support vector machine	0.65	0.65	0.65	0.66
Ensemble-Sharpened	*0.69**	*0.70**	*0.71**	*0.71**

4.3. Real world data

The third dataset is from the CoIL challenge 2000 dataset, which is a set of real world business problem called 'response modeling' in the marketing domain. The purpose of the problem is to predict which customers are potentially interested in a caravan insurance policy. The customers who have been targeted in a preliminary campaign are labeled either respondent ('+1') or non-respondent ('−1'), whereas the others, who have been excluded from the campaign and thus do not have labels, are called unlabeled ('?'). Because of the limited marketing cost for the campaign, only a small part of the customer data is labeled, with most of the customer data remaining unlabeled.

The CoIL challenge 2000 dataset consists of 134 descriptions of customers including demographic, monetary and frequency information. See Table 3. The original training set contains 5,822 customers including the label information on whether or not they have a caravan insurance policy, and the original test set contains 4,000 customers. In our experiment, 2,000 labeled customers in the training set and 2,000 unlabeled customers in the test set were randomly selected. The numbers of labeled data used were 800, 1,200, 1,600 and 2,000, and each dataset was evaluated by taking 30 iterations. We compared the proposed method, *Ensemble-Sharpened*, with the four individual graphs obtained from demographic, monetary, frequency, and a dataset merging the three data sources (denoted as 'All'), respectively. For the individual graphs, the values of model parameter (k, μ) were obtained at the best performance by five cross validation searching over $(k, \mu) \in \{5, 7, 10, 20, 30, 40, 50\} \times \{0.01, 0.05, 0.1, 1.0, 50, 100, 500\}$. For the proposed method, on the other hand, an ensemble network of 147 $(= 7\times7\times3)$ sharpened graphs-from every combination of parameters (k, μ) and the three data sources, was built without parameter selection procedure.

The box plot of Fig. 7 presents the distributions of the performance over 30 iterations. The AUC of *Ensemble-Sharpened* shows more robustness to experimental repetitions and higher accuracy regardless of different numbers of labeled data whereas the AUC of the *Single-Original* are less stable (individual graphs from demographic, monetary, frequency, all).

Table 4 compares the performance of *Ensemble-Sharpened* with those of different types of classification model: logistic regression, k-nearest neighbor, support vector machine. Logistic regression produced 0.53∼0.55 of the AUC, which is slightly better than random guessing. K-nearest neighbor and Support Vector Machine, on the other hand, produced 0.65∼0.68 of the AUC, which is regarded acceptable as an accuracy for real-world marketing data. However, it may be skeptical for domain experts when actually using those models in practical problems. *Ensemble-Sharpened* improves upon the supervised models for any amount of labeled data, from 800 to 4000, achieving the highest accuracies around 0.70 of the AUC. Moreover, *Ensemble-Sharpened* built with a small number of labeled data points produces results comparable to other models with a much larger number of labeled data: The AUC of *Ensemble-Sharpened* in the case of 800 data points already outperforms those of other three models trained with 2000 data points. This implies that *Ensemble-Sharpened* performs reasonably well especially when only

AUTHOR COPY

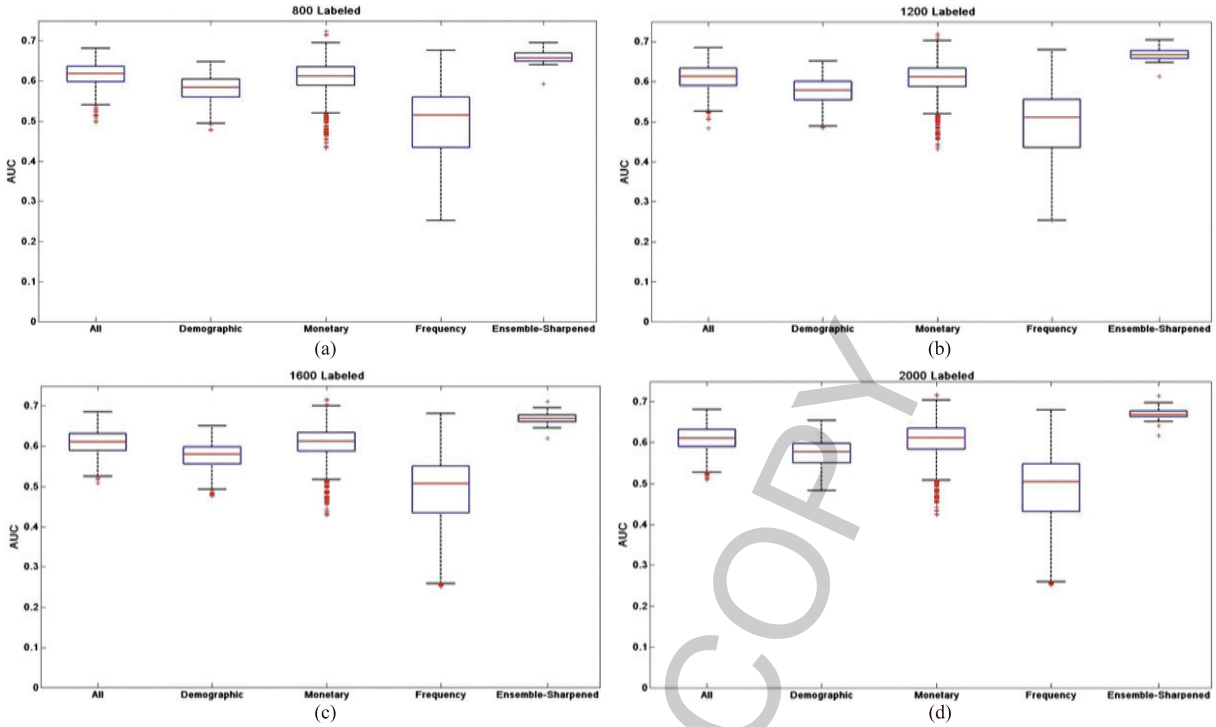

Fig. 7. AUC Comparison on the real world dataset: Single-Original vs. Ensemble-Sharpened. (Colours are visible in the online version of the article; http://dx.doi.org/10.3233/IDA-130585)

a small number of labeled data points are available. On the other hand, the AUC of *Ensemble-Sharpened* tends to increase with the number of labeled data points. This shows that even if the marketing cost for the campaign allows the most part of the customer data labeled by domain experts, *Ensemble-Sharpened* still will be a good choice as a prediction model over the supervised models because of its accuracy and no requirement for parameter tuning.

5. Conclusion

In this paper, we proposed to use ensemble learning and graph sharpening for graph-based semi-supervised learning. Instead of using a single graph specified to a certain value of the model parameter in a trial-and-error fashion, we employed a graph ensemble of which committee members trained with various values of parameter. Ensemble learning stabilizes the performance by reducing the error variance-and-bias of individual learners, and renders the parameter selection procedure less critical. In addition, the accuracy of an individual graph was improved by graph-sharpening due to its ability to remove noisy or unnecessary edges from the original graph, which enhances the robustness to noise in the dataset. When we applied both methods to an artificial problem and six real-world problems, their synergy significantly improved the performance.

Several future research directions need to be addressed. First, a comparative study involving various SSL algorithms can be considered [37]. One obstacle is that many of them are computationally inefficient compared with the graph-based SSL that we employed in this paper. Since the proposed method deals with multiple learners in parallel, computational efficiency should be taken into account first. Second, to

AUTHOR COPY

avoid the difficulty in parameter selection, we proposed to use an ensemble network of individual graphs set to various values of parameter. However, instead of simple ensemble integration of multiple graphs, a more sophisticated algorithm can be adopted as an alternative such as the convex optimization based integration as in [38]. But then, the parameter selection for integration will remain as another future work. Lastly, we found that the proposed method is well-suited for response modeling in the marketing domain. However, to be a proper marketing tool, the domain specific analyses using the lift or response chart and profitability, should be further exploited.

Acknowledgements

The authors would like to gratefully acknowledge support from Post Brain Korea 21 and the research grant from National Research Foundation of the Korean Government (2009-0065043/2012-0000994).

AUTHOR COPY

References

[1] O. Chapelle et al., Cluster kernels for semi-supervised learning, *Advances in Neural Information Processing Systems* **15** (2003), 585–592.
[2] X. Zhu, Semi-supervised learning with graphs, PA 15213, Carnegie Mellon, Pittsburgh, 2005.
[3] O. Chapelle et al., *Semi-Supervised Learning* , Cambridge, England: MIT Press, 2006.
[4] J. Wang et al., On efficient large margin semisupervised learning: Method and theory, *Journal of Machine Learning Research* **10** (2009), 719–742.
[5] X. Zhu, Semi-supervised learning literature survey, Technical Report TR-1530, Wisconsin-Madson, 2008.
[6] H. Shin and K. Tsuda, Prediction of protein function from networks, in: *Semi-Supervised Learning*, O. Chapelle et al., eds, ed: MIT press, 2006, pp. 339–352.
[7] H. Shin et al., Graph sharpening plus graph integration: A synergy that improves protein functional classification, *Bioinformatics* **23** (2007), 3217–3224.
[8] A. Subramanya and J. Bilmes, Soft-supervised learning for text classification, in: *Conference on Empirical Methods in Natural Language Processing*, Honolulu, Hawaii, (2008), 1090–1099.
[9] R.K. Ando and T. Zhang, A high-performance semi-supervised learning method for text chunking, in: *Annual Meeting on Association for Computational Linguistics*, Ann Arbor, Michigan, (2005), 1–9.
[10] L. Wei and E. Keogh, Semi-supervised time series classification, in: *International Conference on Knowledge Discovery and Data Mining*, Philadelphia, USA, (2006), 748–753.
[11] E. Bair and R. Tibshirani, Semi-supervised methods to predict patient survival from gene expression data, *PLoS Biology* **2** (2004), 511–522.
[12] Y.-C. Gong and C.-L. Chen, Semi-supervised method for gene expression data Classification cation with gaussian fields and harmonic functions, in: *International Conference on Pattern Recognition*, Tampa, FL, (2008), 1–4.
[13] N. Morsillo et al., Semi-supervised learning of visual classifiers from web images and text, in: *International Joint Conference on Artificial Intelligence*, California, USA, (2009), 1169–1174.
[14] A. Celikyilmaz et al., A graph-based semi-supervised learning for question-answering, in: *Annual Meeting of the Association for Computational Linguistics*, Singapore, (2009), 719–727.
[15] R. Liu et al., A graph-based semi-supervised learning algorithm for web page classification, in: *International Conference on Intelligent Systems Design and Applications*, China, (2006), 856–860.
[16] Y. Bengio, Gradient-based optimization of hyperparameters, *Neural Computation* **12** (2000), 1889–1900.
[17] L. Breiman, Bagging predictors, *Machine Learning* **24** (1996) 123–140.
[18] M.P. Perrone, Improving regression estimation: Averaging methods for variance reduction with extensions to general convex measure optimization, *PhD Thesis, Brown University, Providence, RI* 1993.
[19] A.J.C. Sharkey and N.E. Sharkey, Combining diverse neural nets, *The Knowledge Engineering Review* **12** (1997), 231–247.
[20] K. Tumer and J. Ghosh, Error correlation and error reduction in ensemble classifiers, *Connection Science* **8** (1996), 385–404.
[21] H. Shin et al., Graph sharpening, *Expert Systems with Applications*, 2010.
[22] H. Shin et al., Graph sharpening, *Expert Systems with Applications* **37** (2010), 7870–7879.

[23] J. He et al., Graph-based semi-supervised learning as a generative model, in: *International Joint Conference on Artificial Intelligence*, Hyderabad, India, (2007), 2492–2497.
[24] D. Zhou et al., Learning with local and global consistency, *Advances in Neural Information Processing Systems* **16** (2004), 321–328.
[25] Y. Song et al., Semi-supervised discriminative classification with application to tumorous tissues segmentation of MR brain images, *Pattern Analysis and Applications* **12** (2009), 99–115.
[26] M. Belkin et al., Regression and regularization on large, in: *COLT 2004 LNCS (LNAI)*, J. Shawe-Taylor and Y. Singer, eds, **3120** (2003), 624–638.
[27] D.A. Spielman and S.-H. Teng, Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time, *Journal of the ACM* **51** (2004), 385–463.
[28] Y. Freund and R.E. Schapire, Experiments with a new boosting algorithm, *Machine Learning: Proceedings of the Thirteenth International Conference* (1996), 148–156.
[29] H. Shin and S. Cho, Pattern selection using the bias and variance of ensemble, *Journal of the Korean Institute of Industrial Engineers* **28** (2001), 112–127.
[30] Y. Freund and R.E. Schapire, Experiments with a new boosting algorithm, in: *Proceedings 13th International Conference on Machine Learning* (1996), 148–156.
[31] T.K. Ho, The random subspace method for constructing decision forests, *IEEE Trans Pattern Analysis and Machine Intelligence* **20** (1998), 832–844.
[32] M. Skurichina and R.P.W. Duin, Bagging, boosting and the random subspace method for linear classifiers, *Pattern Analysis and Applications* **5** (2002), 121–135.
[33] M. Vuk and T. Curk, ROC curve, lift chart and calibration plot, *Metodoloski Zvezki* **3** (2006), 89–108.
[34] J. Huang and C.X. Ling, Using AUC and accuracy in evaluating learning algorithms, *IEEE Educational Activities Department* **17** (2005), 299–310.
[35] T. Fawcett, An introduction to ROC analysis, *Pattern Recognition Letters* **27** (2006), 861–874.
[36] J. Demsar, Statistical comparisons of classifiers over multiple data sets, *Journal of Machine Learning Research* **7** (2006), 1–30.
[37] H. Shin and K. Tsuda, Prediction of protein function from networks, in: *Semi-Supervised Learning*, O. Chapelle et al., eds, ed Cambridge, Massachusetts, London, England: The MIT Press, 2006, pp. 361–376.
[38] K. Tsuda et al., Fast protein classification with multiple networks, *Bioinformatics* **21** (2005), 59–65.

AUTHOR COPY