Cost-Sensitive Double Updating Online Learning and Its Application to Online Anomaly Detection

Peilin Zhao* Steven C.H. Hoi†

Abstract

Although both *cost-sensitive classification* and *online learning* have been well studied separately in data mining and machine learning, there was very few comprehensive study of cost-sensitive online classification in literature. In this paper, we formally investigate this problem by directly optimizing cost-sensitive measures for an online classification task. As the first comprehensive study, we propose the Cost-Sensitive Double Updating Online Learning (CSDUOL) algorithms, which explores a recent double updating technique to tackle the online optimization task of cost-sensitive classification by maximizing the weighted sum or minimizing the weighted misclassification cost. We theoretically analyze the cost-sensitive measure bounds of the proposed algorithms, extensively examine their empirical performance for cost-sensitive online classification tasks, and finally demonstrate the application of our technique to solve online anomaly detection tasks.

1 Introduction

Online learning has been studied extensively in machine learning. Most existing online learning techniques are however not suitable to solve a cost-sensitive classification task, an important problem for data mining which takes the misclassification costs into consideration [8, 5]. This is because most existing online learning studies [14] often concern the performance of an online classification algorithm in terms of prediction *mistake rate* or *accuracy*, which is obviously *cost-insensitive* and thus *inappropriate* for many real applications in data mining, especially for cost-sensitive classification tasks where datasets are often class-imbalanced and the misclassification costs of instances from different classes can be very different [24, 3, 7].

To address the above challenge of cost-sensitive classification, researchers especially in data mining literature have proposed more meaningful metrics, such as the weighted sum of *sensitivity* and *specificity* [11, 2] and the weighted *misclassification cost* [8, 1]. Over the past decades, substantial research efforts have been devoted to developing batch classification algorithms to improve the cost-sensitive measures, including the weighted sum of sensitivity and specificity and the weighted misclassification cost metrics [8, 1]. However, these batch classification algorithms often suffer poor efficiency and scalability when solving large-scale problems, which thus are unsuitable for online classification applications.

Although both *cost-sensitive classification* and *online learning* have been studied extensively in data mining and machine learning communities, respectively, there was very few comprehensive study of cost-sensitive online classification in both data mining and machine learning literature. In this paper, we formally investigate this problem by directly optimizing cost-sensitive measures for an online classification task. As the first comprehensive study, we propose the "Cost-Sensitive Double Updating Online Learning" (CSDUOL) algorithms based on the Double Updating Online Learning (DUOL) technique [28] to tackle the online optimization task of maximizing the weighted sum or minimizing the weighted misclassification cost. We theoretically analyze the cost-sensitive measure bounds of the proposed algorithms, extensively examine their empirical performance of cost-sensitive online classification tasks, and finally demonstrate the application of our technique for solving online anomaly detection tasks.

The rest of this paper is organized as follows. Section 2 briefly reviews related work. Section 3 formulates the problem, presents our algorithms, and theoretically analyzes the bounds of the proposed algorithms. Section 4 discusses our experimental results. Section 5 shows an application to online anomaly detection tasks, and finally Section 6 concludes this paper.

2 Related Work

Our work is mainly related to two groups of research in data mining and machine learning communities, that is, cost-sensitive classification in data mining literature, and online learning in machine learning literature.

Cost-sensitive classification has been extensively studied in data mining and machine learning. To address this problem, researchers have proposed a vari-

*Nanyang Technological University, peilinzhao@ntu.edu.sg.
†Nanyang Technological University, chhoi@ntu.edu.sg.

ety of cost-sensitive metrics. The well-known examples include the weighted sum of *sensitivity* and *specificity* [11, 2], and the weighted *misclassification cost* that takes cost into consideration when measuring classification performance [8, 1]. As a special case, when the weights are both equal to 0.5, the weighted sum of sensitivity and specificity is reduced to the well-known *balanced accuracy* [2]. Over the past decades, various batch learning algorithms have been proposed for cost-sensitive classification in literature [21, 22, 5, 8, 19, 18].

Online learning has been extensively studied in machine learning community. Various online learning methods have been actively proposed in literature [20, 17, 4, 13, 27, 30, 12, 14]. Examples include the well-known Perceptron algorithm [20, 9], the recent Passive-aggressive (PA) learning [4], and many other recently proposed algorithms, many of which usually follow the principle of large margin learning [10, 15, 6]. Most online learning algorithms are cost-insensitive, except the prediction-based PA algorithm ('CPA$_{ML}$') [4] and the perceptron algorithm with uneven margin ('PAUM') [16]. However, very few existing work had attempted to directly optimize the two cost-sensitive metrics in an online learning setting, except some very recent work [25] which adopts a linear model and thus differs considerably from the DUOL algorithm used in this work. Finally, we note that our work is very different from another recent online learning study [29], which aims to optimize AUC, but cannot be guaranteed to optimize the cost-sensitive measures in our study.

3 Cost-Sensitive Online Classification

3.1 Problem Formulation Without loss of generality, let us consider an online binary classification problem. Formally, let us denote by $\mathbf{x}_t \in \mathrm{R}^d$ the instance received at the t-th learning step, and $f_{t-1} \in \mathcal{H}_\kappa$ a linear prediction model learned from the previous $t-1$ training examples. We also denote the prediction for the t-th instance as $\hat{y}_t = sign(f_{t-1}(\mathbf{x}_t))$, while the value $|f_{t-1}(\mathbf{x}_t)|$, known as the "margin", is used as the confidence of the learner on the prediction. The true label for instance $\mathbf{x}_t$ is denoted as $y_t \in \{-1, +1\}$. The learner made a mistake if and only if $\hat{y}_t \neq y_t$.

We now consider a sequence of training examples $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ for online learning. Then, for convenience, we denote by $\mathcal{M}$ the set of indexes that correspond to the trials of misclassification:

$$\mathcal{M} = \{t \ | y_t \neq \mathrm{sign}(f_t(\mathbf{x}_t)), \ \forall t \in [T]\},$$

where $[T] = \{1, \ldots, T\}$. Similarly, we denote by $\mathcal{M}_p = \{t \ | t \in \mathcal{M} \ and \ y_t = +1\}$ the set of indexes for false negatives, and $\mathcal{M}_n = \{t \ | t \in \mathcal{M} \ and \ y_t = -1\}$ the set of indexes for false positives.

Further, we introduce notation $M = |\mathcal{M}|$ to denote the number of mistakes, $M_p = |\mathcal{M}_p|$ to denote the number of false negatives, and $M_n = |\mathcal{M}_n|$ to denote the number of false positives. Also we use notation $\mathcal{I}_T^p = \{i \in [T] | y_i = +1\}$ to denote the set of indexes of the positive examples, $\mathcal{I}_T^n = \{i \in [T] | y_i = -1\}$ to denote the set of indexes of negative examples, $T_p = |\mathcal{I}_T^p|$ to denote the number of positive examples, and $T_n = |\mathcal{I}_T^n|$ to denote the number of negative examples. We adopt the following performance metrics:

$$sensitivity = \frac{T_p - M_p}{T_p}, \quad specificity = \frac{T_n - M_n}{T_n},$$
$$accuracy = \frac{T - M}{T}.$$

where *sensitivity* is defined as the ratio between the number of true positives $T_p - M_p$ and the total number of positives; *specificity* is defined as the ratio between $T_n - M_n$ and the total number of negatives; and *accuracy* is defined as the ratio between the number of correctly classified examples and the total number of examples.

Consider an online binary classification task, without loss of generality, we assume positive class is the rare class, i.e., $T_p \leq T_n$, the number of positive examples is smaller than the number of negative examples. For simplicity, we also assume that $\kappa(\mathbf{x}_t, \mathbf{x}_t) \leq 1$. For traditional online learning, the performance is measured by the prediction accuracy (or mistake rate equivalently) over the sequence of examples. This is inappropriate for imbalanced data because a trivial learner that simply classifies any example as negative could achieve a quite high accuracy for a highly imbalanced dataset. Thus, a more appropriate metric is to measure the *sum* of weighted *sensitivity* and *specificity*, i.e.,

$$sum = \eta_p \times sensitivity + \eta_n \times specificity \tag{3.1}$$

where $\eta_p + \eta_n = 1$ and $0 \leq \eta_p, \eta_n \leq 1$ are two parameters to trade off between sensitivity and specificity. Notably, when $\eta_p = \eta_n = 0.5$, the corresponding *sum* is the well known balanced accuracy. In general, the higher the *sum* value, the better the classification performance. Besides, another approach is to measure the total misclassification cost suffered by the algorithm, which is defined as:

$$cost = c_p \times M_p + c_n \times M_n \tag{3.2}$$

where $c_p + c_n = 1$ and $0 \leq c_p, c_n \leq 1$ are the misclassification cost parameters for positive and negative classes, respectively. The lower the *cost* value, the better the classification performance.

3.2 Algorithms In this section, we propose the Cost-Sensitive Double Updating Online Learning algorithms for cost-sensitive classification by optimizing two

cost-sensitive measures. Before presenting our algorithms, we first prove the following important proposition that motivates our solution.

PROPOSITION 1. *Consider a cost-sensitive classification problem, the goal of maximizing the weighted sum in (3.1) or minimizing the weighted cost in (3.2) is equivalent to minimizing the following objective:*

$$\sum_{y_t=+1} \theta \mathrm{I}_{(y_t f(\mathbf{x}_t)<0)} + \sum_{y_t=-1} \mathrm{I}_{(y_t f(\mathbf{x}_t)<0)}$$

where $\theta = \frac{\eta_p T_n}{\eta_n T_p}$ *for the maximization of the weighted sum, and* $\theta = \frac{c_p}{c_n}$ *for the minimization of the weighted misclassification cost.*

Proof. Firstly, by analyzing the function of the weighted sum in (3.1), we can derive the following:

$$\begin{aligned} sum &= \eta_p \frac{T_p - M_p}{T_p} + \eta_n \frac{T_n - M_n}{T_n} \\ &= 1 - \frac{\eta_n}{T_n}\Big[\frac{\eta_p T_n}{\eta_n T_p} \sum_{y_t=+1} \mathrm{I}_{(y_t f(\mathbf{x}_t)<0)} + \sum_{y_t=-1} \mathrm{I}_{(y_t f(\mathbf{x}_t)<0)}\Big], \end{aligned}$$

where I_π is the indicator function that outputs 1 if the statement π holds and 0 otherwise. Thus, maximizing *sum* is equivalent to minimizing

$$\frac{\eta_p T_n}{\eta_n T_p} \sum_{y_t=+1} \mathrm{I}_{(y_t f(\mathbf{x}_t)<0)} + \sum_{y_t=-1} \mathrm{I}_{(y_t f(\mathbf{x}_t)<0)}.$$

Secondly, by analyzing the function of the weighted cost in (3.2), we can also derive the following:

$$\begin{aligned} cost &= c_p M_p + c_n M_n \\ &= c_n \Big[\frac{c_p}{c_n} \sum_{y_t=+1} \mathrm{I}_{(y_t f(\mathbf{x}_t)<0)} + \sum_{y_t=-1} \mathrm{I}_{(y_t f(\mathbf{x}_t)<0)}\Big] \end{aligned}$$

Thus, minimizing *cost* is equivalent to minimizing

$$\frac{c_p}{c_n} \sum_{y_t=+1} \mathrm{I}_{(y_t f(\mathbf{x}_t)<0)} + \sum_{y_t=-1} \mathrm{I}_{(y_t f(\mathbf{x}_t)<0)}.$$

Thus, the proposition holds by setting $\theta = \frac{\eta_p T_n}{\eta_n T_p}$ for sum, and $\theta = \frac{c_p}{c_n}$ for cost.

Proposition 1 gives the explicit objective function for optimization, but the indicator function is not convex. To facilitate the online optimization task, we replace the indicator function by its convex surrogate, i.e., the following modified hinge loss function:

$$\ell(f;(\mathbf{x},y)) = \max(0, (\theta * \mathrm{I}_{(y=1)} + \mathrm{I}_{(y=-1)}) - y f(\mathbf{x}))$$

As a result, we can formulate the optimization problem for cost-sensitive classification as follows:

$$\mathcal{F}_T(f) = \frac{1}{2}\|f\|^2_{\mathcal{H}_\kappa} + C\sum_{t=1}^{T} \ell(f;(\mathbf{x}_t,y_t)), \tag{3.3}$$

where $\|f\|^2_{\mathcal{H}_\kappa}$ is introduced to regularize the complexity of the linear classifier and C is a positive penalty parameter of the cumulative loss. The idea of the above formulation is somewhat similar to the biased formulation of batch SVM for learning with imbalanced datasets [1].

Now our goal is to find an online learning solution to tackle the above convex optimization (3.3). To this end, we propose to explore double updating online learning to tackle this problem. Specifically, we consider trial t in an online learning task where the training example (x_a, y_a) is misclassified (i.e., $y_a f(\mathbf{x}_a) \leq 0$). Similar to DUOL for regular binary classification, we introduce an auxiliary example $(\mathbf{x}_b, y_b)$ from the existing support vectors that obey the following conditions:

- $y_b f(\mathbf{x}_b) \leq 0$, that is, support vector $(\mathbf{x}_b, y_b)$ is misclassified by the current classifier $f(\mathbf{x})$,
- $k(\mathbf{x}_b, \mathbf{x}_a) y_a y_b \leq -\rho$ where $\rho \in (0,1)$ is a predetermined threshold, that is, support vector $(\mathbf{x}_b, y_b)$ "**conflicts**" with the new misclassified example $(\mathbf{x}_a, y_a)$.

To facilitate the analysis, we also denote

$$\begin{aligned} &k_a = \kappa(\mathbf{x}_a, \mathbf{x}_a),\ k_b = \kappa(\mathbf{x}_b, \mathbf{x}_b), \\ &k_{ab} = \kappa(\mathbf{x}_a, \mathbf{x}_b),\ w_{ab} = y_a y_b k_{ab}. \end{aligned}$$

According to the assumption of auxiliary example, we have $w_{ab} = k_{ab} y_a y_b \leq -\rho$. Finally, we denote by $\widehat{\gamma}_b$ the weight for the auxiliary example $(\mathbf{x}_b, y_b)$ that is used in the current classifier $f(\mathbf{x})$, by γ_a and γ_b the updated weights for $(\mathbf{x}_a, y_a)$ and $(\mathbf{x}_b, y_b)$, respectively, and by d_{γ_b} the difference $\gamma_b - \widehat{\gamma}_b$.

Following the framework of dual formulation for online learning, the following lemma shows how to compute Δ_t, that is, the improvement in the objective function of dual biased SVM by adjusting weights for $(\mathbf{x}_a, y_a)$ and $(\mathbf{x}_b, y_b)$.

LEMMA 3.1. *The maximal improvement in the objective function of dual biased SVM by adjusting weights for* $(\mathbf{x}_a, y_a)$ *and* $(\mathbf{x}_b, y_b)$*, denoted by* Δ_t*, is computed by solving the following optimization problem:*

$$\Delta_t = \max_{\gamma_a, d_{\gamma_b}} \left\{ h(\gamma_a, d_{\gamma_b}) : 0 \leq \gamma_a \leq C,\ -\widehat{\gamma}_b \leq d_{\gamma_b} \leq C - \widehat{\gamma}_b \right\}, \tag{3.4}$$

where

$$\begin{aligned} h(\gamma_a, d_{\gamma_b}) &= \gamma_a(\theta_a - y_a f(\mathbf{x}_a)) + d_{\gamma_b}(\theta_b - y_b f(\mathbf{x}_b)) \\ &- \frac{k_a}{2}\gamma_a^2 - \frac{k_b}{2} d_{\gamma_b}^2 - w_{ab}\gamma_a d_{\gamma_b}. \end{aligned} \tag{3.5}$$

Proof. It is straightforward to verify that the dual function of $\min_{f\in\mathcal{H}_\kappa} \frac{1}{2}\|f\|^2_{\mathcal{H}_\kappa} + C\sum_{i=1}^t \max(0, \theta_i - y_i f(\mathbf{x}_i))$, where $\theta_i = \theta * \mathrm{I}_{(y_i=1)} + \mathrm{I}_{(y_i=-1)}$, denoted by $\mathcal{D}_t(\gamma_1, ..., \gamma_t)$, is computed as follows,

$$\mathcal{D}_t(\gamma_1, \ldots, \gamma_t) \quad = \quad \sum_{i=1}^t \gamma_i\theta_i - \frac{1}{2}\|\sum_{i=1}^t \gamma_i y_i \kappa(\mathbf{x}_i, \cdot)\|^2_{\mathcal{H}_\kappa},$$

where $\gamma_i \in [0, C]$, $i = 1, \ldots, t$ and $f_{t-1} = \sum_{i=1}^{t-1} \gamma_i y_i \kappa(\mathbf{x}_i, \cdot)$ is the current classifier. Thus

$$\begin{aligned} & h(\gamma_a, d_{\gamma_b}) \\ &= \mathcal{D}(\gamma_1, \ldots, \widehat{\gamma}_b + d_{\gamma_b}, \ldots, \gamma_{t-1}, \gamma_a) - \mathcal{D}(\gamma_1, \ldots, \widehat{\gamma}_b, \ldots, \gamma_{t-1}) \\ &= \gamma_a(\theta_a - y_a f_{t-1}(\mathbf{x}_a)) + d_{\gamma_b}(\theta_b - y_b f_{t-1}(\mathbf{x}_b)) - \frac{1}{2}\gamma_a^2\kappa(\mathbf{x}_a, \mathbf{x}_a) \\ &\quad - \frac{1}{2}d^2_{\gamma_b}\kappa(\mathbf{x}_b, \mathbf{x}_b) - \gamma_a d_{\gamma_b} y_a y_b \kappa(\mathbf{x}_a, \mathbf{x}_b). \end{aligned}$$

The theorem below bounds the bounding constant Δ when C is sufficiently large.

THEOREM 1. *Assume* $C \geq \max(\hat{\gamma}_b + \frac{\theta_b + \rho\theta_a}{1-\rho^2}, \frac{\theta_a + \rho\theta_b}{1-\rho^2})$ *with* $\rho \in [0, 1)$ *for the selected auxiliary example* $(\mathbf{x}_b, y_b)$*, we have the following bound for the bounding constant:*

$$\Delta \geq \frac{\theta_a^2 + 2\rho\theta_a\theta_b + \theta_b^2}{2(1-\rho^2)}.$$

Proof. First, we show $d_{\gamma_b} \geq 0$. This is because for given $\gamma_a \geq 0$, the optimal solution for d_{γ_b}, given by

$$d_{\gamma_b} = \frac{\theta_b - y_b f(\mathbf{x}_b) - w_{ab}\gamma_a}{k_b},$$

is positive because $y_b f(\mathbf{x}_b) \leq 0$ and $w_{ab} \leq -\rho$. Using the fact $k_a, k_b \leq 1$, $\gamma_a, d_{\gamma_b} \geq 0$, $y_a f(\mathbf{x}_a) \leq 0$, $y_b f(\mathbf{x}_b) \leq 0$, and $w_{a,b} \leq -\rho$, we have

$$h(\gamma_a, d_{\gamma_b}) \geq \gamma_a\theta_a + d_{\gamma_b}\theta_b - \frac{1}{2}\gamma_a^2 - \frac{1}{2}d^2_{\gamma_b} + \rho\gamma_a d_{\gamma_b}.$$

Thus, Δ is bounded as

$$\Delta \geq \max_{\gamma_a\in[0,C], d_{\gamma_b}\in[0, C-\widehat{\gamma}_b]} \gamma_a\theta_a + d_{\gamma_b}\theta_b - \frac{1}{2}(\gamma_a^2 + d^2_{\gamma_b}) + \rho\gamma_a d_{\gamma_b}.$$

Under the condition that $C \geq \max(\hat{\gamma}_b + \frac{\theta_b + \rho\theta_a}{1-\rho^2}, \frac{\theta_a + \rho\theta_b}{1-\rho^2})$, it is easy to verify that the optimal solution for the above problem is $\gamma_a = \frac{\theta_a + \rho\theta_b}{1-\rho^2}$ and $d_{\gamma_b} = \frac{\theta_b + \rho\theta_a}{1-\rho^2}$.

We refer to the case as a **strong double update** when the condition of Theorem 1 is satisfied. We have the following theorem for the general case when we only have $C \geq \max(\theta, 1)$.

THEOREM 2. *Assume* $C \geq \max(\theta, 1)$*. We have the following bound for* Δ *when updating the weight for the misclassified example* $(\mathbf{x}_a, y_a)$ *and the auxiliary example* $(\mathbf{x}_b, y_b)$*:*

$$\Delta \geq \frac{\theta_a^2}{2} + \frac{1}{2}\min((\theta_b + \rho\theta_a)^2, (C - \widehat{\gamma})^2).$$

Proof. By setting $\gamma_a = \theta_a$, we have $h(\gamma_a, d_{\gamma_b})$ computed as

$$h(\gamma_a = 1, d_{\gamma_b}) \geq \frac{\theta_a^2}{2} + (\theta_b + \rho\theta_a)d_{\gamma_b} - \frac{1}{2}d^2_{\gamma_b}.$$

Hence, Δ is lower bounded by

$$\begin{aligned} \Delta &\geq \frac{\theta_a^2}{2} + \max_{d_{\gamma_b}\in[0, C-\widehat{\gamma}]} \left((\theta_b + \rho\theta_a)d_{\gamma_b} - \frac{1}{2}d^2_{\gamma_b}\right) \\ &\geq \frac{\theta_a^2}{2} + \frac{1}{2}\min((\theta_b + \rho\theta_a)^2, (C - \widehat{\gamma})^2). \end{aligned}$$

Although Theorem 1 and Theorem 2 show that the double update strategy could significantly improve the bounding constant Δ, it is applicable only when there exists an auxiliary example. Below, we extend the double update strategy to the cases when there is no auxiliary example. Specifically, we relax the condition for performing double update as follows: there exists $(\mathbf{x}_b, y_b) \in \mathcal{D}$ that (i) $w_{ab} \leq -\rho$, (ii) $y_b f_{t-1}(\mathbf{x}_b) \leq \theta_b$, and (iii) $C \geq \max(\hat{\gamma}_b + \frac{\rho\theta_a}{1-\rho^2}, \frac{\theta_a}{1-\rho^2})$. We refer to these cases as **weak double update**.

THEOREM 3. *Assume* $w_{ab} \leq -\rho$*,* $y_b f_{t-1}(\mathbf{x}_b) \leq \theta_b$ *and* $C \geq \max(\hat{\gamma}_b + \frac{\rho\theta_a}{1-\rho^2}, \frac{\theta_a}{1-\rho^2})$*, we have the following bound for the bounding constant*

$$\Delta \geq \frac{\theta_a}{2(1-\rho^2)}.$$

Proof. Following the definitions and assumptions, we have

$$\begin{aligned} \Delta &= \max_{\gamma_a, d_{\gamma_b}} h(\gamma_a, d_{\gamma_b}) \geq h(\frac{\theta_a}{1-\rho^2}, \frac{\rho\theta_a}{1-\rho^2}) \\ &\geq \frac{\theta_a}{1-\rho^2}\theta_a + 0 - \frac{1}{2}(\frac{\theta_a}{1-\rho^2})^2 - \frac{1}{2}(\frac{\rho\theta_a}{1-\rho^2})^2 + \rho\frac{\theta_a^2}{1-\rho^2}\frac{\rho\theta_a}{1-\rho^2} \\ &= \frac{\theta_a}{2(1-\rho^2)}. \end{aligned}$$

Now solving the optimization problem 3.4 is the key to the double update. The following proposition provides the optimal solution to the problem 3.4.

PROPOSITION 2. *Denote* $\ell_a := \theta_a - y_a f(\mathbf{x}_a)$ *and* $\ell_b := \theta_b - y_b f(\mathbf{x}_b)$*. Assume* $\ell_a, \ell_b \geq 0$*,* $k_a, k_b > 0$ *and* $w_{ab} \leq 0$*, then the solution* (γ_a, d_{γ_b}) *of optimization problem (3.4) is as follows:*

$$\begin{cases} (C, C - \hat{\gamma}_b) & \text{if } (k_a C + w_{ab}(C - \hat{\gamma}_b) - \ell_a) < 0 \text{ and} \\ & \quad (k_b(C - \hat{\gamma}_b) + w_{ab}C - \ell_b) < 0 \\ (C, \frac{\ell_b - w_{ab}C}{k_b}) & \text{if } \frac{w_{ab}^2 C - w_{ab}\ell_b - k_a k_b C + k_b \ell_a}{k_b} > 0 \text{ and} \\ & \quad \frac{\ell_b - w_{ab}C}{k_b} \in [-\hat{\gamma}_b, C - \hat{\gamma}_b] \\ (\frac{\ell_a - w_{ab}(C-\hat{\gamma}_b)}{k_a}, C - \hat{\gamma}_b) & \text{if } \frac{\ell_a - w_{ab}(C-\hat{\gamma}_b)}{k_a} \in [0, C] \text{ and} \\ & \quad \ell_b - k_b(C - \hat{\gamma}_b) - w_{ab}\frac{\ell_a - w_{ab}(C-\hat{\gamma}_b)}{k_a} > 0 \\ (\frac{k_b\ell_a - w_{ab}\ell_b}{k_a k_b - w_{ab}^2}, \frac{k_a\ell_b - w_{ab}\ell_a}{k_a k_b - w_{ab}^2}) & \text{if } (\frac{k_b\ell_a - w_{ab}\ell_b}{k_a k_b - w_{ab}^2}, \frac{k_a\ell_b - w_{ab}\ell_a}{k_a k_b - w_{ab}^2}) \in \\ & \quad [0, C] \times [-\hat{\gamma}_b, C - \hat{\gamma}_b] \end{cases}.$$

We skip the proof for Proposition 2 as it is is similar to regular binary DUOL [28]. Algorithm 1 summarizes the proposed Cost-Sensitive Double Updating Online Learning (DUOL) algorithm. In this algorithm, to efficiently find the auxiliary example $(\mathbf{x}_b, y_b)$, we introduce a variable f_t^i for each support vector to keep track of its classification score. Parameter ρ is used to trade off between efficiency and efficacy for DUOL: the smaller ρ the more double updates will be performed.

Algorithm 1 The proposed Cost-Sensitive Double Updating Online Learning (CSDUOL) algorithms.

Initialize $S_0 = \emptyset$, $f_0 = 0$ bias parameter $\theta = \frac{\eta_p T_n}{\eta_n T_p}$ for "sum" and $\theta = \frac{c_p}{c_n}$ for "cost";
for $t = 1, 2, \ldots, T$ **do**
  Receive a new instance $\mathbf{x}_t$;
  Predict $\hat{y}_t = \text{sign}(f_{t-1}(\mathbf{x}_t))$;
  Receive its label y_t and compute $\theta_t = \theta * \mathrm{I}_{(y_t=1)} + \mathrm{I}_{(y_t=-1)}$;

  $\ell_t = max\{0, \theta_t - y_t f_{t-1}(\mathbf{x}_t)\}$;
  if $\ell_t > 0$ **then**
    $w_{min} = \infty$;
    for $\forall i \in S_{t-1}$ **do**
      if $(f_{t-1}^i \le \theta_i)$ **then**
        if $(y_i y_t \kappa(\mathbf{x}_i, \mathbf{x}_t) \le w_{min})$ **then**
          $w_{min} = y_i y_t \kappa(\mathbf{x}_i, \mathbf{x}_t)$;
          $(\mathbf{x}_b, y_b) = (\mathbf{x}_i, y_i)$;
        end if
      end if
    end for
    $f_{t-1}^t = y_t f_{t-1}(\mathbf{x}_t)$;
    $S_t = S_{t-1} \cup \{t\}$;
    if $(w_{min} \le -\rho)$ **then**
      Compute γ_t and d_{γ_b} by solving the optimization (3.4);

      for $\forall i \in S_t$ **do**
        $f_t^i \leftarrow f_{t-1}^i + y_i \gamma_t y_t \kappa(\mathbf{x}_i, \mathbf{x}_t) + y_i d_{\gamma_b} y_b \kappa(\mathbf{x}_i, \mathbf{x}_b)$;
      end for
      $f_t = f_{t-1} + \gamma_t y_t \kappa(\mathbf{x}_t, \cdot) + d_{\gamma_b} y_b \kappa(\mathbf{x}_b, \cdot)$;
    else
      $\gamma_t = min(C, \ell_t / \kappa(\mathbf{x}_t, \mathbf{x}_t))$;
      for $\forall i \in S_t$ **do**
        $f_t^i \leftarrow f_{t-1}^i + y_i \gamma_t y_t \kappa(\mathbf{x}_i, \mathbf{x}_t)$;
      end for
      $f_t = f_{t-1} + \gamma_t y_t \kappa(\mathbf{x}_t, \cdot)$;
    end if
  else
    $f_t = f_{t-1}$; $S_t = S_{t-1}$;
    for $\forall i \in S_t$ **do**
      $f_t^i \leftarrow f_{t-1}^i$;
    end for
  end if
end for
Return f_T, S_T

Finally, we give the bound analysis for the CSDUOL algorithm. We denote by $\mathcal{M}_d^s(\rho)$, $\mathcal{M}_d^w(\rho)$ and $\mathcal{M}_s$ the sets of indexes for the cases of *strong*, *weak* and *single* double updating, respectively, that is, $\mathcal{M}_d^s(\rho) = \{t | \exists$ auxiliary example $(\mathbf{x}_b, y_b)$ s.t. $C \ge \max(\hat{\gamma}_b + \frac{\theta_b + \rho\theta_t}{1-\rho^2}, \frac{\theta_t + \rho\theta_b}{1-\rho^2})$ for $(\mathbf{x}_t, y_t)$, $t \in \mathcal{M}\}$, $\mathcal{M}_d^w(\rho) = \{t \in \mathcal{M}/\mathcal{M}_d^s(\rho) | \exists$ $(\mathbf{x}_b, y_b)$ s.t. $w_{ab} \le -\rho$, $y_b f_{t-1}(\mathbf{x}_b) \le \theta_b$ and $C \ge \max(\hat{\gamma}_b + \frac{\rho\theta_t}{1-\rho^2}, \frac{\theta_t}{1-\rho^2})\}$, $\mathcal{M}_s = \mathcal{M}/\left[\mathcal{M}_d^s(\rho) \bigcup \mathcal{M}_d^w(\rho)\right]$. Note that in set $\mathcal{M}_d^s(\rho)$, for the convenience of analysis, we only consider the subset of strong updates when the condition $C \ge \max(\hat{\gamma}_b + \frac{\theta_b + \rho\theta_t}{1-\rho^2}, \frac{\theta_t + \rho\theta_b}{1-\rho^2})$ is satisfied.

THEOREM 4. *Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ be a sequence of examples, where $\mathbf{x}_t \in \mathbb{R}^d$, $y_t \in \{-1, +1\}$ and $\kappa(\mathbf{x}_t, \mathbf{x}_t) \le 1$ for all t, and assume $C \ge \max(\theta, 1)$. Then the proposed CSDUOL algorithm satisfies the following inequality, for $\rho \in [0, 1)$*

$$\theta^2 M_p + M_n \le \min_{f \in \mathcal{H}_\kappa} 2\mathcal{F}_T(f) - \sum_{t \in \mathcal{M}_d^s(\rho)} \frac{2\rho\theta_t\theta_b + \theta_b^2 + \theta_t^2\rho^2}{1-\rho^2} - \sum_{t \in \mathcal{M}_d^w(\rho)} \frac{\theta_t^2\rho^2}{1-\rho^2}.$$

Proof. According to Theorem 1 and 3, we have

$$\min_{t \in \mathcal{M}_d^s(\rho)} \Delta_t \ge \frac{\theta_t^2 + 2\rho\theta_t\theta_b + \theta_b^2}{2(1-\rho^2)}, \quad \min_{t \in \mathcal{M}_d^w(\rho)} \Delta_t \ge \frac{\theta_t^2}{2(1-\rho^2)}.$$

Moreover, according to Theorem 2, we have $\Delta_t \ge \theta_t^2/2, \forall t \in \mathcal{M}$. Putting them together, we have

$$\sum_{t \in \mathcal{M}_s} \frac{\theta_t^2}{2} + \sum_{t \in \mathcal{M}_d^s(\rho)} \frac{\theta_t^2 + 2\rho\theta_t\theta_b + \theta_b^2}{2(1-\rho^2)} + \sum_{t \in \mathcal{M}_d^w(\rho)} \frac{\theta_t^2}{2(1-\rho^2)} \le \min_{f \in \mathcal{H}_\kappa} \mathcal{F}_T(f).$$

Using the fact $\mathcal{M} = \mathcal{M}_s \bigcup \mathcal{M}_d^w(\rho) \bigcup \mathcal{M}_d^s(\rho)$,

$$\sum_{t \in \mathcal{M}} \frac{\theta_t^2}{2} \le \min_{f \in \mathcal{H}_\kappa} \mathcal{F}_T(f) - \sum_{t \in \mathcal{M}_d^s(\rho)} \frac{2\rho\theta_t\theta_b + \theta_b^2 + \theta_t^2\rho^2}{2(1-\rho^2)} - \sum_{t \in \mathcal{M}_d^w(\rho)} \frac{\theta_t^2\rho^2}{2(1-\rho^2)}.$$

We complete the proof using $\theta_t = \theta$, if $y_t = +1$, otherwise $\theta_t = 1$.

Now our goal is to analyze the performance of the proposed algorithm in terms of the imbalance metrics. We first consider the weighted sum of sensitivity and specificity, i.e., $sum = \eta_p \times sensitivity + \eta_n \times specificity$, where $\eta_p + \eta_n = 1$ and $\eta_p \ge \eta_n > 0$. The following theorem gives the bound of "sum" by CSDUOL.

THEOREM 5. *Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ be a sequence of examples, where $\mathbf{x}_t \in \mathbb{R}^d$, $y_t \in \{-1, +1\}$ and $\kappa(\mathbf{x}_t, \mathbf{x}_t) \le 1$ for all t. By setting $\theta = \frac{\eta_p T_n}{\eta_n T_p} \ge 1$, and assuming $C \ge \max(\theta, 1)$, for any $f \in \mathcal{H}_\kappa$, we have the following bound for the proposed CSDUOL algorithm:*

$$sum \ge 1 - \frac{\eta_n}{T_n}\Big\{ \min_{f \in \mathcal{H}_\kappa} 2\mathcal{F}_T(f) - \sum_{t \in \mathcal{M}_d^s(\rho)} \frac{2\rho\theta_t\theta_b + \theta_b^2 + \theta_t^2\rho^2}{1-\rho^2} - \sum_{t \in \mathcal{M}_d^w(\rho)} \frac{\theta_t^2\rho^2}{1-\rho^2} \Big\}.$$

The proof can be found in the supplemental file `http://csduol.stevenhoi.org/CSDUOL_sup.pdf`.

In the above approach, the parameter θ is set to $\frac{\eta_p T_n}{\eta_n T_p}$, in which the ratio $\frac{T_n}{T_p}$ may be unavailable in

advance. To alleviate this issue, we consider another approach using the cost based performance metric. Specifically, we propose to set $\theta = \frac{c_p}{c_n}$, where c_p and c_n are the predefined cost parameters of false negative and false positive, respectively. We assume $c_p + c_n = 1$ and $0 \leq c_n < c_p$ since we would prefer to improve the accuracy of predicting the rare positive examples. By this setting, the following theorem gives us the cost bound of the proposed CSDUOL algorithm.

THEOREM 6. *Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ be a sequence of examples, where $\mathbf{x}_t \in \mathbb{R}^d$, $y_t \in \{-1, +1\}$ and $\kappa(\mathbf{x}_t, \mathbf{x}_t) \leq 1$ for all t. By setting $\theta = \frac{c_p}{c_n} \geq 1$, and assuming $C \geq \max(\theta, 1)$, for any $f \in \mathcal{H}_\kappa$, the overall cost made by the proposed CSDUOL algorithm over this sequence of examples is bounded as follows:*

$$\begin{aligned} cost \quad \leq \quad & c_n \Big\{ \min_{f \in \mathcal{H}_\kappa} 2\mathcal{F}_T(f) - \sum_{t \in \mathcal{M}_d^s(\rho)} \frac{2\rho\theta_t\theta_b + \theta_b^2 + \theta_t^2\rho^2}{1-\rho^2} \\ & - \sum_{t \in \mathcal{M}_d^w(\rho)} \frac{\theta_t^2\rho^2}{1-\rho^2} \Big\}. \end{aligned}$$

The proof can be found in the supplemental file `http://csduol.stevenhoi.org/CSDUOL_sup.pdf`.

4 Experiments of Cost-Sensitive Online Classification

This section is to evaluate the empirical performance of the two proposed algorithms. To facilitate our discussions, we denote by CSDUOL$_{sum}$ the proposed CSDUOL algorithm that aims to maximize the weighted sum of sensitivity and specificity, and CSDUOL$_{cost}$ the proposed CSDUOL algorithm that aims to minimize the overall misclassification cost. More details about our experiments can be found in our website `http://csduol.stevenhoi.org/`.

4.1 Experimental Testbed and Setup We compare two CSDUOL algorithms with the regular DUOL and a number of state-of-the-art online learning algorithms, including:

- "Perceptron": the kernel Perceptron algorithm [20],
- "ALMA$_p(\alpha)$": **A**pproximate **L**arge **M**argin **A**lgorithm [10],
- "ROMMA": the **R**elaxed **O**nline **M**aximum **M**argin **A**lgorithm [17],
- "PA-I": the PA-I version of **P**assive-**A**ggressive algorithm [4],
- "PAUM": the **P**erceptron **A**lgorithm with **U**neven **M**argin [16], and
- "CPA$_{ML}$": the **C**ost-sensitive **P**assive-**A**ggressive algorithm based on **M**ax-**L**oss update method [4].

Table 1: List of binary datasets in our experiments.

dataset	#Examples	#Features	#Pos:#Neg
a7a	16100	123	1:3.1
german	1000	24	1:2.3
spambase	4601	57	1:1.5
w7a	24692	300	1:32.4

To examine the performance, we test all the algorithms on a number of benchmark datasets from web machine learning repositories [1]. For space limitation, we randomly choose some of them for our following discussions, which are listed in Table 1.

To enable fair comparisons, all algorithms follow the same experimental settings. Specifically, for all the algorithms, we set the penalty parameter C as 10 and adopt the same Gaussian kernel with $\sigma = 8$. For the ALMA$_p(\alpha)$ algorithm, $p = 2$ and $\alpha = 0.9$. For the proposed CSDUOL$_{sum}$ algorithm, we set $\eta_p = \eta_n = 1/2$ and $\theta = \frac{\eta_p T_n}{\eta_n T_p}$ for all cases, while for CSDUOL$_{cost}$, we set $c_p = 0.95$ and $c_n = 0.05$ and $\theta = \frac{c_p}{c_n}$. For PAUM, the parameters are set as $\tau_+ = \theta$, $\tau_- = 1$ and $\eta = 1$; for CPA$_{ML}$, $\rho(-1, 1)$ is set to 1 and $\rho(1, -1)$ is set to θ. The threshold ρ of DUOL and CSDUOL is set to 0.

All the experiments were run over 20 random permutations on each dataset. The results are reported by averaging over these 20 runs. We evaluate the online classification performance by the weighted **sum** of sensitivity and specificity, and the weighted **cost**.

4.2 Evaluation of Weighted Sum Performance We first evaluate the weighted sum performance. The left part of Table 2 summarizes the results. We can draw the following observations.

Firstly, by examining the *sum* results, we found that CSDUOL$_{sum}$ always achieves the best for all the datasets, which significantly outperforms all the online algorithms, including two cost-sensitive online algorithms (PAUM and CPA) and the regular DUOL. This shows the proposed CSDUOL$_{sum}$ algorithm is effective in optimization of weighted sum.

Secondly, the number of support vectors of CSDUOL$_{sum}$ is comparable with the regular DOUL algorithm, less than the PA and CPA$_{ML}$ algorithms, while more than the other algorithms. This shows that the proposed technique does not suffer storing more support vectors as a cost for improving the performance.

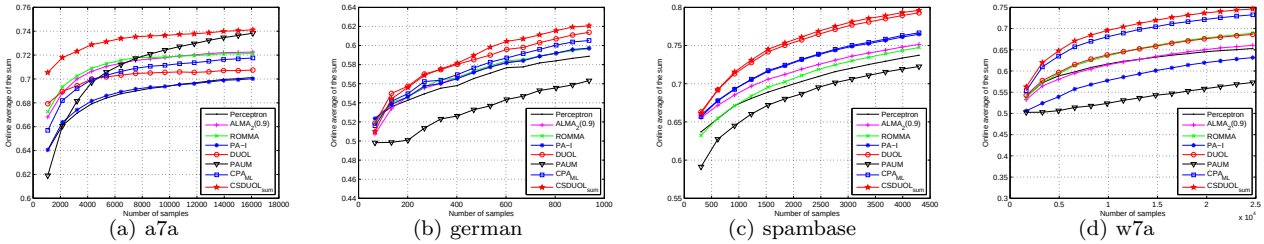
Thirdly, according to the running time results, we observe that CSDUOL$_{sum}$ is overall as efficient as the state-of-the-art online learning algorithms.

Finally, Figure 1 shows the changes of online average *sum* performance, from which we observe that the CSDUOL$_{sum}$ algorithms consistently outperform the other algorithms in the entire online learning process.

[1] `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`

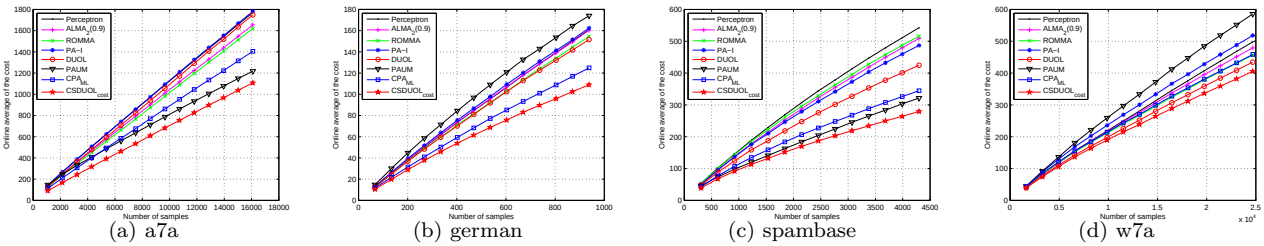
Table 2: Evaluation of the performance of CSDUOL and other existing algorithms.

Algorithm	sum on 'a7a'				cost on 'a7a'		
	Sum(%)	Support Vectors (#)	Time (s)		Cost	Support Vectors (#)	Time (s)
Perceptron	70.125 ± 0.337	3542.95 ± 39.866	2.132		1771.255 ±20.021	3542.950 ±39.866	2.157
$ALMA_2(0.9)$	72.256 ± 0.213	3586.15 ± 23.002	2.227		1654.950 ±12.299	3586.150 ±23.002	2.276
ROMMA	72.129 ± 0.44	3417.4 ± 51.542	2.175		1621.060 ±26.830	3417.400 ±51.542	2.211
PA-I	70.017 ± 0.436	6768.95 ± 57.628	4.444		1783.005 ±26.927	6768.950 ±57.628	4.400
DUOL	70.746 ± 0.355	6299.85 ± 49.036	9.122		1749.400 ±25.573	6299.850 ±49.036	8.948
PAUM	73.832 ± 0.33	5659.75 ± 22.627	3.717		1218.215 ±15.904	6527.700 ±21.796	4.417
CPA_{ML}	71.759 ± 0.35	6769.75 ± 57.819	4.691		1405.200 ±20.894	6769.500 ±56.947	4.666
CSDUOL	**74.115** ± 0.337	6425.1 ± 50.054	9.494		**1108.005** ±17.175	6442.150 ±42.521	9.528
Algorithm	sum on 'german'				cost on 'german'		
	Sum(%)	Support Vectors (#)	Time (s)		Cost	Support Vectors (#)	Time (s)
Perceptron	58.867 ± 1.116	347.6 ± 9.467	0.017		171.720 ±4.719	347.600 ±9.467	0.017
$ALMA_2(0.9)$	59.786 ± 0.994	394.75 ± 9.244	0.03		170.400 ±5.626	394.750 ±9.244	0.030
ROMMA	59.739 ± 1.178	347.25 ± 10.088	0.031		164.565 ±5.367	347.250 ±10.088	0.031
PA-I	59.750 ± 1.258	721.1 ± 12.994	0.027		173.050 ±5.977	721.100 ±12.994	0.027
DUOL	61.339 ± 1.12	656.9 ± 10.208	0.085		161.825 ±6.148	656.900 ±10.208	0.086
PAUM	56.621 ± 1.27	566.4 ± 9.185	0.023		183.715 ±8.171	599.300 ±1.559	0.024
CPA_{ML}	60.658 ± 1.276	721.75 ± 12.674	0.045		132.580 ±4.509	719.750 ±13.408	0.045
CSDUOL	**62.213** ± 1.392	657.9 ± 9.846	0.087		**116.025** ±5.627	668.450 ±13.032	0.090
Algorithm	sum on 'spambase'				cost on 'spambase'		
	Sum(%)	Support Vectors (#)	Time (s)		Cost	Support Vectors (#)	Time (s)
Perceptron	74.061 ± 0.515	1137.5 ± 22.596	0.207		573.190 ±11.575	1137.500 ±22.596	0.207
$ALMA_2(0.9)$	75.464 ± 0.602	1544.1 ± 19.62	0.329		537.460 ±15.673	1544.100 ±19.620	0.328
ROMMA	75.102 ± 0.566	1096.1 ± 24.809	0.262		544.330 ±13.056	1096.100 ±24.809	0.261
PA-I	76.789 ± 0.474	2854 ± 29.088	0.486		513.580 ±12.683	2854.000 ±29.088	0.491
DUOL	79.571 ± 0.382	2432.3 ± 27.058	0.974		447.400 ±12.762	2432.300 ±27.058	0.985
PAUM	72.591 ± 0.464	2275.5 ± 18.724	0.382		339.665 ±14.697	2943.350 ±13.112	0.508
CPA_{ML}	77.015 ± 0.466	2851.45 ± 28.956	0.559		362.100 ±13.095	2839.800 ±29.433	0.565
CSDUOL	**79.913** ± 0.354	2443.85 ± 24.314	0.991		**293.560** ±12.845	2636.000 ±24.917	1.127
Algorithm	sum on 'w7a'				cost on 'w7a'		
	Sum(%)	Support Vectors (#)	Time (s)		Cost	Support Vectors (#)	Time (s)
Perceptron	65.305 ± 0.831	994.4 ± 23.569	1.134		497.960 ±11.905	994.400 ±23.569	1.191
$ALMA_2(0.9)$	66.062 ± 0.685	1031.05 ± 15.33	1.305		479.345 ±9.348	1031.050 ±15.330	1.295
ROMMA	68.888 ± 0.6	1026.75 ± 21.511	1.337		456.595 ±8.542	1026.750 ±21.511	1.325
PA-I	63.155 ± 0.405	2842.6 ± 39.4	2.708		518.230 ±5.825	2842.600 ±39.400	2.688
DUOL	68.617 ± 0.626	2228.4 ± 41.031	2.732		434.400 ±8.467	2228.400 ±41.031	2.720
PAUM	57.285 ± 0.481	1477.95 ± 0.605	1.52		585.855 ±6.526	1477.950 ±0.605	1.576
CPA_{ML}	73.241 ± 0.699	2841.5 ± 39.91	2.955		458.700 ±8.042	2843.100 ±41.551	2.920
CSDUOL	**74.609** ± 0.616	2675.95 ± 41.082	3.448		**405.640** ±9.594	2547.750 ±32.301	3.227



(a) a7a (b) german (c) spambase (d) w7a

Figure 1: Evaluation of online "sum" performance of the CSDUOL_{sum} algorithm.



(a) a7a (b) german (c) spambase (d) w7a

Figure 2: Evaluation of online "cost" performance of the CSDUOL_{cost} algorithm.

4.3 Evaluation of Weighted Cost Performance We further evaluate the performance of the CSDUOL$_{cost}$ algorithm. The right part of Table 2 summarizes the results of total cost evaluation. From the results, we can also draw several observations.

First of all, among all the algorithms, we found that the proposed CSDUOL$_{cost}$ algorithm achieves significantly less total misclassification *cost* than the other algorithms for all the cases. For example, on the dataset "spambase", the total misclassification cost made by CSDUOL$_{cost}$ is about a half of those made by Perceptron algorithm. This demonstrates the proposed technique can effectively minimize the cost measure.

Further, the number of support vectors of CSDUOL$_{cost}$ is comparable with that of regular DOUL, less than those of PA and CPA$_{ML}$, while more than those of the other algorithms. For running time, we observe that CSDUOL$_{cost}$ is generally as efficient as the state-of-the-art online learning algorithms.

Finally, Figure 2 shows the changes of online average *cost* performance, from which we observe that the CSDUOL$_{cost}$ algorithms consistently outperform the other algorithms in the entire online learning process.

5 Application to Online Anomaly Detection

The proposed cost-sensitive online classification technique can be potentially applied to a wide range of applications. In this section, we show an application of the proposed algorithms to tackle online anomaly detection tasks. Below we first introduce the applications followed by presenting our empirical results.

5.1 Application Domains and Testbeds. We apply our technique to the following domains:

- Bioinformatics: We apply our algorithm to solve a bioinformatics problem using the "Code-RNA" dataset [23]. The objective of this task is to develop a computational method to detect novel non-coding RNAs from some large sequenced genomes. Non-coding RNAs are defined as anomalies and others are considered as normal instances.
- Finance: We apply our algorithm to a credit card approval problem in finance domain. In particular, we consider the well-known Australia credit card data set with 690 instances from an Australian credit company, in which the task is to distinguish credit-worthy from non credit-worthy customers.
- Medical Imaging: We apply our algorithm to solve medical image anomaly detection tasks. We consider the "Wisconsin Breast Cancer" dataset [26], for which the goal is to detect breast cancer from medical images of a fine needle aspirate (FNA) of a breast mass. For this task, the class "benign" is assigned as the normal class, and the class "malignant" is the anomaly class.

Table 3 summarizes the details of these data sets.

Table 3: Data Sets for Online Anomaly Detection.

Dataset Name	#Examples	#Features	#Outlier:#Normal
Cod-RNA	271617	8	1:2.00
Australian	690	14	1:1.25
Breast Cancer	683	10	1:1.86

Table 4: Evaluation of balanced accuracy performance for online anomaly detection.

Algorithm	Cod-RNA		
	Balanced Accuracy(%)	Support Vectors (#)	Time (s)
Perceptron	85.137 ± 0.260	1318.700 ± 23.052	0.567
ALMA$_2$(0.9)	87.087 ± 0.241	1330.100 ± 20.870	0.614
ROMMA	87.923 ± 0.227	1060.650 ± 20.717	0.546
PA-I	86.940 ± 0.293	3347.250 ± 34.860	1.226
DUOL	89.371 ± 0.209	2134.250 ± 31.217	1.251
PAUM	86.547 ± 0.294	2830.050 ± 27.594	1.143
CPA$_{ML}$	87.670 ± 0.254	3347.350 ± 35.006	1.360
CSDUOL$_{sum}$	**90.248** ± 0.273	2262.900 ± 31.792	1.367
Algorithm	Australian		
	Balanced Accuracy(%)	Support Vectors (#)	Time (s)
Perceptron	76.623 ± 1.294	159.350 ± 8.821	0.009
ALMA$_2$(0.9)	79.486 ± 1.098	160.800 ± 7.223	0.017
ROMMA	78.245 ± 0.908	148.400 ± 6.278	0.019
PA-I	77.707 ± 1.251	351.350 ± 8.536	0.013
DUOL	79.524 ± 1.218	283.250 ± 11.350	0.033
PAUM	77.343 ± 1.043	281.450 ± 6.669	0.012
CPA$_{ML}$	77.800 ± 1.340	351.500 ± 8.757	0.024
CSDUOL$_{sum}$	**80.214** ± 0.950	285.700 ± 10.352	0.034
Algorithm	Wisconsin Breast Cancer		
	Balanced Accuracy(%)	Support Vectors (#)	Time (s)
Perceptron	91.876 ± 0.685	50.400 ± 4.248	0.007
ALMA$_2$(0.9)	93.150 ± 0.465	53.250 ± 3.007	0.015
ROMMA	93.389 ± 0.701	40.650 ± 4.043	0.016
PA-I	93.923 ± 0.493	152.250 ± 8.819	0.010
DUOL	94.859 ± 0.578	89.000 ± 10.618	0.016
PAUM	93.541 ± 0.652	112.950 ± 3.993	0.010
CPA$_{ML}$	94.368 ± 0.589	152.050 ± 8.407	0.018
CSDUOL$_{sum}$	**95.529** ± 0.431	92.250 ± 10.852	0.017

5.2 Empirical Evaluation Results. We apply our algorithm to solve anomaly detection tasks on the real-world datasets as shown in Table 3. For performance metric, we evaluate the anomaly detection performance using the *balanced accuracy*, which is able to avoid inflated performance estimates on imbalanced datasets which are very common in anomaly detection tasks. Table 4 summarizes the experimental results. From the results, we can draw the following two observations.

Firstly, among all the existing algorithms, the two cost-sensitive algorithms (PAUM and CPA$_{ML}$) generally perform better than their regular versions. However, the improvements are not always consistent and significant across different datasets. Such observations validate the importance of investigating more effective cost-sensitive online learning algorithms.

Secondly, among all the compared algorithms, it is obvious that CSDUOL$_{sum}$ significantly outperforms the other algorithms on all the datasets. The promising result shows the advantage of the proposed algorithm for real-world online anomaly detection tasks.

6 Conclusion

This paper investigated a new framework of Cost-Sensitive Online Classification that directly optimizes some cost-sensitive metrics. Specifically, we proposed two effective cost-sensitive DUOL algorithms based on the recent Double Updating Online Learning (DUOL) techniques, theoretically analyzed their cost-sensitive bounds, and finally examined their empirical performance extensively, including their applications to online anomaly detection tasks. Our encouraging results show that our algorithms outperform the existing algorithms for cost-sensitive online classification tasks.

Acknowledgements

This work was partly supported by Singapore MOE tier-1 grant (RG33/11), and Singapore NRF IDM research grant (MDA/IDM/2012/8/8-2VOL01).

References

[1] Rehan Akbani, Stephen Kwek, and Nathalie Japkowicz. Applying support vector machines to imbalanced datasets. In *ECML*, pages 39–50, 2004.

[2] Kay Henning Brodersen, Cheng Soon Ong, Klaas Enno Stephan, and Joachim M. Buhmann. The balanced accuracy and its posterior distribution. In *ICPR*, pages 3121–3124, 2010.

[3] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *JAIR*, 16:321–357, 2002.

[4] Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *J. Mach. Learn. Res. (JMLR)*, 7:551–585, 2006.

[5] Pedro Domingos. Metacost: a general method for making classifiers cost-sensitive. In *KDD'99*, pages 155–164, San Diego, CA, USA, 1999. ACM.

[6] Mark Dredze, Koby Crammer, and Fernando Pereira. Confidence-weighted linear classification. In *ICML*, pages 264–271, 2008.

[7] Chris Drummond and Robert C. Holte. C4.5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling. In *ICML'03 Workshop on Learning from Imbalanced Data Sets*, pages 1–8, 2003.

[8] Charles Elkan. The foundations of cost-sensitive learning. In *IJCAI*, pages 973–978, 2001.

[9] Yoav Freund and Robert E. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296, 1999.

[10] Claudio Gentile. A new approximate maximal margin classification algorithm. *J. Mach. Learn. Res. (JMLR)*, 2:213–242, 2001.

[11] Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2000.

[12] Steven C. H. Hoi, Rong Jin, Peilin Zhao, and Tianbao Yang. Online multiple kernel classification. *Machine Learning*, 90(2):289–316, 2013.

[13] Steven C. H. Hoi, Jialei Wang, and Peilin Zhao. Exact soft confidence-weighted learning. In *ICML*, 2012.

[14] Steven C.H. Hoi, Jialei Wang, and Peilin Zhao. *LIBOL: A Library for Online Learning Algorithms*. Nanyang Technological University, 2012.

[15] Jyrki Kivinen, Alexander J. Smola, and Robert C. Williamson. Online learning with kernels. *IEEE Tran. on Signal Processing*, 52(8):2165–2176, 2004.

[16] Yaoyong Li, Hugo Zaragoza, Ralf Herbrich, John Shawe-Taylor, and Jaz S. Kandola. The perceptron algorithm with uneven margins. In *ICML*, pages 379–386, 2002.

[17] Yi Li and Philip M. Long. The relaxed online maximum margin algorithm. In *NIPS*, pages 498–504, 1999.

[18] Yu-Feng Li, James T. Kwok, and Zhi-Hua Zhou. Cost-sensitive semi-supervised support vector machine. In *AAAI*, 2010.

[19] Aurélie C. Lozano and Naoki Abe. Multi-class cost-sensitive boosting with p-norm loss functions. In *KDD'08*, pages 506–514, 2008.

[20] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–407, 1958.

[21] Ming Tan. Cost-sensitive learning of classification knowledge and its applications in robotics. *Mach. Learn.*, 13(1):7–33, October 1993.

[22] Peter D. Turney. Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *JAIR*, 2:369–409, 1995.

[23] Andrew V. Uzilov, Joshua M. Keegan, and David H. Mathews. Detection of non-coding rnas on the basis of predicted secondary structure formation free energy change. *BMC Bioinformatics*, 7:173, 2006.

[24] K. Veropoulos, C. Campbell, and N. Cristianini. Controlling the sensitivity of support vector machines. In *IJCAI*, pages 55–60, 1999.

[25] Jialei Wang, Peilin Zhao, and Steven C. H. Hoi. Cost-sensitive online classification. In *ICDM*, pages 1140–1145, 2012.

[26] Street W.N. Wolberg, W.H. and O.L. Mangasarian. Machine learning techniques to diagnose breast cancer from fine-needle aspirates. *Cancer Letters*, 77:163–171, 1994.

[27] Peilin Zhao and Steven C. H. Hoi. Bduol: Double updating online learning on a fixed budget. In *ECML/PKDD (1)*, pages 810–826, 2012.

[28] Peilin Zhao, Steven C. H. Hoi, and Rong Jin. Double updating online learning. *Journal of Machine Learning Research*, 12:1587–1615, 2011.

[29] Peilin Zhao, Steven C. H. Hoi, Rong Jin, and Tianbao Yang. Online auc maximization. In *ICML*, pages 233–240, 2011.

[30] Peilin Zhao, Jialei Wang, Pengcheng Wu, Rong Jin, and Steven C. H. Hoi. Fast bounded online gradient descent algorithms for scalable kernel-based online learning. In *ICML*, 2012.