Kybernetes

Take full advantage of unlabeled data for sentiment classification

Lei La, Shuyan Cao, Liangjuan Qin,

Take full advantage of unlabeled data for sentiment classification

Lei La
School of Information Technology and Management, University of International Business and Economics, Beijing, China

Shuyan Cao
Department of Information Management, University of International Business and Economics, Beijing, China, and

Liangjuan Qin
School of Information Technology and Management, University of International Business and Economics, Beijing, China

Abstract

Purpose – As a foundational issue of social mining, sentiment classification suffered from a lack of unlabeled data. To enhance accuracy of classification with few labeled data, many semi-supervised algorithms had been proposed. These algorithms improved the classification performance when the labeled data are insufficient. However, precision and efficiency are difficult to be ensured at the same time in many semi-supervised methods. This paper aims to present a novel method for using unlabeled data in a more accurate and more efficient way.

Design/methodology/approach – First, the authors designed a boosting-based method for unlabeled data selection. The improved boosting-based method can choose unlabeled data which have the same distribution with the labeled data. The authors then proposed a novel strategy which can combine weak classifiers into strong classifiers that are more rational. Finally, a semi-supervised sentiment classification algorithm is given.

Findings – Experimental results demonstrate that the novel algorithm can achieve really high accuracy with low time consumption. It is helpful for achieving high-performance social network-related applications.

Research limitations/implications – The novel method needs a small labeled data set for semi-supervised learning. Maybe someday the authors can improve it to an unsupervised method.

Practical implications – The mentioned method can be used in text mining, image classification, audio processing and so on, and also in an unstructured data mining-related field. Overcome the problem of insufficient labeled data and achieve high precision using fewer computational time.

Social implications – Sentiment mining has wide applications in public opinion management, public security, market analysis, social network and related fields. Sentiment classification is the basis of sentiment mining.

Originality/value – According to what the authors have been informed, it is the first time transfer learning be introduced to AdaBoost for semi-supervised learning. Moreover, the improved AdaBoost uses a totally new mechanism for weighting.

Keywords Social networks, Boosting, Semi-supervised learning, Sentiment mining, Unlabeled data

Paper type Research paper

Kybernetes
Vol. 47 No. 3, 2018
pp. 474-486

0368-492X
DOI 10.1108/K-08-2016-0196

This work was supported in part by "Beijing Municipal Social Science Foundation" No. 16GLC067 and "Beijing Municipal Science and Technology Project", no. D171100003317001 (Key technology research and development of Beijing animation derivatives personalized consumer service).

1. Introduction and related works

Sentiment mining is attracting increasing attention from researchers with the rapid development of social networking services (SNSs). Sentiment mining techniques can provide many benefits to the society as a whole. Governments can amend or repeal policies according to online complaints, enterprises can analyze the popularity of products and determine future production according to customers' comments, sociologists can determine the well-being index of different communities using sentiment mining and so on.

Sentiment classification is the foundation of many sentiment-mining tasks (Wang, 2011). Before the further processing of data, sentiments should be classified as cheerful and despondent, supporting and opposing, positive and negative, etc. The shortage of labeled data is a considerable challenge in sentiment classification. Traditionally, a high-precision supervised classification requires a large amount of labeled data for model training. However, due to the explosive growth of data in internet, it is almost impossible to obtain sufficient labeled training data for each domain, especially in the scope of sentiment classification. Moreover, the high labor cost makes manual annotation unacceptable.

To achieve higher performance when labeled data are scarce, scholars have proposed a number of semi-supervised algorithms that use unlabeled data as auxiliary training data (Nikitidis *et al.*, 2012) to improve classification accuracy (Kingma *et al.*, 2014).

COREG (Isaac *et al.*, 2015) is a co-training-based (Zhang *et al.*, 2014) semi-supervised algorithm that can achieve high classification accuracy by proposing a taxonomy based on the main characteristics of the data. The performance of the algorithm in terms of its transductive and inductive classification capabilities was measured empirically in an exhaustive study involving a large number of data sets with different ratios of labeled data. However, the complex mechanism designed for regression enhanced the computational consumption.

Astorino and Fuduli (2015) proposed a three-step semi-supervised algorithm for pattern classification when labeled data are lacking. Support vector machine-based learning (SVM – support vector machine) is integrated with an evolutionary stochastic algorithm called the firefly algorithm as a relevance feedback approach in a region-based image retrieval system. The experimental results proved the effectiveness of the algorithm. Because its core classifier, SVM, is a high time-overhead algorithm (Zhang *et al.*, 2015) and three steps are required in the classification, the algorithm has a high runtime cost.

Kullback-Leibler divergence (KLD)-based classification (Subramanya and Bilmes, 2011) is a general algorithm that can be used in many fields such as text classification, phone recognition and image processing. This method is a graph-based semi-supervised learning method based on minimizing the KLD between discrete probability measures that encode class membership probabilities. Experiments revealed low time consumption, but unfortunately, simulations revealed that the precision for text classification is not ideal.

Cecotti (2016) presented a graph-based algorithm for semi-supervised learning that uses adaptive k-nearest neighbors (kNN) (Patel and Thakur, 2016) as its base classifier. A kNN graph is obtained with a sample deformation model that considers local deformations. During the active learning procedure, the user first labels the vertices with the highest number of neighbors. The expert labels the examples that are more likely to propagate their labels to a high number of close neighbors, and the examples are automatically labeled by a label propagation function. The procedure is repeated until all samples are labeled. The experimental results revealed simultaneously high accuracy and efficiency. However, this algorithm may not be suitable for sentiment classification because it is overly sensitive to the proportion of labeled data in each category.

Transfer learning (Lu *et al.*, 2015) is an ingenious methodology (Perlich *et al.*, 2015) that uses labeled data in the old domain as additional training data for classification in new

domains with insufficient labeled data (Long *et al.*, 2014). Online-multitask-boosting (OMB) (Wang and Pineau, 2016) is a powerful algorithm in this field. OMB extends existing transfer and multitask learning algorithms to allow their use in any time setting. Two novel online boosting algorithms for transfer learning and multitask learning, respectively, were designed to take advantage of knowledge of instances in other domains. Although the convergence rate of OMB is not very high, it supports the potential utility of boosting methodology in semi-supervised classification.

To solve the above problems, this paper proposes a novel weighting mechanism for transfer learning inspired by OMB but using completely different classification strategies. The aim is to achieve higher accuracy and convergence speed with a shorter running time and reduced labor cost for sentiment mining.

The rest of this paper is organized as follows. Section 2 reviews the training data weighting method and proposes a novel method based on boosting to use unlabeled data efficiently. In Section 3, the weighting method for base classifiers is discussed, and a more rational method of combining classifiers to generate a strong classifier is designed. The complete form of the novel sentiment classification algorithm is presented and analyzed in Section 4. Section 5 describes the application of the novel method and evaluates the experimental results. Finally, Section 6 summarizes the paper.

2. Methodology: using unlabeled data efficiently

Traditional classification algorithms attempt to construct a very strong classifier for categorization. Boosting methodology abandons this strategy and instead attempts to integrate a group of weak classifiers together using a weighting mechanism. AdaBoost is among the most classic boosting algorithms.

2.1 Review of AdaBoost

Because by Schapire and Singer (Schapire and Singer, 2000), AdaBoost has attracted extensive attention from scholars because of its high performance. It is defined as follows (Lua *et al.*, 2015).

Let X be the feature space and $D = \{(d_1, c_1), (d_2, c_2), \ldots\}$ be the training data, where $d_i \in X$ are the training document representations and $c_i \in \{1, -1\}$ are the category assignments. A base classifier is an algorithm that produces a weak hypothesis h: $X \rightarrow \{\pm 1\}$ given the training data D together with a weight distribution W upon it. The capability of a hypothesis is measured by its error:

$$\varepsilon(h, W) = \sum\nolimits_{i:h(d_i)\neq c_i} W(i) \tag{1}$$

where ε presents the sum of the weights of misclassified documents. The work steps of AdaBoost are as follows:

- Initialize the weight distribution $W_1(i) = 1/|\mathrm{D}|$for all i.
- Perform iterative loops for $t <$ T and train a weak classifier ht using the current weight W_t.
- Assume a parameter α_t as:

$$\alpha_t = 1/2 \ln \frac{1 - \varepsilon(h_t, W_t)}{\varepsilon(h_t, W_t)} \tag{2}$$

- Update the weights as:

$$W_{t+1}(i) = Z_t \cdot W_t(i) \tag{3}$$

Where Z is a normalization factor.

- Output the final strong classifier as:

$$H(d) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(d)\right) \tag{4}$$

This weighting mechanism of AdaBoost gives greater weight to difficult documents to reduce the probability of their misclassification in the next iteration. Although AdaBoost consequently achieves high precision in traditional supervised classification tasks, it is not suitable for semi-supervised learning because its accuracy is highly dependent on the amount of labeled training data.

2.2 Unlabeled data selection

Similar to many semi-supervised algorithms, we attempt to generate a coarsely labeled data set using a small number of accurately labeled training samples. Therefore, a two-stage boosting process is required for classification. The first stage is to select suitable unlabeled data as supplementary training data, and the second stage is data classification using training sets containing a mix of accurately labeled data and coarsely labeled data.

The small accurately labeled set is used as training data, and the large number of unlabeled data are used as testing data, similar to the process for the original AdaBoost. To perform auxiliary training data selection and weighting, the category of the unlabeled data must first be defined. In this paper, we use the plural category of a document in T iterations as follows:

$$C'(i) = \underset{1 \le t \le T}{Mode}\left(C_t(i)\right) \tag{5}$$

where $C'(i)$ is the true category of testing data i used in the system, and C_t is the category determined by the system in tth loop. The plural category is used because the plural is insensitive to noise compared with the mean and median. Moreover, in two-class problems, the median does not exist, and rounding of the mean will seriously reduce the precision.

In contrast to traditional AdaBoost, in semi-supervised categorization, when an unlabeled sample is misclassified, its weight is not increased arbitrarily because the classification error may reveal a different distribution. In unlabeled data selection, the purpose is to select the data with a distribution most similar to the accurately labeled data as supplementary training data. Because distribution similarity is related to the correctly classified rate [11], the category variance of an unlabeled sample caused by base classifiers can be used as the metric of similarity. Let σ be the distribution similarity of unlabeled data in the objective domain, which can be calculated as:

$$\sigma(i) = \frac{\left|\sum_{j=1}^{M} C_j(i) - C'(i)\right|}{M} \tag{6}$$

where $C_j(i)$ is the classification result of the ith document given by the jth weak learner, and M is the number of base classifiers. When using the definition of initial weights of documents, the weights can be updated according to $\sigma(i)$ as:

$$W_{t+1}(i) = \frac{\sum_{i=1}^{N} \sigma}{N\sigma_t(i)} W_t(i) \cdot Z_t \tag{7}$$

where N is the number of unlabeled data. Actually, the plural category $C'(i)$ is the coarse label of candidate data i of the unlabeled set, and the weights of the unlabeled samples can be regarded as the credibility of their categories assigned automatically by the machine.

The above steps will result in automatic labeling by the machine. Furthermore, we can obtain a credibility sort table of the machine-made labels. Several different strategies can be used to use these coarse-labeled data. The simplest strategy is to use all coarse-labeled data as additional training samples; this strategy will have the lowest runtime complexity. However, some candidate data should usually be excluded as noise because their labels are so suspicious.

Two methods can be used for noise filtering. It is easy to imagine that in many cases, a percentage of noise data exists in the unlabeled set. Therefore, we can set a proportion k and decide to save or discard the data according to it. In other words, only the top k percentage of the coarse-labeled data will be used as the additional training sample. This first method is that described above.

However, in some situations, large gaps exist for a proportion of the noise data. To deal with this problem, a lower bound β could be introduced as:

$$\beta = \lambda \cdot \frac{N \cdot \min\limits_{1 \le i \le N} \left(W_T(i)\right) + \sum_{i=1}^{N} \left(W_T(i)\right)}{2N} \tag{8}$$

where $W_T(i)$ is the final weight of sample i after T iterations, and λ is an empirical parameter that can be adjusted adaptively according to the error rate. When the weight of a coarse labeled sample is lower than β, it will be excluded from the training set. This strategy can achieve the highest precision in theory, but its computational consumption is also higher than those of the other two methods.

The coarse labeled samples are then used together with the accurately labeled samples as the training data. A training course similar to that in AdaBoost can subsequently be used for the base learner's training process.

The above method labels the candidate data automatically. Furthermore, the system selects machine-labeled data according to the credibility of their labels. The selected data will increase the accuracy of semi-supervised classification, which is quite valuable for sentiment classification because labeled data in this field are insufficient and costly.

3. Methodology: weighting mechanism for weak classifiers

Although novel strategies are proposed for training data, work remains to achieve boosting in a semi-supervised sentiment classification. Weight-allocation methods for weak classifiers of previous boosting-based algorithms cannot be used directly when the training sets contain coarse-labeled data.

3.1 Evaluating the capability of base classifiers

The base learner weighting mechanisms of AdaBoost algorithm family members are more or less similar to each other. The basic steps of weak classifier weighting are as shown in Algorithm 1:

Algorithm 1:

Input: sum of iterations T, training data set $S = \{S_1, S_2, \ldots, S_N\}$, the sum of weak classifiers M

1. begin
2. sample learning
3. for ($t = 1$, $t <= T$, $t++$)
4. for($i = 1$, $i <= N$, $i++$)

```
5.        for(j = 1, j<= M, j++)
6.           if(C_t^j(i) == C(i))
7.              W_{t+1}(j) = W_t(j) + Δ_1
8.           else
9.              W_{t+1}(j) = W_t(j) − Δ_2
10.   Output W(j)
11. end
```

As shown in Algorithm 1, in traditional AdaBoost, the judgment of whether a weak classifier is powerful is the number of training samples classified correctly by it, which is obviously not a comprehensive weighting strategy. If a question in answered incorrectly by a student, two possibilities exist: the question is too difficult or the student is too ignorant. However, only one possibility is considered in traditional AdaBoost: the document is too difficult.

To comprehensively consider weak learner weighting, the difficulties of training data should be taken into account. Because the weights of training data represent their difficulty, the weight increment of a base classifier should grow with the increasing weight of the training sample that can be classified correctly by it and vice versa. The initial weights of weak classifiers can be defined as:

$$\omega_1(j) = 1/M \tag{9}$$

where M is the number of weak classifiers. Therefore, the weights of base learners can be updated as:

$$\omega_{t+1}(j) = \frac{\sum_{i=1}^{n} W_t(i)}{\sum_{j=1}^{N-n} W_t'(i)} \omega_t(j) \cdot Z_t \tag{10}$$

where n is the number of training samples classified correctly by the jth weak classifier in iteration $t + 1$ and $W_t'(i)$ is the weight of ith training data misclassified by classifier j in the tth iterative loop. In this way, the difficulties of training samples are taken into consideration to enhance the performance of the combined strong classifier. In addition, this strategy causes no additional runtime complexity compared with original AdaBoost.

3.2 Addressing noise-sensitive classifiers

The performance of base classifiers can be roughly divided into four levels as shown in Table I.

The last level, which is a weak classifier, can classify a difficult sample correctly, but the misclassification of easy training data reveals that this base learner is probably noise-sensitive. The first three conditions can be addressed by equation (10), but the last case is beyond the scope of its consideration. However, the role of noise-sensitive classifiers must be limited because of the presence of excessive noise in unlabeled data.

We define D_a as the additional training set and D_b as the base training set. When a training document $D_i \in D_a$ is considered an easy or difficult sample, the system will not change its

Difficulty of training data		Classification results		Conclusion
Difficult	Easy	Right	Right	Strong
Difficult	Easy	Wrong	Right	Common
Difficult	Easy	Wrong	Wrong	Poor
Difficult	Easy	Right	Wrong	Noise-sensitive

Table I. Performance levels of weak classifiers

weight to prevent wrong decisions. When a training document $D_i \in D_b$ is considered an easy or difficult sample, a weight-adjustment strategy similar to that of traditional AdaBoost is used to improve accuracy. When $D_i \in D_a$ is considered to have a significantly different distribution than the test set, its weight will be reduced. Distractors will be ignored as noise.

The above steps will be operated T times until the algorithm converges. In this way, the system can distinguish between data with different distributions that include difficult data and give training data weights in a more reasonable way. In the case of differences in distribution, some noisy data should be excluded. To achieve this purpose, a threshold σ of weight is introduced into the system as:

$$\sigma = \frac{\sum_{i=1}^{m} \omega_i + m \cdot \min\limits_{1 \le i \le m} \omega_i}{2m} \tag{11}$$

When the weight of additional training data is lower than σ, the data are moved out of the training set. Moreover, the iterative weighting function for additional training data is modified to reduce the system's time consumption. The training data's initial weight is defined in function (12).

$$\omega_i = \begin{cases} 1/n, \ 1 \le i \le n \\ 1/m, \ n+1 \le i \le n+m \end{cases} \tag{12}$$

The weight is updated following the below function:

$$\omega_i^{t+1} = \begin{cases} \omega_i^t - (-1)^{|C'(Di)-C(Di)|} \cdot \lambda_1, \ 1 \le i \le n \\ \omega_i^t + (-1)^{|C'(Di)-C(Di)|} \cdot \lambda_2, n+1 \le i \le n+m \end{cases} \tag{13}$$

where $C'(D_i)$ is the category of D_i given by the classifier, $C(D_i)$ is the real category of D_i, and t is the current round of iteration. The weight changes λ_1 and λ_2 are defined as:

$$\lambda_1 = 1 \Big/ \left(1 + \sqrt{2 \ln n / T}\right) \tag{14}$$

$$\lambda_1 = 1 \Big/ \left(1 + \sqrt{2 \ln n / T}\right) \tag{15}$$

where T is the number of iteration rounds. Obviously, the classification errors are bounded. Thus, the base classifier can output the final classification results of test documents in a reasonable weighted way.

To minimize the influence of noise-sensitive classifiers, a validator can be used. The training data are divided into two parts: if the error rate of a weak learn in the easier part is much higher than the error rate in the harder part, the weak classifier will be regarded as noise-sensitive. In this case, the weights should be reduced as:

$$\omega'(j) = (1/1 - \delta) \cdot \omega(j) \tag{16}$$

where δ is the difference in the error rates of the two parts of the training data. When the weak classifier belongs to Level 4, using function (16) can limit its influence on the classification results and make the system more reliable.

3.3 Reducing the dependence on accurately labeled data

Because the training set is a combination of accurately labeled data and supplementary data, dependence on accurately labeled data should be avoided. Dependence on accurately labeled data means that a weak learner has very good performance when the training samples are accurately labeled but very low precision when the training samples are coarse-labeled. The classifiers will cause the classification to have a high requirement for the proportion of accurately labeled data and violate the idea of semi-supervised learning.

Fortunately, in the proposed method and most boosting-based methods, training samples are input sequentially. Therefore, the error rates of a weak learner when dealing with accurately labeled training data and auxiliary training data can be distinguished similar to noise-sensitive classifier discovery. The work in this section can be integrated to obtain the final method for the weight allocation of weak classifiers. The flowchart of the weak classifier weighting mechanism in this article is shown in Figure 1.

4. Complete form of FullBoost

Having designed the strategies for additional training selection and base learner weighting, a novel semi-supervised algorithm for sentiment classification can be proposed. This algorithm is called FullBoost in this article. The alternative strategies for coarse-labeled data selection are called FullBoost.A, FullBoost.B and FullBoost.C, respectively.

As in the analysis in the last section, the weights of the training data must be considered in updating the base classifiers' weights to improve performance. Similarly, misclassification of a training sample should not indicate an immediate need to increase its weight. Instead, the reason for misclassification should be analyzed: either the training data are confusing or the weak learner has low capability. Therefore, the weights of the weak classifiers should also be taken into account during updating of the training samples' weights.

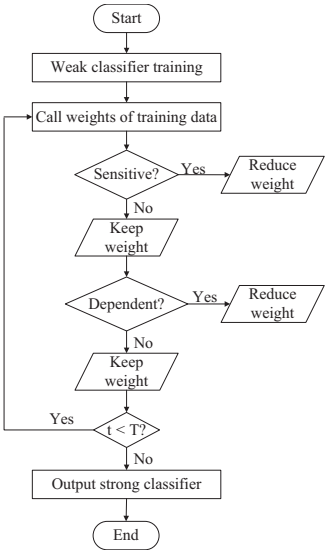

Figure 1.
Flowchart of weak classifier weighting

To avoid falling into endless loops, the order of weight updating must be set for the system, and cross-iterative steps should be used. The detailed work flow of Pseudo code of FullBoost are as follows (Algorithm 2):

```
Algorithm 2: FullBoost for semi-supervised sentiment classification
Input: base training set S1 = {S1(1), S1(2),..., S1(N1)} Additional
    training set S2 = {S2(1), S2(2),..., S2(N2)} iteration times T
Output: Final strong classifier H
begin
  for(t = 1, t <= T, t++)
    for (i = 1, j <= N1, i++)
        unlabeled data classification
    C'(i) calculation
  calculating the credibity WT(i) of C'(i)
  supplementary data selection according to WT(i)
  for(t = 1, t < = T, t++)
      for(i = 1, j <= N1 + N2, i++)
          for(j = 1, j <= M, j++)
              training C(j) with S(i)
              call weights of S(i) for computation
              if(Ct^j(i) = C(i)
                  ω(t+1)(j) = ωt(j) + Δ1
                  W(t+1)(i) = Wt(i)
              else
                  ω(t+1)(j) = ωt(j) − Δ2
                  W(t+1)(i) = Wt(i) + Δ'
      if (C(j) is noise-sensitive \?\ is ALD depentent)
         ω(t+1)(j) = (1/1 − δ)ωt(j)
      else
          keep the weight of classifier j
   combine weak learners according to their weights
   output strong classifier H
end
```

Thus, a novel semi-supervised algorithm is fully proposed. The strategies designed in this paper ensured its high precision in theory. The computational complexity of the original AdaBoost is $O(T \times M \times N)$ (Landesa-Vazquez and Alba-Castro, 2012). Because FullBoost has an additional unlabeled data selection procedure, its runtime complexity R_F is:

$$R_F = O(2T \cdot (N_1 + N_2) \cdot M) \tag{17}$$

The computational complexity of the novel algorithm presented in this article has the same order of magnitude as the original AdaBoost and the ability of machine-made annotation. Theoretically, the time efficiency of FullBoost is ideal.

All formulas and parameters are bounded in FullBoost. Furthermore, no missing condition exists in the novel algorithm. Therefore, once the error rates of weak classifiers are less than $1/n$, where n is the number of categories of the classification task, the algorithm converges (Gordon *et al.*, 2015). Note that the high convergence rate may result in even lower time consumption by the novel algorithm than the original AdaBoost.

5. Experiment, illustrative example and analysis

To evaluate the performance of the novel algorithm, simulations and experiments are performed to test its accuracy, time consumption and convergence rate. Some classic classification algorithms are used for comparison.

5.1 Simulation for evaluation of time consumption

Time consumption is an important judgment for performance evaluation. Many theoretically ideal classification algorithms do not become popular due to huge time overhead. Therefore, a simulation is performed to test the time consumptions of FullBoost.A, FullBoost.B and FullBoost.C and compare them with other classic algorithms. The comparison results are shown in Figure 2.

As shown in Figure 2, the three versions of FullBoost have significantly lower time overhead than CORGE and Semi-SVM (Leng *et al.*, 2013). Moreover, FullBoost.A and FullBoost.B are more efficient than TrAdaBoost, and FullBoost.C is nearly equivalent to TrAdaBoost. Although the time consumption of FullBoost is slightly higher than that of AdaBoost, FullBoost can perform classification when the labeled data are insufficient. Logarithmic coordinates are used on the *X*-axis to facilitate display.

5.2 Experiment for precision evaluation

The convergence rate is an important indicator for assessing performance. The high precision of a classification algorithm must not depend highly on a large number of iterations. The results of experiments to test the convergence rate of FullBoost are shown in Figure 3.

In Figure 3, the *X*-axis is the number of loops, and the *Y*-axis is the percentage of error. The results reveal that all three versions of FullBoost have a fast convergence rate.

We use a small set of labeled SNS data and a large unlabeled set of documents downloaded from Reuters-21578, the Sougo Chinese corpus and Web crawling of the internet to test the accuracy of FullBoost. The sentiments of the training and testing documents are divided into eight categories. The detailed results are shown in Table II.

As shown in Table II, FullBoost has higher precision than many classic semi-supervised algorithms because FullBoost uses a comprehensive weighting strategy for both data selection and the classifier ensemble. The novel weighting method enhances accuracy, and the threshold introduced to limit the influence of noisy sensitive learners further improves the algorithm's performance. The experimental results are consistent with the theoretical

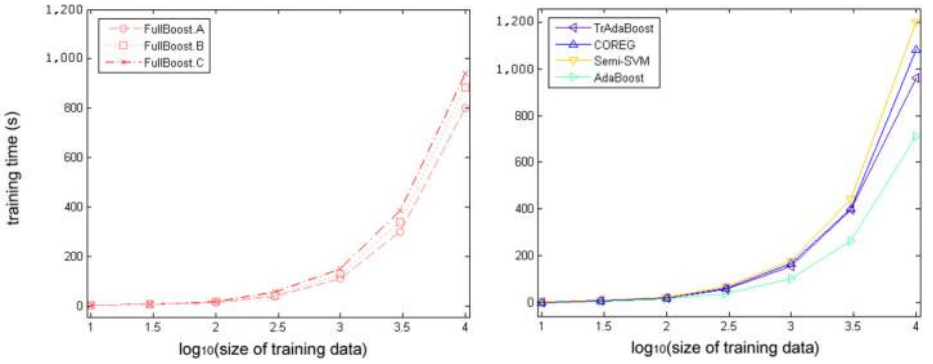

Figure 2.
Time consumption of the compared algorithms

analysis indicating that FullBoost.C should have the highest accuracy among the three versions of FullBoost.

6. Conclusions and future work

A novel semi-supervised classification algorithm based on boosting methodology, FullBoost, is proposed in this paper. A self-training strategy is designed to achieve machine labeling. The machine-labeled data are used as auxiliary training data to train the classifier with accurately labeled base training data. In addition, we propose a novel, more reasonable method for combining the weak classifiers into a strong classifier. Updating of the weights of the training samples and base learners is achieved simultaneously in a cross-iterative process. In this way, the classifiers can take full advantage of the training data, and the strong classifier is constructed in a comprehensive manner. The experimental results reveal that the novel algorithm achieves higher precision than many classic semi-supervised algorithms at lower time cost and with a high convergence rate.

Significantly, FullBoost has higher accuracy, higher convergence speed, lower time consumption and lower labor cost than traditional semi-supervised algorithms. In a word, it is a very helpful tool for sentiment mining, given the marked insufficiency of labeled data (Li and Zhou, 2015) and the high labor cost of manual labeling (Manek *et al.*, 2016). The proposed method will be valuable in social mining-related fields, such as improving the

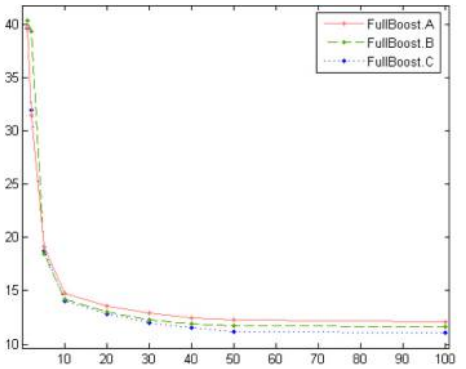

Figure 3.
Convergence rate of FullBoost

Algorithms \ Categories	Happy	Sad	Optimistic	Pessimistic	Positive	Negative	Endorsed	Against
FullBoost.A	0.876	0.879	0.881	0.882	0.880	0.885	0.879	0.873
FullBoost.B	0.881	0.880	0.879	0.888	0.887	0.889	0.881	0.886
FullBoost.C	0.884	0.891	0.894	0.889	0.896	0.887	0.884	0.887
TradaBoost	0.871	0.868	0.865	0.866	0.861	0.872	0.864	0.870
COREG	0.863	0.852	0.859	0.860	0.857	0.854	0.851	0.858
Semi-SVM	0.873	0.859	0.862	0.858	0.872	0.859	0.863	0.862
Web-based Self-training	0.854	0.850	0.852	0.853	0.848	0.853	0.856	0.857

Table II.
Precision comparison of FullBoost and several classic algorithms

performance of intelligent marketing, intelligent crisis management, automatic public opinion surveys, and so on.

However, the experimental results reveal that when the percentage of labeled data is higher than 85 per cent, the accuracy of FullBoost is not significantly higher than that of traditional AdaBoost. Therefore, the novel algorithm should be further modified to construct a complete framework that can deal perfectly with both semi-supervised and supervised problems. These modifications will be undertaken as future work in this field. In the complete framework, semi-supervised learning is undoubtedly the most difficult component and has now been solved by this study.

References

Astorino, A. and Fuduli, A. (2015), "Support vector machine polyhedral separability in semisupervised learning", *Journal of Optimization Theory and Applications*, Vol. 164 No. 3, pp. 1039-1050.

Cecotti, H. (2016), "Active graph based semi-supervised learning using image matching: application to handwritten digit recognition", *Pattern Recognition Letters*, Vol. 73 No. 2, pp. 76-82.

Gordon, D., Hendler, D. and Rokach, L. (2015), "Fast and space-efficient Shapelets-based time-series classification", *Intelligent Data Analysis*, Vol. 19 No. 4, pp. 953-981.

Isaac, T., García, S. and Herrera, F. (2015), "Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study", *Knowledge and Information Systems*, Vol. 42 No. 4, pp. 245-284.

Kingma, D.P., Mohamed, S., Rezende, D.J. and Welling, M. (2014), "Semi-supervised learning with deep generative models", *Advances in Neural Information Processing Systems*, Montréal, pp. 3581-3589.

Landesa-Vazquez, I. and Alba-Castro, J.L. (2012), "Shedding light on the asymmetric learning capability of AdaBoost", *Pattern Recognition Letters*, Vol. 33 No. 1, pp. 247-255.

Leng, Y., Xub, X. and Qi, G. (2013), "Combining active learning and semi-supervised learning to construct SVM classifier", *Knowledge-Based Systems*, Vol. 44 No. 8, pp. 121-131.

Li, Y.F. and Zhou, Z.H. (2015), "Towards making unlabeled data never hurt", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 37 No. 1, pp. 175-188.

Long, M., Wang, J., Ding, G., Pan, S.J. and Yu, P.S. (2014), "Adaptation regularization: a general framework for transfer learning", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 26 No. 5, pp. 1076-1089.

Lu, J., Behbood, V., Hao, P., Zuo, H., Xue, S. and Zhang, G. (2015), "Transfer learning using computational intelligence: a survey", *Knowledge-Based Systems*, Vol. 80 No. 5, pp. 14-23.

Lua, J., Hub, H. and Bai, Y. (2015), "Generalized radial basis function neural network based on an improved dynamic particle swarm optimization and AdaBoost algorithm", *Neurocomputing*, Vol. 152, pp. 305-315.

Manek, A.S., Shenoy, P.D., Mohan, M.C. and Venugopal, K.R. (2016), "Aspect term extraction for sentiment analysis in large movie reviews using Gini Index feature selection method and SVM classifier", *World Wide Web*, Vol. 20 No. 2, pp. 1-20.

Nikitidis, S., Nikolaidis, N. and Pitas, I. (2012), "Multiplicative update rules for incremental training of multiclass support vector machines", *Pattern Recognition*, Vol. 45 No. 5, pp. 1838-1852.

Patel, H. and Thakur, G.S. (2016), "A hybrid weighted nearest neighbor approach to mine imbalanced data", *Proceedings of the 12th International Conference on Data Mining (ICDM), IEEE, Las Vegas*, pp. 106-111.

Perlich, C., Dalessandro, B., Raeder, T., Stitelman, O. and Provost, F. (2015), "Machine learning for targeted display advertising: transfer learning in action", *Machine Learning*, Vol. 95 No. 1, pp. 103-127.

Schapire, R.E. and Singer, Y. (2000), "BoosTexter: a boosting-based system for text categorization", *Machine Learning*, Vol. 39 No. 3, pp. 135-168.

Subramanya, A. and Bilmes, J. (2011), "Semi-supervised learning with measure propagation", *Journal of Machine Learning Research*, Vol. 12 Nos 1/2, pp. 3311-3370.

Wang, B. and Pineau, J. (2016), "Online boosting algorithms for anytime transfer and multitask learning", *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, AAAI, Austin Texas*, pp. 3038-3044.

Wang, X. (2011), "Bilingual co-training for sentiment classification of Chinese product reviews", *Computational Linguistics*, Vol. 37 No. 3, pp. 587-616.

Zhang, M., Tang, J. and Zhang, X. (2014), "Addressing cold start in recommender systems: a semi-supervised co-training algorithm", *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, New York, NY, Gold Coast*, pp. 73-82.

Zhang, Z., Zhao, M. and Chow, T.W.S. (2015), "Graph based constrained semi-supervised learning framework via label propagation over adaptive neighborhood", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 27 No. 9, pp. 2362-2376.

Further reading

Mukherjee, I., Rudin, C. and Schapire, R.E. (2015), "The rate of convergence of AdaBoost", *The Journal of Machine Learning Research*, Vol. 14, pp. 2315-2347.

Corresponding author

Lei La can be contacted at: lalei.ece@hotmail.com