Adaptive Sparse Confidence-Weighted Learning for Online Feature Selection

Yanbin Liu,[1,2] **Yan Yan,**[2] **Ling Chen,**[2] **Yahong Han,**[3] **Yi Yang**[2]
[1]SUSTech-UTS Joint Centre of CIS, Southern University of Science and Technology
[2]Centre for Artificial Intelligence, University of Technology Sydney
[3]College of Intelligence and Computing, Tianjin University
{csyanbin, yanyan.tju}@gmail.com, ling.chen@uts.edu.au, yahong@tju.edu.cn, yi.yang@uts.edu.au

Abstract

In this paper, we propose a new online feature selection algorithm for streaming data. We aim to focus on the following two problems which remain unaddressed in literature. First, most existing online feature selection algorithms merely utilize the first-order information of the data streams, regardless of the fact that second-order information explores the correlations between features and significantly improves the performance. Second, most online feature selection algorithms are based on the balanced data presumption, which is not true in many real-world applications. For example, in fraud detection, the number of positive examples are much less than negative examples because most cases are not fraud. The balanced assumption will make the selected features biased towards the majority class and fail to detect the fraud cases. We propose an Adaptive Sparse Confidence-Weighted (ASCW) algorithm to solve the aforementioned two problems. We first introduce an ℓ_0-norm constraint into the second-order confidence-weighted (CW) learning for feature selection. Then the original loss is substituted with a cost-sensitive loss function to address the imbalanced data issue. Furthermore, our algorithm maintains multiple sparse CW learner with the corresponding cost vector to dynamically select an optimal cost. We theoretically enhance the theory of sparse CW learning and analyze the performance behavior in F-measure. Empirical studies show the superior performance over the state-of-the-art online learning methods in the online-batch setting.

1 Introduction

Online learning typically receives and processes a single instance at a time. It has become extremely popular and been employed in many applications such as video-ad allocation (Sumita et al. 2017). In order to deal with high dimensional data streams, online feature selection (OFS) has been proposed to select a fixed number of features for prediction by an online learning fashion.

Existing online feature selection algorithms usually apply the first-order updating rule (Wang et al. 2014; Han et al. 2016). For example, OFS (Wang et al. 2014) modified the first-order Perceptron (Rosenblatt 1958) algorithm by applying truncation. However, feature interactions are ignored by these algorithms. Prior studies in online learning have attested the effectiveness of second-order algorithms, such as confidence-weighted (CW) learning (Crammer, Dredze, and Pereira 2009), with a covariance structure exploring the feature correlations. Due to the high computation cost of covariance matrix, very few methods (Tan et al. 2016) have been advanced for second-order online feature selection.

While class imbalance is prevalent in real-world applications, it remains to be under-studied in the context of online feature selection. Current online learning methods usually combine first-order updating rules with cost-sensitive learning to deal with class imbalance (Wang, Zhao, and Hoi 2014; Zhao and Hoi 2013; Yan et al. 2017). In this sense, how to decide appropriate cost values is the key challenge in these methods. While most algorithms adopt fixed or ad-hoc schemes to compute costs from the given data, OMCSL (Yan et al. 2017) trains a number of classifiers with various costs and achieves improved performance.

To the best of our knowledge, no previous work has uncovered the problem of online feature selection with the presence of class imbalance. Motivated by this, we propose an Adaptive Sparse CW algorithm (ASCW) for imbalanced online-batch feature selection. Specifically, our method simultaneously maintains multiple sparse CW learners. For each learner, we assign a unique cost vector to its objective function. As the online training proceeds, we incrementally update the target measure for each learner in an online manner. For each online-batch, we choose the best performer for prediction. The main contributions of our paper are summarized as follows:

- We propose an adaptive sparse CW method for feature selection on imbalanced online-batch data. Unlike previous approaches that use a fixed or ad-hoc cost vector, our method dynamically chooses the best cost from a set of candidates by incrementally updating the target performance for each learner.
- We enhance the theory of the existing sparse CW feature selection algorithm and analyze the performance behavior for F-measure.
- Empirical studies demonstrate the efficacy of the proposed algorithm. Further results show that our algorithm is capable to automatically choose a cost that is sufficiently close to the best one.

The remainder of the paper is organized as follows. We first briefly review related work and then present the problem

formulation and the cost-sensitive sparse CW algorithm for imbalanced feature selection. Next, we show how the adaptive strategy chooses a cost from candidates and provide theoretical analysis. We further discuss our experimental results and finally, conclude the paper.

2 Related Work

Online learning has been extensively studied in machine learning community (Crammer et al. 2006; Crammer, Dredze, and Pereira 2009; Wang, Zhao, and Hoi 2012; Crammer, Kulesza, and Dredze 2009; Ma et al. 2010; Dong et al. 2019). First-order algorithms (Crammer et al. 2006; Zinkevich 2003) usually ignore the direction and scale of parameter updates. Confidence-weighted (CW) learning (Crammer, Dredze, and Pereira 2009) addresses this issue by assuming a Gaussian distribution over weights with mean $\boldsymbol{\mu} \in \mathbb{R}^d$ and covariance $\boldsymbol{\Sigma} \in \mathbb{R}^{d\times d}$ with theoretical guarantees in terms of mistake bounds. However, the aggressive update rules based on separable data assumption may cause over-fitting for noisy data. *Adaptive Regularization of Weights* (AROW) (Crammer, Kulesza, and Dredze 2009) relaxes such separable assumption by employing a soft-margin squared hinge loss plus a confidence penalty. As another solution, *Soft Confidence-weighted* (SCW) (Wang, Zhao, and Hoi 2012) assigns adaptive margins for different instances.

Cost-sensitive approaches have been proposed to deal with imbalanced online learning problem, such as CSOGD (Wang, Zhao, and Hoi 2014), CSOAL (Zhao and Hoi 2013), and MBPA (Han et al. 2016). They either utilize PA (Crammer et al. 2006) or OGD (Zinkevich 2003) updating rules, which only consider the first-order information and ignore covariance structure. ACOG (Zhao et al. 2018) adopts the idea of adaptive regularization to incorporate the second-order information, which is similar to our method. However, they use ad-hoc cost values computed from the training instances while we dynamically choose the optimal cost from a set of candidates.

Many online feature selection methods have been proposed recently (Wu et al. 2013; Zhou et al. 2017; Yu et al. 2014; Wang et al. 2014; Han et al. 2016; Wu et al. 2017; Tan et al. 2016), most of which are first-order methods. For example, OFS (Wang et al. 2014) adopts the first-order Perceptron (Rosenblatt 1958) updating rule and MBPA (Han et al. 2016) utilizes the first-order PA (Crammer et al. 2006) updating rule. Based on CW learning, (Wu et al. 2017) tries to incorporate the diagonal elements of the covariance matrix for online feature selection. However, the feature correlations is not fully explored by only using diagonal information. Compared with the above methods, we not only explore feature correlations by incorporating second-order covariance structure but also select features that can better fit the imbalanced measures due to the adaptive cost-selection strategy.

3 Imbalanced Online-Batch CW Learning

3.1 Notations

We first present some notations. Let superscript ^{T} represent transpose, $\mathbf{0}$ be a vector/matrix with all zeros, $\|\cdot\|_p$ denote the l_p-norm of a vector, $\mathrm{diag}(\cdot)$ be the diagonal matrix, $A \odot B$ stand for the element-wise product of A and B, and $\mathbb{I}(b)$ be an indicator function, where $\mathbb{I}(b) = 1$ if b is true and 0 otherwise. Let $[n] = \{1, \ldots, n\}$. $\{\mathbf{X}_h, \mathbf{y}_h\}$ denote examples received at the h-th iteration, where $\mathbf{X}_h \in \mathbb{R}^{d\times N_h}$ and $\mathbf{y}_h \in \{-1, 1\}^{N_h}$. $\boldsymbol{\mu}_h$ and $\boldsymbol{\Sigma}_h$ respectively represent model weights and covariance at the h-th iteration. We denote $f_h(\mathbf{X}_h) : \mathbb{R}^{d\times N_h} \to \mathbb{R}^{N_h}$ as the prediction function at the h-th iteration and $f_h = f_h(\mathbf{X}_h)$ as the predictions.

3.2 Cost-Sensitive Learning for Imbalanced Data

For traditional confidence-weighted learning (Crammer, Dredze, and Pereira 2009; Wang, Zhao, and Hoi 2012) and high-dimensional online feature selection such as (Tan et al. 2016), the cumulative mistake is optimized by the hinge loss as: $\ell(\boldsymbol{\mu}; (\mathbf{x}_i, y_i)) = \max(0, 1 - y_i \boldsymbol{\mu}^T \mathbf{x}_i)$. However, for imbalanced feature selection, this loss function ignores cost asymmetry between the majority classes and the minority ones. Thus, we propose the cost-sensitive loss function to deal with the imbalanced problem: $\ell_c(\boldsymbol{\mu}; (\mathbf{x}_i, y_i)) = c_+\mathbb{I}(y_i = 1)\ell(\boldsymbol{\mu}; (\mathbf{x}_i, y_i)) + c_-\mathbb{I}(y_i = -1)\ell(\boldsymbol{\mu}; (\mathbf{x}_i, y_i))$. Let $D_i = c_+\mathbb{I}(y_i = 1) + c_-\mathbb{I}(y_i = -1)$. Then, $\ell_c(\boldsymbol{\mu}; (\mathbf{x}_i, y_i)) = D_i\ell(\boldsymbol{\mu}; (\mathbf{x}_i, y_i))$. Moreover, we also propose $\ell_c^2(\boldsymbol{\mu}; (\mathbf{x}_i, y_i)) = D_i\ell(\boldsymbol{\mu}; (\mathbf{x}_i, y_i))^2$ as the cost-sensitive squared hinge loss.

Thus, how to choose c_+ and c_- is the key issue for imbalanced learning. We will describe the choice strategy in section 5, together with a theoretical analysis in detail.

3.3 Online-Batch CW Learning

Inspired by AROW (Crammer, Kulesza, and Dredze 2009) and cost-sensitive learning (Wang, Zhao, and Hoi 2014), we propose an algorithm to estimate $\boldsymbol{\mu}, \boldsymbol{\Sigma}$ at the h-th iteration for online-batch data.

Fix $\boldsymbol{\mu}$ and update $\boldsymbol{\Sigma}$. We learn $\boldsymbol{\Sigma}$ for the following problem:

$$\min_{\boldsymbol{\Sigma}} \mathrm{D}_{\mathrm{KL}}(\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \| \mathcal{N}(\boldsymbol{\mu}_{h-1}, \boldsymbol{\Sigma}_{h-1})) + \frac{C}{2}\sum_{i=1}^{N_h} \mathbf{x}_i^T \boldsymbol{\Sigma} \mathbf{x}_i \,, \tag{1}$$

where $\mathrm{D}_{\mathrm{KL}} := \frac{1}{2}\log(\frac{\det \boldsymbol{\Sigma}_{h-1}}{\det \boldsymbol{\Sigma}}) + \frac{1}{2}\mathrm{Tr}(\boldsymbol{\Sigma}_{h-1}^{-1}\boldsymbol{\Sigma}) + \frac{1}{2}(\boldsymbol{\mu}_{h-1} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}_{h-1}^{-1}(\boldsymbol{\mu}_{h-1} - \boldsymbol{\mu}) - \frac{d}{2}$. Using KKT condition, we have

$$\boldsymbol{\Sigma}^{-1} = \boldsymbol{\Sigma}_{h-1}^{-1} + C\mathbf{X}_h\mathbf{X}_h^T \,. \tag{2}$$

Fix $\boldsymbol{\Sigma}$ and update $\boldsymbol{\mu}$. Once we get $\boldsymbol{\Sigma}$, we can learn $\boldsymbol{\mu}$ by the following problem:

$$\min_{\boldsymbol{\mu}} \frac{1}{2}(\boldsymbol{\mu} - \boldsymbol{\mu}_{h-1})^T \boldsymbol{\Sigma}_h^{-1}(\boldsymbol{\mu} - \boldsymbol{\mu}_{h-1}) + \frac{C}{q}\sum_{i=1}^{N_h} D_i\ell(\boldsymbol{\mu}; (\mathbf{x}_i, y_i))^q \,, \tag{3}$$

where $q = 1$ or 2.

Since $\boldsymbol{\Sigma}$ is positive semidefinite (PSD), it can be rewritten as $\boldsymbol{\Sigma} = \boldsymbol{\gamma}^2$. We introduce $\mathbf{w} := \boldsymbol{\gamma}^{-1}\boldsymbol{\mu}$, $\mathbf{w}_{h-1} := \boldsymbol{\gamma}^{-1}\boldsymbol{\mu}_{h-1}$ and $\widehat{\mathbf{x}}_i := \boldsymbol{\gamma}\mathbf{x}_i$, then problem (3) can be reformulated:

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w} - \mathbf{w}_{h-1}\|_2^2 + \frac{C}{q}\sum_{i=1}^{N_h} D_i\ell(\mathbf{w}; (\widehat{\mathbf{x}}_i, y_i))^q \,. \tag{4}$$

In order to solve problem (4), we assume an online setting, i.e., each example comes sequentially from $i = 1$ to N_h. This setting is similar to PA (Crammer et al. 2006). Thus we can come up with the solution as follows:

$$\mathbf{w} = \mathbf{w}_{h-1} + \tau_i y_i \mathbf{x}_i \,, \tag{5}$$

$$\tau_i = \min(\ell(\mathbf{w}; (\widehat{\mathbf{x}}_i, y_i))/\|\widehat{\mathbf{x}}_i\|_2^2, CD_i) \,, \text{ for } q = 1 \,; \tag{6}$$

$$\tau_i = \ell(\mathbf{w}; (\widehat{\mathbf{x}}_i, y_i))/(\|\widehat{\mathbf{x}}_i\|_2^2 + 1/(2CD_i)) \,, \text{ for } q = 2 \,. \tag{7}$$

4 Sparse CW for Feature Selection

4.1 Feature Selection by Sparsity Index $\boldsymbol{\eta}$

The proposed online-batch CW learning algorithm maintains the full covariance matrix $\boldsymbol{\Sigma}$. It is thus not appropriate for very high-dimensional data. In practice, high-dimensional data often exhibits the property of having many zero values and only a small number of features are relevant (Ma et al. 2009). Usually, only the relevant features and their interactions are significant for specific applications. Based on these observations, we propose the sparse feature selection algorithm in this section.

In order to find the most relevant features, we introduce an index vector $\boldsymbol{\eta} = \{0, 1\}^d$ and apply it to the feature vector $\mathbf{x}$ as $(\boldsymbol{\eta} \odot \mathbf{x})$. Here $\boldsymbol{\eta}_j = 1$ if feature j is selected and $\boldsymbol{\eta}_j = 0$ otherwise. In this situation, hinge loss is expressed as:

$$\ell(\boldsymbol{\mu}, \boldsymbol{\eta}; (\mathbf{x}_i, y_i)) = \max(0, 1 - y_i \boldsymbol{\mu}^T (\boldsymbol{\eta} \odot \mathbf{x}_i)) \,. \tag{8}$$

Thus $\ell_c(\boldsymbol{\mu}, \boldsymbol{\eta}; (\mathbf{x}_i, y_i)) = D_i \ell(\boldsymbol{\mu}, \boldsymbol{\eta}; (\mathbf{x}_i, y_i))$.

Considering our aim for feature selection, we impose an ℓ_0-norm constraint on $\boldsymbol{\eta}$ to induce the sparsity property, i.e., $\|\boldsymbol{\eta}\|_0 \leq r$ (where $r \ll d$). In convenience, let $\boldsymbol{\Lambda} := \{\boldsymbol{\eta} | \boldsymbol{\eta} \in \{0, 1\}^d, \|\boldsymbol{\eta}\|_0 \leq r\}$ be the set of all candidate $\boldsymbol{\eta}$. So there are $|\boldsymbol{\Lambda}| = \sum_{i=0}^{r} \binom{d}{i}$ feasible $\boldsymbol{\eta}$ in total, which is exponential. In the following, we will incorporate $\boldsymbol{\eta}$ into the online-batch CW learning and solve it gradually.

At first, as in Section 3.3, we assume $\boldsymbol{\mu}$ and $\boldsymbol{\eta}$ are given, and solve for $\boldsymbol{\Sigma}$. Accordingly, we incorporate $\boldsymbol{\eta}$ into equation (1):

$$\min_{\boldsymbol{\Sigma}} \mathrm{D}_{\mathrm{KL}}(\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \| \mathcal{N}(\boldsymbol{\mu}_{h-1}, \boldsymbol{\Sigma}_{h-1})) + \frac{C}{2} \sum_{i=1}^{N_h} (\boldsymbol{\eta} \odot \mathbf{x}_i)^T \boldsymbol{\Sigma} (\boldsymbol{\eta} \odot \mathbf{x_i}) \,. \tag{9}$$

Let $\mathbf{X}_h^r = \mathrm{diag}(\boldsymbol{\eta}) \mathbf{X}_h$. Applying the KKT condition on $\boldsymbol{\Sigma}$, the closed form solution is:

$$\boldsymbol{\Sigma}(\boldsymbol{\eta})^{-1} = \boldsymbol{\Sigma}_{h-1}^{-1} + C(\mathbf{X}_h^r)(\mathbf{X}_h^r)^T \,. \tag{10}$$

Once we have $\boldsymbol{\Sigma}(\boldsymbol{\eta})$, we incorporate $\boldsymbol{\eta}$ into formulation (3) and obtain the following problem:

$$\min_{\boldsymbol{\eta} \in \boldsymbol{\Lambda}} \min_{\boldsymbol{\mu}} \frac{1}{2} (\boldsymbol{\mu} - \boldsymbol{\mu}_{h-1})^T \boldsymbol{\Sigma}_h(\boldsymbol{\eta})^{-1} (\boldsymbol{\mu} - \boldsymbol{\mu}_{h-1}) + \frac{C}{q} \sum_{i=1}^{N_h} D_i \ell(\boldsymbol{\mu}, \boldsymbol{\eta}; (\mathbf{x_i}, y_i))^q \,, \tag{11}$$

where $q = 1$ or 2.

Problem (11) is a mixed integer problem including $\boldsymbol{\eta}$ and $\boldsymbol{\mu}$, which is hard to solve. Here, we employ the convex relaxation proposed in (Tan, Wang, and Tsang 2010) and apply the KKT condition to transform it into the dual form as a standard convex problem (detailed development can be found in Appendix A.1):

$$\max_{\theta \in \mathbb{R}, \boldsymbol{\alpha} \in \mathcal{A}} \theta \,, \quad \text{s.t. } \theta \leq f(\boldsymbol{\alpha}, \boldsymbol{\eta}) \,, \quad \forall \boldsymbol{\eta} \in \boldsymbol{\Lambda} \,. \tag{12}$$

Here, $f(\boldsymbol{\alpha}, \boldsymbol{\eta})$ is defined as: $f(\boldsymbol{\alpha}, \boldsymbol{\eta}) = -\frac{1}{2} \mathbf{g}(\boldsymbol{\alpha}, \boldsymbol{\eta})^T \boldsymbol{\Sigma}_h(\boldsymbol{\eta}) \mathbf{g}(\boldsymbol{\alpha}, \boldsymbol{\eta}) - (q-1) \frac{\tilde{\boldsymbol{\alpha}}^T \tilde{\boldsymbol{\alpha}}}{2C} + \sum_{i=1}^{N_h} \alpha_i - \boldsymbol{\mu}_{h-1}^T \mathbf{g}(\boldsymbol{\alpha}, \boldsymbol{\eta})$ where $\mathbf{g}(\boldsymbol{\alpha}, \boldsymbol{\eta}) := \sum_{i=1}^{N_h} \alpha_i y_i (\boldsymbol{\eta} \odot \mathbf{x}_i)$, $\boldsymbol{\alpha} \in \mathbb{R}^{N_h}$ is the dual variable with regard to equation (8), $\forall i \in [N_h]$ and $\mathcal{A} := \{\boldsymbol{\alpha} \in \mathbb{R}^{N_h} | 0 \leq \alpha_i \leq U\}$ is the domain of $\boldsymbol{\alpha}$ (here, $U = CD_i$ for $q = 1$ and $U = \infty$ for $q = 2$). At last, $\tilde{\boldsymbol{\alpha}} = [\alpha_1 / D_1^{1/2}, \ldots, \alpha_{N_h} / D_{N_h}^{1/2}]$.

4.2 Optimization

Problem (12) has exponential number of constraints as $\sum_{i=0}^{r} \binom{d}{i}$, making it difficult to directly solve. Fortunately, not all constraints in (12) are active at optimality. Alternatively, we can efficiently solve this problem by cutting plane algorithm (Kortanek and No 1993), which iteratively generate a pool of sparse feature subsets to constitute the constraints in (12).

Instead of considering all $T = \sum_{i=1}^{r} \binom{d}{i}$ constraints, we iteratively seek an active constraint until some stopping conditions are encountered. Given the previously estimated $\boldsymbol{\alpha}$, the most-violated constraint can be found by solving the following problem:

$$\begin{aligned} \boldsymbol{\eta}_t &= \arg\min_{\boldsymbol{\eta} \in \boldsymbol{\Lambda}} f(\boldsymbol{\alpha}, \boldsymbol{\eta}) \\ &= \arg\max_{\boldsymbol{\eta} \in \boldsymbol{\Lambda}} \mathbf{g}(\boldsymbol{\alpha}, \boldsymbol{\eta})^T \boldsymbol{\Sigma}_h(\boldsymbol{\eta}) \mathbf{g}(\boldsymbol{\alpha}, \boldsymbol{\eta}) + 2\boldsymbol{\mu}_{h-1}^T \mathbf{g}(\boldsymbol{\alpha}, \boldsymbol{\eta}) \,. \end{aligned} \tag{13}$$

Let $\mathbf{s} = \sum_{i=1}^{N_h} \alpha_i y_i \mathbf{x}_i$, then $\mathbf{g}(\boldsymbol{\alpha}, \boldsymbol{\eta}) = \boldsymbol{\eta} \odot \mathbf{s}$. Problem (13) can be reformulated:

$$\boldsymbol{\eta}_t = \arg\max_{\boldsymbol{\eta} \in \boldsymbol{\Lambda}} (\boldsymbol{s}^T \boldsymbol{\Sigma}_{\boldsymbol{h}}(\boldsymbol{\eta}) + 2\boldsymbol{\mu}_{h-1}^T)(\boldsymbol{\eta} \odot \boldsymbol{s}) \,. \tag{14}$$

Let $\mathbf{m} = (\boldsymbol{s}^T \boldsymbol{\Sigma}_h(\boldsymbol{\eta}) + 2\boldsymbol{\mu}_{h-1}^T) \odot \boldsymbol{s}$, then this problem can be solved by finding the r features with the largest score (e.g. m_j), and setting the corresponding η_j to 1 and the rest to 0. In other words, m_j measures the importance of the j-th feature and acts as the feature score.

After we obtained an active constraint $\boldsymbol{\eta}_t$, it can be added to the active set $\boldsymbol{\Lambda}_t = \boldsymbol{\Lambda}_{t-1} \cup \{\boldsymbol{\eta}_t\}$, then we can solve the following subproblem w.r.t constraints defined by $\boldsymbol{\Lambda}_t$:

$$\max_{\theta \in \mathbb{R}, \boldsymbol{\alpha} \in \mathcal{A}} \theta, \quad \text{s.t. } \theta \leq f(\boldsymbol{\alpha}, \boldsymbol{\eta}), \quad \forall \boldsymbol{\eta} \in \boldsymbol{\Lambda}_t \,. \tag{15}$$

Problem (14) and (15) are solved alternatively and stop when: (1) $|\theta_t - \theta_{t-1}|/|\theta_t| \leq \epsilon$, where ϵ is small tolerance value; (2) after $m = \lceil p/r \rceil$ iterations in order to choose p features.

4.3 Proximal Dual Coordinate Ascent for Subproblem (15)

Subproblem (15) regarding dual variable $\boldsymbol{\alpha}$ is hard and expensive to directly optimize. So in the following, we give a proximal-dual coordinate ascent based method to efficiently solve it. Let $K = |\boldsymbol{\Lambda}_t|$ be the number of active constraints. For each constraint $\boldsymbol{\eta}_k \in \boldsymbol{\Lambda}_t$, we take out the corresponding data, previous model parameter, model parameter and covariance matrix w.r.t $\boldsymbol{\eta}_k$ as: $\mathbf{x}_i^k \in \mathbb{R}^r$, $\boldsymbol{\mu}_k^{h-1} \in \mathbb{R}^r, \boldsymbol{\mu}_k \in \mathbb{R}^r$ and $\boldsymbol{\Sigma}_k \in \mathbb{R}^{r\times r}$. Furthermore, let $\boldsymbol{\gamma}_k$ be the square root of $\boldsymbol{\Sigma}_k$, $\mathbf{w}_k := \boldsymbol{\gamma}_k^{-1}\boldsymbol{\mu}_k, \mathbf{w}_k^{h-1} = \boldsymbol{\gamma}_k^{-1}\boldsymbol{\mu}_k^{h-1}, \widehat{\mathbf{x}}_i^k = \boldsymbol{\gamma}_k^T\mathbf{x}_i^k$. If we denote $\mathbf{w} = [\mathbf{w}_k]_{k=1}^K, \mathbf{w}_{h-1} = [\mathbf{w}_k^{h-1}]_{k=1}^K$ and $\widehat{\mathbf{x}}_i = [\widehat{\mathbf{x}}_i^k]_{k=1}^K$. The loss function $\ell(\mathbf{w}, \boldsymbol{\eta}; (\widehat{\mathbf{x}}_i, y_i)) = \max(0, 1 - \sum_{k=1}^K y_i \mathbf{w}_k^T \widehat{\mathbf{x}}_i^k) = \max(0, 1 - y_i \mathbf{w}^T \widehat{\mathbf{x}}_i)$.

The formulation of subproblem (15) is formally similar to the dual format of some problems. By using KKT condition, we can obtain the primal form of subproblem (15) (detailed development can be found in Appendix A.2):

$$\min_{\mathbf{w}} \frac{1}{2}(\sum_{k=1}^K \|\mathbf{w}_k - \mathbf{w}_k^{h-1}\|)^2 + \frac{C}{q}\sum_{i=1}^{N_h} D_i \ell(\mathbf{w}, \boldsymbol{\eta}; (\widehat{\mathbf{x}}_i, y_i))^q . \tag{16}$$

Problem (16) is non-smooth due to the $\ell_{2,1}^2$-norm regularizer. To make this problem tractable, we make some modifications and apply a proximal-dual coordinate ascent method (Shalev-Shwartz and Zhang 2014) to find a nearly accurate solution of (16) effectively. At first, we introduce a small regularization term $\frac{\sigma}{2}\|\mathbf{w} - \mathbf{w}_{h-1}\|^2$ (i.e., $\sigma \ll 1$) and address the following optimization problem:

$$\min_{\mathbf{w}} \frac{\sigma}{2}\|\mathbf{w} - \mathbf{w}_{h-1}\|^2 + \frac{1}{2}(\sum_{k=1}^K \|\mathbf{w}_k - \mathbf{w}_k^{h-1}\|)^2 + \frac{C}{q}\sum_{i=1}^{N_h} D_i \ell(\mathbf{w}, \boldsymbol{\eta}; (\widehat{\mathbf{x}}_i, y_i))^q . \tag{17}$$

Remark 1. If $\mathbf{w}^*$ is an $\frac{\epsilon}{2}$-accurate minimizer of (17) and the σ we are choosing is sufficiently small, then $\mathbf{w}^*$ is also an ϵ-accurate solution of (16) (Shalev-Shwartz and Zhang 2014). Therefore, the optimal values of problems (16) and (17) are very close.

Let $\Omega(\mathbf{w}) := \frac{\sigma}{2}\|\mathbf{w}-\mathbf{w}_{h-1}\|^2 + \frac{1}{2}(\sum_{k=1}^K \|\mathbf{w}_k - \mathbf{w}_k^{h-1}\|)^2$, and $L_i(\mathbf{w}^T\widehat{\mathbf{x}}_i) := \frac{1}{q} D_i \ell(\mathbf{w}, \boldsymbol{\eta}; (\widehat{\mathbf{x}}_i, y_i))^q$. Note here Ω is strongly convex and L_i is γ-Lipschitz for some $\gamma > 0$. Let $\Omega^*(\mathbf{z}) = \max_{\mathbf{w}} \mathbf{w}^T\mathbf{z} - \Omega(\mathbf{w})$ be the conjugate of $\Omega(\mathbf{w})$, and L_i^* be the conjugate of L_i. Then we can come up with the conjugate dual of problem (17) (detailed development can be found in Appendix A.3):

$$\max_{\boldsymbol{\alpha}\geq \mathbf{0}} H(\boldsymbol{\alpha}) , \tag{18}$$

where $H(\boldsymbol{\alpha}) = -\Omega^*(C\sum_{i=1}^{N_h} \alpha_i \widehat{\mathbf{x}}_i) - C\sum_{i=1}^{N_h} L_i^*(-\alpha_i)$.

Following (Shalev-Shwartz and Zhang 2014), we define $\mathbf{z}(\boldsymbol{\alpha}) = C\sum_{i=1}^{N_h} \alpha_i \widehat{\mathbf{x}}_i$, then $\mathbf{w}(\boldsymbol{\alpha}) = \nabla^*\Omega(\mathbf{z}(\boldsymbol{\alpha}))$. Here, $\nabla^*\Omega(\mathbf{z}(\boldsymbol{\alpha}))$ denotes the gradient of the conjugate of Ω. According to the property of conjugate, it is also the solution of $\Omega^*(\mathbf{z}) = \max_{\mathbf{w}} \mathbf{w}^T\mathbf{z} - \Omega(\mathbf{w})$.[1] Similarly, we assume an online setting as for problem (4). Finally, we give the full algorithm for solving the imbalanced feature selection problem in Algorithm 1.

Algorithm 1 Imbalanced sparse CW in online-batch manner

Require: Parameters $C > 0, H, r$
  Initialize $\boldsymbol{\alpha} = \frac{1}{N_h}\mathbf{1}, \boldsymbol{\mu}_0 = \mathbf{0}, \boldsymbol{\Sigma}_0 = \mathbf{I}$.
  for $h = 1 : H$ **do**
    Get a batch of data $\{\mathbf{X}_h, \mathbf{y}_h\}$,where $\mathbf{X}_h \in \mathbb{R}^{d\times N_h}$
    Compute $\boldsymbol{\Sigma}_h$ by (10) and $\boldsymbol{\gamma}$ by eigen-decomposition.
    Initialize $\Lambda_0 = \emptyset$ and $t = 1$
    while stopping conditions not meet **do**
      Find $\boldsymbol{\eta}_t$ by solving (14). Let $\Lambda_t = \Lambda_{t-1} \cup \boldsymbol{\eta}_t$.
      Compute $\mathbf{w}_k^{h-1}, \mathbf{w}_{h-1}$ according to Λ_t.
      Initialize $\mathbf{z} = \mathbf{0}, \mathbf{w} = \mathbf{w}_{h-1}$.
      for $i = 1 : N_h$ **do**
        Compute $D_i, \widehat{\mathbf{x}}_i^k$.
        Compute loss $\ell = \max(0, 1 - \sum_{k=1}^K y_i \mathbf{w}_k^T \widehat{\mathbf{x}}_i^k)$.
        if $\ell > 0$ **then**
          Compute $\alpha_i = \min(\ell/(C\|\widehat{\mathbf{x}}_i\|_2^2), D_i)$ for $q = 1$ or $\alpha_i = \ell/(C\|\widehat{\mathbf{x}}_i\|_2^2 + 0.5/D_i)$ for $q = 2$.
          Compute $\mathbf{z} = \mathbf{z} + C\alpha_i\widehat{\mathbf{x}}_i$.
          Compute $\mathbf{w} = \mathbf{w} + \nabla^*\Omega(\mathbf{z})$.
        end if
      end for
      Update $\mathbf{w}_h = \mathbf{w}, t = t + 1$.
    end while
    Update $\boldsymbol{\mu_h} = \boldsymbol{\gamma}\mathbf{w}_h$.
  end for

4.4 Discussions

We emphasize that the proposed algorithm enhances the theory of existing sparse CW (Tan et al. 2016; Wu et al. 2017) methods. First, with the introduction of cost-sensitive loss function in section 3.2, we can select features that better fit the imbalanced measures. Moreover, instead of fixing the cost, we adaptively choose the best cost from candidates and theoretically validate the optimality of our selection method in section 5. Second, in Equation (3) and (11), we employ $\boldsymbol{\mu}_{h-1}$ as the initialization when updating $\boldsymbol{\mu}$, while (Tan et al. 2016) uses $\mathbf{0}$. Thus $\boldsymbol{\mu}_{h-1}$ acts as the warm-start initialization and further influences on Equation (13) and (14) for solving $\boldsymbol{\eta}_t$. (Tan et al. 2016) assumes $\Sigma_h(\boldsymbol{\eta})$ to be an identity matrix when solving for $\boldsymbol{\eta}_t$. In fact, it is unclear if this assumption holds in practice. In contrast, we relax such assumption in Eq (13) and (14). Particularly, computing $\boldsymbol{\eta}_t$ reduces to a simple sorting problem in Eq (14). In intuition, our method takes more advantages of the information from the previous online-batch through $\boldsymbol{\mu}_{h-1}$ and $\boldsymbol{\Sigma}_h(\boldsymbol{\eta})$.

[1] This problem can be efficiently solved using Algorithm2 of (Martins et al. 2011). Due to space limitation, we give the detailed algorithm in Appendix A.4.

Algorithm 2 Multiple Cost-Sensitive Learning.

Require: the number of models K
  Initialize $M_1^j = 0, \boldsymbol{\mu}_1^j = 0, \boldsymbol{\Sigma}_1^j = \mathbf{I}, \forall j \in [K]$.
  for $h = 1 : H$ **do**
    Get a batch of data $\{\mathbf{X}_h, \mathbf{y}_h\}$,where $\mathbf{X}_h \in \mathbb{R}^{d \times N_h}$
    Let $k = \arg\max_{j=1,\ldots,K} M_h^j$.
    Sample a model $\boldsymbol{\mu}_h^* = \boldsymbol{\mu}_h^k$.
    Predict for a batch of data $\mathbf{f}_h = \text{sign}((\boldsymbol{\mu}_h^*)^T \mathbf{X}_h)$
    for $j = 1, \ldots, K$ **do**
      Update model $\boldsymbol{\mu}_h^j$ and $\boldsymbol{\Sigma}_h^j$ by running Algorithm 1.
      Compute M_{h+1}^j according to M_h^j, $\boldsymbol{f}_h$, and $\mathbf{y}_h$.
    end for
  end for

5 Multiple Cost-Sensitive Learning

In section 3 and section 4, we propose the cost-sensitive sparse CW algorithm. However, how to decide the value of c_+ and c_- remains an issue. Some previous works use ad-hoc approaches to set up the values (Wang, Zhao, and Hoi 2014; Sahoo, Hoi, and Zhao 2016; Zhao and Hoi 2013). However, there is no guarantee that these approaches can achieve optimal performance for various imbalanced measures such as F-measure, AUPRC, and AUROC.

To solve this problem, we propose a strategy which maintains multiple cost-sensitive vectors. The motivation is that if multiple cost vectors $\mathbf{c} = (c_+, c_-)$ is tracked and maintained simultaneously, there must exist one setting that can best fit the data. For convenience, we assume $c_+ + c_- = 1$ to eliminate one parameter and thus $c_+ \in (0, 1)$. In order to maintain the multiple c_+, we divide $(0, 1)$ into K evenly distributed values $\theta_1, \ldots, \theta_K$, i.e., $\theta_j = j/(K+1)$ and set $c_+^j = 1 - \theta_j/2$, then the cost-sensitive loss is denoted as:

$$\ell_c^j(\boldsymbol{\mu}_j; (\mathbf{x}_i, y_i)) = (1 - \theta_j/2)\mathbb{I}(y_i = 1)\ell(\boldsymbol{\mu}_j; (\mathbf{x}_i, y_i)) + (\theta_j/2)\mathbb{I}(y_i = -1)\ell(\boldsymbol{\mu}_j; (\mathbf{x}_i, y_i)) . \quad (19)$$

With this strategy, we can maintain and track K learners with the corresponding costs simultaneously: $(\theta_1, \boldsymbol{\mu}_1), \ldots, (\theta_K, \boldsymbol{\mu}_K)$. At the h-th online-batch, we update the current target measure of the j-th learner, denoted by M_h^j. Different from (Yan et al. 2017), we apply the greedy criterion to select the best performer according to $\{M_h^1, ..., M_h^K\}$ from K candidates for prediction at the h-th online-batch. With this criterion, we do not need to introduce extra hyper-parameter, and we can analyze the performance guarantee in a different way.

We update the target measures (e.g., F-measure, AUROC, and AUPRC) only using the current measure M_h^j, current predictions $\boldsymbol{f}_h$, and labels $\mathbf{y}_h$, which is efficient without storing all $\boldsymbol{f}_h$ and $\mathbf{y}_h$. Due to the space limitation, we put the detailed updating formulations in Appendix A.5. We summarize the multiple cost-sensitive algorithm in Algorithm 2.

5.1 Theoretical Analysis in F-measure

we define the following notations for binary classification:

$\boldsymbol{a}(\theta) = [1 - \frac{\theta}{2}, \frac{\theta}{2}]$ and $\Delta = \frac{\theta_j - \theta_{j+1}}{2} = \frac{1}{2K}$,
P_1 : the marginal probability of the positive instances ,
$\boldsymbol{E}(h) = [\texttt{fn}, \texttt{fp}]$: false negative and false positive ,
$F^* = \max_{\boldsymbol{e}} F(\boldsymbol{e})$: the maximum F-measure ,
$F(\boldsymbol{\mu})$: the F-measure achieved by $\boldsymbol{\mu}$.

Proposition 1. Assume that $\{\boldsymbol{\mu}_h^1, ..., \boldsymbol{\mu}_h^K\}$ minimizes the cost-sensitive loss to a certain degree, then the F-measure achieved by Algorithm 2 has the following lower bound as long as h increases:

$$\max_{j=1,\ldots,K} F(\boldsymbol{\mu}_h^j) \geq F^* - \Delta - \frac{\epsilon_0}{P_1} ,$$

where $k = \arg\max_{j=1,\ldots,K} F(\boldsymbol{\mu}_h^j)$ and $\langle \boldsymbol{a}(\theta_k), \boldsymbol{E}(\boldsymbol{\mu}_h^k) \rangle \leq \min_{\boldsymbol{\mu}} \langle \boldsymbol{a}(\theta_k), \boldsymbol{E}(\boldsymbol{\mu}) \rangle + \epsilon_0$. The full proof is in Appendix A.6.

6 Experiments

In this section, we evaluate the proposed ASCW algorithm on three imbalanced measures, i.e., F-measure, AUROC, and AUPRC and compare with various online learning and feature selection methods.

6.1 Experimental Testbed

We conduct experiments on three widely-used high-dimensional benchmarks and sample with different ratios to construct nine imbalance configurations, as shown in Table 1. In order to construct imbalanced configurations from the original datasets, we adopt two strategies. Firstly, for binary datasets (real-sim and rcv1), we fix the negative class and sample from the positive class to satisfy specific ratios (1:5, 1:10 and 1:20). Secondly, for the multi-class dataset (news20), we set class1 as positive class and select class2-6, class2-11 and class2-20 as negative class respectively.

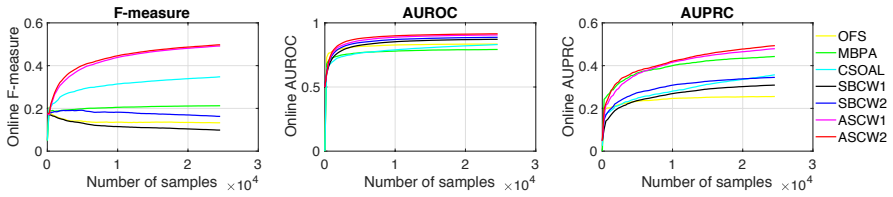
Table 1: Datasets Statistics

Datasets	d	N_{train}	# nonzeros per example	#Pos:#Neg
real-sim	20,958	32,309	52	1:5, 1:10, 1:20
rcv1	47,236	20,242	74	1:5, 1:10, 1:20
news20	62,061	15,935	80	1:5, 1:10, 1:19

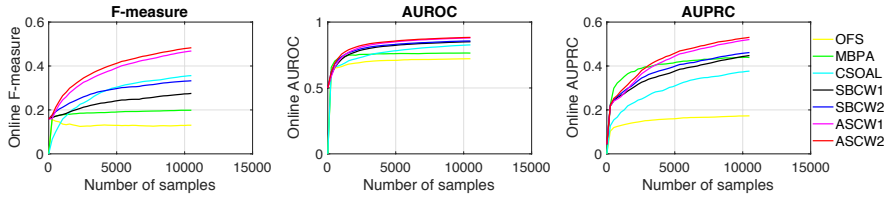
6.2 Comparison Algorithms

We compare the following algorithms:

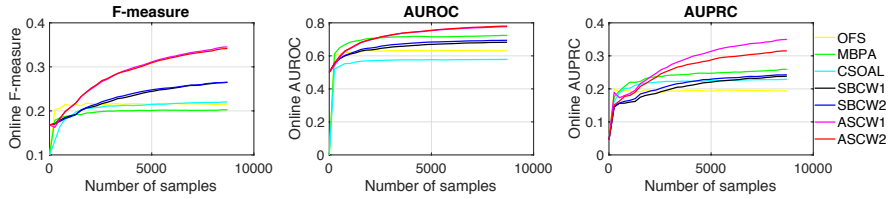
- OFS (Wang et al. 2014): The state-of-the-art first-order online feature selection via sparse projection.
- MBPA (Han et al. 2016): Margin-based passive aggressive method for online feature selection.
- CSOAL (Zhao and Hoi 2013): A cost-sensitive online active learning method.



(a) real-sim



(b) rcv1



(c) news20

Figure 1: Online performance with imbalance ration 1:10 (other ratios in Appendix A.7) for different performance measures

- SBCW1 and SBCW2 (Tan et al. 2016): Two variations of the sparse online-batch feature selection method.
- FGM (Tan, Tsang, and Wang 2014): The full-batch high-dimensional feature selection method which generates a pool of violated sparse feature subsets and combines them via efficient Multiple Kernel Learning (MKL) algorithm.
- L1SVM (Yuan et al. 2010): ℓ_1-norm SVM by Liblinear.
- ASCW1 and ASCW2: The proposed algorithm with hinge ($q = 1$) and squared hinge ($q = 2$) loss.

SBCW and ASCW consider second-order structure while others only optimize first-order information.

6.3 Experimental Results

As shown in Table 1, the number of nonzeros per example varies from 52 to 80 in different datasets, so in the experiments we set the selected feature dimension to 50 for all algorithms except that for CSOAL we set query ratio to be 1%.[2] Following (Hoi, Wang, and Zhao 2014), we repeat all online learning experiments 20 times with random permutation of training data. For full batch methods (FGM, L1SVM), we follow the default settings.

Batch Size. In Algorithm 1, $\boldsymbol{\mu}$ is updated in a pure online manner and $\boldsymbol{\Sigma}$ is updated in an online-batch manner. To explain the necessity of the online-batch update and explore proper batch size, we perform experiments on news20 with various batch sizes, as shown in Table 2. The best performance is achieved with batch size=1 (the strict online case).

[2]Actually, 1% of all examples contain more information than 50 features of entire features for all datasets.

Table 2: Test performance on news20 with various batch sizes

Ratio	Batch	news20			
		F	AUROC	AUPRC	Time (s)
1:5	256	0.5614	0.8474	0.6011	0.71
	64	0.5640	0.8536	0.6163	2.40
	1	0.9125	0.9922	0.9769	913.86
1:10	256	0.4239	0.8114	0.4306	1.39
	64	0.4275	0.8096	0.4309	4.53
	1	0.8549	0.9872	0.9479	3034.95
1:19	256	0.3349	0.8223	0.3126	2.86
	64	0.3031	0.7952	0.2867	9.63
	1	0.8080	0.9796	0.9068	13950.78

However, the time cost is unbearable. The performance of batch size=256 is close to that of 64, but 256 is 3∼4 times faster. We thus set batch size=256 in remaining experiments.

Online Performance. To compare the online performances, we evaluate three measures on all datasets. The results of ratio 1:10 are shown in Figure 1. It can be seen that ASCW outperforms all other methods when the number of samples increases. Moreover, the F-measure of ASCW outperforms all other methods with a large margin.

Test Performance. We report the test performances of all algorithms under different imbalance ratios in Table 3. It is observed that ASCW outperforms all other algorithms on most settings for the three performance measures. Also, the improvements of ASCW on F-measure are higher than that on AUROC and AUPRC.

We attribute the good online and test performance of

Table 3: Average test performance over models trained on 20 random data permutations

Ratio	Methods	real-sim			rcv1			news20		
		F-measure	AUROC	AUPRC	F-measure	AUROC	AUPRC	F-measure	AUROC	AUPRC
1:5	OFS	0.0279	0.8943	0.5707	0.0215	0.8207	0.5212	0.3744	0.7706	0.4919
	MBPA	0.3742	0.8435	0.6886	0.3404	0.8365	0.6646	0.3193	0.7239	0.3930
	CSOAL	0.6248	0.9163	0.6868	**0.6366**	0.9183	0.7203	0.3579	0.5915	0.3597
	FGM	0.5334	0.9103	0.7366	0.4115	0.7235	0.4381	0.2754	0.5930	0.2411
	L1SVM	0.4501	0.9127	0.6892	0.5552	0.8906	0.7308	0.4135	0.8028	0.5664
	SBCW1	0.3778	0.9295	0.7161	0.4156	0.8913	0.7083	0.4115	0.7934	0.5298
	SBCW2	0.4363	0.9390	0.7357	0.4078	0.9056	0.7255	0.4703	0.8122	0.5474
	ASCW1	**0.7036**	**0.9434**	**0.7395**	**0.6409**	**0.9185**	**0.7398**	**0.5614**	**0.8474**	**0.6011**
	ASCW2	**0.6948**	**0.9464**	**0.7521**	0.6355	**0.9312**	**0.7887**	**0.5698**	**0.8529**	**0.6095**
1:10	OFS	0.0021	0.8537	0.2964	0.0001	0.7794	0.2480	0.2237	0.7139	0.2928
	MBPA	0.2394	0.8203	0.5564	0.2085	0.7505	0.4509	0.2096	0.7203	0.2618
	CSOAL	0.3931	0.8801	0.4528	0.4342	0.8869	0.5051	0.2526	0.6013	0.2567
	FGM	0.3580	0.9071	**0.6279**	0.2968	0.7565	0.3432	0.1942	0.6078	0.1450
	L1SVM	0.0816	0.8655	0.3473	0.2830	0.8485	0.4968	0.3986	0.7636	0.4056
	SBCW1	0.0710	0.8994	0.3778	0.3510	0.8767	0.5557	0.3193	0.7198	0.2895
	SBCW2	0.1241	0.9173	0.4273	0.3906	0.8990	**0.6016**	0.3063	0.7235	0.2821
	ASCW1	**0.5621**	**0.9422**	0.5914	**0.5649**	**0.9060**	0.5880	**0.4239**	**0.8114**	**0.4306**
	ASCW2	**0.5662**	**0.9457**	**0.6073**	**0.6005**	**0.9190**	**0.6252**	**0.4312**	**0.8188**	**0.4341**
1:20 (1:19)	OFS	0.0041	0.8295	0.1559	0.0000	0.7623	0.1191	0.1525	0.7160	0.1969
	MBPA	0.1144	0.7578	0.3941	0.1231	0.7129	0.3170	0.1283	0.7419	0.2017
	CSOAL	0.2128	0.8156	0.1776	0.2386	0.8512	0.2856	0.2570	0.6519	0.2358
	FGM	0.1827	0.8660	0.4540	0.1761	0.7224	0.2055	0.1622	0.6981	0.1282
	L1SVM	0.0000	0.8223	0.1264	0.0000	0.8310	0.2264	0.3154	0.7499	**0.2991**
	SBCW1	0.0554	0.8678	0.1947	0.2422	0.8508	0.3668	0.2538	0.7328	0.2161
	SBCW2	0.0802	0.9069	0.2466	0.2857	0.8726	0.4404	0.2455	0.7427	0.1964
	ASCW1	**0.3893**	**0.9178**	**0.4803**	**0.4565**	**0.9042**	**0.5189**	**0.3349**	**0.8223**	**0.3126**
	ASCW2	**0.4236**	**0.9372**	**0.4582**	**0.5081**	**0.9078**	**0.5073**	**0.3245**	**0.8173**	0.2937

Table 4: Average estimated error of cost $\hat{c}_+$ by Algorithm 2 and optimal cost c_+^*.

Ratio	Methods	real-sim			rcv1			news20		
		F-measure	AUROC	AUPRC	F-measure	AUROC	AUPRC	F-measure	AUROC	AUPRC
1:5	ASCW1	0.0008	0.0000	0.0000	0.0090	0.0032	0.0021	0.0073	0.0058	0.0140
	ASCW2	0.0056	0.0031	0.0078	0.0082	0.0208	0.0345	0.0198	0.0020	0.0103
1:10	ASCW1	0.0000	0.0006	0.0014	0.0014	0.0084	0.0201	0.0006	0.0017	0.0119
	ASCW2	0.0071	0.0006	0.0040	0.0111	0.0235	0.0117	0.0033	0.0069	0.0233
1:20 (1:19)	ASCW1	0.0000	0.0000	0.0000	0.0542	0.0076	0.0354	0.0069	0.0024	0.0089
	ASCW2	0.0000	0.0023	0.0011	0.0357	0.0345	0.0501	0.0107	0.0019	0.0148

ASCW to two main reasons. First, our algorithm is capable of selecting a close-to-optimal cost vector $[c_+, c_-]$, which makes it perform better on imbalanced measures. Moreover, there is a theoretical guarantee on the lower bound of F-measure. It explains the higher improvements of F-measure compared with AURROC and AUPRC. Second, our algorithm employs covariance structure that can better capture the interplays among features to find more effective features.

6.4 Optimal Cost Vector

In preposition 1, we theoretically analyze the lower-bound of the F-measure achieved by Algorithm 2. In order to quantitatively verify that our algorithm can choose near to optimal cost vector $[c_+, c_-]$, we perform cost-sensitive feature selection by Algorithm 1 with costs vary among $c_+ = \{0.55, 0.60, \ldots, 0.95\}$ and choose the best cost according to overall online performance, denoted by c_+^*. To compare our selected cost with the best performance cost, we average c_+ sampled in Algorithm 2 in the last 20 iterations (>5000 examples) as an estimation of the best cost, denoted as $\hat{c}_+$. Then we compute the estimated errors as: $|c_+^* - \hat{c}_+|$ and present the results on Table 4. We can observe that the estimated errors of our algorithm and the optimal one is very close with the search length of 0.05, thus verifying the accurate estimation of our algorithm for the optimal cost.

7 Conclusion

Many real-world applications process data in an online-batch manner and suffer from the skewed distribution. In this paper, we propose an adaptive sparse CW algorithm to deal with the feature selection problem on imbalanced online-batch data. Our algorithm simultaneously learns multiple base classifiers with their own costs. With the data comes sequentially in each online-batch, the aimed measure is updated incrementally for each classifier in an online-batch manner. Among all the classifiers, we choose the one with

the best performance for prediction. We theoretically enhance the theory of the existing sparse CW feature selection algorithm and analyze the performance behavior regarding F-measure. Experimental results show the superior performance of ASCW and its ability for selecting the satisfactory cost vector.

Acknowledgments

Yanbin Liu, Yan Yan, Ling Chen and Yi Yang are in part supported by Data to Decisions CRC (D2D CRC). Yahong Han is in part supported by the NSFC (under Grant U1509206, 61472276, 61876130).

References

Crammer, K.; Dekel, O.; Keshet, J.; Shalev-Shwartz, S.; and Singer, Y. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research* 7(Mar):551–585.

Crammer, K.; Dredze, M.; and Pereira, F. 2009. Exact convex confidence-weighted learning. In *Advances in Neural Information Processing Systems*, 345–352.

Crammer, K.; Kulesza, A.; and Dredze, M. 2009. Adaptive regularization of weight vectors. In *Advances in neural information processing systems*, 414–422.

Dong, X.; Yan, Y.; Tan, M.; Yang, Y.; and Tsang, I. W. 2019. Late fusion via subspace search with consistency preservation. *IEEE Transactions on Image Processing* 28(1):518–528.

Han, C.; Tan, Y.-K.; Zhu, J.-H.; Guo, Y.; Chen, J.; and Wu, Q.-Y. 2016. Online feature selection of class imbalance via pa algorithm. *Journal of Computer Science and Technology* 31(4):673–682.

Hoi, S. C.; Wang, J.; and Zhao, P. 2014. Libol: A library for online learning algorithms. *The Journal of Machine Learning Research* 15(1):495–499.

Kortanek, K. O., and No, H. 1993. A central cutting plane algorithm for convex semi-infinite programming problems. *SIAM Journal on optimization* 3(4):901–918.

Ma, J.; Saul, L. K.; Savage, S.; and Voelker, G. M. 2009. Identifying suspicious urls: an application of large-scale online learning. In *Proceedings of the 26th annual international conference on machine learning*, 681–688. ACM.

Ma, J.; Kulesza, A.; Dredze, M.; Crammer, K.; Saul, L.; and Pereira, F. 2010. Exploiting feature covariance in high-dimensional online learning. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 493–500.

Martins, A. F. T.; Smith, N.; Xing, E.; Aguiar, P.; and Figueiredo, M. 2011. Online learning of structured predictors with multiple kernels. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 507–515.

Rosenblatt, F. 1958. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review* 65(6):386.

Sahoo, D.; Hoi, S.; and Zhao, P. 2016. Cost sensitive online multiple kernel classification. In *Asian Conference on Machine Learning*, 65–80.

Shalev-Shwartz, S., and Zhang, T. 2014. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. In *International Conference on Machine Learning*, 64–72.

Sumita, H.; Kawase, Y.; Fujita, S.; Fukunaga, T.; and Center, R. A. 2017. Online optimization of video-ad allocation. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 423–429. AAAI Press.

Tan, M.; Yan, Y.; Wang, L.; Van Den Hengel, A.; Tsang, I. W.; and Shi, Q. J. 2016. Learning sparse confidence-weighted classifier on very high dimensional data. In *AAAI*, 2080–2086.

Tan, M.; Tsang, I. W.; and Wang, L. 2014. Towards ultrahigh dimensional feature selection for big data. *The Journal of Machine Learning Research* 15(1):1371–1429.

Tan, M.; Wang, L.; and Tsang, I. W. 2010. Learning sparse svm for feature selection on very high dimensional datasets. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, 1047–1054.

Wang, J.; Zhao, P.; Hoi, S. C.; and Jin, R. 2014. Online feature selection and its applications. *IEEE Transactions on Knowledge and Data Engineering* 26(3):698–710.

Wang, J.; Zhao, P.; and Hoi, S. C. 2012. Exact soft confidence-weighted learning. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, 107–114. Omnipress.

Wang, J.; Zhao, P.; and Hoi, S. C. 2014. Cost-sensitive online classification. *IEEE Transactions on Knowledge and Data Engineering* 26(10):2425–2438.

Wu, X.; Yu, K.; Ding, W.; Wang, H.; and Zhu, X. 2013. Online feature selection with streaming features. *IEEE transactions on pattern analysis and machine intelligence* 35(5):1178–1192.

Wu, Y.; Hoi, S. C.; Mei, T.; and Yu, N. 2017. Large-scale online feature selection for ultra-high dimensional sparse data. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 11(4):48.

Yan, Y.; Yang, T.; Yang, Y.; and Chen, J. 2017. A framework of online learning with imbalanced streaming data. In *AAAI*, 2817–2823.

Yu, K.; Wu, X.; Ding, W.; and Pei, J. 2014. Towards scalable and accurate online feature selection for big data. In *Data Mining (ICDM), 2014 IEEE International Conference on*, 660–669. IEEE.

Yuan, G.-X.; Chang, K.-W.; Hsieh, C.-J.; and Lin, C.-J. 2010. A comparison of optimization methods and software for large-scale l1-regularized linear classification. *Journal of Machine Learning Research* 11(Nov):3183–3234.

Zhao, P., and Hoi, S. C. 2013. Cost-sensitive online active learning with application to malicious url detection. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 919–927. ACM.

Zhao, P.; Zhang, Y.; Wu, M.; Hoi, S. C.; Tan, M.; and Huang, J. 2018. Adaptive cost-sensitive online classification. *IEEE Transactions on Knowledge and Data Engineering*.

Zhou, P.; Hu, X.; Li, P.; and Wu, X. 2017. Online feature selection for high-dimensional class-imbalanced data. *Knowledge-Based Systems* 136:187–199.

Zinkevich, M. 2003. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, 928–936.