Group-based Learning — A Boosting Approach

Weijian Ni
Nankai University
No. 94 Weijin Road,
Tianjin, China 300071
niweijian@gmail.com

Jun Xu, Hang Li
Microsoft Research Asia
No. 49 Zhichun Road,
Beijing, China 100190
{junxu,hangli}@microsoft.com

Yalou Huang
Nankai University
No. 94 Weijin Road,
Tianjin, China 300071
yellow@nankai.edu.cn

ABSTRACT

This paper points out that many machine learning problems in IR should be and can be formalized in a novel way, referred to as 'group-based learning'. In group-based learning, it is assumed that training data as well as testing data consist of groups. The classifier is created and utilized across groups. Furthermore, evaluation in testing and also in training are conducted at group level, with the use of evaluation measures defined on a group. This paper addresses the problem and presents a Boosting algorithm to perform the new learning task. The algorithm, referred to as AdaBoost.Group, is proved to be able to improve accuracies in terms of group-based measures during training.

1. INTRODUCTION

In many IR learning problems, data naturally forms groups and training and testing should be performed on the basis of groups. For example, in email spam filtering, emails are associated with users (receivers), and we can group emails according to users. Emails of different users may differ largely in content and style, while emails of the same user may be quite similar. Furthermore, evaluations in training and testing should be conducted at the user level. For example, F_1 scores should be calculated based on individual users, and we want to enhance the F_1 scores of all users.

In this paper, we aim to (1) give a formal definition to such kind of learning task, which we call group based learning, and (2) propose a learning method that can perform the task. First, group-based learning is formalized in a decision theoretic manner. (1) Training data as well as testing data are assumed to be comprised of different groups. The groups are supposed to be generated i.i.d. according to a fixed but unknown probability distribution. (2) The goal of learning is to construct a single classifier which can be applied across groups. (3) Evaluation in testing and also in training are conducted at group level. Second, a Boosting algorithm for group-based learning is proposed. The algorithm, named as AdaBoost.Group, manages to optimize the total performance in terms of *any* evaluation measures defined on a group.

2. GROUP-BASED LEARNING

We give a formal description of the group-based learning problem. Without loss of generality, we consider binary classification. Let $\mathcal{G}$ denote the space of groups, $\mathcal{X}$ the space of instances ($\mathcal{X} \subseteq \mathbb{R}^d$), and $\mathcal{Y}$ the space of labels ($\mathcal{Y} = \{+1, -1\}$ currently). Furthermore, let $\mathcal{Z}$ denote the space of all possible instance and label pairs:$\mathcal{Z} = \cup_{i=1}^{\infty}(\mathcal{X} \times \mathcal{Y})^i$. Let $(g, (\mathbf{x}, \mathbf{y})) \in \mathcal{G} \times \mathcal{Z}$ be a group g and its associated instance-label pairs $(\mathbf{x}, \mathbf{y})$, where $\mathbf{x} = (x_1, \cdots, x_{m(g)})$, $x_i \in \mathcal{X}$, $\mathbf{y} = (y_1, \cdots, y_{m(g)})$, and $m(g)$ denotes the number of instances.

In training, we are given a set of training data $S = \{(g_i, (\mathbf{x}_i, \mathbf{y}_i))\}_{i=1}^n$. It is supposed that $(g_i, (\mathbf{x}_i, \mathbf{y}_i))$ is generated i.i.d. according to the joint probability distribution $\mathbb{P}_{\mathcal{G}\times\mathcal{Z}}$. The goal of learning is to construct a classifier $f \in \mathcal{F}$ to predict y on the basis of x. The learning problem turns out to be that of minimizing the risk function

$$R^{\Delta}(f) = \int_{\mathcal{G}\times\mathcal{Z}} \Delta((g, (\mathbf{x}, \mathbf{y})); f) d\Pr(g, (\mathbf{x}, \mathbf{y})), \tag{1}$$

where $\Delta((g, (\mathbf{x}, \mathbf{y})); f)$ denotes the loss function. Then the empirical risk function is defined as

$$\hat{R}_S^{\Delta}(f) = \frac{1}{n} \sum_{i=1}^{n} \Delta((g_i, (\mathbf{x}_i, \mathbf{y}_i)); f). \tag{2}$$

It is easy to verify that with certain assumptions group-based learning will become equivalent to instance-based learning. Since $(g, (\mathbf{x}, \mathbf{y})) \sim \mathbb{P}_{\mathcal{G}\times\mathcal{Z}} = \mathbb{P}_{\mathcal{G}} \times \mathbb{P}_{\mathcal{Z}|\mathcal{G}}$, Equation (1) can be written as

$$R^{\Delta}(f) = \int_{\mathcal{G}} \int_{\mathcal{Z}} \Delta((g, (\mathbf{x}, \mathbf{y})); f) d\Pr((\mathbf{x}, \mathbf{y})|g) d\Pr(g). \tag{3}$$

We further assume that the total loss Δ over each group g can be decomposed linearly into a sum of losses δ over instances within g and that δ is independent of g, and further assuming that (x, y) is generated i.i.d. according to $\mathbb{P}_{\mathcal{X}\times\mathcal{Y}|\mathcal{G}}$, (3) can be written as

$$R^{\delta}(f) = \int_{\mathcal{G}} \int_{\mathcal{X}\times\mathcal{Y}} \delta(f(x), y) d\Pr(x, y|g) d\Pr(g). \tag{4}$$

The total loss then becomes an instance-based loss function:

$$\hat{R}_S^{\delta}(f) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{m(g)} \sum_{j=1}^{m(g)} \delta(f(x_{ij}), y_{ij}). \tag{5}$$

Group-based learning is related to multi-instance learning [1] (groups in this case correspond to bags in multi-instance learning); however, there are also differences between the two. In multi-instance learning, the bags are labeled but instances are not, and the task is to predict the label of a new bag. In group-based learning, the instances are labeled and the task is to predict the labels of instances in a new group.

3. ADABOOST.GROUP

We propose a novel group-based learning method. The algorithm, referred to as 'AdaBoost.Group', is a natural extension of the AdaBoost [3] algorithm to group-based learning. Figure 1 gives the pseudo code of the AdaBoost.Group algorithm.

CIKM'08, October 26–30, 2008, Napa Valley, California, USA.
ACM 978-1-59593-991-3/08/10.

Input :$S = \{(g_i, (\mathbf{x}_i, \mathbf{y}_i))\}_{i=1}^n$, performance E, and rounds T
Initialize $D_1(i) = 1/n$
For $t = 1, \cdots, T$

- Train weak classifier h_t with distribution $\boldsymbol{D}_t$ on S.
- Choose $\alpha_t = \frac{1}{2} \ln \frac{\sum_{i=1}^n D_t(i)\{1+E((g_i,(\mathbf{x}_i,\mathbf{y}_i));h_t)\}}{\sum_{i=1}^n D_t(i)\{1-E((g_i,(\mathbf{x}_i,\mathbf{y}_i));h_t)\}}$.
- Create $f_t = \sum_{k=1}^t \alpha_k h_k(x)$.
- Update $\boldsymbol{D}_{t+1}$: $D_{t+1}(i) = \frac{\exp\{-E((g_i,(\mathbf{x}_i,\mathbf{y}_i));f_t)\}}{\sum_{j=1}^n \exp\{-E((g_j,(\mathbf{x}_j,\mathbf{y}_j));f_t)\}}$.

End For
Output hypothesis $f(x) = f_T(x)$.

Figure 1: AdaBoost.Group algorithm.

AdaBoost.Group takes training data $S = \{(g_i, (\mathbf{x}_i, \mathbf{y}_i))\}_{i=1}^n$ as input and takes group-based evaluation measure E and number of iterations T as parameters. AdaBoost.Group runs T rounds and in each round it creates a weak classifier $h_t (t = 1, \cdots, T)$. Finally, it outputs a model f by linearly combining the weak classifiers.

The linear combination of weighted weak classifiers is defined as $f(x) = \sum_{t=1}^T (\alpha_t h_t(x))$, where $h_t(x)$ denotes the t^{th} weak classifiers, α_t the corresponding weight, and T the number of weak classifiers.

At each round t, AdaBoost.Group maintains a distribution of weights $\boldsymbol{D}_t$ over the groups in the training data. Initially, it sets equal weights to the groups. During the training, it increases the weights of those groups that are not predicted well by f_t, the model created so far. As a result, the learning at the next round will be focused on the creation of a weak classifier that can work on the prediction of those 'hard' groups.

At each round, a weak classifier h_t is constructed based on training data with weight distribution $\boldsymbol{D}_t$. The 'goodness' of a weak classifier is measured by the group-based evaluation measure E, which is assumed to take values from $[0, 1]$.

Once a weak classifier h_t is built, AdaBoost.Group chooses a weight $\alpha_t > 0$ for h_t. Intuitively, α_t measures the importance of h_t.

It can be proved that there exists a lower bound on the total performance of AdaBoost.Group with respect to training data, as shown in Theorem 1.

THEOREM 1. *The following bound exists for AdaBoost.Group:*

$$\frac{1}{n}\sum_{i=1}^n E((g_i, (\mathbf{x}_i, \mathbf{y}_i)); f_T) \geq 1 - \left(\prod_{t=1}^T e^{-\zeta_t^{min}} \sqrt{1-\varphi_t^2}\right),$$

where $\varphi_t = \sum_{i=1}^n D_t(i) E((g_i, (\mathbf{x}_i, \mathbf{y}_i)); h_t)$ *and* $\zeta_t^{min} = \min_{i=1,\cdots,n} \zeta_t^i$. *Here* $\zeta_t^i = E((g_i, (\mathbf{x}_i, \mathbf{y}_i)); f_t) - E((g_i, (\mathbf{x}_i, \mathbf{y}_i)); f_{t-1}) - \alpha_t E((g_i, (\mathbf{x}_i, \mathbf{y}_i)); h_t)$.

4. EXPERIMENTS

We conducted experiments to evaluate the performances of AdaBoost.Group on three applications: email spam filtering, acronym interpretation, and protein homology detection. For email spam filtering, the TREC 2005 dataset[1] was used; for acronym interpretation, we created a dataset based on the UCI data 'NSF Research Awards Abstracts 1990-2003'. The dataset contains 495 ground truth acronym-expansion pairs; for protein homology detection, we made use of the dataset in KDD Cup 2004[2]. We randomly divided the groups in these datasets into several even subsets and conducted cross validation. The results reported are those averaged over trials.

[1]http://plg.uwaterloo.ca/ gvcormac/treccorpus/
[2]http://kodiak.cs.cornell.edu/kddcup/datasets.html

Table 1: Experimental results in terms of accuracy.

	Email	Acronym	Protein
AdaBoost.Group (Accuracy)	**0.9567**	**0.9981**	**0.9970**
AdaBoost.Group (F_1)	0.9381	0.9975	0.9966
SVM^{perf} (Accuracy)	0.9442	0.9978	0.9968
SVM^{perf}(F_1)	0.9346	0.9977	0.9953
SVM (cost-sensitive)	0.9481	0.9979	0.9966
AdaBoost (cost-sensitive)	0.9488	0.9974	0.9965
SVM	0.9433	0.9972	0.9965
AdaBoost	0.9436	0.9972	0.9966

Table 2: Experimental results in terms of F_1 score.

	Email	Acronym	Protein
AdaBoost.Group (Accuracy)	0.6983	**0.9160**	0.6708
AdaBoost.Group (F_1)	**0.8645**	0.8918	**0.6848**
SVM^{perf} (Accuracy)	0.4804	0.8990	0.6636
SVM^{perf}(F_1)	0.4674	0.9017	0.6712
SVM (cost-sensitive)	0.5612	0.8849	0.6510
AdaBoost (cost-sensitive)	0.5130	0.8867	0.6601
SVM	0.4804	0.7932	0.6354
AdaBoost	0.4885	0.8298	0.6532

We chose several instance-based methods as baselines. State-of-the-art learning approaches SVM, AdaBoost, SVM^{perf} [2] were selected as baseline. Accuracy and F_1 score were adopted as the measures for training in SVM^{perf}, denoted as 'SVM^{perf} (Accuracy)' and 'SVM^{perf}(F_1)', respectively. We also employed cost-sensitive versions of SVM and AdaBoost, denoted as 'SVM (cost-sensitive)' and 'AdaBoost (cost-sensitive)'. For AdaBoost.Group, E can be a group-based evaluation measure. Here, E was defined as Accuracy or F_1 score, denoted as 'AdaBoost.Group (Accuracy)' or 'AdaBoost.Group (F_1)', respectively.

The experimental results for the tasks of email spam filtering, acronym interpretation, and protein homology detection in terms of accuracy and F_1 are summarized in Table 1 and Table 2, respectively. From the results, we can see that AdaBoost.Group outperforms all the baselines in terms of the group-based evaluation measures. We conducted significance tests (t-test) on the improvements of AdaBoost.Group (Accuracy) and AdaBoost.Group (F_1) over the baselines. Most of the improvements are statistically significant (p-value< 0.05).

5. CONCLUSION

In this paper we have proposed and formalized a new learning problem 'group-based learning'. In group-based learning, all the instances are clustered into groups and the evaluation is conducted at group level. To address the problem we have proposed a novel Boosting algorithm, named AdaBoost.Group. AdaBoost.Group can optimize an upper bound of the total loss function in terms of evaluation measure defined on a group.

6. REFERENCES

[1] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Perez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71, 1997.

[2] T. Joachims. A support vector method for multivariate performance measures. In *ICML'05*, pages 377–384, 2005.

[3] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Mach. Learn.*, 37(3):297–336, 1999.