This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2019.2955078, IEEE Transactions on Knowledge and Data Engineering

Online Adaptive Asymmetric Active Learning with Limited Budgets

Yifan Zhang, Peilin Zhao, Shuaicheng Niu, Jiezhang Cao, Junzhou Huang, Qingyao Wu and Mingkui Tan

Abstract—Online Active Learning (OAL) aims to manage unlabeled datastream by selectively querying the label of data. OAL is applicable to many real-world problems, such as anomaly detection in health-care and finance. In these problems, there are two key challenges: the query budget is often limited; the ratio between classes is highly imbalanced. In practice, it is quite difficult to handle imbalanced unlabeled datastream when only a limited budget of labels can be queried for training. To solve this, previous OAL studies adopt either asymmetric losses or queries (an isolated asymmetric strategy) to tackle the imbalance, and use first-order methods to optimize the cost-sensitive measure. However, the isolated strategy limits their performance in class imbalance, while first-order methods restrict their optimization performance. In this paper, we propose a novel Online Adaptive Asymmetric Active learning algorithm, based on a new asymmetric strategy (merging both asymmetric losses and queries strategies), and second-order optimization. We theoretically analyze its mistake bound and cost-sensitive metric bounds. Moreover, to better balance performance and efficiency, we enhance our algorithm via a sketching technique, which significantly accelerates the computational speed with quite slight performance degradation. Promising results demonstrate the effectiveness and efficiency of the proposed methods.

Index Terms—Active Learning; Online Learning; Class Imbalance; Budgeted Query; Sketching Learning.

1 Introduction

Due to rapid growth of data and computational resources, machine learning addresses more and more practical problems, powering many aspects of modern society [1], [2], [3], [4], [5]. Nevertheless, many machine learning methods require the availability of sufficient off-line data before training, while those off-line samples are required to be i.i.d. [7], [8]. However, in many real-world applications, data comes in an online manner and the i.i.d. assumption may not hold. To address these limitations, online learning has emerged as a powerful tool [9], [10], [11], [12]. It makes no assumptions about the distribution of data and thus is data efficient and adaptable [7], [8].

Most existing online methods assume all samples are labeled, and ignore the labeling cost as well as the budget control problem. However, in many applications like medical diagnosis [2] and malicious URL detection [5], [13], the cost of annotation is often expensive. Hence, it is important to find out samples, which deserve to be labeled from data streams. To handle this task, online active learning (OAL) [14], [15] has emerged. It aims to train a well-performed model by selectively querying only a small number of labels for data streams. Many studies [4], [14], [15] have found that different query rules result in very different performance, which means that the query strategy is very important. Meanwhile, real-world companies usually expect to spend as few funds as possible for data annotation. In other words, we only have a limited budget for label querying. Given a limited budget, we have to select the most informative samples to query so that they can help to train a well-performed model.

- *Y. Zhang, S. Niu, J. Cao, Q. Wu and M. Tan are with South China University of Technology, China. E-mail: {sezyifan, sensc, secaojiezhang}@mail.scut.edu.cn; {qyw, mingkuitan}@scut.edu.cn.*
- *P. Zhao and J. Huang are with Tencent AI Lab, China. Email: peilinzhao@hotmail.com; joehhuang@tencent.com.*
- *P. Zhao and S. Niu are the co-first authors; Corresponding to M. Tan.*

In addition, the class-imbalanced issue seriously affects algorithm performance in real-world applications, such as cancer diagnosis [2], financial credit monitoring [3] and network fraud detection [5]. Existing OAL methods usually train models using balanced *accuracy* or *mistake rate* as metrics. However, they cannot handle the imbalance issue well [10]. To solve this, researchers have proposed more informative metrics, such as the weighted *sum* of *sensitivity* and *specificity*, and the weighted *misclassification cost* [10].

Based on these metrics, a pioneering cost-sensitive online active learning algorithm (CSOAL) [5] was proposed to directly optimize asymmetrically cost-sensitive metrics for OAL. However, this method only adopts a symmetric query rule [14] and ignores the imbalance problem in data selection. Recently, online asymmetric active learning algorithm (OAAL) [4] discovered that using asymmetric query strategy helps to handle imbalanced data better. However, this method overlooks imbalance issues in the optimization process and tends to query more majority data due to the recommended parameter settings [4]. Hence, this method may lead to poor performance on minority data. In comparison, CSOAL is "asymmetric update plus symmetric query", and OAAL is "symmetric update plus asymmetric query". Both algorithms devise the asymmetric strategy from a different and isolated perspective, and thus may perform insufficiently in class-imbalance problems.

In addition, both algorithms only consider first-order information of data streams. However, when scales of different features vary significantly, these methods may converge slowly [9]. As a result, it is difficult for them to achieve a good solution when labeled data is quite limited. To deal with this issue, recent studies [16], [17] have found second-order information (*i.e.*, correlations between features) helps

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2019.2955078, IEEE Transactions on Knowledge and Data Engineering

to enhance online methods significantly.

In this paper, we propose a novel online adaptive asymmetric active (**OA3**) learning algorithm. By exploiting samples' second-order information, we develop a new asymmetric strategy, which considers both model optimization and label queries simultaneously. As a result, the proposed strategy addresses the class imbalance better and thus improves model performance. Moreover, we theoretically analyze the metric bounds of our proposed algorithm for the cases within budgets and over budgets, respectively.

Although enjoying the advantage of second-order information, our proposed algorithm may run slower than first-order methods, because the update of the correlation matrix is time-consuming. Therefore, it may be inappropriate for applications with quite high-dimensional datasets. To address this issue, we further propose two efficient variants of OA3 based on sketching techniques [20], [21], [22], [23].

We empirically evaluate the proposed methods on real-world datasets. Encouraging results confirm their effectiveness, efficiency and stability. We also examine the influences of algorithm parameters. Extensive results validate the algorithm characteristics.

The rest of this paper is organized as follows. We first present the problem formulation and the proposed algorithm in Section 2, followed by the theoretical analyses in Section 3. Next, we propose two efficient variants based on sketching techniques in Section 4. We empirically evaluate the proposed algorithms in Section 5, and conclude the paper in Section 6. Due to the page limitation, we put related work in Supplementary D.

2 Setup and Algorithm

In this section, we firstly introduce the problem formulation of the online active learning problem for budgeted imbalanced data. Next, we present the scheme of the proposed Online Adaptive Asymmetric Active (OA3) Learning algorithm. Lastly, relying on samples' second-order information, we propose a new asymmetric strategy, which consists of an asymmetric update rule and an asymmetric query rule.

2.1 Problem Formulation

Without loss of generality, we consider online binary classification under limited query budgets here. Streaming data comes in one by one $\{x_1, x_2, ..., x_T\}$, where $x_t \in \mathbb{R}^d$ is a d-dimensional sample at time t, and T is the total quantity of samples. Note that all samples are unlabeled, and there is a limited query budget B for obtaining class label $y \in \{-1, 1\}$. The main task is to learn a well-performed online linear classifier $w \in \mathbb{R}^d$ with only limited labeled data. Moreover, the prediction of the classifier is $\hat{y} = \mathrm{sign}(\mathrm{w}^\top \mathrm{x})$.

Primarily, we define some notations: $\mathcal{M} = \{t \mid y_t \neq \mathrm{sign}(w_t^\top x_t), \forall t \in [T]\}$ is the mistake index set, $\mathcal{M}_p = \{t \in \mathcal{M} \text{ and } y_t = +1\}$ is the positive set of mistake index and $\mathcal{M}_n = \{t \in \mathcal{M} \text{ and } y_t = -1\}$ is the negative one. In addition, we set $M = |\mathcal{M}|$, $M_p = |\mathcal{M}_p|$ and $M_n = |\mathcal{M}_n|$ to denote the number of total mistakes, positive mistakes and negative mistakes. Moreover, we denote the index sets of all positive samples and all negative samples by $\mathcal{I}_T^p = \{i \in [T] | y_i = +1\}$ and $\mathcal{I}_T^n = \{i \in [T] | y_i = -1\}$, where $T_p = |\mathcal{I}_T^p|$ and $T_n = |\mathcal{I}_T^n|$ denote the number of positive samples and negative samples. For convenience, we assume the positive class as the minority class, *i.e.*, $T_p \leq T_n$.

Traditional online algorithms often optimize *accuracy* or *mistake rate*, which treats samples from different class equally. These metrics, however, may be inappropriate for imbalanced data, since a trivial learner by simply classifying all data as negative can still achieve high accuracy. To address this issue, a more suitable metric is to measure the *sum* of weighted *sensitivity* and *specificity*, *i.e.*,

$$sum = \alpha_p \times \frac{T_p - M_p}{T_p} + \alpha_n \times \frac{T_n - M_n}{T_n}, \tag{1}$$

where $\alpha_p, \alpha_n \in [0, 1]$ are trade-off parameters between *sensitivity* and *specificity*, and $\alpha_p + \alpha_n = 1$. Note that when $\alpha_p = \alpha_n = 0.5$, the *sum* metric becomes the famous *balanced accuracy*. In general, the higher the *sum* value, the better the classification performance.

In addition, another metric is to measure the weighted misclassification *cost* suffered by the model, *i.e.*,

$$cost = c_p \times M_p + c_n \times M_n, \tag{2}$$

where $c_p, c_n \in [0, 1]$ denote the cost weights for positive and negative instances, and $c_p + c_n = 1$. The lower the *cost* value, the better the classification performance.

2.2 Algorithm Scheme

Inspired by the adaptive confidence weight technique [17], [25], we exploit samples' second order information. Assume that the online model satisfies a multivariate Gaussian distribution [17], *i.e.*, $w \sim \mathcal{N}(\mu, \Sigma)$, where μ is the mean vector and Σ is the covariance matrix of distribution. Without loss of generality, the mean value μ_i represents the model's knowledge of the weight for feature i, while $\Sigma_{i,i}$ encodes the confidence of feature i. Generally, the smaller $\Sigma_{i,i}$, the more confidence the model has in the mean weight value μ_i. The covariance term $\Sigma_{i,j}$ keeps the correlations between weights i and j. Given a definite Gaussian distribution, it is more practical to simply use the mean vector $\mu = \mathbb{E}[w]$ to make predictions, *i.e.*, $\hat{y} = \mathrm{sign}(\mu^\top x)$ [16], [17], [19], where we denote $p = \mu^\top x$ as the predictive margin.

Formally, when receiving a new sample x_t in the t-th round, the learner needs to make a prediction $\hat{y}_t$ and decide whether to query the true label y_t. If deciding to query, the learner will consume one unit budget, and then update the predictive vector μ_t based on the received painful loss from (x_t, y_t). Otherwise, the model will ignore x_t. Note that the above process is performed within the limited query budget B. Once the available budget goes down to zero, the learner will stop querying true labels and thus stop updating. We summarize the algorithm scheme in Algo. 1.

Considering the class-imbalanced issue, there are two main challenges when designing this active algorithm.

1) **How to update** the model in an asymmetric way to obtain a well-performed model, which is described in Subsection 2.3.

2) **How to query** the most informative samples asymmetrically, which is described in Subsection 2.4.

Algorithm 1 Online Adaptive Asymmetric Active (OA3) Learning algorithm.

Input budget B; learning rate η; regular parameter γ.
Initialization $\mu_1 = 0$, $\Sigma_1 = I$, $B_1 = 0$.

```
for t = 1 → T do
    Receive an example x_t ∈ R^d;
    Compute p_t = μ_t^⊤ x_t;
    Make the prediction ŷ_t = sign(p_t);
    Draw a variable Z_t = Query(p_t) ∈ {0, 1};
    if Z_t = 1 and B_t < B then
        Query the true label y_t ∈ {−1, +1};
        B_{t+1} = B_t + 1;
        μ_{t+1}, Σ_{t+1} = Update(μ_t, Σ_t; x_t, y_t).
    else
        B_{t+1} = B_t, μ_{t+1} = μ_t, Σ_{t+1} = Σ_t.
    end if
end for
```

2.3 Adaptive Asymmetric Update Rule

To solve the class imbalance issue, our objective is either to maximize *sum* metric in Eq. (1) or to minimize *cost* metric in Eq. (2). Both objectives are equivalent to minimizing the following objective [10]:

$$\sum_{y_t=+1} \rho \mathbb{I}_{(y_t(\mu^\top x_t)<0)} + \sum_{y_t=-1} \mathbb{I}_{(y_t(\mu^\top x_t)<0)}, \tag{3}$$

where $\rho = \frac{\alpha_p T_n}{\alpha_n T_p}$ for maximizing *sum* metric and $\rho = \frac{c_p}{c_n}$ for minimizing *cost* metric, while $\mathbb{I}_{(\cdot)}$ is the indicator function.

However, this objective is non-convex. To facilitate the optimization, we replace the indicator function with its convex variants, *i.e.*,

$$\ell_t(\mu) = \big(\rho \mathbb{I}_{(y=+1)} + \mathbb{I}_{(y=-1)}\big) \max\big\{0, 1 - y_t(\mu^\top x_t)\big\}. \tag{4}$$

At round t, when receiving a sample x_t and querying its label y_t, we can naturally exploit second-order information of data by minimizing the following objective [16], [17], *i.e.*,

$$D_{KL}(\mathcal{N}(\mu, \Sigma) || \mathcal{N}(\mu_t, \Sigma_t)) + \eta \ell_t(\mu) + \frac{1}{2\gamma} x_t^\top \Sigma x_t, \tag{5}$$

where η is the learning rate, γ is the regularized parameter and D_{KL} denotes the Kullback-divergence, *i.e.*,

$$\begin{aligned} &D_{KL}\big(\mathcal{N}(\mu, \Sigma) || \mathcal{N}(\mu_t, \Sigma_t)\big) \\ &= \frac{1}{2}\log\Big(\frac{\det \Sigma_t}{\det \Sigma}\Big) + \frac{1}{2}\mathrm{Tr}(\Sigma_t^{-1}\Sigma) + \frac{1}{2}||\mu_t - \mu||^2_{\Sigma_t^{-1}} - \frac{d}{2}. \end{aligned}$$

Specifically, the objective of Eq. (5) helps to reach the trade-off between distribution divergence (first term), loss function (second term) and model confidence (third term). In other words, the objective tends to make the least adjustment to minimize the painful loss and optimize the model confidence. However, this optimization does not have closed-form solution. To address this issue, we replace the loss $\ell(\mu)$ with its first order Taylor expansion $\ell(\mu_t) + g_t^\top(\mu - \mu_t)$, where $g_t = \partial \ell_t(\mu_t)$. We then obtain the final optimization objective by removing constant terms, *i.e.*,

$$f_t(\mu, \Sigma) = D_{KL}(\mathcal{N}(\mu, \Sigma) || \mathcal{N}(\mu_t, \Sigma_t)) + \eta g_t^\top \mu + \frac{1}{2\gamma} x_t^\top \Sigma x_t, \tag{6}$$

which is much easier to be solved. We solve this optimization problem in two steps, iteratively:

- Update the mean parameter:

$$\mu_{t+1} = \arg\min_{\mu} f_t(\mu, \Sigma);$$

- If $\ell_t(\mu_t) \neq 0$, update the covariance matrix:

$$\Sigma_{t+1} = \arg\min_{\Sigma} f_t(\mu, \Sigma).$$

For the first step, setting the derivative $\partial_\mu f_t(\mu_{t+1}, \Sigma)$ to zero will give:

$$\Sigma_t^{-1}(\mu_{t+1} - \mu_t) + \eta g_t = 0 \implies \mu_{t+1} = \mu_t - \eta \Sigma_t g_t.$$

Second, setting derivative $\partial_\Sigma f_t(\mu, \Sigma_{t+1})$ to zero gives:

$$-\Sigma_{t+1}^{-1} + \Sigma_t^{-1} + \frac{x_t x_t^\top}{\gamma} = 0 \implies \Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \frac{x_t x_t^\top}{\gamma}. \tag{7}$$

Based on the Woodbury identity [26], we have:

$$\Sigma_{t+1} = \Sigma_t - \frac{\Sigma_t x_t x_t^\top \Sigma_t}{\gamma + x_t^\top \Sigma_t x_t}. \tag{8}$$

Apparently, the update of the predictive vector relies on the confidence Σ. Thus, we update the mean vector μ_t based on the updated covariance matrix Σ_{t+1}, which will be more accurate and aggressive [16], [19], *i.e.*,

$$\mu_{t+1} = \mu_t - \eta \Sigma_{t+1} g_t. \tag{9}$$

We summarize the adaptive asymmetric update strategy in Algo. 2.

Algorithm 2 Adaptive Asymmetric Update Strategy: **Update**$(\mu_t, \Sigma_t; x_t, y_t)$.

Input $\rho = \frac{\alpha_p T_n}{\alpha_n T_p}$ for *sum* or $\rho = \frac{c_p}{c_n}$ for *cost*;

```
Receive a sample (x_t, y_t);
Compute the loss ℓ_t(μ_t), based on Equation (4);
if ℓ_t(μ_t) > 0 then
    Σ_{t+1} = Σ_t − (Σ_t x_t x_t^⊤ Σ_t) / (γ + x_t^⊤ Σ_t x_t);
    μ_{t+1} = μ_t − η Σ_{t+1} g_t, where g_t = ∂ℓ_t(μ_t).
else
    μ_{t+1} = μ_t, Σ_{t+1} = Σ_t.
end if
Return μ_{t+1}, Σ_{t+1}.
```

Time Complexity Analysis. Time complexities of updating for μ and Σ are both $\mathcal{O}(Td^2)$, so the overall time complexity of this update strategy is $\mathcal{O}(Td^2)$, where d is the dimension of data. Nevertheless, the update efficiency of OA3 is slower than first-order algorithms $\big(\mathcal{O}(Td)\big)$, especially when handling high-dimensional datasets. To promote the efficiency, we propose a **diagonal version** of the update strategy (the pseudo-code is put in Supplementary B.1), which accelerates the efficiency to $\mathcal{O}(Td)$. Specifically, in this variant, only the diagonal entries of Σ are maintained and updated in each round.

Remark. *We employ the adaptive asymmetric update rule for OA3 to pursue high performance with faster convergence. Nevertheless, it is not the only choice. Other classic techniques can also be used,* e.g., *online gradient descent [10] and online margin-based strategies [9], [11].*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2019.2955078, IEEE Transactions on Knowledge and Data Engineering

2.4 Asymmetric Query Strategy

In a pioneering study [14], one classic sampling method was proposed to query labels based on a Bernoulli random variable $Z_t \in \{0,1\}$. That is, querying the label for a given instance only when $Z_t = 1$. To be specific, the probability of sampling Z_t depends on the absolute value of the predictive margin $|p_t|$, *i.e.*,

$$\Pr(Z_t = 1) = \frac{\delta}{\delta + |p_t|},$$

where $\delta > 0$ is the query bias. In this equation, when the absolute predictive margin $|p_t|$ is low, the probability to query the label of sample x_t is relatively high. This is because when the sample's prediction is close to the classification hyperplane (small $|p_t|$), the sample is difficult to be classified and hence is more valuable to be queried (high $\Pr(Z_t = 1)$).

However, this query rule ignores the imbalance of data and treats predictions of two imbalanced classes equally. To address this limitation, inspired by recent work [4], we employ an asymmetric strategy to query labels, *i.e.*,

$$\Pr(Z_t = 1) = \begin{cases} \frac{\delta_+}{\delta_+ + |p_t|}, & \text{if } p_t \geq 0; \\ \frac{\delta_-}{\delta_- + |p_t|}, & \text{if } p_t < 0; \end{cases}$$

where $\delta_+ > 0$ and $\delta_- > 0$ denote query biases for positive and negative predictions, respectively.

However, this asymmetric query strategy heavily depends on the absolute value of margin p_t, which is directly calculated by the model μ_t. As a result, the query decisions may be inaccurate when μ_t is not precise enough [31].

To address this issue, we use samples' second order information to enhance the query strategy and improve the robustness of the query judgement. We first define the variance of a model on the sample x_t as $v_t = x_t^\top \Sigma_t x_t$. It represents the familiarity of the model with the current sample through previous experience. Based on v_t, we then define the query confidence:

$$c_t = -\frac{1}{2}\frac{\eta \rho_{max}}{\frac{1}{v_t} + \frac{1}{\gamma}}, \tag{10}$$

where $\rho_{max} = \max\{1, \rho\}$. We highlight that this equation is helpful for the theoretical analysis. Moreover, the confidence c_t directly depends on the variance v_t. Based on this equation, when the model has been well trained on some instances similar to the current sample x_t (*i.e.*, low variance v_t), the model would be confident of this sample (*i.e.*, large confidence c_t).

Based on the predictive margin and the confidence, we obtain the final query parameter:

$$q_t = |p_t| + c_t. \tag{11}$$

Moreover, both the learning rate η and regularized parameter γ in Eq. (10) can be understood as trade-off factors in Eq. (11).

Relying on above analyses, we propose an improved asymmetric query strategy:

$$\Pr(Z_t = 1) = \begin{cases} \frac{\delta_+}{\delta_+ + q_t}, & \text{if } p_t \geq 0; \\ \frac{\delta_-}{\delta_- + q_t}, & \text{if } p_t < 0. \end{cases}$$

To be specific, when $q_t{>}0$, the query decision of the model is very confident, so we directly draw a Bernoulli variable based on this equation. If $q_t{\leq}0$, the query decision is unconfident of the current sample, so we decide to query the true label whatever the value of p_t, *i.e.*, obtaining $Z_t{=}1$ by setting $q_t{=}0$ (see the above equation).

We summarize the proposed asymmetric query strategy in Algo. 3.

Algorithm 3 Asymmetric Query Strategy: **Query**(p_t).

Input $\rho_{max} = \max\{1, \rho\}$; query bias ($\delta_+$,$\delta_-$) for positive and negative predictions.

Compute the variance $v_t = x_t^\top \Sigma_t x_t$;
Compute the query parameter $q_t = |p_t| - \frac{1}{2}\frac{\eta \rho_{max}}{\frac{1}{v_t} + \frac{1}{\gamma}}$;
if $q_t \leq 0$ **then**
    Set $q_t = 0$;
end if
if $p_t \geq 0$ **then**
    $p_t^+ = \frac{\delta_+}{\delta_+ + q_t}$;
    Draw a Bernoulli variable $Z_t{\in}\{0,1\}$ with p_t^+.
else
    $p_t^- = \frac{\delta_-}{\delta_- + q_t}$;
    Draw a Bernoulli variable $Z_t{\in}\{0,1\}$ with p_t^-.
end if
Return Z_t.

We can obtain the expected number of queried samples without budget limitations as follows.

Proposition 1. Based on the proposed asymmetric query strategy, the expected number of requested samples without a budget is:

$$\sum \mathbb{I}_{(q_t \leq 0)} + \sum_{\substack{q_t > 0 \\ p_t \geq 0}} \frac{\delta_+}{\delta_+ + q_t} + \sum_{\substack{q_t > 0 \\ p_t < 0}} \frac{\delta_-}{\delta_- + q_t}.$$

3 Theoretical Analysis

We next analyze the proposed algorithm in terms of its mistake bound and two cost-sensitive metric bounds, for the cases within budgets and over budgets, respectively. Before that, we first show a lemma, which facilitates the analysis within budgets. Due to the page limitation, all proofs are put in Supplementary A.

For convenience, we introduce the following notations:

$$M_t = \mathbb{I}_{(\hat{y}_t \neq y_t)},\ \rho = \frac{\alpha_p T_n}{\alpha_n T_p} \text{ or } \frac{c_p}{c_n},$$

$$\rho_t{=}\rho\mathbb{I}_{(y_t=+1)}{+}\mathbb{I}_{(y_t=-1)}, \rho_{max}{=}\max\{1,\rho\}, \rho_{min}{=}\min\{1,\rho\}.$$

Lemma 1. *Let $(x_1, y_1), ..., (x_T, y_T)$ be a sequence of input samples, where $x_t \in \mathbb{R}^d$ and $y_t \in \{-1,+1\}$ for all t. Let T_B be the round that runs out of the budgets,* i.e., *$B_{T_B+1} = B$. For any $\mu \in \mathbb{R}^d$ and any $\delta > 0$, OA3 algorithm satisfies:*

$$\sum_{t=1}^{T_B} M_t Z_t(\delta{+}q_t) \leq \frac{\delta}{\rho_{min}} \sum_{t=1}^{T_B} \ell_t(\mu) + \frac{1}{\eta \rho_{min}} Tr(\Sigma_{T_B+1}^{-1}) \times \left[M(\mu) + (1-\delta)^2 ||\mu||^2\right],$$

where $M(\mu) = \max_t ||\mu_t {-} \mu||^2$.

Based on Lemma 1, we obtain the following three theorems for the case **within budgets**.

Theorem 1. *Let $(x_1, y_1), ..., (x_T, y_T)$ be a sequence of input samples, where $x_t \in \mathbb{R}^d$ and $y_t \in \{-1, +1\}$ for all t. Let T_B be the round that runs out of the budgets,* i.e., *$B_{T_B+1} = B$. For any $\mu \in \mathbb{R}^d$, the expected mistake number of OA3 within budgets is bounded by:*

$$\mathbb{E}\Big[\sum_{t=1}^{T_B} M_t\Big] = \mathbb{E}\Big[\sum_{\substack{t=1\\ y_t=+1}}^{T_B} M_t + \sum_{\substack{t=1\\ y_t=-1}}^{T_B} M_t\Big]$$
$$\leq \frac{1}{\rho_{min}}\Big[\sum_{t=1}^{T_B} \ell_t(\mu) + \frac{1}{\eta} D(\mu) Tr(\Sigma_{T_B+1}^{-1})\Big],$$

where $D(\mu) = \max\Big\{\frac{M(\mu)+(1-\delta_+)^2||\mu||^2}{\delta_+}, \frac{M(\mu)+(1-\delta_-)^2||\mu||^2}{\delta_-}\Big\}$.

This mistake bound helps to analyze the weighted sum performance under limited budgets.

Theorem 2. *Under the same condition in Theorem 1, by setting $\rho = \frac{\alpha_p T_n}{\alpha_n T_p}$, the proposed OA3 within budgets satisfies for any $\mu \in \mathbb{R}^d$:*

$$\mathbb{E}\Big[sum\Big] \geq 1 - \frac{\alpha_n \rho_{max}}{T_n \rho_{min}}\Big[\sum_{t=1}^{T_B} \ell_t(\mu) + \frac{1}{\eta} D(\mu) Tr(\Sigma_{T_B+1}^{-1})\Big].$$

Remark. *By setting $\alpha_p = \alpha_n = 0.5$, we can easily obtain the bound of the balanced accuracy.*

Note that α_n cannot be set to zero, because $\rho = \frac{\alpha_p T_n}{\alpha_n T_p}$. One restriction is that we could not acquire $\frac{T_n}{T_p}$ in advance in real-world tasks. To overcome this limitation, we can choose $cost$ metric as an alternative, where $\rho = \frac{c_p}{c_n}$. In this sense, engineers need not worry $\frac{T_n}{T_p}$ any more. Next, we bound the cumulative $cost$ performance under limited budgets.

Theorem 3. *Under the same condition in Theorem 1, by setting $\rho = \frac{c_p}{c_n}$, the proposed OA3 within budgets satisfies for any $\mu \in \mathbb{R}^d$:*

$$\mathbb{E}\Big[cost\Big] \leq \frac{c_n \rho_{max}}{\rho_{min}}\Big[\sum_{t=1}^{T_B} \ell_t(\mu) + \frac{1}{\eta} D(\mu) Tr(\Sigma_{T_B+1}^{-1})\Big].$$

Note that c_n cannot be set to zero, since $\rho = \frac{c_p}{c_n}$.

By now, we have analyzed OA3 algorithm within budgets. Next, we analyze OA3 for the case **over budgets**.

Theorem 4. *Let $(x_1, y_1), ..., (x_T, y_T)$ be a sample stream, where $x_t \in \mathbb{R}^d$ and $y_t \in \{-1, +1\}$. Let T_B be the round that uses up the budgets,* i.e., *$B_{T_B+1} = B$. For any $\mu \in \mathbb{R}^d$, the expected mistakes of OA3 over budgets is bounded by:*

$$\mathbb{E}\Big[\sum_{T_B+1}^{T} M_t\Big] \leq \sum_{T_B+1}^{T}\Big[\frac{\ell_t(\mu)}{\rho_{min}} + y_t x_t^\top \mu_{T_B+1}\Big],$$

where μ_{T_B+1} is the predictive vector of model, trained by all the previous queried samples.

Now, we bound the weighted sum and misclassification $cost$ after running out of budgets.

Theorem 5. *Under the same condition in Theorem 4, by setting $\rho = \frac{\alpha_p T_n}{\alpha_n T_p}$, the sum performance of OA3 over budgets satisfies for any $\mu \in \mathbb{R}^d$:*

$$\mathbb{E}\Big[sum\Big] \geq 1 - \frac{\alpha_n \rho_{max}}{T_n} \sum_{T_B+1}^{T}\Big[\frac{\ell_t(\mu)}{\rho_{min}} + y_t x_t^\top \mu_{T_B+1}\Big].$$

Theorem 6. *Under the same condition in Theorem 4, by setting $\rho = \frac{c_p}{c_n}$, the misclassification cost of OA3 over budgets satisfies for any $\mu \in \mathbb{R}^d$:*

$$\mathbb{E}\Big[cost\Big] \leq c_n \rho_{max} \sum_{T_B+1}^{T}\Big[\frac{\ell_t(\mu)}{\rho_{min}} + y_t x_t^\top \mu_{T_B+1}\Big].$$

4 Enhanced Algorithm With Sketching

As mentioned above, OA3 requires $\mathcal{O}(Td^2)$ time complexity. The diagonal version accelerates the time complexity to $\mathcal{O}(Td)$. However, it cannot enjoy the correlation information between different dimensions of samples. When instances have low *effective rank*, the regret bound of OA3$_{diag}$ may be much worse than its full-matrix version, because it lacks enough dependence on data dimensionality [22]. Unfortunately, many real-world high-dimensional datasets have such low-rank settings with abundant correlations among features. For these datasets, it is more appropriate to consider the complete feature correlations (*i.e.*, adopting its full-matrix version) and also the efficiency issue.

To better balance performance and efficiency, we propose two efficient variants of our OA3 algorithm, which use the sketch method to approximate the inverse of the covariance matrix. Specifically, we first propose Sketched Online Adaptive Asymmetric Active (SOA3) Learning algorithm in Subsection 4.1 and then present its sparse version (SSAO3) in Subsection 4.2.

4.1 Sketched Algorithm

By exploiting the Oja's sketch method, we propose a SAO3 algorithm [20], [27] to accelerate our algorithm when facing high-dimensional datasets.

The Oja's sketch [28] is a method to compute the dominant eigenvalues and corresponding eigenvectors of a $n \times n$ matrix A. In Oja's method, matrix A has the following property: A itself is unknown but there is an available sequence $A_k, k=1, 2, ...$ with $E\{A_k\} = A$ for all k. In our proposed OA3 algorithm, the computation of inverse covariance matrix Σ^{-1} in Eq. 7 can be transformed into this formula. Thus, the Oja's sketch method can be introduced to our OA3 algorithm.

In SOA3, we exploit Oja's sketch to search m carefully selective directions and use them to approximate our second-order inverse covariance matrix. Here, m is a small constant and called as the sketch size. According to Eqs. (8-9), we know the updating rule of the model weight μ:

$$\mu_{t+1} = \mu_t - \eta \Sigma_{t+1} g_t,$$

and the incremental formula of the covariance matrix:

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \frac{x_t x_t^\top}{\gamma},$$

which can be expressed in another way:

$$\Sigma_{t+1}^{-1} = I_d + \sum_{i=1}^{t} \frac{x_i x_i^\top}{\gamma}, \tag{12}$$

where d is the dimensionality of instance. Let $X \in \mathbb{R}^{t\times d}$ be a matrix, whose t-th row is $\hat{x}_t^\top$, where we define $\hat{x}_t = \frac{x_t}{\sqrt{\gamma}}$ as the *to-sketch vector*. Hence, Eq. (12) can be written as:

$$\Sigma_{t+1}^{-1} = I_d + X_t^\top X_t.$$

The idea of sketching is to maintain a sketch matrix $S_t \in \mathbb{R}^{m\times d}$, where $m \ll d$. When m is chosen so that $S_t^\top S_t$ can approximate $X_t^\top X_t$ well, the Eq. (12) can be redefined as:

$$\Sigma_{t+1}^{-1} = I_d + S_t^\top S_t.$$

By Woodbury identity [26], we have:

$$\Sigma_{t+1} = I_d - S_t^\top H_t S_t, \tag{13}$$

where $H_t = (I_m + S_t S_t^\top)^{-1} \in \mathbb{R}^{m\times m}$. We rewrite the updating rule of μ:

$$\mu_{t+1} = \mu_t - \eta(g_t - S_t^\top H_t S_t g_t). \tag{14}$$

We summarize SOA3 in Algo. 4.

Algorithm 4 Sketched Online Adaptive Asymmetric Active (SOA3) Learning algorithm.

Input budget B; learning rate η; regularized parameter γ; sketch size m; bias $\rho = \frac{\alpha_p * T_n}{\alpha_n * T_p}$ for sum and $\rho = \frac{c_p}{c_n}$ for $cost$.
Initialization $\mu_1=0$, $B_1=0$.
Initialization $(S_0, H_0) \leftarrow$ **SketchInit**(m); (See Algo. 5)
for $t = 1 \to T$ **do**
  Receive sample x_t;
  Compute $p_t = \mu_t^\top x_t$;
  Make the prediction $\hat{y}_t = sign(p_t)$;
  Draw Z_t=**SketchQuery**$(p_t) \in \{0, 1\}$; (See Algo. 6)
  if $Z_t = 1$ and $B_t < B$ **then**
    Query the true label $y_t \in \{-1, +1\}$, $B_{t+1}=B_t+1$;
    Compute the loss $\ell_t(\mu_t)$, based on Equation (4);
    Compute the $t\text{-}sketch\ vector$ $\hat{x}_t = \frac{x_t}{\sqrt{\gamma}}$;
    $(S_t, H_t) \leftarrow$ **SketchUpdate**$(\hat{x})$; (See Algo. 5)
    if $\ell_t(\mu_t) > 0$ **then**
      $\mu_{t+1} = \mu_t - \eta(g_t - S_t^\top H_t S_t g_t)$, where $g_t = \partial_\mu \ell_t(\mu_t)$;
    else
      $\mu_{t+1} = \mu_t$;
    end if
  else
    $\mu_{t+1}=\mu_t$, $B_{t+1}=B_t$, $S_t=S_{t-1}$, $H_t=H_{t-1}$.
  end if
end for

We next discuss how to maintain matrices S_t and H_t efficiently via sketching technique. Specifically, with *to-sketch vector* x_t as input, the eigenvalues and eigenvectors of sequential data are computed by online gradient descent.

At round t, let the diagonal matrix $\Lambda_t \in \mathbb{R}^{m\times m}$ contain m estimated eigenvalues and let $V_t \in \mathbb{R}^{m\times d}$ denote the corresponding eigenvectors. The update rules of Λ_t and V_t using Oja's algorithm are defined as:

$$\Lambda_t = (I_m - \Gamma_t)\Lambda_{t-1} + \Gamma_t diag\{V_{t-1}\hat{x}_t\}^2, \tag{15}$$

$$V_t \xleftarrow{orth} V_{t-1} + \Gamma_t V_{t-1}\hat{x}_t \hat{x}_t^\top, \tag{16}$$

where $\Gamma_t \in \mathbb{R}^{m\times m}$ is a diagonal matrix whose diagonal elements are learning rates. In this paper, we set $\Gamma_t = \frac{1}{t} I_m$. The "$\xleftarrow{orth}$" operator represents an orthonormalizing step[1]. Hence, the sketch matrices can be obtained by:

$$S_t = (t\Lambda)^{\frac{1}{2}} V_t, \tag{17}$$

$$H_t = diag\{\frac{1}{1 + t\Lambda_{1,1}}, ..., \frac{1}{1 + t\Lambda_{m,m}}\}.$$

The rows of V_t are always orthonormal and thus H_t is an efficiently maintainable diagonal matrix. We summarize the Oja's sketching technique in Algo. 5.

Algorithm 5 Oja's Sketch for SOA3

Input m, $\hat{x}$ and stepsize matrix Γ_t.
Internal State t, Λ, V and H.
SketchInit(m)
  Set $t = 0, S = 0_{m\times d}, H = I_m, \Lambda = 0_{m\times m}$ and V to any $m \times d$ matrix with orthonormal rows;
  Return (S, H).

SketchUpdate$(\hat{x})$
  Update $t \leftarrow t + 1$;
  Update $\Lambda = (I_m - \Gamma_t)\Lambda + \Gamma_t diag\{V\hat{x}\}^2$;
  Update $V \xleftarrow{orth} V + \Gamma_t V \hat{x}\hat{x}^\top$;
  Set $S = (t\Lambda)^{\frac{1}{2}} V$;
  Set $H = diag\{\frac{1}{1+t\Lambda_{1,1}}, ..., \frac{1}{1+t\Lambda_{m,m}}\}$;
  Return (S, H).

Remark. The time complexity of this sketched updating strategy is $O(m^2 d)$ per round because of the orthonormalizing operation. One can update the sketch every m rounds to improve time complexity to $O(md)$ [29].

Last, we discuss the asymmetric query strategy in SOA3. In Section 2.4, we compute variance $v_t = x_t^\top \Sigma_t x_t$ to enhance the query strategy. For the same purpose, in SOA3, we compute variance based on Eq. (13):

$$v_t = x_t^\top (I_d - S_{t-1}^\top H_{t-1} S_{t-1}) x_t. \tag{18}$$

Based on Eq. (18) and Algo. 3, we summarize the sketched query strategy in Algo. 6.

4.2 Sparse Sketched Algorithm

In many real-world applications, data is sparse that $||x_t||_0 \leq s$ for all t, and $s \ll d$ is a small constant. For most first-order online methods, computational complexities depend on s rather than d. However, SOA3 cannot enjoy this sparsity, because the sketch matrix S_t will become dense quickly due to the orthonormalizing updating of V_t. To address this, we propose a sparse variant of SOA3 to achieve a purely sparsity-dependent time cost, named Sparse Sketched Online Adaptive Asymmetric (SSOA3) Learning.

The key of SSOA3 is to decompose the estimated eigenvectors V_t and predictive vector μ_t so that they can keep sparse. Specifically, there are two main modifications: (1) the eigenvectors V_t are modified as $V_t = F_t U_t$, where $F_t \in \mathbb{R}^{m\times m}$ is an orthonormalizing matrix so that $F_t U_t$ is orthonormal, and $U_t \in \mathbb{R}^{m\times d}$ is a sparsely updatable

1. For sake of simplicity, $V_t + \Gamma_{t+1} V_t \hat{x}_t \hat{x}_t^\top$ is assumed as full rank with rows all the way, so that the $\xleftarrow{orth}$ operation always keeps the same dimensionality of V_t.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2019.2955078, IEEE Transactions on Knowledge and Data Engineering

Algorithm 6 Sketched Asymmetric Query Strategy: **SketchQuery**(p_t).

Input $\rho_{max} = \max\{1, \rho\}$; query bias (δ_+, δ_-) for positive and negative predictions.

```
Compute the variance v_t = x_t^⊤ (I_d − S_{t−1}^⊤ H_{t−1} S_{t−1}) x_t;
Compute the query parameter q_t = |p_t| − (1/2) · ηρ_max / (1/v_t + 1/γ);
if q_t ≤ 0 then
    Set q_t = 0;
end if
if p_t ≥ 0 then
    p_t^+ = δ_+ / (δ_+ + q_t);
    Draw a Bernoulli variable Z_t ∈ {0, 1} with p_t^+.
else
    p_t^- = δ_- / (δ_- + q_t);
    Draw a Bernoulli variable Z_t ∈ {0, 1} with p_t^-.
end if
Return Z_t.
```

direction. (2) the weight μ_t are split as $\bar{\mu}_t + U_{t-1}^\top b_t$, where $b_t \in \mathbb{R}^m$ maintains the weight on the subspace captured by V_{t-1} (the same as U_{t-1}), and $\bar{\mu}_t$ captures the weight on the complementary subspace.

Next, we describe the way to update $\bar{\mu}_t$ and b_t sparsely in detail. Firstly, according to Eq. (14) and $S_t = (t\Lambda)^{\frac{1}{2}} V_t = (t\Lambda)^{\frac{1}{2}} F_t U_t$, we have:

$$
\begin{aligned}
\mu_{t+1} =& \mu_t - \eta(I_d - S_t^\top H_t S_t) g_t \\
=& \bar{\mu}_t + U_{t-1}^\top b_t - \eta g_t + \eta U_t^\top F_t^\top (t\Lambda H_t) F_t U_t g_t \\
=& \underbrace{[\bar{\mu}_t - \eta g_t - (U_t - U_{t-1})^\top b_t]}_{\bar{\mu}_{t+1}} + U_t^\top \underbrace{[b_t + \eta F_t^\top (t\Lambda H_t) F_t U_t g_t]}_{b_{t+1}}.
\end{aligned}
$$

Based on this equation, we define the updating rule of $\bar{\mu}_t$:

$$
\bar{\mu}_{t+1} = \bar{\mu}_t - \eta g_t - (U_t - U_{t-1})^\top b_t,
$$

and define the updating rule of b_t:

$$
b_{t+1} = b_t + \eta F_t^\top (t\Lambda_t H_t) F_t U_t g_t.
$$

Based on the above, we summarize SSOA3 in Algo. 7, where the pseudo-code of sparse Oja's algorithms is provided in Supplementary B.2 due to the page limitation.

Next, we discuss how to update Λ_t, U_t and F_t. First, we rewrite Eq. (15) based on $V_t = F_t U_t$:

$$
\Lambda_t = (I_m - \Gamma_t)\Lambda_{t-1} + \Gamma_t diag\{F_{t-1} U_{t-1} \hat{x}_t\}^2.
$$

and rewrite Eq. (16) in the same way:

$$
\begin{aligned}
F_t U_t \xleftarrow{orth}& F_{t-1} U_{t-1} + \Gamma_t F_{t-1} U_{t-1} \hat{x}_t \hat{x}_t^\top, \\
=& F_{t-1}(U_{t-1} + F_{t-1}^{-1} \Gamma_t F_{t-1} U_{t-1} \hat{x}_t \hat{x}_t^\top),
\end{aligned}
$$

where $U_t = U_{t-1} + \delta_t \hat{x}_t^\top$ and $\delta_t = F_{t-1}^{-1} \Gamma_t F_{t-1} U_{t-1} \hat{x}_t$. Note that $U_t - U_{t-1}$ is a sparse rank-one matrix, which makes the update of $\bar{\mu}_t$ efficient.

Since the update of F_t is to enforce $F_t U_t$ orthonormal, we apply the Gram-Schmidt algorithm to F_{t-1} in a Banach space, where the inner product is defined as $\langle a, b \rangle = a^\top K_t b$ and $K_t = U_t U_t^\top$ is the Gram matrix (see Supplementary B.2). Consequently, we can update K_t efficiently based on the update of U_t:

Algorithm 7 Sparse Sketched Online Adaptive Asymmetric Active (SSOA3) Learning Algorithm.

Input budget B; learning rate η; regularized parameter γ; sketch size m; bias $\rho = \frac{\alpha_p * T_n}{\alpha_n * T_p}$ for "sum", $\rho = \frac{c_p}{c_n}$ for "cost".

Initialization $\bar{\mu}_1 = 0_{d\times 1}$, $b_1 = 0_{m\times 1}$, $B_1 = 0$;

Initialization $(\Lambda_0, F_0, U_0, H_0) \leftarrow$ **SparseSketchInit**(m);

```
for t = 1 → T do
    Receive sample x_t;
    Compute p_t = μ_t^⊤ x_t;
    Make the prediction ŷ_t = sign(p_t);
    Draw a variable Z_t = SparseSketchQuery(p_t) ∈ {0, 1};
    if Z_t = 1 and B_t < B then
        Query the true label y_t ∈ {−1, +1}, B_{t+1} = B_t + 1;
        Compute the loss ℓ_t(μ_t), based on Equation (4);
        Compute the t-sketch vector x̂_t = x_t / √γ;
        (Λ_t, F_t, U_t, H_t, δ_t) ← SparseSketchUpdate(x̂);
        if ℓ_t(μ_t) > 0 then
            μ̄_{t+1} = μ̄_t − η g_t − x̂_t δ_t^⊤ b_t, where g_t = ∂ℓ_t(μ_t);
            b_{t+1} = b_t + η F_t^⊤ (tΛ_t H_t) F_t U_t g_t;
            μ_{t+1} = μ̄_{t+1} + U_t^⊤ b_{t+1};
        else
            μ_{t+1} = μ_t,  μ̄_{t+1} = μ̄_t,  b_{t+1} = b_t;
        end if
    else
        μ_{t+1} = μ_t,  μ̄_{t+1} = μ̄_t,  b_{t+1} = b_t,  B_{t+1} = B_t;
        (Λ_t, F_t, U_t, H_t, δ_t) = (Λ_{t−1}, F_{t−1}, U_{t−1}, H_{t−1}, δ_{t−1}).
    end if
end for
```

$$
\begin{aligned}
K_t =& U_t U_t^\top, \\
=& (U_{t-1} + \delta_t \hat{x}_t^\top)(U_{t-1} + \delta_t \hat{x}_t^\top)^\top, \\
=& K_{t-1} + U_{t-1} \hat{x}_t \delta_t^\top + \delta_t \hat{x}_t^\top U_{t-1}^\top + \delta_t \hat{x}_t^\top \hat{x}_t \delta_t^\top.
\end{aligned}
$$

We summarize the Sparse Oja's algorithm for OA3 in Supplementary B.2.

Remark. Note that both the updates of $\bar{\mu}_t$ and b_t require $O(m^2 + ms)$ time complexity. The most time-consuming step is the update of F_t, which needs $O(m^3)$. Furthermore, the prediction $\mu_t^\top x_t = \bar{\mu}_t^\top x_t + b_t^\top U_{t-1} x_t$ can be computed in $O(ms)$ time. So, the overall time complexity of the sparse sketched update rule is $O(m^3 + ms)$.

Like SOA3, SSOA3 also computes the variance v_t in an approximate way. Based on the decomposition of estimated eigenvectors $V_t = F_t U_t$ and Eq. (18), we have:

$$
v_t = x_t^\top (I_d - U_{t-1}^\top F_{t-1}^\top (t-1)\Lambda_{t-1} H_{t-1} F_{t-1} U_{t-1}) x_t.
$$

Based on this equation and Algo. 3, we summarize the sparse sketched asymmetric query strategy in Supplementary B.2.

5 Experiments

In this section, we evaluate the performance and characteristics of the proposed algorithms[2].

2. The source code will be released after acceptance.

5.1 Experimental Testbed and Setup

We compare **OA3** and its variants (**OA3**$_{diag}$, **SOA3**, **SSOA3**) with several state-of-the-art online active learning methods: (1) Online Passive-aggressive Active Algorithm (**PAA**) [30]; (2) Online Asymmetric Active Algorithm (**OAAL**) [4]; (3) Cost-Sensitive Online Active Algorithm (**CSOAL**) [5]; (4) Second-order Online Active Algorithm (**SOAL**) [31] and its cost-sensitive variant (**SOAL-CS**) [31].

All algorithms are evaluated on four benchmark datasets. The statistics are summarized in Table 1. Specifically, the first three datasets are obtained from LIBSVM[3] and the fourth dataset is obtained from KDD Cup 2008[4].

For data preprocessing, all samples are normalized by $x_t \leftarrow \frac{x_t}{\|x_t\|_2}$, which is extensively used in online learning, since samples are obtained sequentially. When budgets are run out, both the query and update of the corresponding method will stop.

For fair comparisons, all algorithms use the same experimental settings. We set $\alpha_p = \alpha_n = 0.5$ for sum, and $c_p = 0.9$ and $c_n = 0.1$ for $cost$. The value of ρ is set to $\frac{\alpha_p * T_n}{\alpha_n * T_p}$ for sum and $\frac{c_p}{c_n}$ for $cost$. In addition, query biases $(\delta, \delta_+, \delta_-)$ and learning rates η for all algorithms are selected from $[10^{-5}, 10^{-4}, ..., 10^4, 10^5]$ using cross validations. By default, the regularization parameter γ is set to 1 for all second order algorithms (*i.e.*, SOAL and OA3 based algorithms). For our sketched algorithms, we focus on the case that the sketch size m is fixed as 5, although our methods can be easily generalized by setting different sketch size like [20], while other implementation details are similar to [20].

On each dataset, experiments are conducted over 20 random permutations of data. Results are averaged over these 20 runs and 4 metrics are employed: *sensitivity*, *specificity*, weighted *sum* of sensitivity and specificity, and weighted *cost* of misclassification. All algorithms are implemented in MATLAB on a 3.40GHz Windows machine.

TABLE 1: Datasets for Evaluation of OA3 Algorithms

Dataset	#Examples	#Features	#Pos:#Neg
protein	17766	357	1:1.7
Sensorless	58509	48	1:10
w8a	64700	300	1:32.5
KDDCUP08	102294	117	1:163.19

5.2 Evaluation on Fixed Query Budgets

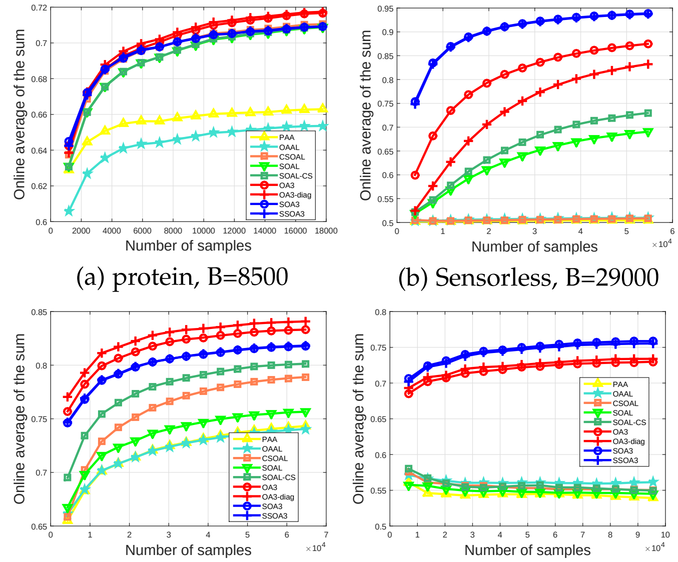
We first evaluate all algorithms under fixed budgets. Figs. 1 and 2 show the development of *sum* and *cost* performance, respectively and Table 2 summarizes more details.

First, OAAL (asymmetric query) and CSOAL (asymmetric update) outperform PAA (symmetric rule) in most cases. This comparison shows the importance of asymmetric strategies for imbalanced data in online active learning.

Second, as expected, all second order based algorithms (SOAL and OA3) outperform the first-order algorithms (PAA, OAAL and CSOAL) in most cases, which confirms the effectiveness of second-order information.

Third, the proposed OA3 algorithms outperform all baselines with smaller standard deviations, which demonstrates the effectiveness and stability of our methods.

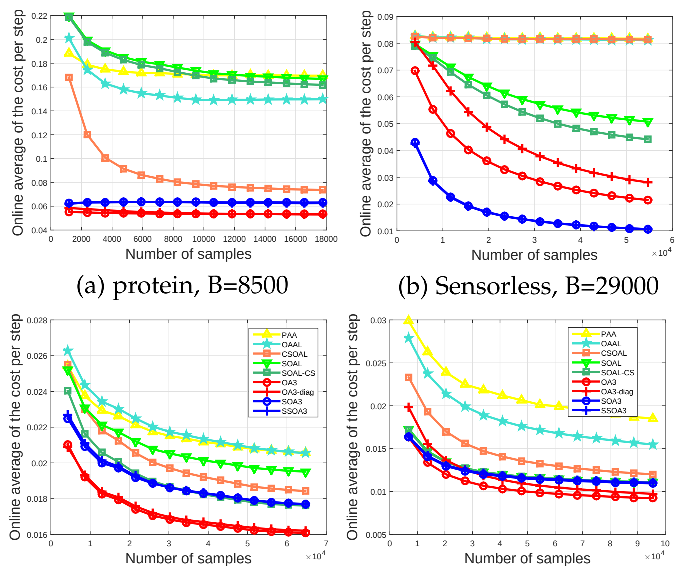
3. https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/.
4. http://www.kdd.org/kdd-cup/view/kdd-cup-2008/Data.



(a) protein, B=8500 (b) Sensorless, B=29000

(c) w8a, B=32000 (d) KDDCUP08, B=50000

Fig. 1: Evaluation of sum with fixed budget.



(a) protein, B=8500 (b) Sensorless, B=29000

(c) w8a, B=32000 (d) KDDCUP08, B=50000

Fig. 2: Evaluation of cost with fixed budget.

According to comparisons in terms of *sensitivity* and *specificity*, our proposed algorithms achieve the best *sensitivity* on all datasets and produce fairly good *specificity* on most datasets. This indicates that our algorithms pay more attention to minority samples, which are usually more important in practical tasks.

Last, we compare the efficiency of the proposed methods. From Table 2, OA3$_{diag}$, SOA3 and SSOA3 are much efficient than the original OA3 with quite slight performance degradation. Specifically, when datasets are quite high-dimensional, SOA3 and SSOA3 are further faster than OA3$_{diag}$. These observations imply that SOA3 and SSOA3 are better choices for balancing performance and efficiency.

5.3 Evaluation on Varying Query Budgets

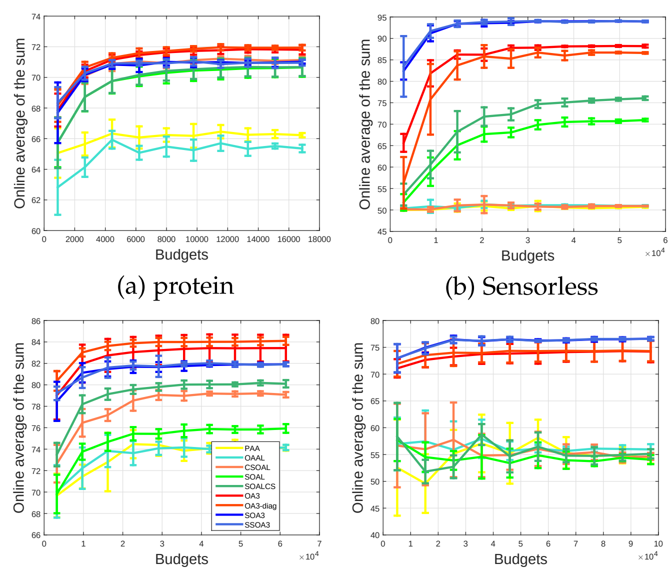
In this subsection, we compare the performance of all algorithms with varying query budgets. Figs. 3 and 4 show the results in terms of *sum* and *cost* respectively.

TABLE 2: Evaluation of the OA3 algorithms

Algorithm	"sum" on protein				"$cost$" on protein			
	Sum(%)	Sensitivity(%)	Specificity (%)	Time(s)	Cost	Sensitivity(%)	Specificity (%)	Time(s)
PAA	66.263 ± 0.553	61.455 ± 4.455	71.072 ± 3.732	0.274	3092.285 ± 248.215	61.882 ± 3.777	70.754 ± 3.357	0.274
OAAL	65.930 ± 0.574	68.953 ± 3.442	62.907 ± 3.664	0.275	2657.710 ± 210.120	67.306 ± 3.142	74.343 ± 2.305	0.277
CSOAL	71.073 ± 0.304	69.565 ± 3.144	72.582 ± 2.714	0.258	1309.025 ± 75.994	89.110 ± 1.285	47.167 ± 2.204	0.263
SOAL	70.863 ± 0.734	62.576 ± 5.437	**79.150 ± 4.177**	12.411	2959.845 ± 356.434	62.586 ± 5.356	**79.163 ± 4.104**	13.095
SOAL-CS	70.920 ± 0.734	63.115 ± 5.490	78.725 ± 4.268	12.616	2876.015 ± 370.944	63.853 ± 5.600	78.152± 4.478	13.736
OA3	**71.673 ± 0.348**	**72.608 ± 4.700**	70.738 ± 4.838	12.511	949.855 ± 5.615	**99.691 ± 0.099**	3.106 ± 0.963	8.870
OA3$_{diag}$	71.531 ± 0.334	72.114 ± 1.270	70.949 ± 1.393	7.008	**938.795 ± 9.116**	99.144 ± 0.188	8.480 ± 1.864	5.318
SOA3	71.028 ± 0.247	70.658 ± 2.766	71.398 ± 2.485	1.395	1117.690 ± 38.451	95.041 ± 0.912	21.421 ± 3.461	1.360
SSOA3	70.907 ± 0.304	70.197 ± 3.102	71.616 ± 2.967	1.004	1137.265 ± 45.136	94.474 ± 0.971	23.749 ± 3.238	0.944
Algorithm	"sum" on Sensorless				"$cost$" on Sensorless			
	Sum(%)	Sensitivity(%)	Specificity (%)	Time(s)	Cost	Sensitivity(%)	Specificity (%)	Time(s)
PAA	50.411 ± 0.487	8.049 ± 9.730	92.772 ± 8.926	0.426	4789.080 ± 85.260	11.395 ± 14.573	89.707± 14.189	0.432
OAAL	51.661 ± 0.377	39.208 ± 0.790	64.114 ± 0.683	0.441	4809.425 ± 40.441	38.309 ± 0.874	65.102 ± 0.578	0.445
CSOAL	50.582 ± 0.279	9.172 ± 8.983	91.993 ± 8.670	0.405	4774.760 ± 30.883	7.069 ± 2.585	93.870 ± 1.949	0.409
SOAL	69.381 ± 1.524	40.079 ± 3.141	**98.683 ± 0.179**	0.826	2904.685 ± 145.436	40.854 ± 3.215	**98.621 ± 0.298**	0.824
SOAL-CS	73.426 ± 1.118	49.253 ± 2.339	97.598 ± 0.338	0.874	2555.670 ± 113.284	49.267 ± 2.570	97.612± 0.362	0.871
OA3	87.944 ± 0.516	89.649 ± 0.973	86.238 ± 1.148	0.966	1219.940 ± 57.251	89.113 ± 0.948	86.863 ± 0.844	0.985
OA3$_{diag}$	86.268 ± 0.744	87.469 ± 2.042	85.067 ± 1.374	0.792	1364.925 ± 63.246	87.365 ± 1.190	85.710 ± 1.682	0.761
SOA3	**94.067 ± 0.238**	95.860 ± 0.318	92.274 ± 0.578	0.862	607.540 ± 34.277	**95.458 ± 0.484**	92.666 ± 0.823	0.863
SSOA3	94.011 ± 0.349	**95.870 ± 0.394**	92.151 ± 0.747	0.920	**605.860 ± 29.903**	95.441± 0.472	92.713 ± 0.707	0.913
Algorithm	"sum" on w8a				"$cost$" on w8a			
	Sum(%)	Sensitivity(%)	Specificity (%)	Time(s)	Cost	Sensitivity(%)	Specificity (%)	Time(s)
PAA	74.121 ± 1.196	57.716 ± 2.564	90.526 ± 0.213	0.990	1334.555 ± 29.329	57.512 ± 2.309	90.514 ± 0.270	0.982
OAAL	73.919 ± 0.848	57.850 ± 1.798	89.987 ± 0.163	1.022	1327.965 ± 16.675	56.945 ± 1.033	90.776 ± 0.040	1.034
CSOAL	78.929 ± 0.586	68.078 ± 1.307	89.780 ± 0.159	0.976	1193.555 ± 11.979	66.699 ± 0.967	90.214 ± 0.110	0.979
SOAL	75.688 ± 0.531	60.476 ± 1.071	**90.899 ± 0.042**	3.340	1262.290 ± 17.663	60.212 ± 1.031	**90.917 ± 0.028**	3.458
SOAL-CS	80.076 ± 0.380	69.912 ± 0.781	90.241 ± 0.050	3.763	1137.000 ± 14.635	69.268 ± 0.944	90.403 ± 0.053	3.752
OA3	83.313 ± 0.951	**84.369 ± 0.565**	82.257 ± 2.365	14.118	**1042.260 ± 7.455**	**79.240 ± 0.679**	89.149 ± 0.194	9.002
OA3$_{diag}$	**84.129 ± 0.260**	83.101 ± 0.435	85.157 ± 0.742	6.691	1048.075 ± 7.826	79.020 ± 0.477	89.117 ± 0.126	5.398
SOA3	81.759 ± 0.455	80.864 ± 0.760	82.655 ± 1.303	4.556	1150.620 ± 10.609	76.234 ± 0.974	88.256 ± 0.266	4.364
SSOA3	81.803 ± 0.265	80.365 ± 0.788	83.241 ± 0.816	3.398	1149.620 ± 8.379	76.079 ± 0.946	88.315 ± 0.273	3.332
Algorithm	"sum" on KDDCUP08				"$cost$" on KDDCUP08			
	Sum(%)	Sensitivity(%)	Specificity (%)	Time(s)	Cost	Sensitivity(%)	Specificity (%)	Time(s)
PAA	53.433 ± 4.439	48.475 ± 8.820	58.391 ± 2.286	1.065	1863.705 ± 155.865	23.900 ± 2.803	85.866 ± 1.624	1.082
OAAL	56.054 ± 1.589	27.006 ± 2.978	85.101 ± 1.273	1.034	1567.705 ± 72.265	21.701 ± 3.897	88.899 ± 0.845	1.045
CSOAL	55.891 ± 3.842	25.257 ± 8.550	**86.526 ± 1.625**	0.938	1206.975 ± 27.689	13.756 ± 1.855	92.885 ± 0.275	0.974
SOAL	53.768 ± 2.346	26.942 ± 5.182	80.594 ± 1.574	3.292	1126.580 ± 37.343	9.222 ± 0.525	93.926 ± 0.390	5.879
SOAL-CS	54.775 ± 2.515	28.363 ± 5.433	81.188 ± 1.693	3.468	1129.770 ± 36.692	9.342 ± 0.543	93.888 ± 0.383	5.935
OA3	73.189 ± 2.510	90.859 ± 2.354	55.520 ± 3.729	7.066	**931.645 ± 60.664**	35.947 ± 1.946	**94.369 ± 0.507**	4.467
OA3$_{diag}$	73.598 ± 2.144	87.844 ± 1.286	59.353 ± 3.817	2.699	977.920 ± 13.775	**38.644 ± 1.276**	93.765 ± 0.119	2.708
SOA3	74.200 ± 1.088	88.965 ± 1.810	59.434 ± 3.124	3.779	1063.260 ± 13.663	25.939 ± 3.965	93.627 ± 0.285	3.928
SSOA3	**75.642 ± 1.560**	**90.971 ± 1.329**	60.313 ± 3.234	3.376	1056.040 ± 16.917	25.778 ± 2.852	93.706 ± 0.296	3.970

In detail, our algorithms achieve good performance over a wide range of budgets in terms of both metrics. This observation further demonstrates the effectiveness and stability of our algorithms. Moreover, it suggests that our algorithms can help real-world companies with different labeling budgets.

In addition, when the query budget falls, the standard deviation of each algorithm increases. This observation implies that the randomness of samples plays an important role in performance, especially when the budget is limited, which validates the importance of query strategies.



(a) protein (b) Sensorless

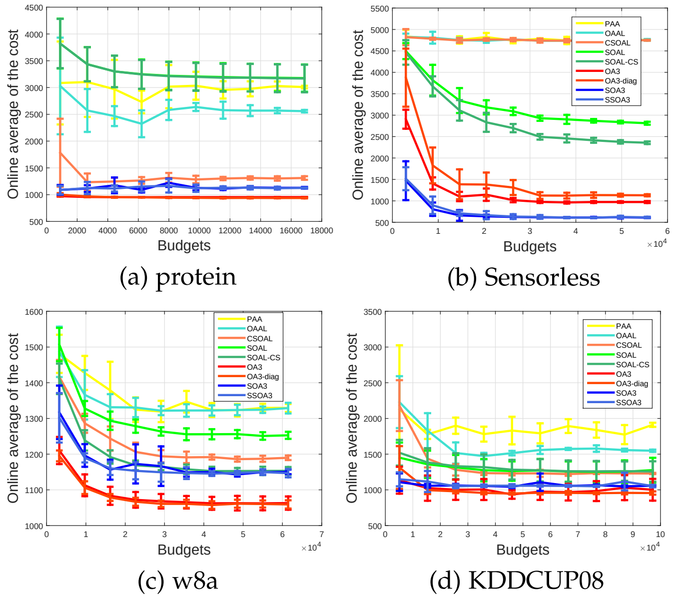
(c) w8a (d) KDDCUP08

Fig. 3: Evaluation of sum with varying budgets.

5.4 Evaluation of Algorithm Properties

We have evaluated performance of the proposed algorithms in previous experiments, where promising results confirm their superiority. Next, we further examine their unique properties, including the influence of query biases, cost weights and learning rates. The examinations contribute to better understanding and applications of the proposed methods. Due to the page limitation, we only report the

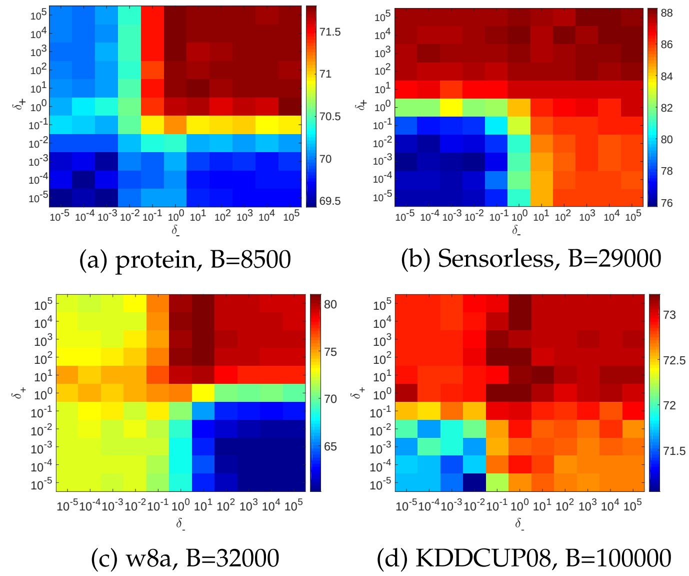
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2019.2955078, IEEE Transactions on Knowledge and Data Engineering



(a) protein (b) Sensorless

(c) w8a (d) KDDCUP08

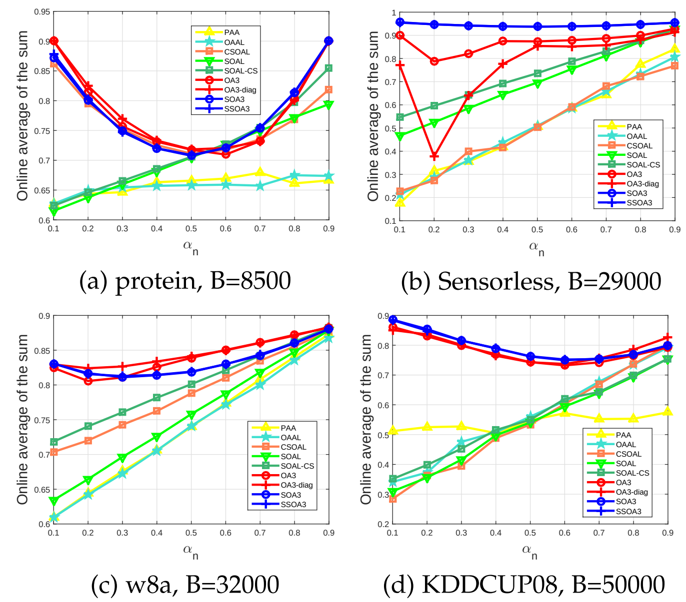
Fig. 4: Evaluation of cost with varying budgets.



(a) protein, B=8500 (b) Sensorless, B=29000

(c) w8a, B=32000 (d) KDDCUP08, B=100000

Fig. 5: Performance of sum with varying query parameters.



(a) protein, B=8500 (b) Sensorless, B=29000

(c) w8a, B=32000 (d) KDDCUP08, B=50000

Fig. 6: Performance of sum with varying cost weights.

results of *sum* metric, while more results based on *cost* metric are put in Supplementary C. Each experiment focuses on only one variable, while all other variable settings are fixed and similar with previous experiments.

5.4.1 Evaluation Between Query Biases

We first examine the influences of query biases on OA3 under a limited budget. All query biases (δ_+, δ_-) are selected from $[10^{-5}, 10^{-4}, ..., 10^4, 10^5]$.

The best results (i.e., deep red color in Fig. 5) are often obtained when $\delta_+ \in \{10^2, 10^3, 10^4, 10^5\}$ and $\delta_- \in \{1, 10\}$. This suggests, when querying more samples with the positive prediction (i.e., $\delta_+ \geq \delta_-$), OA3 can achieve better results. In other words, when paying more attention to positive predictions, the model will query more positive samples which are more informative in imbalanced tasks.

This observation is different from the discussions in OAAL [4], where the authors argued that δ_- should be larger than δ_+. The main difference is that our algorithm considers asymmetric strategies in both optimization and queries; while OAAL considers only the asymmetric query. As a result, our method can query more positive samples (*i.e.*, minority) due to the algorithm characteristics.

In addition, when both δ_+ and δ_- are large (i.e., the upper right corners in Fig. 5), our algorithms achieve fairly good performance. In this setting, the algorithm tends to query each observed sample and degrades to the "First come first served" strategy. This means that our algorithms with weak query strategy can also perform well. Moreover, when both δ_+ and δ_- are small (i.e., the bottom left corner), the model tends to ignore the samples, so the algorithm performance decreases significantly.

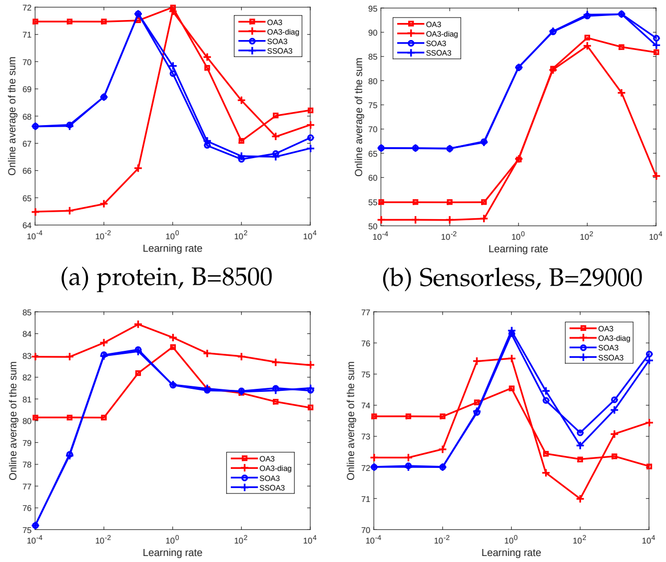
5.4.2 Evaluation of Cost Weights

In this subsection, we evaluate the influence of different cost weights, *i.e.*, α_n, where α_p=1−α_n. Fig. 6 summarizes the results of *sum* metric under a fixed budget. We find that our proposed algorithms consistently outperform all other algorithms with different weights. This observation shows that OA3 based algorithms have a wide selection range of cost weights, which further validates the effectiveness of the proposed methods.

5.4.3 Evaluation of Learning Rates

We next evaluate the influence of different learning rates to the proposed methods, where the learning rate η is selected from $[10^{-4}, 10^{-3}, ..., 10^3, 10^4]$.

Fig. 7 shows a suitable range of learning rates for different datasets, which provides a candidate choice of the learning rate for algorithm engineers. To be specific, OA3 algorithms achieve the best result on most datasets. Moreover, SOA3 and SSOA3 perform well on most datasets and sometimes even better than OA3. Considering that SOA3 and SSOA3 are more efficient than OA3, we conclude that the sketched versions of OA3 are favorable choices to balance performance and efficiency.



(a) protein, B=8500 (b) Sensorless, B=29000

(c) w8a, B=32000 (d) KDDCUP08, B=50000

Fig. 7: Performance of sum with varying learning rates.

5.5 Evaluation on High Dimensional Datasets

In this subsection, we further evaluate our methods on two higher dimensional datasets (CIFAR-10[5] and Internet Advertisements[6] (IAD)). The dimensionality of CIFAR-10 is 3072 and that of IAD is 1558. To be specific, we construct the CIFAR-10 dataset by randomly sampling 20% samples from the CIFAR-10 training set, and randomly take one class as the positive class and set other classes as the negative class. All images ($\mathbb{R}^{32\times32\times3}$) are squeezed to the vectors ($\mathbb{R}^{3072}$). In addition, we clean IAD by removing the samples that have null attributes. Other settings are the same as the previous experiments. Due to the page limits, we only report the results of *sum* metric, while more results based on *cost* metric and additional analysis are put in Supplementary C.

As shown in Table 3, OA3-based algorithms achieve the best performance on both datasets. However, the running time of OA3 is much longer than first-order methods. In contrast, the proposed sketched variants (i.e., SOA3 and SSOA3) are much faster than OA3. Meanwhile, they also perform very well and sometimes even better than OA3. In this sense, SOA3 and SSOA3 are favorable choices when handling real-world high-dimensional datasets.

6 Conclusion

In this paper, we have proposed a novel online adaptive asymmetric active learning algorithm to handle imbalanced and unlabeled datastream under limited query budgets. Relying on samples' second-order information, we develop a new asymmetric strategy, which integrates both the asymmetric losses and query strategies. We theoretically analyze the mistake and cost-sensitive metric bounds of the proposed algorithm, for the cases within budgets and over budgets.

To overcome the time-consuming problem of second-order methods, we further propose a sketch variant of our method, which can be developed as a sparse sketch approach. We empirically evaluate the proposed algorithms in real-world datasets. Promising results confirm the effectiveness, efficiency and stability of the proposed methods. In the future, we will extend the linear classifier to a nonlinear one with kernel methods.

5. https://www.cs.toronto.edu/~kriz/cifar.html
6. https://archive.ics.uci.edu/ml/datasets.php

7 Acknowledgement

This work was partially supported by National Natural Science Foundation of China (NSFC) 61602185, key project of NSFC (No. 61836003), Program for Guangdong Introducing Innovative and Enterpreneurial Teams 2017ZT07X183, Guangdong Provincial Scientific and Technological Funds under Grants 2018B010107001, Tencent AI Lab Rhino-Bird Focused Research Program (No. JR201902).

References

[1] N. Abe, B. Zadrozny, J. Langford. Outlier detection by active learning. *In SIGKDD*,2006, pp. 504-509.
[2] S. C. Hoi, R. Jin, J. Zhu, M. R. Lyu. Batch mode active learning and its application to medical image classification. *In ICML*, 2006.
[3] S. O. Moepya, S. S. Akhoury, F. V. Nelwamondo. Applying cost-sensitive classification for financial fraud detection under high class-imbalance. *In IEEE ICDM*, 2014, pp. 183-192.
[4] X. Zhang, T. Yang, P. Srinivasan. Online asymmetric active learning with imbalanced data. *In SIGKDD*, 2016, pp. 2055-2064. s
[5] P. Zhao, S. C. Hoi. Cost-sensitive online active learning with application to malicious URL detection. *In SIGKDD*, 2013, pp. 919-927.
[6] J. Wang, P. Zhao, S. C. Hoi. Exact soft confidence-weighted learning. *In ICML*, 2012, pp. 107-114.
[7] M. Dundar, B. Krishnapuram, J. Bi, R. B. Rao. Learning classifiers when the training data is not IID. *In IJCAI*, 2007, pp. 756-761.
[8] G. Hulten, L. Spencer, P. Domingos. Mining time-changing data streams. *In SIGKDD*, 2001, pp. 97-106.
[9] N. Cesa-Bianchi, A. Conconi, C. Gentile. A second-order perceptron algorithm. *SIAM Journal on Computing*, 2005, No. 3, pp. 640-668.
[10] J. Wang, P. Zhao and S. C. H. Hoi. Cost-sensitive online classification. *IEEE Transactions on Knowledge and Data Engineering*, 2014.
[11] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, Y. Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 2006, pp. 551-585.
[12] Y. Freund, R. E. Schapire. Large margin classification using the perceptron algorithm. *Machine learning*, 1999, Vol. 37, pp. 277-296.
[13] P. Zhao, S. C. Hoi. Cost-sensitive double updating online learning and its application to online anomaly detefction. *In SIAM ICDM*, 2013, pp. 207-215.
[14] N. Cesa-Bianchi, C. Gentile, L. Zaniboni. Worst-case analysis of selective sampling for linear classification. *Journal of Machine Learning Research*, 2006, No. 7, pp. 1205-1230.
[15] V. S. Sheng, F. Provost, P. G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. *In SIGKDD*, 2008, pp. 614-622.
[16] P. Zhao, Y. Zhang, M. Wu, S. C. Hoi, M. Tan, J. Huang. Adaptive cost-sensitive online classification. *IEEE Transactions on Knowledge and Data Engineering*, 2018.
[17] K. Crammer, A. Kulesza, M. Dredze. Adaptive regularization of weight vectors. *In NIPS*, 2009, pp. 414-422.
[18] K. Crammer, M. Dredze, F. Pereira. Exact convex confidence-weighted learning. *In NIPS*, 2009, pp. 345-352.
[19] P. Zhao, F. Zhuang, M. Wu, X. Li, and S. C. H. Hoi. Cost-sensitive online classification with adaptive regularization and its applications. *In IEEE ICDM*, 2015, pp. 649-658.
[20] H. Luo, A. Agarwal, N Cesa-Bianchi. Efficient second order online learning by sketching. *In NIPS*, 2016, pp. 902-910.
[21] Woodruff, P. David. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 2014.
[22] G. Krummenacher, B. McWilliams, Y. Kilcher, J. M. Buhmann, N. Meinshausen. Scalable adaptive stochastic optimization using random projections. *In NIPS*, 2016, pp. 1750-1758.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2019.2955078, IEEE Transactions on Knowledge and Data Engineering

TABLE 3: Sum evaluation on high-dimensional datasets

Algorithm	“sum“ on CIFAR-10				“sum“ on IAD			
	Sum(%)	Sensitivity(%)	Specificity (%)	Time(s)	Sum(%)	Sensitivity(%)	Specificity (%)	Time(s)
PAA	54.271 ± 2.255	18.060 ± 15.303	90.482 ± 15.269	0.513	52.570 ± 3.171	14.216 ± 12.900	90.924 ± 6.777	0.064
OAAL	63.510 ± 0.525	60.149 ± 1.977	66.872 ± 1.363	0.475	58.965 ± 1.420	55.123 ± 1.865	62.807 ± 2.379	0.061
CSOAL	56.331 ± 3.485	21.552 ± 14.413	91.110 ± 8.913	0.466	55.783 ± 5.938	19.338 ± 13.048	92.228 ± 1.235	0.059
SOAL	56.161 ± 0.765	18.229 ± 2.197	**94.092 ± 0.863**	529.586	74.496 ± 5.005	52.892 ± 10.550	**96.099 ± 0.984**	21.655
SOAL-CS	60.133 ± 0.839	27.582 ± 1.993	92.685 ± 0.532	584.077	78.259 ± 0.830	63.235 ± 2.127	93.282 ± 1.162	28.633
OA3	**72.858 ± 0.561**	**79.393 ± 3.627**	66.322 ± 3.994	1948.292	80.398 ± 0.624	76.201 ± 3.238	84.595 ± 2.594	44.674
OA3$_{diag}$	64.992 ± 1.830	64.448 ± 8.104	65.535 ± 6.215	152.623	81.359 ± 0.514	79.926 ± 2.227	82.792 ± 2.307	10.355
SOA3	67.959 ± 2.945	69.463 ± 8.768	66.456 ± 10.743	66.739	81.484 ± 2.133	85.515 ± 6.608	77.453 ± 7.641	4.084
SSOA3	68.678 ± 2.049	70.308 ± 6.538	67.047 ± 8.636	52.126	**82.101 ± 7.677**	**91.005 ± 5.876**	73.197 ± 19.614	3.215

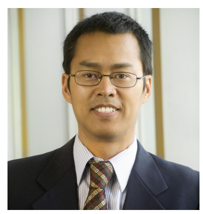
[23] D. Wang, P. Wu, P. Zhao, Y. Wu, C. Miao, S. C. Hoi. High-dimensional data stream classification via sparse online learning. *In IEEE ICDM*, 2014, pp. 1007-1012.
[24] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. *In ICML*, 2003, pp. 928-936.
[25] M.Dredze, K.Crammer, F.Pereira.Confidence-weighted linear classification.*In ICML*, 2008, 264-271.
[26] R. Horn. Matrix analysis. *Cambridge University Express*, 1985.
[27] E. Oja. Simplified neuron model as a principal component analyzer. Journal of Mathematical biology, 1982, pp. 267-273.
[28] E. Oja, J. Karhunen. On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix. *Journal of Mathematical Analysis and Applications*, 1985, vol. 106, pp. 69-84.
[29] M. Hardt, E. Price, The noisy power method: A meta algorithm with application. *In NIPS*, 2014, pp. 2861-2869.
[30] J. Lu, P. Zhao, S. C. Hoi. Online passive-aggressive active learning. *Machine Learning*, 2016, Vol. 103, No. 2, pp. 141-183.
[31] S. Hao, J. Lu, P. Zhao, C. Zhang, S. C. Hoi, C. Miao. Second-order online active learning and its applications. *IEEE Transactions on Knowledge and Data Engineering*, 2017.
[32] J. Attenberg, F. Provost. Why label when you can search? Alternatives to active learning for applying human resources to build classification models under extreme class imbalance. *In SIGKDD*, 2010, pp. 423-432.

Yifan Zhang is working toward the M.E. degree in the School of Software Engineering, South China University of Technology, China. He received the B.E. degree from the Southwest University, China, in 2017. His research interests are broadly in machine learning, with high self-motivation to design explainable and robust algorithms for data-limited problems and decision-making tasks.

Peilin Zhao is currently a Principal Researcher at Tencent AI Lab, China. Previously, he has worked at Rutgers University, Institute for Infocomm Research (I2R), Ant Financial Services Group. His research interests include: Online Learning, Deep Learning, Recommendation System, Automatic Machine Learning, etc. He has published over 90 papers in top venues, including JMLR, ICML, KDD, etc. He has been invited as a PC member, reviewer or editor for many international conferences and journals, such as ICML, JMLR, etc. He received his bachelor degree from Zhejiang University, and his Ph.D. degree from Nanyang Technological University.

Shuaicheng Niu is currently a Ph.D candidate in South China University of Technology (SCUT), China, School of Software Engineering. He received the Bachelor degree in mathematics from Southwest Jiaotong University (SWJTU), China, in 2018. His research interests include Machine Learning, Reinforcement Learning and Neural Network Architecture Search.

Jiezhang Cao is a master of the School of Software Engineering, South China University of Technology, China. He received the B.S. degree in statistics from the Guangdong University of Technology, China, 2017. His current research interests include machine learning, generative model.



Junzhou Huang is an Associate Professor in the Computer Science and Engineering department at the University of Texas at Arlington. He received the B.E. degree from Huazhong University of Science and Technology, China, the M.S. degree from Chinese Academy of Sciences, China, and the Ph.D. degree in Rutgers university. His major research interests include machine learning, computer vision and imaging informatics. He was selected as one of the 10 emerging leaders in multimedia and signal processing by the IBM T.J. Watson Research Center in 2010. He received the NSF CAREER Award in 2016.

Qingyao Wu received the Ph.D. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 2013. He was a Post-Doctoral Research Fellow with the School of Computer Engineering, Nanyang Technological University, Singapore, from 2014 to 2015. He is currently a Professor with the School of Software Engineering, South China University of Technology, Guangzhou, China. His current research interests include machine learning, data mining, big data research.

Mingkui Tan received his Bachelor Degree in Environmental Science and Engineering in 2006 and Master degree in Control Science and Engineering in 2009, both from Hunan University in Changsha, China. He received the Ph.D. degree in Computer Science from Nanyang Technological University, Singapore, in 2014. From 2014-2016, he worked as a Senior Research Associate on computer vision in the School of Computer Science, University of Adelaide, Australia. Since 2016, he has been with the School of Software Engineering, South China University of Technology, China, where he is currently a Professor. His research interests include machine learning, sparse analysis, deep learning and large-scale optimization.