Pattern Recognition Letters 68 (2015) 22–26

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

An efficient online active learning algorithm for binary classification☆

Dehua Liu[a], Peng Zhang[a,*], Qinghua Zheng[a]

Xi'an Jiaotong University, Xianning West Road 28, Xi'an 710049, China

ARTICLE INFO

Article history:
Received 5 May 2015
Available online 28 August 2015

Keywords:
Online active learning
Binary classification
Margin-based criterion
Iteratively decreased threshold

ABSTRACT

Active learning is an important class of machine learning where labels are queried when necessary. Most active learning algorithms need to iteratively retrain the classifier when new labeled data are obtained. Such a batch learning process can incur a high overhead in both time and memory. In this paper, we propose a new online active learning algorithm for binary classification. Our algorithm uses the margin-based criterion, which compares the margin of instances with a threshold to decide whether it should be queried. Especially, we propose Iteratively Decreased Threshold (IDT), a new threshold update method for the margin-based criterion. By iteratively decreasing the threshold with IDT, our algorithm can effectively reduce the number of queried instances. In addition, as evaluating the margin-based criterion involves only simple inner productions, our algorithm is also very efficient to evaluate. We compare our algorithm with other state-of-the-art online active learning algorithms on six data sets, demonstrating that it requires less queries to achieve the same classification accuracy, and incurs a smaller computation overhead at the same time.

1. Introduction

In traditional supervised learning, all instances should be annotated in prior to training, and the classifier is not allowed to query new annotated data points. Such a passive learning approach has the limitation that it is difficult to annotate a large data set. Active learning, on the other hand, allows the learner to dynamically query valuable data points (instances) for annotation, and iteratively refine the classifier. Many theoretical results suggest that active learning can effectively reduce the number of annotated instances [2,11,12,23]. In addition, many studies have also empirically verified the advantage of active learning over passive learning [1,13–15,20].

However, most batch algorithms used in active learning are computationally expensive. The reason is that in every iteration, the classifier needs to be retrained to calculate the next query. In many situations, the data set is very big, while the computation resource is limited. For example, it is expensive to implement an OCR system on a small mobile device with batch algorithms.

Due to the low complexity in both computation and memory of online learning, it has been used to make active learning more efficient [15]. Online learning can date back to about 50 years ago, when algorithms like perceptron [17] and stochastic gradient descent (SGD) [22] were proposed. In online learning, updating the classifier only requires the latest instance and $O(d)$ simple arithmetic operations, where d is the dimension of instance. One can refer to [19] for comprehensive review of online learning.

Generally, there are two steps in online active learning. The first step defines a selection (or sampling) criterion which measures the usefulness of instances, and based on this criterion selects the most informative instance to label. In the second step, the classifier is updated based on the newly added instances and labels. In this paper, we focus on the binary classification problem, where instances are supposed to be classified into two categories. We assume that the classifier is linear and is of the form $w^\top x + b$, where the sign categorizes an instance. Without loss of generality, we assume $b = 0$, which can be easily achieved by normalizing the data at the origin. We think it is reasonable to restrict the classifier to be linear, since any data set can be linearly separable if we map them to a higher dimensional feature space. Even in the nonlinearly separable case, linear classifier is a feasible choice.

As noted above, selection criterion is a critical component for active learning. There are many existing criterions proposed for active binary classification, e.g., uncertainty sampling [2], query by committee [11], expected error reduction [9]. However, many of these are fit for batch learning.

In this paper, we first propose *Iteratively Decreased Threshold* (IDT), a new threshold update method for margin-based selection criterion used in active learning. Then, we combine this criterion with the classical SGD updating rule to propose a new online active learning algorithm, simply named *IDT+SGD*. In our IDT+SGD algorithm, when there is a new instance, the margin of the instance is calculated. If

☆ This paper has been recommended for acceptance by Dr. D. Dembele.

* Corresponding author.: Tel.: +86 29 82668642(8020).
E-mail addresses: dehual@mail.xjtu.edu.cn (D. Liu), p-zhang@xjtu.edu.cn, p-zhang@mail.xjtu.edu.cn (P. Zhang), qhzheng@mail.xjtu.edu.cn (Q. Zheng).

http://dx.doi.org/10.1016/j.patrec.2015.08.010
0167-8655/

the margin is smaller than a given threshold, the instance will be selected; otherwise the instance is discarded without consideration. For selected instances, IDT+SGD queries their labels and uses SGD to update the classifier. At the end of each iteration, the threshold is updated based on the estimation of current classification error. Since evaluating the IDT margin-based criterion only involves simple inner productions, IDT+SGD is quite efficient to implement.

We experiment with six benchmark data sets, and find that compared with other online active learning algorithms, IDT+SGD achieves either better or comparable classification performance, depending on which data sets are used. Experimental results also show that IDT+SGD has a lower computation overhead compared with many other online active learning algorithms that depend on matrix inverse operations [6,7,10].

The remainder of the paper is organized as follows. In Section 2 we review some online binary active learning algorithms. Then, we propose our margin-based online active learning algorithm in Section 3. We empirically verify the effectiveness of our algorithm, and evaluate its computation time in Section 4. Finally, we conclude in Section 5.

2. Related work

Online active binary classification involves several steps $t = 1, 2, 3, \ldots$. At each step, the learner receives an instance $x_t \in \mathbb{R}^d$, and decides whether its label y_t should be queried. If the label is queried, the learner updates the classifier.

Monteleoni and Kaariainen [15] compared several online active algorithms, which are classified based on their sample selection methods and updating rules. Sample selection methods include DKM [8], CBGZ [5], and random; updating rules include perceptron and verified perceptron [8]. Both DKM and CBGZ are based on margin. In DKM, the absolute value of an instance margin is compared with a threshold, and the threshold is cut to its half if consecutive predictions are all correct (the length of sequence is predefined as a hyperparameter). In CBGZ, the learner queries the label with probability $\frac{b}{b+|\hat{p}|}$, where $\hat{p} = w^\top x$ is the margin of instance and b is a constant. Previous results [15] show that perceptron outperforms verified perceptron when combined with all three sample selection rules, and that when using perceptron as the updating rule, DKM and CBGZ have almost the same performance.

Apart from margin-based online active learning, there are some other algorithms, such as regularized least square based algorithms [6,10]. In these algorithms, the sample selection methods and updating rules are both defined through the least squares. Consider the BBQ algorithm [6] for example. Let $S_{t-1} = [x_1, \ldots x_{N_{t-1}}]$, $Y = [y_1, \ldots, y_{N_{t-1}}]^\top$, and $A_t = I + S_{t-1}S_{t-1}^\top + x_t x_t^\top$. $r_t = x_t^\top X_t^{-1} x_t$. If $r_t > t^{-\kappa}$, the corresponding label is queried, and parameter of linear classifier is updated as $w_t = A_t^{-1} S_{t-1} Y_{t-1}$. The authors empirically show that this algorithm has a big improvement over random sampling. Combining the margin idea and regularized least square, [10] proposes a compound algorithm. They show theoretically that both their regret and sample complexity bounds have a strict improvement over previous algorithms. But these two regularized least square based algorithms are both computationally expensive since matrix inverse is involved in every iteration.

As the linear classifier $f(x) = w^\top x$ is parameterized by vector w, it is reasonable to give w a prior. [7] assumes that w satisfy a multi-variant Gaussian distribution $\mathcal{N}(w_t|\mu_t, \Sigma_t)$. The likelihood $P(y_i|x_i, w)$ is given by the probit function $\phi(y_i w^\top x)$, where $\phi(\cdot)$ is the cumulative distribution function of the standard Gaussian distribution. Then the parameter w is determined by posterior, and the authors devise an online updating rule for w. It is shown that this algorithm is more powerful in the dynamic problem than in setting where instances are i.i.d. Similarly as the above two algorithms, this algorithm also requires expensive matrix inverse operations. Moreover, the matrices must be positive definite, which is generally not guaranteed in practice.

3. Margin-based active learning algorithm

Before introducing our sample selection criterion, let us first define what margin means. There are several definitions for margin in binary classification. Suppose the values of labels are either 1 or -1, then the margin of an instance can be defined as $p(y = 1|x) - p(y = -1|x)$ [18], where $p(y|x)$ is the conditional probability of category y given x. If the margin of an instance is around zero, then the instance is quite close to the separator of two classes. As another definition, the margin of instance x is defined as $w^\top x$, where w is the parameter of classifier. For convenience, we assume w to be of unit length. Then $|w^\top x|$ measures the distance of x to the separator. Here we adopt the second definition, since it is easy to calculate and has a direct geometric intuition. This definition has already been used in many previous works [3,5,8]. We assume the optimal linear classifier $h_*(x) = w_*^\top x$. In the nonlinear separable case, w_* or h_* has a classification error ν which is the smallest error of linear classifiers.

3.1. Overview

The basic idea of our algorithm is as follows. For each new instance, we compare its margin with a threshold. If the margin is smaller than the threshold, the label of the instance is queried; otherwise the instance is discarded. The threshold takes the form of $c(2\nu + \epsilon)$, and decreases at every iteration. Here, ϵ is the access error (the classification error minus the Bayesian error). The reason is that if active learning is effective, the number of labeled and unlabeled instances should be almost the same as passive learning when they perform equally well on classification. We will give a more detailed discussion on how to determine the threshold.

After we decide to query the label of an instance, the parameter of linear classifier is updated using SGD, which been widely used in online learning, due to its simplicity and reliable performance. Define $\ell(f(x_i), y_i)$ as the loss function that measures the cost of prediction $f(x_i)$, where y_i is the true label. If there are n instances, then the empirical risk is defined as $R_n(f) = \frac{1}{n}\sum_{i=1}^n \ell(f(x_i), y_i)$. In our linear setting, $f(x) = w^\top x$, so $R_n(f)$ can be written as $\frac{1}{n}\sum_{i=1}^n \ell(w^\top x_i, y_i)$.

Gradient descent is a batch algorithm used to minimize $R_n(f)$ with respect to w, and SGD is simply the online version of gradient descent. In each iteration, SGD updates the classifier as:

$$w_{t+1} = w_t - \gamma_t \frac{\partial}{\partial w}\ell(w_t^\top x_t, y_t).$$

Here we choose hinge loss as our loss function, i.e., $\ell(w^\top x, y) = (\gamma - y w^\top x)_+$, and let $\gamma_t = \frac{1}{\sqrt{t}}$. Then, the SGD updating rule is:

$$w_{t+1} = w_t + \frac{1}{\sqrt{t}} y_t x_t, \text{ if } y_t w_t^\top x_t < \frac{1}{\sqrt{t}}; \text{ or } w_t, \text{ otherwise.} \quad (1)$$

3.2. The IDT+SGD algorithm

Our IDT+SGT online active learning algorithm is summarized as Algorithm 1.

Line 3–6 calculate the mean of all queried instances, and map x_t into a ball centered at the origin. Line 7–8 calculate the maximum norm R_t of all queried instances, and use it to normalize x_t. Note that both the mean and maximum norm are online estimates, as we cannot determine them until all the instances are queried. Line 9 calculates the margin of x as $w^\top x$, and compares it with the current threshold s_t. If the margin is smaller than the current threshold, then Line 10–11 apply SGD to update the classifier w_t, and Line 12 normalizes the new w_t. ϵ_t in line 15 is the upper bound estimation of access error. This estimation is based on a conjecture that at iteration t, the access

Algorithm 1 Iteratively Decreased Threshold (IDT) with SGD.

1: initialize w_1 (random), $mean := 0$, $s_1 := 1$, $R_0 := 1$, $r_0 := 0$, $\sigma_0 := 0$
2: **for** $t = 1, 2, 3, \ldots$ **do**
3: $mean := (1 - 1/t)mean + x_t/t$
4: **if** $t > 1$ **then**
5: $x_t := x_t - mean$
6: **end if**
7: $R_t := \max(R_{t-1}, \|x_t\|_2)$
8: $x_t := x_t/R_t$
9: **if** $|w_t^\top x_t| < s_t$ **then**
10: **if** $y_t w_t^\top x_t < \gamma$ **then**
11: $w_{t+1} := w_t + \frac{1}{\sqrt{t}} y_t x_t$
12: $w_{t+1} := w_{t+1}/\|w_{t+1}\|_2$
13: **end if**
14: **end if**
15: $\epsilon_t := \log(t)/\sqrt{t}$
16: $s_{t+1} := c(2\nu + \epsilon_t)$
17: **end for**

error of Algorithm 1 does not exceed the passive online SGD with t labeled instances. Formally, the conjecture is stated as follows.

Conjecture 1. Algorithm 1 has smaller or nearly the same classification error compared to SGD after accessing the same number of instances (both labeled and unlabeled instances are counted).

We will use experiments to empirically verify it in Section 4. Recall that access error of passive online SGD is of the order $\log(t)/\sqrt{t}$ [4]. We use it to update the threshold s_t in Line 16. The determination of hyper-parameters ν, c and γ will be discussed in Section 4.

3.3. The determination of s_t

First, let us state our goal when determining s_t. Let w_* be the Bayesian classifier with error rate ν. Then, our goal is to ensure the property that for instances $\{x : |w_t^\top x| > s_t\}$, our classifier w_t gives the same predication as w_*. This means the instances that are not informative to our classifier (w_t and w_* gives the same classification) would be filtered out by the threshold. Clearly, to reduce the number of queried instance, we should also make s_t as small as possible to filter out the most instances.

We continue to introduce a candidate for s_t. For iteration t, define θ_t as the angle of w_t and w_*. Clearly, if $|w_t^\top x| > \theta_t$, $w_t(x)$ and $w_*(x)$ predict the same label. This can be equivalently stated as $\{x : \text{sign}(h_t(x)) \neq \text{sign}(h_*(x))\} \subset \{x : |w_t^\top x| \leq \theta_t\}$. Thus, we can set $s_t \geq \theta_t$.

In the following, we give an upper bound for θ_t. Let P be the distribution of instances, there exists a constant c_1 such that

$$P\{x : |w_t^\top x| \leq \theta_t\} \leq c_1 P\{x : \text{sign}(h_t(x)) \neq \text{sign}(h_*(x))\} \quad (2)$$

For the right hand of Eq. (2), we have:

$$\begin{aligned} &P\{x : \text{sign}(h_t(x)) \neq \text{sign}(h_*(x))\} \\ &= \mathbb{E}[yh_t(x) > 0 \wedge yh_*(x) < 0] + \mathbb{E}[yh_t(x) < 0 \wedge yh_*(x) > 0] \\ &\leq \epsilon_t + 2\mathbb{E}[yh_t(x) > 0 \wedge yh_*(x) < 0] \leq 2\nu + \epsilon_t \end{aligned}$$

For the left hand of Eq. (2), as the data satisfy uniform distribution, we have:

$$\theta_t \leq c_2 P\{x : |w_t^\top x| \leq \theta_t\}$$

for some constant c_2. Putting the above equations together, we have:

$$\theta_t \leq c(2\nu + \epsilon_t),$$

where $c = c_1 c_2$. Thus, we set $s_t = c(2\nu + \epsilon_t)$.

For cases where distribution of data points are not uniform, the above discussion still holds, if there exists constant $\alpha > 0$, $\beta > 0$ such that $P\{x : |w_t^\top x| \leq \theta_t\} \leq \alpha P\{x : h_t(x) \neq h_*(x)\}$ and $P\{x : |w^\top x| < r\} \geq \beta r$.

4. Experimental results

Many online active algorithms have been proposed recently [6,7,18]. In the algorithm proposed by Chu et al. [7], one needs to inverse a positive definite matrix. Although the matrix can be replaced by diagonal form, the positive definiteness of matrix is still not guaranteed. As a result, it is not suitable for many data sets. Although [18] has given a perfect theoretical analysis, their active algorithm usually queries all the instances. Because the active query criterion is $\theta_t^2 \geq \Delta_t^2 = (w_{t-1}^\top x_t)^2$ and $\theta_t^2 = x_t^\top A_{t-1}^{-1} x_t (1 + 4\sum_{i=1}^{t-1} z_i r_i + 36\log(t/\delta))$ is generally much larger than Δ_t^2. For the above reasons, here we choose to compare our algorithm with

- The standard SGD algorithm [22];
- Cesa09 [6];
- DKM+SGD, a DKM active learning algorithm [8] with SGD as the updating rule;
- CBGZ+SGD, another online active learning algorithm [5], also with SGD as the updating rule.

One reason that we use SGD as the updating rule for DKM and CBGZ is that many studies show SGD performs better than perceptron and verified perceptron used in [8]. In addition, it is more fair to compare our algorithm (IDT+SGD) with algorithms also using SGD. Here the SGD updating rule is given by Eq. (1).

Our experiments use six data sets, namely protein01, protein02, splice, a2a, a3a, a4a. All of them are downloaded from the LIBSVM website[1]. Here, protein01, protein02 are two subsets of data set protein consisting of three classes, where Protein01 consists of categories 0 and 1, and protein02 consists of categories 0 and 2. The goal is to predict the protein secondary structure [21]. splice is from UCI repository[2]. The goal is to recognize two types of splice junctions in DNA sequences. a2a, a3a, a4a come from UCI Adult data set[3]. The goal is to predict whether a person has an income greater than $50,000. The original data has 14 attributes and is transformed yielding 123 features [16]. For each problem, we merge the original training set with the test set, and reshuffle them. Then, we randomly select a subset as the training set, and the rest as the test set. All the problems are non-separable.

There are some hyper-parameters in each of the four active learning algorithms: constant c and Bayesian error ν in algorithm IDT+SGD, R to control the decrease of threshold in DKM+SGD, constant b to estimate the probability of sampling in CBGZ+SGD, and κ to control the query rate in Cesa09. For each data set, we tune these hyper-parameters with 5-fold cross-validation on a separate holdout set. The holdout set is randomly chosen from the whole training set, and its size is given in the second column of Table 1. For example, "CV:5000" in problem protein01 means 5000 instances are randomly selected for 5-fold cross-validation (4000 instances for training and the remaining 1000 for test). The SGD updating rule used in all algorithms (except Cesa09) has a hyper-parameter γ, which is also determined through cross-validation. Since it is impossible to find a common optimal γ for all these four algorithms, we choose to tune it only for passive SGD. We find that the average error rates on the six data sets are almost the same when $\gamma \in [0.03, 0.06]$, and become worse when γ is out of this interval. In our experiments, we choose $\gamma = 0.03$ for all algorithms except Cesa09.

[1] http://www.csie.ntu.edu.tw/~cjlin/libsvm/
[2] http://archive.ics.uci.edu/ml/datasets
[3] http://archive.ics.uci.edu/ml/datasets/Adult

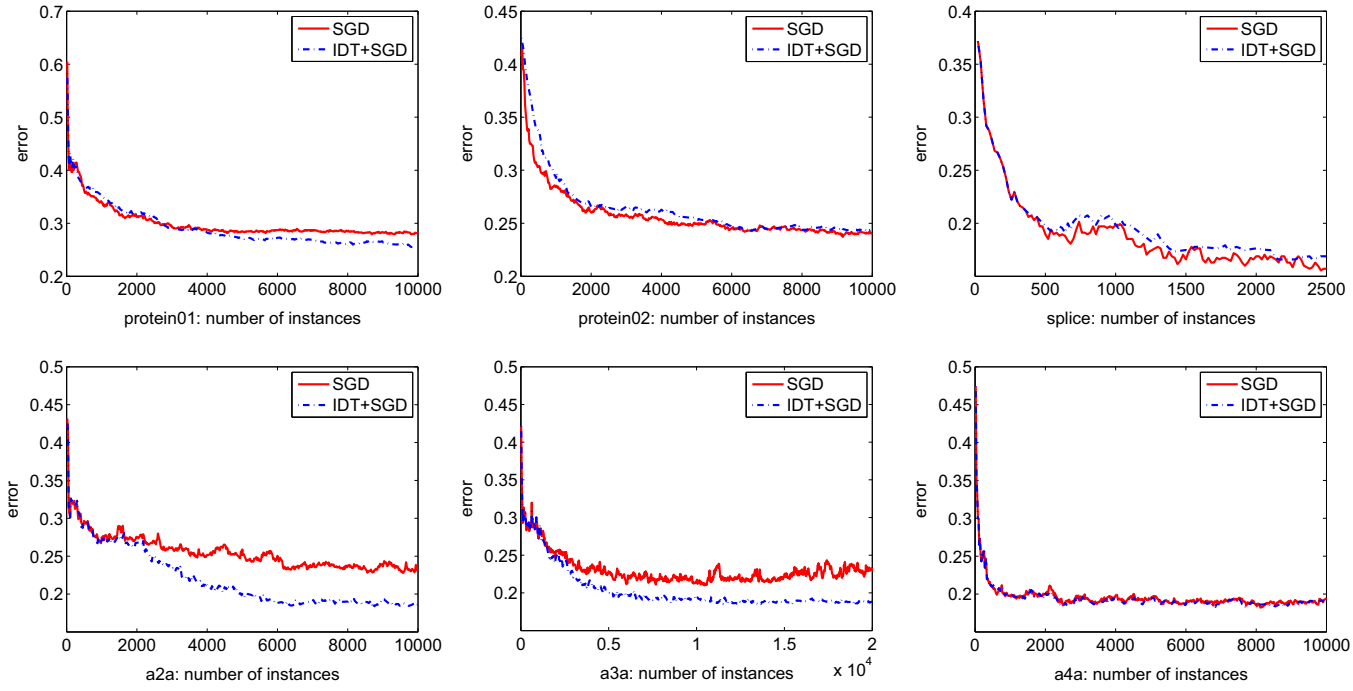

Fig. 1. Error rates of IDT+SGD and SGD, when they access the same number of instances (both labeled and unlabeled instances are counted).

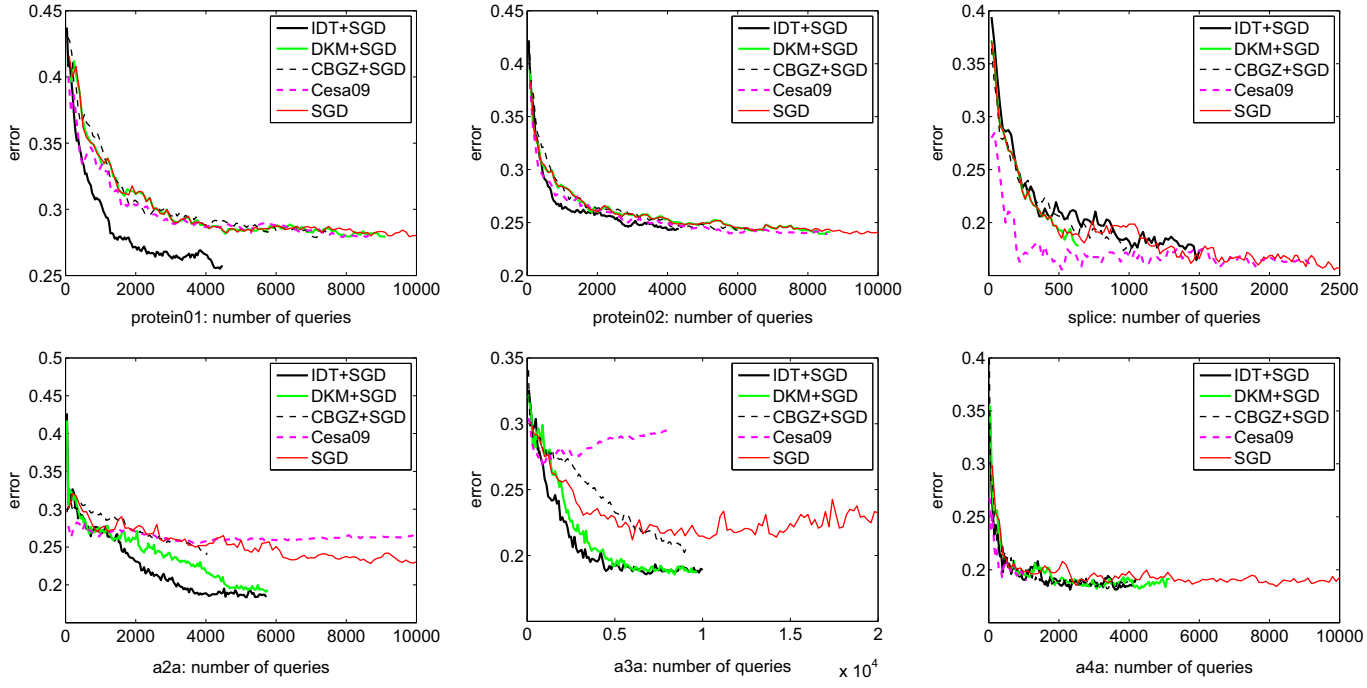

Fig. 2. Comparison of IDT+SGD with the other four learning algorithms on error rate.

4.1. Verification of conjecture 1

In Section 3, we give the conjecture that the threshold s_t is updated so that IDT+SGD has classification error no larger than native SGD if they access the same number of instances (both labeled and unlabeled instances are counted). Here, we verify this conjecture via experiments using six data sets. The results are reported in Fig. 1. We can observe that for problems protein01, a2a, and a3a, IDT+SGD has a better performance than SGD, and in the other three problems their performance is similar. Especially for problem a4a, their learning curves are almost identical. Note that here we are not comparing our algorithm (IDT+SGD) with SGD, as the comparison would be unfair. For SGD, it needs to query for all instances, while IDT+SGD only selectively query a subset of them. Rather, we are only aimed to verify our conjecture that IDT+SGD has comparable performance with SGD, if they access the same number of instances.

4.2. Comparison on learning performance

We continue to compare our algorithm with other online active learning algorithms, in terms of learning performance. For each data set, we run all the algorithms 5 times, and prior to each run, we reshuffle the data set for randomness. The average learning curves are shown in Fig. 2, where n is the size of training set, the x-axis

Table 1
Comparison of IDT+SGD with the other four learning algorithms on running time.

Data	Data information			CPU time (s)				
	Train size	Test size	Dimension	IDT+SGD	DKM+SGD	CBGZ+SGD	Cesa09	SGD
protein01	10000 (CV:5000)	6561	357	0.45	0.70	0.66	31.32	0.75
protein02	10000 (CV:5000)	9136	357	0.49	0.85	0.71	28.40	0.97
splice	2500 (CV:1250)	675	60	0.02	0.01	0.02	0.38	0.02
a2a	10000 (CV:5000)	5000	123	0.12	0.11	0.15	4.77	0.15
a3a	20000 (CV:10000)	9000	123	0.42	0.37	0.47	6.49	0.62
a4a	10000 (CV:5000)	3306	123	0.09	0.09	0.13	2.45	0.11

represents the number of queries, and the y-axis represents the classification error. Note that SGD is the baseline for all these algorithms: an algorithm is only considered effective if its learning curves is below that of native SGD.

We can observe that IDT+SGD has a remarkable advantage over SGD on data set protein01, a2a, a3a, and small advantage over SGD on data set protein02. In addition, the advantage of our algorithm over native SGD is more remarkable than that in Fig. 1. For example, for problem protein01, two curves are similar in Fig. 1, while the curve of IDT+SGD is strictly under that of SGD in Fig. 2. The reason is that in Fig. 1, both labeled and unlabeled instances are counted for IDT+SGD, while in Fig. 2, only labeled instances are counted.

Also we can observe that DKM+SGD algorithm performs better than SGD on data sets a2a and a3a, but a little worse than IDT+SGD. CBGZ+SGD has an advantage over SGD only on data set a3a. Cesa09 has an obvious advantage over SGD only on data set splice, but worse than SGD on data set a2a and a3a. In sum, our IDT+SGD algorithm has generally better or comparable performance over other online active learning algorithms, namely DKM+SGD, CBGZ+SGD, and Cesa09.

4.3. Comparison on computation overhead

Then, we evaluate the computation overhead of our algorithm, in terms of running time. The experiments are done using MATLAB R2011a, on a Windows desktop with Intel quad-core i5-3470CPU@3.2 GHz. For comparison, we also include the other four algorithms. The results are reported in Table 1, which shows that our IDT+SGD algorithm is generally more efficient than Cesa09 and SGD, and has roughly the same performance as DKM+SGD, CBGZ+SGD.

We explain a little bit more about Table 1. First, note that the running time of a learning algorithm depends on two factors: (1) the number of iterations, i.e., the number of queried instances; (2) the computation complexity of each iteration. Since Cesa09 has a much higher complexity per iteration than the other four algorithms (due to the matrix inversion operations), its running time is always the longest. The running times of the other four algorithms largely depend on the number of queried instances, which is represented as the largest x-axis value of each curve in Fig. 2. We can see that IDT+SGD only queries half (or even fewer) of all instances for data sets except splice, while its passive counterpart SGD always needs to query all instances. Thus, IDT+SGD generally has a smaller running time than SGD. Finally, note that IDT+SGD, DKM+SGD, and CBGZ+SGD query roughly the same number of instances for data sets a2a, a3a and a4a, thus their running times are quite close for these data sets.

5. Conclusion

In this paper, we proposed a new online active learning algorithm for binary classification. The key of our algorithm is a new threshold updating method used in margin based criterion in which the threshold is decreased at the end of each iteration, based on the estimation of error rate. The label of the instance is queried only if it has a margin smaller than the threshold. Experiments on six data sets show that our algorithm is effective in reducing the number of queries compared to other online active learning algorithms, and is more efficient in computation.

Acknowledgments

Dehua Liu, Peng Zhang, and Qinghua Zheng are with the MOE KLINNS Lab, Department of Computer Science and Technology, Xi'an Jiaotong University. This work is supported in part by NSFC (Nos. 91118005, 91218301, 61221063, 61402357), the Fundamental Research Funds for the Central Universities, and MOE IRT 13035.

References

[1] O.M. Aodha, N.D. Campbell, J. Kautz, G.J. Brostow, Hierarchical subquery evaluation for active learning on a graph, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2014, pp. 564–571.
[2] M.-F. Balcan, A. Beygelzimer, J. Langford, Agnostic active learning, in: Proceedings of the 23th International Conference on Machine Learning, ACM, 2006, pp. 65–72.
[3] M.-F. Balcan, A. Broder, T. Zhang, Margin based active learning, in: Learning Theory, Springer, 2007, pp. 35–50.
[4] L. Bottou, Stochastic gradient descent tricks, in: Neural Networks: Tricks of the Trade, Springer, 2012, pp. 421–436.
[5] N. Cesa-Bianchi, C. Gentile, L. Zaniboni, Worstcase analysis of selective sampling for linear-threshold algorithms, in: Proceedings of the NIPS, 2004.
[6] N. Cesa-Bianchi, C. Gentile, F. Orabona, Robust bounds for classification via selective sampling, in: Proceedings of the 26th Annual International Conference on Machine Learning, ACM, 2009, pp. 121–128.
[7] W. Chu, M. Zinkevich, L. Li, A. Thomas, B. Tseng, Unbiased online active learning in data streams, in: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2011, pp. 195–203.
[8] S. Dasgupta, A.T. Kalai, C. Monteleoni, Analysis of perceptron-based active learning, in: Proceedings of 18th Annual Conference on Learning Theory, 2005.
[9] S. Dasgupta, D. Hsu, C. Monteleoni, A general agnostic active learning algorithm, in: Proceedings of the Advances in Neural Information Processing Systems, vol. 3, 2007.
[10] O. Dekel, C. Gentile, K. Sridharan, Selective sampling and active learning from single and multiple teachers, J. Mach. Learn. Res. 13 (1) (2012) 2655–2697.
[11] Y. Freund, H.S. Seung, E. Shamir, N. Tishby, Selective sampling using the query by committee algorithm, Mach. Learn. 28 (1997) 133–168.
[12] S. Hanneke, Activized learning: transforming passive to active with improved label complexity, J. Mach. Learn. Res. 13 (2012) 1469–1587.
[13] S.-S. Ho, H. Wechsler, Query by transduction, IEEE Trans. Pattern Anal. Mach. Intell. 30 (9) (2008) 1557–1571.
[14] A.J. Joshi, F. Porikli, N.P. Papanikolopoulos, Scalable active learning for multiclass image classification, IEEE Trans. Pattern Anal. Mach. Intell. 34 (11) (2012) 2259–2273.
[15] C. Monteleoni, M. Kaariainen, Practical online active learning for classification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR'07, IEEE, 2007, pp. 1–8.
[16] J. Platt, et al., Fast training of support vector machines using sequential minimal optimization, Adv. Kernel Methods Support Vector Learn. 3 (1999).
[17] F. Rosenblatt, The perceptron: a probabilistic model for information storage and organization in the brain, Psychol. Rev. 65 (6) (1958) 386.
[18] B. Settles, Active learning, Synth. Lect. Artif. Intell. Mach. Learn. 6 (1) (2012) 1–114.
[19] S. Shalev-Shwartz, Online learning and online convex optimization, Found. Trends Mach. Learn. 4 (2) (2011) 107–194.
[20] S. Tong, D. Koller, Support vector machine active learning with applications to text classification, J. Mach. Learn. Res. 2 (2002) 45–66.
[21] J.-Y. Wang, Application of support vector machines in bioinformatics Ph.D. thesis, National Taiwan University, 2002.
[22] B. Widrow, M.E. Hoff, et al., Adaptive switching circuits, in: Ire Wescon Conv.Record, Defense Technical Information Center, 1960.
[23] L. Yang, S. Hanneke, Activized learning with uniform classification noise, in: Proceedings of the 30th International Conference on Machine Learning, 2013, pp. 370–378.