Boosting Linear Perceptrons for Unbalanced Data

Jangmin O

Artificial Intelligence Lab (SCAI)
School of Computer Science & Engineering
Seoul National University

jmoh@scai.snu.ac.kr

Byoung-Tak Zhang

Artificial Intelligence Lab (SCAI)
School of Computer Science & Engineering
Seoul National University

btzhang@scai.snu.ac.kr

Abstract

In information retrieval, lack of positive examples is a main cause of poor performance. In this case most learning algorithms may not capture proper characteristics in the data leading to low recall. To solve the problem of unbalanced data, we propose a boosting method that uses linear perceptrons as weak learners. The perceptrons are trained on local data sets. The proposed algorithm is applied to a text filtering problem for which only a small portion of positive examples is available. In the experiment on category "crude" of the Reuters-21578 document set, the boosting method achieved the recall of 80.8%, which is 37.2% improvement over multilayer perceptrons' with comparable precision.

KEYWORDS: Unbalanced data, AdaBoost, Boosting linear perceptrons

1 Introduction

Text filtering is a task which offers a user documents relevant to his interest about a large input document set [4]. For example, we can consider an agent which filters emails and presents filtered emails to the user. The agent must filter emails according to the user's interest.

General performance measures in information retrieval are recall/precision [9, 11, 12].

	+	–
+	a	b
–	c	d

Table 1 performance measure in IR.

Table 1 shows the matrix of recall/precision. The row represents the result IR system produces and the column represents the target value. Recall is a/(a+c) and precision is a/(a+b). Recall is the ratio of '+' examples which system correctly classified to real '+' examples. Precision is measured as the ratio of correctly classified '+' examples to examples which system answered as '+'. Since the user using a text filtering agent can only receive '+' classified documents, it is important to make the system have high performance in both recall and precision.

Text filtering intrinsically has imbalance between positive examples and negative examples. The portion of positive examples is typically much smaller than that of negative examples, because it is highly probable that most of documents are not relevant to a user's interest. These data are called unbalanced data. General learning algorithms such as multiplayer perceptrons and naïve Bayesian may produce poor result on unbalanced data.

In this paper, we introduce a boosting method based on linear perceptrons as a approach for solving the unbalanced data problem. The experimental results show that the proposed boosting method outperforms ordinary multiplayer perceptrons in terms of the recall performance of the text filtering system.

The rest of this paper is organized as follows. Section 2 describes boosting methods and boosting linear perceptrons. Section 3 shows the experimental results. Finally, Section 4 summarizes the results and draws conclusion.

2 Boosting Methods

Boosting is regarded as a kind of "committee machines". The committee machines are based on principle of divide and conquer, which is a strategy that a complex computational task is solved by dividing it into a number of computationally simple tasks and then combining the solutions to those tasks. In boosting, each expert is trained on data sets with entirely different distributions [2, 5]. It is a general method that can be used to improve the performance of any learning algorithm [1, 2, 5].

2.1 AdaBoost

AdaBoost is a kind of boosting methods which resamples examples from adaptively adjusted data distribution in order to train weak learner. The only requirement of weak learner is that the error rate of it should only be slightly less than 0.5 on entire training data. If all weak learners meet this requirement, then the training error of the combined learner decreases exponentially fast with the number of weak learners

that are combined [10]. AdaBoost adaptively adjusts data distribution according to the errors of the weak learner on entire training data. The bound on performance of AdaBoost depends only on the performance of the weak learner on those distributions that are adaptively adjusted during the learning process.

In the view of statistical learning theory, the error of a model can be decomposed of bias and variance terms [7].

$$mse(f(\mathbf{x},\mathbf{w})) = bias^2(f(\mathbf{x},\mathbf{w})) + \mathrm{var}(f(\mathbf{x},\mathbf{w}))$$

According to bias/variance dilemma, as the complexity of model increases, the bias term can be reduced but the variance term will be increased. So, a good model must balance between bias and variance terms. AdaBoost can produce more and more complex model through combining many weak learners and naturally can reduce bias term. But AdaBoost can also reduce variance term, which may be seen suspicious. Schapire analyzed this phenomenon with respect to the distribution of margins of the training examples. He shows that AdaBoost is effective at increasing the margins of the training examples, which is analogous to the case of Support Vector Machines (SVMs). SVMs aim to maximize the minimal margin and AdaBoost aim to minimize an exponential weighting of the examples as a function of their margin [3, 6, 8].

2.1 Boosting Linear Perceptrons

Boosting Linear Perceptrons is a AdaBoost algorithm using linear perceptrons as a weak learner. Linear perceptron is a binary linear separator and have error rate less than 0.5 in most cases.

$$\mathbf{wx} + b = 0$$

Since training process of linear perceptrons is very simple and fast, it is desirable as a weak learner.

Figure 1 shows the algorithm "Boosting Linear Perceptrons (BLP)". Here the target value is assumed to binary, +1 or –1. The learning process of BLP is based on the distribution of training data. First, the distribution is initialized with uniform distribution, $1/N$, where N is the size of training data. And T loops are iterated. In t-th iteration, the N data are sampled from current distribution of data. We use the roulette-wheel selection method for sampling. The t-th weak learner is trained using this sampled data. After training of that learner , the error ε_t between original data and classified results of weak learner t is calculated and the weight of the weak learner is also calculated. Next, the data distribution is updated so that easy examples, which mean that the weak learner can classify correctly, should be given lower probability but hard examples should be given higher probability. As a result, in next iteration, the weak learner concentrates on more hard examples.

Finally after T iterations, all learned weak learners are combined to compose a committee machine.

There are some heuristics about training of boosting. If data is so hard problem for weak learner, then in certain iteration, the error of weak learner on entire data set might be 0.5! If this situation happens the weight of weak learner gets 0, which doesn't have any influence on combined learner. Moreover next distribution can't be changed. Next weak learner using this unchanged distribution will become the same previous weak learner having error 0.5. So learning iteration must be stopped in this case before T iterations.

Input: $(x_1, y_1),\ldots,(x_N, y_N)$, $y_i=\{-1,+1\}$

Initialize $D_1(i) = 1/N$;

Do next loop for $t = 1, \ldots, T$.

- Generate N training data from distribution D_t
- Produce hypothesis h_t by training perceptron NN_t

  h_t: $X \rightarrow \{-1: NN_t < 0, +1: NN_t > 0\}$
- Compute the error and the weight of NN_t

$$\varepsilon_t = \sum_{NN_t(\mathbf{x}_i)\neq y_i} D_t(i)$$

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$$

- Update distribution D_t

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t}, & \text{if} \quad NN_t(\mathbf{x}_i) = y_i \\ e^{+\alpha_t}, & \text{if} \quad NN_t(\mathbf{x}_i) \neq y_i \end{cases}$$

Output:

$$H(\mathbf{x}) = \mathrm{sgn}\left(\sum_{t=1}^{T} \alpha_t NN_t(\mathbf{x})\right)$$

Figure 1 Algorithms of Boosting Linear Perceptrons.

3 Experiments

3.1 Methods

We experiment the text filtering on a document set "Reuters-21578". This document set has been widely used in text classification as typical performance measure set. It consists of five category sets which also consist of many sub-categories. In this paper, we experiment on three categories, "earn", "grain", and "crude" which belong to the category set, "TOPICS". For the "TOPICS", there are good three training test

divisions, “ModLewis”, “ModApte”, and “ModHayes”. We choose ModApte split. In this split, the number of training data is 8,762 and the number of test data is 3,009.

Document is transformed as a vector form of which elements are *tfidf* [6, 9]. $tf(w_i,\text{doc}_i)$ is the term frequency which means count of w_i occurring in document doc_i. And $idf(w_i)$ is inverse document frequency and is described as follows.

$$idf(w_i) = \log (N / df(w_i))$$

Where N is number of total documents and df is document frequency which means number of documents where w_i is occurring.

The dictionary used in representing document set is composed of 8,754 words. Original words are more than 20 thousands. In order to reduce the size of words, we use stop list and stemming algorithm.

Table 2 shows the ratio of positive examples in training and test data.

	train	test
earn	32.4%	34.6%
grain	4.7%	4.2%
crude	4.2%	5.2%

Table 2 Ratio of positive examples.

Category “earn” has good balance between positive examples and negative examples. But there are few positive examples in “grain” and “crude”.

In order to compare the performance of BLP with that of Multilayer perceptrons, we use a neural network with following specs. It consists of three layers. Hidden layer has 5 neurons and output layer has 1 neuron. Hidden layer uses hypertangent activation function and output layer uses linear activation function. Learning rate is fixed at 0.2 and momentum is fixed at 0.5. It’s learning process is repeated until LMS (Least Mean Square) error becomes lower than 0.005 or until learning epoch becomes 500.

In the case of BLP, the learning rate and momentem of linear perceptrons is fixed same to the case of multilayer perceptrons. And training is stopped when LMS error becomes lower than 0.01 or when the learning epoch is reached to 25. The reason why stopping condition of BLP on LMS error or learning epoch is less tighter than that of multilayer perceptrons is that this loose condition is sufficient to the requirement of weak learner. The number of boosting iterations is chosen to 100.

In order to compare the performance of BLP with the other learning algorithm, we experiment same task using naïve bayes classifier which is very simple and shows reasonable performance. It is trained using same dictionary as that of multilayer perceptrons and BLP.

To compare three learning algorithms, the F1 measure is introduced. F1 measure is also called recall/precision break-even point.

$$F1 = 2 \times \text{recall} \times \text{precision} / (\text{recall} + \text{precision})$$

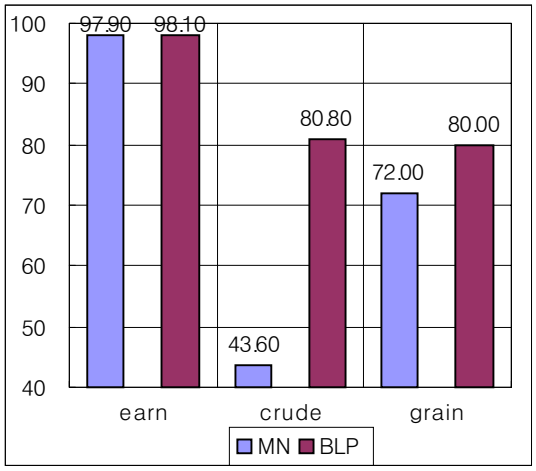

Figure 2 Comparison of recall performances between multilayer perceptrons and BLP.

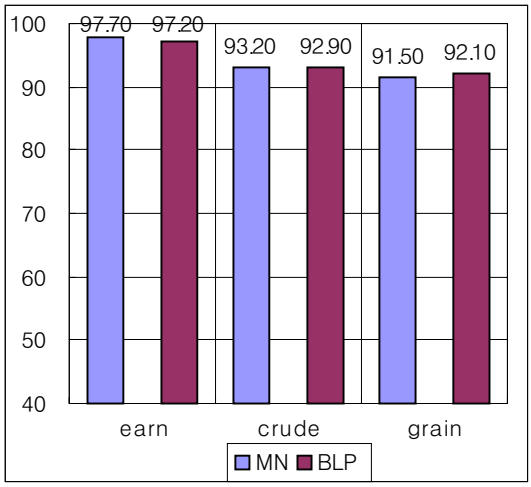

Figure 3 Comparison of precision performances between multilayer perceptrons and BLP.

F1	earn	crude	grain
MN	**97.80**	59.41	80.59
BLP	97.65	**86.43**	**85.62**
Naïve	97.70	57.72	77.46

Table 3 Comparison in terms of the F1 measure between 3 learning algorithms: Naïve means Naïve bayes classifier.

3.2 Results

Generalization performance about test data is shown in figure2 and figure 3. In above two figures, MN means Multilayer Perceptrons and BLP means Boosting Linear Perceptrons.

In problems such as pattern classification, the multiplayer perceptrons are known to have good precision. As figure 3 shows, the precision performance of the multiplayer perceptrons is comparable to the BLP's.

But in the case of recall, BLP is superior to multiplayer perceptrons. On the category, "earn", which has good balance between positive examples and negative examples, multiplayer perceptrons show good recall performance. However on the category, "crude", which has imbalance between positive examples and negative examples, the recall of multiplayer perceptrons is not good.

BLP is well adapted to this unbalanced problem. On "crude", there are only 370 positive examples in 8,762 training examples. Multiplayer perceptrons are known as learning algorithm which can capture the global characteristic of data. On unbalanced data, they may not find the critical information contained in positive examples.

BLP shows recall 80.80% which is very higher than 43.60% of multiplayer perceptrons. At each stage of boosting, linear perceptron as weak learner tries to find global characteristic about it's data as a neural network. But the data is not global in the view of total training data. It is local data that is sampled from data distribution of total training data. So we can say that each weak learner captures the local characteristic in total training data. And the weak learner in later stage of boosting captures more local characteristic. Because as boosting steps are repeated the hard examples are sampled more frequently, the weak learner more concentrated on hard examples. And finally produced committee machine can become a strong learner which can capture maximum information of data.

Figure 4 shows F1 statistics of 3 algorithms. Multilayer perceptrons are comparable with naïve bayes classifier. BLP is superior to others in recall and is comparable with others in precision.

4 Summary and Conclusions

In this paper, we proposed a boosting algorithm based on linear perceptrons (BLP) for unbalanced data. BLP was applied to the problem of text filtering. In text filtering, the lack of positive examples is too heavy a burden for recall. Our experimental results show that boosting methods are promising to solve the unbalanced problem in supervised learning of neural networks.

Boosting may fail to achieve good performance when the error of the weak learners is higher than 0.5 on the entire training data. Future work includes improving the learning algorithm in this case and employing more powerful weak learners to cover data sets having different characteristics.

Acknowledgments

This research was supported by Brain Science and Engineering Research Program sponsored by Korea Ministry of Science and Technology.

References

[1] L. Larkey and W. Croft, Combining Classifiers in Text Categorization, In *Proc. of SIGIR*, pp289-297, 1996.

[2] R.E. Schapire, A Brief Introduction to Boosting, In *Proc. of 6th IJCAI*, pp.1401-1401, 1999.

[3] R.E. Schapire, Y. Freund, P. Bartlett, and W.S. Lee, Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods, *The Annals of Statistics*, 26(5): 1651-1686, 1998.

[4] R.E. Schapire, Y. Singer, and A. Singhal, "Boosting and Rocchio applied to text filtering", In *Proc. of SIGIR-98*, pp.251-223, 1998.

[5] S. Haykin, *Neural Networks second edition*, Prentice-Hall, Inc., 1997.

[6] T. Joachims, Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In *Proc. of the ECML*, pp. 137-142, 1998.

[7] V. Cherkassky and F. Mulier, *Learning from Data Concepts*, Theory and Methods, John Wiley & Sons, Inc., 1997.

[8] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer Verlag, New York Inc., 1997.

[9] W.B. Frakes and R. Baeze-Yates, *Information Retrieval Data Structures & Algorithms*, Prentice-Hall, Inc., 1997.

[10] Y. Freund and R.E. Shapire, A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119-139, 1997.

[11] Y. Yang, An Evaluation of Statistical Approaches to Text Categorization, Technical Report CMU-CS-97-127, 1997.

[12] Y. Yang and J. Pederson, Feature Selection in Statistical Learning of Text Categorization. In *Proc. of ICML*, pp. 412-420, 1997.