

9-2016

# Soft confidence-weighted learning


Jialei WANG  
*University of Chicago*

Peilin ZHAO  
*Institute for Infocomm Research, Singapore*

HOI, Steven C. H.  
*Singapore Management University, CHHOI@smu.edu.sg*

**DOI:** <https://doi.org/10.1145/2932193>

Follow this and additional works at: [https://ink.library.smu.edu.sg/sis\\_research](https://ink.library.smu.edu.sg/sis_research)

 Part of the [Databases and Information Systems Commons](#), and the [Theory and Algorithms Commons](#)

---

## Citation

WANG, Jialei; ZHAO, Peilin; and HOI, Steven C. H.. Soft confidence-weighted learning. (2016). *ACM Transactions on Intelligent Systems and Technology*. 8, (1), 1-32. Research Collection School Of Information Systems.

**Available at:** [https://ink.library.smu.edu.sg/sis\\_research/3418](https://ink.library.smu.edu.sg/sis_research/3418)

This Journal Article is brought to you for free and open access by the School of Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email [libIR@smu.edu.sg](mailto:libIR@smu.edu.sg).

## Soft Confidence-Weighted Learning

JIALEI WANG, Department of Computer Science, University of Chicago, USA

PEILIN ZHAO, Institute for Infocomm Research, A\*STAR, Singapore

STEVEN C. H. HOI, School of Information Systems, Singapore Management University, Singapore

Online learning plays an important role in many big data mining problems because of its high efficiency and scalability. In the literature, many online learning algorithms using gradient information have been applied to solve online classification problems. Recently, more effective second-order algorithms have been proposed, where the correlation between the features is utilized to improve the learning efficiency. Among them, Confidence-Weighted (CW) learning algorithms are very effective, which assume that the classification model is drawn from a Gaussian distribution, which enables the model to be effectively updated with the second-order information of the data stream. Despite being studied actively, these CW algorithms cannot handle nonseparable datasets and noisy datasets very well. In this article, we propose a family of Soft Confidence-Weighted (SCW) learning algorithms for both binary classification and multiclass classification tasks, which is the first family of online classification algorithms that enjoys four salient properties simultaneously: (1) large margin training, (2) confidence weighting, (3) capability to handle nonseparable data, and (4) adaptive margin. Our experimental results show that the proposed SCW algorithms significantly outperform the original CW algorithm. When comparing with a variety of state-of-the-art algorithms (including AROW, NAROW, and NHERD), we found that SCW in general achieves better or at least comparable predictive performance, but enjoys considerably better efficiency advantage (i.e., using a smaller number of updates and lower time cost). To facilitate future research, we release all the datasets and source code to the public at <http://libol.stevenhoi.org/>.

CCS Concepts: • **Theory of computation** → **Online learning algorithms**; • **Computing methodologies** → **Supervised learning by classification**; **Online learning settings**;

Additional Key Words and Phrases: Confidence weighted, second-order algorithms, binary classification, multiclass classification

### ACM Reference Format:

Jialei Wang, Peilin Zhao, and Steven C. H. Hoi. 2016. Soft confidence-weighted learning. *ACM Trans. Intell. Syst. Technol.* 8, 1, Article 15 (September 2016), 32 pages.

DOI: <http://dx.doi.org/10.1145/2932193>

## 1. INTRODUCTION

Online learning algorithms [Hoi et al. 2014; Rosenblatt 1958a; Crammer et al. 2006] represent a family of fast and simple machine-learning techniques, which usually make few statistical assumptions and can be applied to a wide range of applications, including malicious URL detection [Zhao and Hoi 2013], anomaly detection [Wang et al. 2014a],

---

This work was supported in part by the Singapore Ministry of Education (MOE) Academic Research Fund (AcRF) Tier 1 Research Grant (C220/MSS14C003) and Microsoft Research Grant.

Authors' addresses: J. Wang, Department of Computer Science, University of Chicago, USA; email: [jialei@uchicago.edu](mailto:jialei@uchicago.edu); P. Zhao, Institute for Infocomm Research, A\*STAR, Singapore; email: [peilinzhao@hotmail.com](mailto:peilinzhao@hotmail.com); S. C. H. Hoi (corresponding author), School of Information Systems, Singapore Management University, Singapore; email: [chhoi@smu.edu.sg](mailto:chhoi@smu.edu.sg).

image retrieval [Wu et al. 2013; Xia et al. 2014], portfolio selection [Li et al. 2012; Li and Hoi 2014], learning to rank [Wang et al. 2015], collaborative filtering [Wang et al. 2013; Lu et al. 2013], and so forth. Typically, an online learner processes one instance at a time and makes simple updates with each incoming example repeatedly. As a result, online algorithms are not only more efficient and scalable but also able to avoid expensive retraining cost when handling new training data, making them more favorite choices for solving large-scale machine-learning tasks toward big data applications.

Online learning has been actively studied in the machine-learning community [Rosenblatt 1958b; Crammer and Singer 2003; Cesa-Bianchi et al. 2004; Crammer et al. 2006; Fink et al. 2006; Zhao et al. 2011a, 2011b; Hoi et al. 2013; Wang et al. 2014b; Zhao et al. 2014], in which a variety of online learning algorithms have been proposed, including a number of first-order algorithms [Rosenblatt 1958a; Crammer et al. 2006]. One of the most popular first-order online approaches is the well-known Perceptron algorithm [Rosenblatt 1958b; Freund and Schapire 1999]. Recently a number of online learning algorithms have been developed based on the criterion of maximum margin [Crammer and Singer 2003; Gentile 2001; Kivinen et al. 2001; Crammer et al. 2006; Li and Long 1999]. One example is the Relaxed Online Maximum Margin algorithm (ROMMA) [Li and Long 1999], which repeatedly chooses the hyperplanes that correctly classify the existing training examples with a large margin. Another representative example is the Passive-Aggressive (PA) algorithm [Crammer et al. 2006]. It updates the classification function when a new example is misclassified or its classification score does not exceed the predefined margin. Empirical studies showed that the maximum margin-based online learning algorithms are generally more effective than the Perceptron algorithm. Despite the difference, these online learning algorithms only update the algorithm based on the first-order information, such as the gradient of the loss. This constraint could significantly limit the performance of online learning.

Recent years have seen a surge of studies on second-order online learning algorithms [Cesa-Bianchi et al. 2005; Dredze et al. 2008; Crammer et al. 2009; Orabona and Crammer 2010; Duchi et al. 2011], which have shown that parameter confidence information can be explored to guide and improve online learning performance [Cesa-Bianchi et al. 2005]. For example, Second-Order Perceptron (SOP) [Cesa-Bianchi et al. 2005] is the first second-order online learning algorithm that can be viewed as an online variant of the whitened Perceptron algorithm by exploiting online correlation matrices of previously seen instances. Later, other second-order online learning algorithms have been proposed. For example, Confidence-Weighted (CW) learning [Dredze et al. 2008; Crammer et al. 2009] maintains a Gaussian distribution over some linear classifier hypotheses and applies it to control the direction and scale of parameter updates [Dredze et al. 2008]. Although CW has formal guarantees in the mistake-bound model [Crammer et al. 2008], it can overfit in certain situations due to its aggressive update rules based on a separable data assumption. Recently, an improved online algorithm, that is, Adaptive Regularization of Weights (AROW) [Crammer et al. 2009; Orabona and Crammer 2010], relaxes such separable assumption by employing an adaptive regularization for each training example based on its current confidence. This regularization comes in the form of minimizing a combination of the Kullback-Leibler divergence between Gaussian distributed weight vectors and a confidence penalty of vectors.

Although AROW [Crammer et al. 2009] is able to improve the original CW [Crammer et al. 2008] learning by handling noisy and nonseparable cases, it is not the exact corresponding soft extending part of CW (like PA with PA-I and PA-II). In particular, the directly added loss and confidence regularization makes AROW lose an important property of Confidence-Weighted learning, that is, Adaptive Margin property [Crammer

et al. 2008]. Following the similar idea of soft margin support vector machines, the adaptive margin assigns different margins for different instances via a probability formulation, which enables CW to gain extra efficiency and effectiveness.

In this work, we extend the confidence-weighted learning for soft margin learning, which makes our Soft Confidence-Weighted (SCW) learning method more robust than the original CW learning when handling noisy and nonseparable data, and more effective and efficient than the state-of-the-art AROW algorithm. We also extend Soft Confidence-Weighted learning algorithms for solving online multiclass classification problems. Finally, an extensive set of experiments show that SCW in general achieves better or at least comparable predictive performance compared with a variety of state-of-the-art algorithms (including AROW, NAROW, and NHERD) but enjoys a considerably better efficiency advantage (i.e., using a smaller number of updates and lower time cost).

The rest of this article is organized as follows. Section 2 proposes Soft Confidence-Weighted (SCW) learning methods for online binary classification setting. Section 3 presents soft confidence-weighted learning methods for an online multiclass classification setting. Section 4 analyzes the mistake bounds and properties of our algorithms. Section 5 conducts an extensive set of empirical experiments on a large set of datasets, and Section 6 concludes this work. Finally, we note that a short conference version of this work appeared in the International Conference on Machine Learning [Wang et al. 2012].

## 2. SOFT CONFIDENCE-WEIGHTED LEARNING FOR BINARY CLASSIFICATION

### 2.1. Overview of Online Binary Classification

Online learning operates on a sequence of data examples with timestamps. At time step  $t$ , the algorithm processes an incoming example  $\mathbf{x}_t \in \mathbb{R}^d$  by first predicting its label  $\hat{y}_t \in \{-1, +1\}$ . After the prediction, the true label  $y_t \in \{-1, +1\}$  is revealed and then the loss  $\ell(y_t, \hat{y}_t)$ , which is the difference between its prediction and the revealed true label  $y_t$ , is suffered. Finally, the loss is used to update the weights of the model based on some criterion. Overall, the goal of online learning is to minimize the cumulative mistake over the entire sequence of data examples.

Our work is closely related to several first- and second-order online learning algorithms, including Passive-Aggressive learning [Crammer et al. 2006], Confidence-Weighted learning [Dredze et al. 2008], and Adaptive Regularization of Weights learning [Crammer et al. 2009]. Next we review the basics of these algorithms.

### 2.2. Passive-Aggressive Learning

As the state-of-the-art first-order online learning algorithm, the optimization of Passive-Aggressive (PA) learning is formulated as

$$\mathbf{w}_{t+1} = \arg \min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{2} \|\mathbf{w} - \mathbf{w}_t\|^2, \quad s.t. \ell(\mathbf{w}; (\mathbf{x}_t, y_t)) = 0, \quad (1)$$

where the loss function is based on the hinge loss:

$$\ell(\mathbf{w}; (\mathbf{x}_t, y_t)) = \begin{cases} 0 & \text{if } y_t(\mathbf{w} \cdot \mathbf{x}_t) \geq 1 \\ 1 - y_t(\mathbf{w} \cdot \mathbf{x}_t) & \text{otherwise.} \end{cases}$$

The previous optimization has the closed-form solution

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \eta_t^{\text{PA}} y_t \mathbf{x}_t, \quad (2)$$

where  $\eta_t^{\text{PA}} = \frac{\ell(\mathbf{w}_t; (\mathbf{x}_t, y_t))}{\|\mathbf{x}_t\|^2}$ . Further, to let PA be able to handle nonseparable instances and be more robust, a slack variable  $\xi$  was introduced into the optimization (Equation (1))

using one of two types of penalty: linear and quadratic, leading to the following two formulations of soft-margin PA algorithms:

$$\begin{aligned}\mathbf{w}_{t+1}^{\text{PA-I}} &= \arg \min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{2} \|\mathbf{w} - \mathbf{w}_t\|^2 + C \ell(\mathbf{w}; (\mathbf{x}_t, y_t)); \\ \mathbf{w}_{t+1}^{\text{PA-II}} &= \arg \min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{2} \|\mathbf{w} - \mathbf{w}_t\|^2 + C \ell(\mathbf{w}; (\mathbf{x}_t, y_t))^2,\end{aligned}$$

where  $C$  is a parameter to trade off between passiveness and aggressiveness. The resulting weight updates to the soft-margin PA algorithms have the same form as that of Equation (2), but different coefficients  $\eta_t$  as follows:

$$\eta_t^{\text{PA-I}} = \min \left\{ C, \frac{\ell(\mathbf{w}_t; (\mathbf{x}_t, y_t))}{\|\mathbf{x}_t\|^2} \right\}, \quad \eta_t^{\text{PA-II}} = \frac{\ell(\mathbf{w}_t; (\mathbf{x}_t, y_t))}{\|\mathbf{x}_t\|^2 + \frac{1}{2C}}.$$

### 2.3. Confidence-Weighted Learning

To better exploit the underlying structure between features, the Confidence-Weighted (CW) learning algorithm assumes a Gaussian distribution of weights with mean vector  $\boldsymbol{\mu} \in \mathbb{R}^d$  and covariance matrix  $\Sigma \in \mathbb{R}^{d \times d}$ . The weight distribution is updated by minimizing the Kullback-Leibler divergence between the new weight distribution and the old one while ensuring that the probability of correct classification is greater than a threshold as follows:

$$(\boldsymbol{\mu}_{t+1}, \Sigma_{t+1}) = \arg \min_{\boldsymbol{\mu}, \Sigma} D_{KL}(\mathcal{N}(\boldsymbol{\mu}, \Sigma), \mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t)), \quad \text{s.t. } Pr_{\mathbf{w} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)}[y_t(\mathbf{w} \cdot \mathbf{x}_t) \geq 0] \geq \eta.$$

This optimization problem has a closed-form solution

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t + \alpha_t y_t \Sigma_t \mathbf{x}_t, \quad \Sigma_{t+1} = \Sigma_t - \beta_t \Sigma_t \mathbf{x}_t^T \mathbf{x}_t \Sigma_t. \quad (3)$$

The updating coefficients are calculated as follows:

$$\alpha_t = \max \left\{ 0, \frac{1}{v_t \zeta} \left( -m_t \psi + \sqrt{m_t^2 \frac{\phi^4}{4} + v_t \phi^2 \zeta} \right) \right\}, \quad \beta_t = \frac{\alpha_t \phi}{\sqrt{u_t} + v_t \alpha_t \phi},$$

where  $u_t = \frac{1}{4}(-\alpha_t v_t \phi + \sqrt{\alpha_t^2 v_t^2 \phi^2 + 4v_t})^2$ ,  $v_t = \mathbf{x}_t^T \Sigma_t \mathbf{x}_t$ ,  $m_t = y_t(\boldsymbol{\mu}_t \cdot \mathbf{x}_t)$ ,  $\phi = \Phi^{-1}(\eta)$  ( $\Phi$  is the cumulative function of the normal distribution),  $\psi = 1 + \frac{\phi^2}{2}$ , and  $\zeta = 1 + \phi^2$ .

### 2.4. Adaptive Regularization of Weights

Unlike the original CW learning algorithm, the AROW learning introduces the adaptive regularization of the prediction function when processing each new instance in each learning step, making it more robust than CW to sudden changes of label noise in the learning tasks. In particular, the optimization of AROW is formulated as follows:

$$(\boldsymbol{\mu}_{t+1}, \Sigma_{t+1}) = \arg \min_{\boldsymbol{\mu}, \Sigma} D_{KL}(\mathcal{N}(\boldsymbol{\mu}, \Sigma), \mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t)) + \frac{1}{2\gamma} \ell^2(\boldsymbol{\mu}; (\mathbf{x}_t, y_t)) + \frac{1}{2\gamma} \mathbf{x}_t^T \Sigma_t \mathbf{x}_t,$$

where  $\ell^2(\boldsymbol{\mu}; (\mathbf{x}_t, y_t)) = (\max\{0, 1 - y_t(\boldsymbol{\mu} \cdot \mathbf{x}_t)\})^2$  and  $\gamma$  is a regularization parameter. The optimization has a closed-form solution similar with CW of Equation (3), but different updating coefficients:

$$\alpha_t = \ell(\boldsymbol{\mu}_t; (\mathbf{x}_t, y_t)) \beta_t, \quad \beta_t = \frac{1}{\mathbf{x}_t^T \Sigma_t \mathbf{x}_t + \gamma}.$$

## 2.5. Soft Confidence Weighted Learning

In this section, we present a new online learning method that aims to address the limitation of the CW and AROW learning.

Following the same problem settings of the Confidence-Weighted learning, we assume the weight vector  $\mathbf{w}$  follows the Gaussian distribution with the mean vector  $\boldsymbol{\mu}$  and the covariance matrix  $\Sigma$ . Notice that the probability constraint in the CW learning, that is,  $Pr_{\mathbf{w} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)}[y_t(\mathbf{w} \cdot \mathbf{x}_t) \geq 0] \geq \eta$ , can be rewritten as  $y_t(\boldsymbol{\mu} \cdot \mathbf{x}_t) \geq \phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t}$ , where  $\phi = \Phi^{-1}(\eta)$ . Further, we introduce a  $\phi$  loss function as follows:

$$\ell^\phi(\mathcal{N}(\boldsymbol{\mu}, \Sigma); (\mathbf{x}_t, y_t)) = \max\left(0, \phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - y_t \boldsymbol{\mu} \cdot \mathbf{x}_t\right).$$

It is easy to verify that satisfying the probability constraint (i.e.,  $y_t(\boldsymbol{\mu} \cdot \mathbf{x}_t) \geq \phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t}$  for any  $\phi > 0$ ) is equivalent to satisfying  $\ell^\phi(\mathcal{N}(\boldsymbol{\mu}, \Sigma); (\mathbf{x}_t, y_t)) = 0$ . Therefore, the optimization problem of the original CW can be rewritten as follows:

$$(\boldsymbol{\mu}_{t+1}, \Sigma_{t+1}) = \arg \min_{\boldsymbol{\mu}, \Sigma} D_{KL}(\mathcal{N}(\boldsymbol{\mu}, \Sigma) \| \mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t)), \quad s.t. \ell^\phi(\mathcal{N}(\boldsymbol{\mu}, \Sigma); (\mathbf{x}_t, y_t)) = 0, \phi > 0.$$

The original CW learning method employs a very aggressive updating strategy by changing the distribution as much as necessary to satisfy the constraint imposed by the current example. Although it results in the rapid learning effect, it could force one to wrongly change the parameters of the distribution dramatically when handling a mislabeled instance. Such an undesirable property makes the original CW algorithm perform poorly in many real-world applications with relatively large noise.

To overcome this limitation of the CW learning problem, we propose an SCW learning method, which aims to soften the aggressiveness of the CW updating strategy. The idea of the SCW learning is inspired by the variants of PA algorithms (PA-I and PA-II) and the adaptive margin. In particular, we formulate the optimization of SCW for learning the soft-margin classifiers as follows:

$$(\boldsymbol{\mu}_{t+1}, \Sigma_{t+1}) = \arg \min_{\boldsymbol{\mu}, \Sigma} D_{KL}(\mathcal{N}(\boldsymbol{\mu}, \Sigma) \| \mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t)) + C \ell^\phi(\mathcal{N}(\boldsymbol{\mu}, \Sigma); (\mathbf{x}_t, y_t)), \quad (4)$$

where  $C$  is a parameter to trade off the passiveness and aggressiveness. We denoted the previous formulation of the Soft Confidence-Weighted algorithm as ‘‘SCW-I’’ for short. Similar to the variant of PA, we can also modify the previous formulation by employing a squared penalty, leading to the second formulation of SCW learning (denoted as ‘‘SCW-II’’ for short):

$$(\boldsymbol{\mu}_{t+1}, \Sigma_{t+1}) = \arg \min_{\boldsymbol{\mu}, \Sigma} D_{KL}(\mathcal{N}(\boldsymbol{\mu}, \Sigma) \| \mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t)) + C \ell^\phi(\mathcal{N}(\boldsymbol{\mu}, \Sigma); (\mathbf{x}_t, y_t))^2. \quad (5)$$

For the optimization of SCW-I, the following proposition gives the closed-form solution.

**PROPOSITION 1.** *The closed-form solution of the optimization problem (Equation (4)) is expressed as follows:*

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t + \alpha_t y_t \Sigma_t \mathbf{x}_t, \quad \Sigma_{t+1} = \Sigma_t - \beta_t \Sigma_t \mathbf{x}_t^T \mathbf{x}_t \Sigma_t,$$

where the updating coefficients are as follows:

$$\alpha_t = \min \left\{ C, \max \left\{ 0, \frac{1}{v_t \zeta} \left( -m_t \psi + \sqrt{m_t^2 \frac{\phi^4}{4} + v_t \phi^2 \zeta} \right) \right\} \right\}, \quad \beta_t = \frac{\alpha_t \phi}{\sqrt{u_t} + v_t \alpha_t \phi},$$

where  $u_t = \frac{1}{4}(-\alpha_t v_t \phi + \sqrt{\alpha_t^2 v_t^2 \phi^2 + 4v_t})^2$ ,  $v_t = \mathbf{x}_t^T \Sigma_t \mathbf{x}_t$ ,  $m_t = y_t(\boldsymbol{\mu}_t \cdot \mathbf{x}_t)$ ,  $\phi = \Phi^{-1}(\eta)$ ,  $\psi = 1 + \frac{\phi^2}{2}$ , and  $\zeta = 1 + \phi^2$ .

Similarly, the following proposition gives the closed-form solution to the optimization of SCW-II.

PROPOSITION 2. *The closed-form solution of the optimization problem (Equation (5)) is*

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t + \alpha_t y_t \boldsymbol{\Sigma}_t \mathbf{x}_t, \quad \boldsymbol{\Sigma}_{t+1} = \boldsymbol{\Sigma}_t - \beta_t \boldsymbol{\Sigma}_t \mathbf{x}_t^T \mathbf{x}_t \boldsymbol{\Sigma}_t,$$

where the coefficients are as follows:

$$\alpha_t = \max \left\{ 0, \frac{-(2m_t n_t + \phi^2 m_t v_t) + \gamma_t}{2(n_t^2 + n_t v_t \phi^2)} \right\}, \quad \beta_t = \frac{\alpha_t \phi}{\sqrt{u_t} + v_t \alpha_t \phi},$$

where  $\gamma_t = \phi \sqrt{\phi^2 m_t^2 v_t^2 + 4n_t v_t (n_t + v_t \phi^2)}$  and  $n_t = v_t + \frac{1}{2C}$ .

The detailed proofs of Propositions 1 and 2 can be found in the appendix section. Finally, Algorithm 1 summarizes the proposed SCW-I and SCW-II algorithms.

---

**ALGORITHM 1: SCW Learning Algorithms (SCW)**

---

**INPUT:** parameters  $C > 0, \eta > 0$ .

**INITIALIZATION:**  $\boldsymbol{\mu}_1 = (0, \dots, 0)^\top, \boldsymbol{\Sigma}_1 = I$ .

**for**  $t = 1, \dots, T$  **do**

    Receive an example  $\mathbf{x}_t \in \mathbb{R}^d$ ;

    Make prediction:  $\hat{y}_t = \text{sgn}(\boldsymbol{\mu}_t \cdot \mathbf{x}_t)$ ;

    Receive true label  $y_t$ ;

    suffer loss  $\ell^\phi(\mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t); (\mathbf{x}_t, y_t))$ ;

**if**  $\ell^\phi(\mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t); (\mathbf{x}_t, y_t)) > 0$  **then**

$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t + \alpha_t y_t \boldsymbol{\Sigma}_t \mathbf{x}_t, \boldsymbol{\Sigma}_{t+1} = \boldsymbol{\Sigma}_t - \beta_t \boldsymbol{\Sigma}_t \mathbf{x}_t^T \mathbf{x}_t \boldsymbol{\Sigma}_t$

        where  $\alpha_t$  and  $\beta_t$  are computed by either Proposition 1 (SCW-I) or Proposition 2 (SCW-II);

**end if**

**end for**

---

### 3. MULTICLASS SOFT CONFIDENCE-WEIGHTED LEARNING

#### 3.1. Overview of Online Multiclass Learning

Similar to online binary classification, online multiclass learning is performed over a sequence of training examples  $(\mathbf{x}_1, Y_1), \dots, (\mathbf{x}_T, Y_T)$ . Unlike binary classification, where  $y_t \in \{-1, +1\}$ , in multiclass learning, each class assignment  $Y_t \subseteq \mathcal{Y}$  could contain multiple class labels, making it a more challenging problem. We use  $\hat{Y}_t$  to represent the class set predicted by the online learning algorithm. Before presenting our algorithm, we first review online multiclass learning [Crammer and Singer 2003; Fink et al. 2006; Shalev-Shwartz and Singer 2007] based on the framework of label ranking [Crammer and Singer 2005].

#### 3.2. Label Ranking for Multiclass Learning

Given an instance  $\mathbf{x}$ , the label ranking approach first computes a score for every class label in  $\mathcal{Y}$  and ranks the classes in descending order of their scores. The predicted class set  $\hat{Y}_t$  is formed by the classes with the highest scores. The objective of label ranking is to ensure that the score of class  $r$  is significantly larger than that of class  $s$  if  $r \in Y_t$  is a true class assignment while  $s \in \mathcal{Y} \setminus Y_t$  is not. The multiclass learning algorithms can be divided into two categories: single-prototype model and multiprototype model. The single-vector model assumes that there exists the class-dependent feature



mapping and maintains a single hypothesis shared by all the classes. The single-prototype model is generally used in some structured prediction problems [Collins 2002]. Confidence-Weighted learning has also been extended to online multiclass learning in the single-prototype model scenario and has shown promising performance in Natural Language Processing applications [Crammer et al. 2009]. The multiprototype model can be applied to a more general multiclass learning problem since there might be a single natural representation for every instance rather than multiple feature representations for each individual class, such as the digit recognition problems. Besides, as shown in Crammer et al. [2006], the multiprototype model could be easily reduced to a single-prototype model. In this article, we focus on the protocol of the multiprototype model [Crammer and Singer 2001, 2003; Crammer et al. 2006] for the design of a multiclass learning algorithm. It learns multiple hypotheses/classifiers  $\mathbf{w}_1, \dots, \mathbf{w}_k$ , one hypothesis for each class in  $\mathcal{Y}$ , leading to a total of  $k$  vectors that are trained for the classification task. Specifically, for trial  $t$ , upon receiving an instance  $x_t$ , the scores of  $k$  classes output by the set of  $k$  hypotheses are given by

$$(\mathbf{w}_{t,1} \cdot \mathbf{x}_t, \dots, \mathbf{w}_{t,k} \cdot \mathbf{x}_t)^\top.$$

We introduce two variables  $r_t$  and  $s_t$  that are defined as follows:

$$r_t = \arg \min_{r \in Y_t} \mathbf{w}_{t,r} \cdot \mathbf{x}_t, \quad s_t = \arg \max_{s \notin Y_t} \mathbf{w}_{t,s} \cdot \mathbf{x}_t,$$

where  $r_t$  and  $s_t$  represent the class of the smallest score among all relevant classes and the class of the largest score among the irrelevant classes, respectively. Using the notation of  $r_t$  and  $s_t$ , the margin with respect to the hypothesis set at trial  $t$  is defined as follows:

$$\Gamma((\mathbf{w}_{t,1}, \dots, \mathbf{w}_{t,k}); (\mathbf{x}_t, Y_t)) = \mathbf{w}_{t,r_t} \cdot \mathbf{x}_t - \mathbf{w}_{t,s_t} \cdot \mathbf{x}_t.$$

Based on the multiprototype model, several online multiclass learning algorithms have been proposed. Representative work includes three kinds of additive updating methods based on Perceptron [Crammer and Singer 2003] and Multiclass Passive Aggressive Learning [Crammer et al. 2006]. Also, the Multiclass Confidence-Weighted Learning in the single-prototype model [Crammer et al. 2009] also can be extended to the multiprototype model easily.

### 3.3. Multiclass Soft Confidence Weighted Learning

In this section, we present a new online multiclass learning method based on the previously described soft confidence-weighted learning idea.

In Multiclass Soft Confidence-Weighted learning, we assume the each prototype vector  $\mathbf{w}_i$  follows the Gaussian distribution with the mean vector  $\boldsymbol{\mu}_i$  and the covariance matrix  $\Sigma_i$ . For simplicity, we assume each prototype  $i \in \{1, \dots, k\}$  shares the same covariance matrix  $\Sigma$ . In a multiclass classification setting, we want to ensure that the lowest prediction score of all the relevant classes is higher than the highest prediction score of all the irrelevant classes, with a high probability which is not lower than a threshold  $\eta$ . In mathematical form, the constraint can be expressed as follows:

$$Pr_{\mathbf{w}_{t,r_t} \sim \mathcal{N}(\boldsymbol{\mu}_{t,r_t}, \Sigma_t), \mathbf{w}_{t,s_t} \sim \mathcal{N}(\boldsymbol{\mu}_{t,s_t}, \Sigma_t)} [(\mathbf{w}_{t,r_t} \cdot \mathbf{x}_t) \geq (\mathbf{w}_{t,s_t} \cdot \mathbf{x}_t)] \geq \eta,$$

where

$$r_t = \arg \min_{r \in Y_t} \boldsymbol{\mu}_{t,r} \cdot \mathbf{x}_t, \quad s_t = \arg \max_{s \notin Y_t} \boldsymbol{\mu}_{t,s} \cdot \mathbf{x}_t.$$

It is easy to see that this probabilistic constraint can be rewritten as  $(\boldsymbol{\mu}_{t,r_t} \cdot \mathbf{x}_t - \boldsymbol{\mu}_{t,s_t} \cdot \mathbf{x}_t) \geq \phi \sqrt{2\mathbf{x}_t^\top \Sigma_t \mathbf{x}_t}$ , where  $\phi = \Phi^{-1}(\eta)$ .



We introduce a  $\phi$  loss function as follows:

$$\ell^\phi(\mathcal{N}(\boldsymbol{\mu}_{t,r_t}, \boldsymbol{\mu}_{t,s_t}, \Sigma_t); (\mathbf{x}_t, Y_t)) = \max\left(0, \phi\sqrt{2\mathbf{x}_t^\top \Sigma_t \mathbf{x}_t} - (\boldsymbol{\mu}_{t,r_t} \cdot \mathbf{x}_t - \boldsymbol{\mu}_{t,s_t} \cdot \mathbf{x}_t)\right).$$

Then, it is easy to verify that satisfying the probability constraint (i.e.,  $(\boldsymbol{\mu}_{t,r_t} \cdot \mathbf{x}_t - \boldsymbol{\mu}_{t,s_t} \cdot \mathbf{x}_t) \geq \phi\sqrt{2\mathbf{x}_t^\top \Sigma_t \mathbf{x}_t}$  for any  $\phi > 0$ ) is equivalent to satisfying  $\ell^\phi(\mathcal{N}(\boldsymbol{\mu}_{t,r_t}, \boldsymbol{\mu}_{t,s_t}, \Sigma_t); (\mathbf{x}_t, Y_t)) = 0$ . Therefore, the optimization problem of the original CW can be rewritten as follows:

$$\begin{aligned} & (\boldsymbol{\mu}_{t+1,r_t}, \boldsymbol{\mu}_{t+1,s_t}, \Sigma_{t+1}) \\ &= \arg \min_{\boldsymbol{\mu}_r, \boldsymbol{\mu}_s, \Sigma} D_{KL}(\mathcal{N}(\boldsymbol{\mu}_r, \Sigma) \|\mathcal{N}(\boldsymbol{\mu}_{t,r_t}, \Sigma_t)) + D_{KL}(\mathcal{N}(\boldsymbol{\mu}_s, \Sigma) \|\mathcal{N}(\boldsymbol{\mu}_{t,s_t}, \Sigma_t)), \\ & \text{s.t. } \ell^\phi(\mathcal{N}(\boldsymbol{\mu}_r, \boldsymbol{\mu}_s, \Sigma_t); (\mathbf{x}_t, Y_t)) = 0, \phi > 0. \end{aligned}$$

The original multiclass CW learning method employs a very aggressive updating strategy by changing the distribution as much as necessary to satisfy the constraint imposed by the current example. Although it results in the rapid learning effect, it could force one to wrongly change the parameters of the distribution dramatically when handling a mislabeled instance. Such an undesirable property makes the original multiclass CW algorithm perform poorly in many real-world applications with relatively large noise.

To overcome this limitation of the CW learning problem, we propose an SCW learning method, which aims to soften the aggressiveness of the CW updating strategy. The idea of SCW learning is inspired by the variants of PA algorithms (PA-I and PA-II) and the adaptive margin. In particular, we formulate the optimization of SCW for learning the soft-margin classifiers as follows:

$$\begin{aligned} & (\boldsymbol{\mu}_{t+1,r_t}, \boldsymbol{\mu}_{t+1,s_t}, \Sigma_{t+1}) \\ &= \arg \min_{\boldsymbol{\mu}_r, \boldsymbol{\mu}_s, \Sigma} D_{KL}(\mathcal{N}(\boldsymbol{\mu}_r, \Sigma) \|\mathcal{N}(\boldsymbol{\mu}_{t,r_t}, \Sigma_t)) + D_{KL}(\mathcal{N}(\boldsymbol{\mu}_s, \Sigma) \|\mathcal{N}(\boldsymbol{\mu}_{t,s_t}, \Sigma_t)) \\ & \quad + C \ell^\phi(\mathcal{N}(\boldsymbol{\mu}_r, \boldsymbol{\mu}_s, \Sigma_t); (\mathbf{x}_t, Y_t)), \end{aligned} \tag{6}$$

where  $C$  is a parameter to trade off the passiveness and aggressiveness. We denoted the previous formulation of the Multiclass Soft Confidence-Weighted algorithm as ‘‘MSCW1’’ for short. Similar to the variant of PA, we can also modify the previous formulation by employing a squared penalty, leading to the second formulation of Multiclass SCW learning (denoted as ‘‘MSCW2’’ for short):

$$\begin{aligned} & (\boldsymbol{\mu}_{t+1,r_t}, \boldsymbol{\mu}_{t+1,s_t}, \Sigma_{t+1}) \\ &= \arg \min_{\boldsymbol{\mu}_r, \boldsymbol{\mu}_s, \Sigma} D_{KL}(\mathcal{N}(\boldsymbol{\mu}_r, \Sigma) \|\mathcal{N}(\boldsymbol{\mu}_{t,r_t}, \Sigma_t)) + D_{KL}(\mathcal{N}(\boldsymbol{\mu}_s, \Sigma) \|\mathcal{N}(\boldsymbol{\mu}_{t,s_t}, \Sigma_t)) \\ & \quad + C \ell^\phi(\mathcal{N}(\boldsymbol{\mu}_r, \boldsymbol{\mu}_s, \Sigma_t); (\mathbf{x}_t, Y_t))^2. \end{aligned} \tag{7}$$

For the optimization of MSCW1, the following proposition gives the closed-form solution.

**PROPOSITION 3.** *The closed-form solution of the optimization problem (Equation (6)) is expressed as follows:*

$$\boldsymbol{\mu}_{t+1,r_t} = \boldsymbol{\mu}_{t,r_t} + \alpha_t \gamma_t \Sigma_t \mathbf{x}_t, \quad \boldsymbol{\mu}_{t+1,s_t} = \boldsymbol{\mu}_{t,s_t} - \alpha_t \gamma_t \Sigma_t \mathbf{x}_t, \quad \Sigma_{t+1} = \Sigma_t - \beta_t \Sigma_t \mathbf{x}_t^T \mathbf{x}_t \Sigma_t,$$

where the updating coefficients are as follows:

$$\alpha_t = \min \left\{ C, \max \left\{ 0, \frac{1}{2v_t \psi} \left( -m_t \psi + \sqrt{m_t^2 \psi^2 - m_t^2 \psi + 2\psi \phi^2 v_t} \right) \right\} \right\}, \quad \beta_t = \frac{\alpha_t \phi}{\sqrt{2u_t + v_t \alpha_t \phi}},$$

where  $u_t = \frac{1}{8}(-\alpha_t v_t \phi + \sqrt{\alpha_t^2 v_t^2 \phi^2 + 8v_t})^2$ ,  $v_t = \mathbf{x}_t^T \Sigma_t \mathbf{x}_t$ ,  $m_t = \boldsymbol{\mu}_{t,r_t} \cdot \mathbf{x}_t - \boldsymbol{\mu}_{t,s_t} \cdot \mathbf{x}_t$ ,  $\phi = \Phi^{-1}(\eta)$ ,  $\psi = 1 + \frac{\phi^2}{2}$ .

Similarly, the following proposition gives the closed-form solution to the optimization of MSCW2.

PROPOSITION 4. *The closed-form solution of the optimization problem (Equation (7)) is*

$$\boldsymbol{\mu}_{t+1,r_t} = \boldsymbol{\mu}_{t,r_t} + \alpha_t \gamma_t \boldsymbol{\Sigma}_t \mathbf{x}_t, \quad \boldsymbol{\mu}_{t+1,s_t} = \boldsymbol{\mu}_{t,s_t} - \alpha_t \gamma_t \boldsymbol{\Sigma}_t \mathbf{x}_t, \quad \boldsymbol{\Sigma}_{t+1} = \boldsymbol{\Sigma}_t - \beta_t \boldsymbol{\Sigma}_t \mathbf{x}_t \mathbf{x}_t^T \boldsymbol{\Sigma}_t.$$

The updating coefficients are as follows:

$$\alpha_t = \max \left\{ 0, \frac{-(2m_t \rho_t + \phi^2 m_t \nu_t) + \gamma_t}{2(\rho_t^2 + \rho_t \nu_t \phi^2)} \right\}, \quad \beta_t = \frac{\alpha_t \phi}{\sqrt{2u_t} + \nu_t \alpha_t \phi},$$

where  $\gamma_t = \phi \sqrt{\phi^2 m_t^2 \nu_t^2 + 8\rho_t \nu_t (\rho_t + \nu_t \phi^2)}$  and  $\rho_t = 2\nu_t + \frac{1}{2C}$ .

The details of the proofs for Propositions 3 and 4 are in the appendix. Finally, Algorithm 2 summarizes the proposed MSCW1 and MSCW2 algorithms.

---

**ALGORITHM 2:** Multiclass SCW Learning Algorithms (MSCW)

---

**INPUT:** parameters  $C > 0, \eta > 0$ .

**INITIALIZATION:**  $\boldsymbol{\mu}_{1,1}, \dots, \boldsymbol{\mu}_{1,k} = (0, \dots, 0)^\top, \boldsymbol{\Sigma}_1 = I$ .

**for**  $t = 1, \dots, T$  **do**

    Receive an example  $\mathbf{x}_t \in \mathbb{R}^d$ ;

    Make prediction:  $\hat{Y}_t = \arg \max_r (\boldsymbol{\mu}_{t,r} \cdot \mathbf{x}_t), r \in [1, k]$ ;

    Receive true label  $Y_t$ ;

    suffer loss  $\ell^\phi(\mathcal{N}(\boldsymbol{\mu}_{t,r_t}, \boldsymbol{\mu}_{t,s_t}, \boldsymbol{\Sigma}_t); (\mathbf{x}_t, Y_t))$ ;

**if**  $\ell^\phi(\mathcal{N}(\boldsymbol{\mu}_{t,r_t}, \boldsymbol{\mu}_{t,s_t}, \boldsymbol{\Sigma}_t); (\mathbf{x}_t, Y_t)) > 0$  **then**

$\boldsymbol{\mu}_{t+1,r_t} = \boldsymbol{\mu}_{t,r_t} + \alpha_t \gamma_t \boldsymbol{\Sigma}_t \mathbf{x}_t, \boldsymbol{\mu}_{t+1,s_t} = \boldsymbol{\mu}_{t,s_t} - \alpha_t \gamma_t \boldsymbol{\Sigma}_t \mathbf{x}_t, \boldsymbol{\Sigma}_{t+1} = \boldsymbol{\Sigma}_t - \beta_t \boldsymbol{\Sigma}_t \mathbf{x}_t \mathbf{x}_t^T \boldsymbol{\Sigma}_t,$

        where  $\alpha_t$  and  $\beta_t$  are computed by either Proposition 3 (MSCW1) or Proposition 4 (MSCW2);

**end if**

**end for**

---

## 4. ANALYSIS AND DISCUSSIONS

We first give an overview of the comparison of the proposed SCW methods with respect to several existing first-order and second-order online learning algorithms, followed by the discussions on the nonlinear extension and the bound analysis.

### 4.1. Comparison with the Existing Methods

Following the study of AROW, we qualitatively examine the properties of different algorithms in Table I. Unlike the previous second-order algorithms, the proposed SCW algorithm enjoys all four salient properties. In particular, SCW improves over the original CW algorithm by adding the capability to handle the nonseparable cases and improves over AROW by adding the adaptive margin property. To the best of our knowledge, SCW is the first second-order online learning method that holds all four properties.

### 4.2. Extension to Nonlinear Cases

Similar to other linear online learning methods, the proposed SCW learning can be extended to nonlinear cases. The following lemma shows the possibility of extending the proposed SCW algorithms to nonlinear cases using kernel tricks.

Table I. Property Comparison of Online Algorithms

Algorithm	Large Margin	Confidence	Non separable	Adaptive Margin
PA	Yes	No	Yes	No
SOP	No	Yes	Yes	No
IELLIP	No	Yes	Yes	No
CW	Yes	Yes	No	Yes
AROW	Yes	Yes	Yes	No
NHERD	Yes	Yes	Yes	No
NAROW	Yes	Yes	Yes	No
SCW	Yes	Yes	Yes	Yes

LEMMA 4.1 (REPRESENTER THEOREM). *The mean  $\boldsymbol{\mu}_i$  and covariance  $\Sigma_i$  parameters computed by the soft confidence-weighted algorithm can be written as linear combinations of the input vectors with coefficients that depend only on inner products of input vectors, that is,*

$$\Sigma_i = \sum_{p,q=1}^{i-1} \pi_{p,q}^{(i)} \mathbf{x}_p \mathbf{x}_q^\top + aI, \quad \boldsymbol{\mu}_i = \sum_p^{i-1} v_p^{(i)} \mathbf{x}_p,$$

where  $v_i^{(i)} = 1$  and  $v_p^{(i+1)} = v_p^{(i)} + \alpha_i y_i \sum_q^{i-1} \pi_{p,q}^{(i)} \mathbf{x}_q^\top \mathbf{x}_i$  for  $p < i$ , and  $\pi_{p,q}^{(i+1)} = -\beta_i \sum_{r,s} \pi_{p,r}^{(i)} \pi_{s,q}^{(i)} \mathbf{x}_r^\top \mathbf{x}_s + \pi_{p,q}^{(i)}$ ,  $\pi_{p,i}^{(i)} = \pi_{i,p}^{(i)} = -\beta_i \sum_{p,r}^{i-1} \pi_{p,r}^{(i)} (\mathbf{x}_r^\top \mathbf{x}_i)$ ,  $\pi_{i,i}^{(i+1)} = -\beta_i$ .

The previous lemma can be proved by induction similar to the proof in Crammer et al. [2008].

### 4.3. Analysis of the Loss Bound

Our analysis begins with the definition of confidence loss, which is used in Crammer et al. [2008]. The loss is a function of the margin  $m_i$  normalized by  $\sqrt{v}$ , that is,  $\tilde{m}_i = \frac{m_i}{\sqrt{v_i}}$ . We modified the confidence loss in Crammer et al. [2008] as an upper-bounded loss by

$$\ell_{\phi_i}(\tilde{m}_i) = \begin{cases} 0 & \tilde{m}_i \geq \phi \\ \min \left\{ f_{\phi}(\tilde{m}_i), \frac{C^2(1+\phi^2)v_i}{\phi^2} \right\} & \tilde{m}_i < \phi, \end{cases}$$

where  $f_{\phi}(\tilde{m}) = \frac{(-\tilde{m}\psi + \sqrt{\tilde{m}^2 \frac{\phi^4}{4} + \phi^2 \zeta})^2}{\phi^2 \zeta}$ . It is easy to see that the loss  $\ell_{\phi}(\tilde{m})$  holds the properties of Lemma 5 in Crammer et al. [2008] for SCW-I.

We have the following loss bound.

THEOREM 1. *Let  $(\mathbf{x}_1, y_1) \dots (\mathbf{x}_n, y_n)$  be an input sequence for SCW-I. Assume there exist  $\boldsymbol{\mu}^*$  and  $\Sigma^*$  such that for all  $i$  for which the algorithm made an update ( $\alpha_i > 0$ ),*

$$\boldsymbol{\mu}^{*T} \mathbf{x}_i y_i \geq \boldsymbol{\mu}_{i+1}^T \mathbf{x}_i y_i, \quad \mathbf{x}_i^\top \Sigma^* \mathbf{x}_i \leq \mathbf{x}_i^\top \Sigma_{i+1} \mathbf{x}_i.$$

Then the following bound holds:

$$\sum_i \ell_{\phi_i}(\tilde{m}_i) \leq \sum_i (\alpha_i)^2 v_i \leq \frac{(1 + \phi^2)}{\phi^2} (-\log \det \Sigma^* + \text{Tr}(\Sigma^*) + \boldsymbol{\mu}^{*T} \Sigma_{n+1}^{-1} \boldsymbol{\mu}^* - d).$$

The previous theorem can be proved by applying Lemma 7 and property 6 in Lemma 5 in Crammer et al. [2008]. If we let  $\ell_{\phi_i}(\tilde{m}_i)$  upper bound the 0 – 1 loss by choosing an

Table II. List of Datasets Used in the Experiments

Dataset	# Training Examples	# Features
splice	1,000	60
svmguid3	1,243	21
Synthetic data	5,000	20
MITface	6,977	361
usps1vsall	7,291	256
mushrooms	8,124	112
mnist1vs2	14,867	784
w7a	24,692	300
codrna	59,535	8
ijcnn1	141,691	22
covtype	581,012	54

appropriate  $C$ , then our mistake number is also bounded by

$$\frac{(1 + \phi^2)}{\phi^2}(-\log \det \Sigma^* + \text{Tr}(\Sigma^*) + \boldsymbol{\mu}^{*T} \Sigma_{n+1}^{-1} \boldsymbol{\mu}^* - d).$$

## 5. EMPIRICAL EVALUATION

### 5.1. Empirical Evaluation on Online Binary Classification

*5.1.1. Datasets and Compared Algorithms.* We adopt a variety of datasets from different domains:

- Synthetic data: We generated this dataset by the method described in Crammer et al. [2008], which is used to examine the effectiveness of second-order algorithms. More specifically, we first generate 5,000 data points in  $\mathbb{R}^{20}$ , where the first two coordinates were drawn from bivariate normal distribution with rotation of  $45^\circ$ , and the remaining 18 coordinates were drawn from independent normal distribution with mean 0 and variance 2. Following Crammer et al. [2009], we also generated another version with 0.1 noise to examine the robustness of second-order algorithms.
- Digital recognition: We use two benchmarks: “USPS”<sup>1</sup> and “MNIST.”<sup>2</sup> For binary classification, we choose “1” versus “all” for “USPS,” and “1” versus “2” for “MNIST.”
- Face data: We use the MIT-CBCL face images.<sup>3</sup>
- Machine-learning datasets: We randomly choose several public machine-learning datasets from LIBSVM.<sup>4</sup>

Table II shows the statistics of the list of datasets used.

We compare the empirical performance of the following online binary class learning algorithms:

- Perceptron**: the Perceptron algorithm in Rosenblatt [1958a]
- ROMMA, agg-ROMMA**: the Relaxed Online Maximum Margin Algorithm and its aggressive version agg-ROMMA [Li and Long 2002]
- PA-I, PA-II**: the Passive Aggressive Online Learning algorithms [Crammer et al. 2006]

<sup>1</sup><http://www-i6.informatik.rwth-aachen.de/~keysers/usps.html>.

<sup>2</sup><http://yann.lecun.com/exdb/mnist/>.

<sup>3</sup><http://cbcl.mit.edu/software-datasets/FaceData2.html>.

<sup>4</sup><http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/>.

- SOP**: the Second-Order Perceptron algorithms [Cesa-Bianchi et al. 2005]
- CW**: the Confidence-Weighted Learning algorithm [Crammer et al. 2008]
- IELLIP**: the Improved Ellipsoid Method for Online Learning [Yang et al. 2009]
- AROW**: the Adaptive Regularization of Weights algorithm [Crammer et al. 2009]
- NHERD**: the Normal HERD algorithm [Crammer and Lee 2010]
- NAROW**: the New AROW algorithm described in Orabona and Crammer [2010]

Following the similar parameter setting methods in Dredze et al. [2008] and Crammer et al. [2009], the parameter  $r$  in AROW, parameter  $b$  in NAROW, and parameters  $C$  in PA-I, PA-II, NHERD, SCW-I, and SCW-II are all determined by cross-validation to select the best one from  $\{2^{-4}, 2^{-3}, \dots, 2^3, 2^4\}$ ; the parameters  $\eta$  in CW, SCW-I, and SCW-II are determined by cross-validation to select the best one from  $\{0.5, 0.55, \dots, 0.9, 0.95\}$ ; and the parameter  $b$  in IELLIP is determined by cross-validation to select the best one from  $\{0.1, 0.2, \dots, 0.9\}$ . After the best parameters are determined, all the experiments were conducted over 20 random permutations for each dataset. All the results were reported by averaging over these 20 runs, and the source codes of all the algorithms are available at Hoi et al. [2014]. We evaluate the performance by three metrics: (1) online cumulative mistake rate (OCMR), where the OCMR at time  $t$  is defined as

$$\text{OCMR}(t) = \frac{\sum_{i=1}^t \mathbb{I}_{(\text{sign}(\mu_i \cdot \mathbf{x}_i) \neq y_i)}}{t},$$

(2) number of updates (which would be closely related to the potential number of support vectors in kernel extension), and (3) running time cost.

*5.1.2. Experimental Results.* Tables III and IV summarize the results of our empirical evaluation, where we only show margin-based second-order learning algorithms due to space limitations. For a more complete comparison, please refer to our supplemental material. The **bold** elements indicate the best performance with paired t-test at the 95% significance level. We have several observations as follows.

First of all, by examining the overall mistakes, we found that second-order algorithms usually outperform first-order algorithms, and margin-based algorithms usually outperform non-margin-based methods. This shows the efficacy of “Large Margin” and “Confidence” properties for learning better classifiers.

Second, by examining the original CW algorithm, we found that it significantly outperforms the first-order algorithms (e.g., Perceptron, ROMMA, and PA algorithms) on the synthetic data without noise, but fails to outperform the first-order algorithms on some real-world datasets that often have noisy data. This empirical result verifies the importance of the “Handling Nonseparable” property in producing robust classifiers when dealing with noisy data.

Further, we found that AROW significantly outperforms CW in many real-world datasets (except mnist). However, AROW usually produces considerably more updates and spends more running time than CW. This verifies the importance of the “adaptive margin” property of both CW and SCW to reduce the number of updates as well as the running time.

Moreover, among all the compared algorithms, SCW often achieves the best or close to the best performance in terms of accuracy, number of updates, and running time cost. Finally, Figures 1, 2, and 3 show the online results of 13 algorithms with respect to varied numbers of samples in online learning processes. The results again validate the advantages of SCW in both efficacy and efficiency among all of the state-of-the-art algorithms.

Table III. Evaluation of the Classification Performance of SCW

Algorithm	svmguide3			codrna		
	Mistake Rate	#Updates	Time(s)	Mistake Rate	#Updates	Time(s)
Perceptron	0.331 ± 0.009	411 ± 10	0.00 ± 0.00	0.200 ± 0.001	11,902 ± 73	0.51 ± 0.02
ROMMA	0.333 ± 0.014	414 ± 17	0.01 ± 0.00	0.161 ± 0.011	9,614 ± 646	0.70 ± 0.03
agg-ROMMA	0.329 ± 0.014	511 ± 26	0.01 ± 0.00	0.161 ± 0.011	9,565 ± 682	0.74 ± 0.03
PA-I	0.236 ± 0.002	770 ± 8	0.01 ± 0.00	0.228 ± 0.001	29,741 ± 176	2.23 ± 0.26
PA-II	0.255 ± 0.007	1137 ± 13	0.01 ± 0.00	0.228 ± 0.001	29,741 ± 176	2.05 ± 0.12
SOP	0.295 ± 0.008	366 ± 9	0.04 ± 0.00	0.106 ± 0.001	6,298 ± 39	1.04 ± 0.01
CW	0.294 ± 0.011	702 ± 13	0.03 ± 0.00	0.157 ± 0.040	9,278 ± 62	1.08 ± 0.01
IELLIP	0.329 ± 0.008	409 ± 9	0.02 ± 0.00	0.221 ± 0.010	13,156 ± 603	1.22 ± 0.04
NHERD	0.224 ± 0.012	1170 ± 21	0.05 ± 0.00	0.089 ± 0.032	32,232 ± 8,679	3.08 ± 1.18
AROW	0.218 ± 0.005	1174 ± 15	0.04 ± 0.00	0.066 ± 0.000	26,055 ± 328	2.04 ± 0.21
NAROW	0.308 ± 0.096	1229 ± 8	0.05 ± 0.00	0.182 ± 0.054	54,557 ± 4,935	6.97 ± 1.42
SCW-I	<b>0.209 ± 0.007</b>	540 ± 13	0.03 ± 0.00	<b>0.065 ± 0.000</b>	7,328 ± 326	0.95 ± 0.02
SCW-II	0.213 ± 0.008	954 ± 50	0.04 ± 0.00	0.066 ± 0.000	12,070 ± 438	1.18 ± 0.04
Algorithm	splice			usps "1" Versus "all"		
	Mistake Rate	#Updates	Time(s)	Mistake Rate	#Updates	Time(s)
Perceptron	0.345 ± 0.014	345 ± 14	0.00 ± 0.00	0.028 ± 0.002	205 ± 11	0.06 ± 0.00
ROMMA	0.365 ± 0.025	364 ± 25	0.01 ± 0.00	0.028 ± 0.003	205 ± 19	0.09 ± 0.00
agg-ROMMA	0.365 ± 0.018	370 ± 16	0.01 ± 0.00	0.028 ± 0.002	227 ± 19	0.10 ± 0.00
PA-I	0.342 ± 0.012	681 ± 11	0.01 ± 0.00	0.020 ± 0.001	676 ± 10	0.07 ± 0.00
PA-II	0.342 ± 0.012	681 ± 11	0.01 ± 0.00	0.020 ± 0.001	676 ± 10	0.07 ± 0.00
SOP	0.280 ± 0.009	279 ± 9	0.19 ± 0.02	0.023 ± 0.001	167 ± 8	33.32 ± 0.13
CW	0.271 ± 0.009	555 ± 9	0.06 ± 0.00	0.013 ± 0.001	493 ± 21	2.72 ± 0.10
IELLIP	0.310 ± 0.013	310 ± 13	0.04 ± 0.00	0.027 ± 0.002	199 ± 12	2.39 ± 0.46
NHERD	0.245 ± 0.010	805 ± 22	0.09 ± 0.01	0.014 ± 0.002	2421 ± 225	11.55 ± 1.00
AROW	0.241 ± 0.006	741 ± 24	0.07 ± 0.00	0.012 ± 0.001	1449 ± 132	7.00 ± 0.56
NAROW	0.269 ± 0.015	717 ± 35	0.08 ± 0.01	0.018 ± 0.003	2153 ± 251	10.29 ± 1.18
SCW-I	<b>0.229 ± 0.006</b>	541 ± 8	0.06 ± 0.00	0.012 ± 0.001	385 ± 9	2.22 ± 0.04
SCW-II	0.240 ± 0.010	479 ± 12	0.05 ± 0.00	<b>0.011 ± 0.001</b>	385 ± 10	2.22 ± 0.05
Algorithm	ijcnn1			w7a		
	Mistake Rate	#Updates	Time(s)	Mistake Rate	#Updates	Time(s)
Perceptron	0.106 ± 0.001	15,052 ± 71	1.09 ± 0.03	0.117 ± 0.001	2,884 ± 14	0.28 ± 0.00
ROMMA	0.101 ± 0.001	14,291 ± 72	1.70 ± 0.04	0.111 ± 0.001	2,741 ± 15	0.38 ± 0.00
agg-ROMMA	0.101 ± 0.001	14,806 ± 98	1.81 ± 0.05	0.106 ± 0.000	3,575 ± 78	0.41 ± 0.00
PA-I	0.077 ± 0.000	28,398 ± 83	2.53 ± 0.18	0.101 ± 0.000	4,750 ± 20	0.33 ± 0.00
PA-II	0.081 ± 0.000	61,085 ± 154	8.19 ± 0.52	0.102 ± 0.000	5,575 ± 17	0.34 ± 0.00
SOP	0.102 ± 0.001	14,478 ± 93	4.79 ± 0.11	0.111 ± 0.000	2,747 ± 7	164.93 ± 0.12
CW	0.093 ± 0.001	30,678 ± 146	4.84 ± 0.14	0.104 ± 0.000	2,432 ± 48	17.16 ± 0.39
IELLIP	0.117 ± 0.002	16,570 ± 352	3.01 ± 0.07	0.114 ± 0.001	836 ± 19	14.90 ± 2.77
NHERD	0.084 ± 0.001	85,104 ± 4,283	25.01 ± 3.07	0.101 ± 0.001	12,348 ± 378	79.16 ± 2.38
AROW	0.081 ± 0.000	73,082 ± 1,272	16.95 ± 1.19	0.099 ± 0.001	10,233 ± 246	65.26 ± 1.50
NAROW	0.099 ± 0.020	105,937 ± 8,231	39.64 ± 6.89	0.108 ± 0.001	23,666 ± 179	150.37 ± 1.17
SCW-I	<b>0.058 ± 0.002</b>	10,561 ± 704	2.45 ± 0.07	<b>0.097 ± 0.000</b>	4,118 ± 23	14.85 ± 0.19
SCW-II	0.072 ± 0.003	21,792 ± 3,840	3.823 ± 0.568	0.099 ± 0.001	5,634 ± 78	24.55 ± 0.49

Table IV. Evaluation of the Classification Performance of SCW

Algorithm	mnist "1" vs "2"			MITface		
	Mistake Rate	#Updates	Time(s)	Mistake Rate	#Updates	Time(s)
Perceptron	0.017 ± 0.001	259 ± 11	0.29 ± 0.01	0.064 ± 0.003	443 ± 22	0.08 ± 0.00
ROMMA	0.023 ± 0.002	335 ± 24	0.34 ± 0.00	0.064 ± 0.006	446 ± 41	0.11 ± 0.00
agg-ROMMA	0.023 ± 0.002	343 ± 24	0.35 ± 0.00	0.062 ± 0.005	487 ± 41	0.12 ± 0.00
PA-I	0.013 ± 0.001	988 ± 24	0.31 ± 0.01	0.050 ± 0.002	1,323 ± 15	0.10 ± 0.00
PA-II	0.013 ± 0.001	988 ± 24	0.31 ± 0.00	0.049 ± 0.002	1,375 ± 18	0.10 ± 0.00
SOP	0.055 ± 0.004	813 ± 55	935.17 ± 3.68	0.044 ± 0.002	307 ± 11	70.32 ± 0.17
CW	0.012 ± 0.000	856 ± 26	67.80 ± 1.64	0.028 ± 0.001	835 ± 19	8.94 ± 0.20
IELLIP	0.019 ± 0.001	285 ± 13	46.98 ± 5.53	0.048 ± 0.002	334 ± 11	14.17 ± 2.75
NHERD	0.108 ± 0.010	5258 ± 415	335.36 ± 24.02	0.025 ± 0.001	3,316 ± 201	33.25 ± 1.99
AROW	0.036 ± 0.001	4519 ± 241	288.43 ± 14.28	0.027 ± 0.001	1,884 ± 134	19.03 ± 1.34
NAROW	0.038 ± 0.002	5819 ± 356	372.04 ± 21.92	0.031 ± 0.002	2,389 ± 215	24.18 ± 2.15
SCW-I	<b>0.011 ± 0.001</b>	868 ± 22	68.50 ± 1.39	0.025 ± 0.001	756 ± 14	8.13 ± 0.17
SCW-II	<b>0.011 ± 0.001</b>	742 ± 34	60.90 ± 2.08	<b>0.024 ± 0.001</b>	774 ± 20	8.32 ± 0.20

Algorithm	mushrooms			covtype		
	Mistake Rate	#Updates	Time(s)	Mistake Rate	#Updates	Time(s)
Perceptron	0.014 ± 0.001	112 ± 8	0.04 ± 0.00	0.470 ± 0.000	273,185 ± 193	234.81 ± 0.88
ROMMA	0.015 ± 0.002	123 ± 14	0.08 ± 0.01	0.472 ± 0.005	274,049 ± 3,147	239.12 ± 7.65
agg-ROMMA	0.011 ± 0.002	327 ± 32	0.09 ± 0.00	0.469 ± 0.006	272,272 ± 3195	237.18 ± 6.89
PA-I	0.006 ± 0.001	689 ± 19	0.06 ± 0.00	0.483 ± 0.001	417,224 ± 113	568.36 ± 2.23
PA-II	0.006 ± 0.001	723 ± 18	0.06 ± 0.00	0.483 ± 0.001	417,224 ± 113	568.62 ± 1.74
SOP	0.004 ± 0.000	36 ± 2	2.09 ± 0.06	0.337 ± 0.001	195,880 ± 427	241.29 ± 1.19
CW	<b>0.002 ± 0.000</b>	315 ± 18	0.289 ± 0.005	0.405 ± 0.001	389,870 ± 2,278	879.78 ± 9.01
IELLIP	0.009 ± 0.001	70 ± 4	0.23 ± 0.02	0.482 ± 0.002	280,320 ± 1,148	362.67 ± 5.66
NHERD	<b>0.002 ± 0.001</b>	3724 ± 448	1.24 ± 0.12	0.259 ± 0.002	521,225 ± 12,104	1,166.93 ± 34.72
AROW	<b>0.002 ± 0.000</b>	1815 ± 185	0.66 ± 0.05	0.243 ± 0.000	531,187 ± 455	1,193.21 ± 4.85
NAROW	<b>0.002 ± 0.000</b>	3340 ± 386	1.13 ± 0.10	0.367 ± 0.009	546,704 ± 8,814	1,269.77 ± 45.75
SCW-I	<b>0.002 ± 0.000</b>	327 ± 21	0.28 ± 0.06	<b>0.233 ± 0.000</b>	238,415 ± 1,917	264.84 ± 2.50
SCW-II	<b>0.002 ± 0.000</b>	152 ± 5	0.24 ± 0.01	0.239 ± 0.000	451,193 ± 3,783	881.01 ± 9.47

Algorithm	Synthetic Data			Synthetic Data with 0.1 Noise		
	Mistake Rate	#Updates	Time(s)	Mistake Rate	#Updates	Time(s)
Perceptron	0.293 ± 0.005	1,467 ± 24	0.02 ± 0.00	0.368 ± 0.005	1,840 ± 23	0.02 ± 0.00
ROMMA	0.048 ± 0.003	240 ± 17	0.04 ± 0.00	0.251 ± 0.005	1,256 ± 23	0.05 ± 0.00
agg-ROMMA	0.047 ± 0.003	243 ± 15	0.04 ± 0.00	0.251 ± 0.005	1,261 ± 23	0.05 ± 0.00
PA-I	0.251 ± 0.006	3,268 ± 23	0.05 ± 0.00	0.355 ± 0.006	3,463 ± 20	0.05 ± 0.00
PA-II	0.251 ± 0.006	3,268 ± 23	0.05 ± 0.00	0.355 ± 0.006	3,463 ± 20	0.05 ± 0.00
SOP	0.031 ± 0.003	156 ± 12	0.10 ± 0.00	0.256 ± 0.004	1,281 ± 20	0.12 ± 0.00
CW	<b>0.017 ± 0.001</b>	262 ± 6	0.07 ± 0.00	0.293 ± 0.005	2,811 ± 33	0.12 ± 0.00
IELLIP	0.256 ± 0.006	1,278 ± 31	0.09 ± 0.00	0.367 ± 0.006	1,835 ± 31	0.15 ± 0.00
NHERD	0.120 ± 0.016	3,202 ± 292	0.13 ± 0.00	0.208 ± 0.017	4,114 ± 150	0.15 ± 0.00
AROW	0.026 ± 0.003	2,128 ± 167	0.08 ± 0.00	<b>0.133 ± 0.003</b>	4,122 ± 60	0.13 ± 0.00
NAROW	0.101 ± 0.019	3,302 ± 315	0.12 ± 0.00	0.236 ± 0.024	4,242 ± 171	0.15 ± 0.00
SCW-I	0.018 ± 0.001	317 ± 8	0.07 ± 0.00	0.135 ± 0.002	1,373 ± 38	0.09 ± 0.00
SCW-II	0.020 ± 0.001	326 ± 6	0.07 ± 0.00	0.145 ± 0.005	2,138 ± 211	0.11 ± 0.00



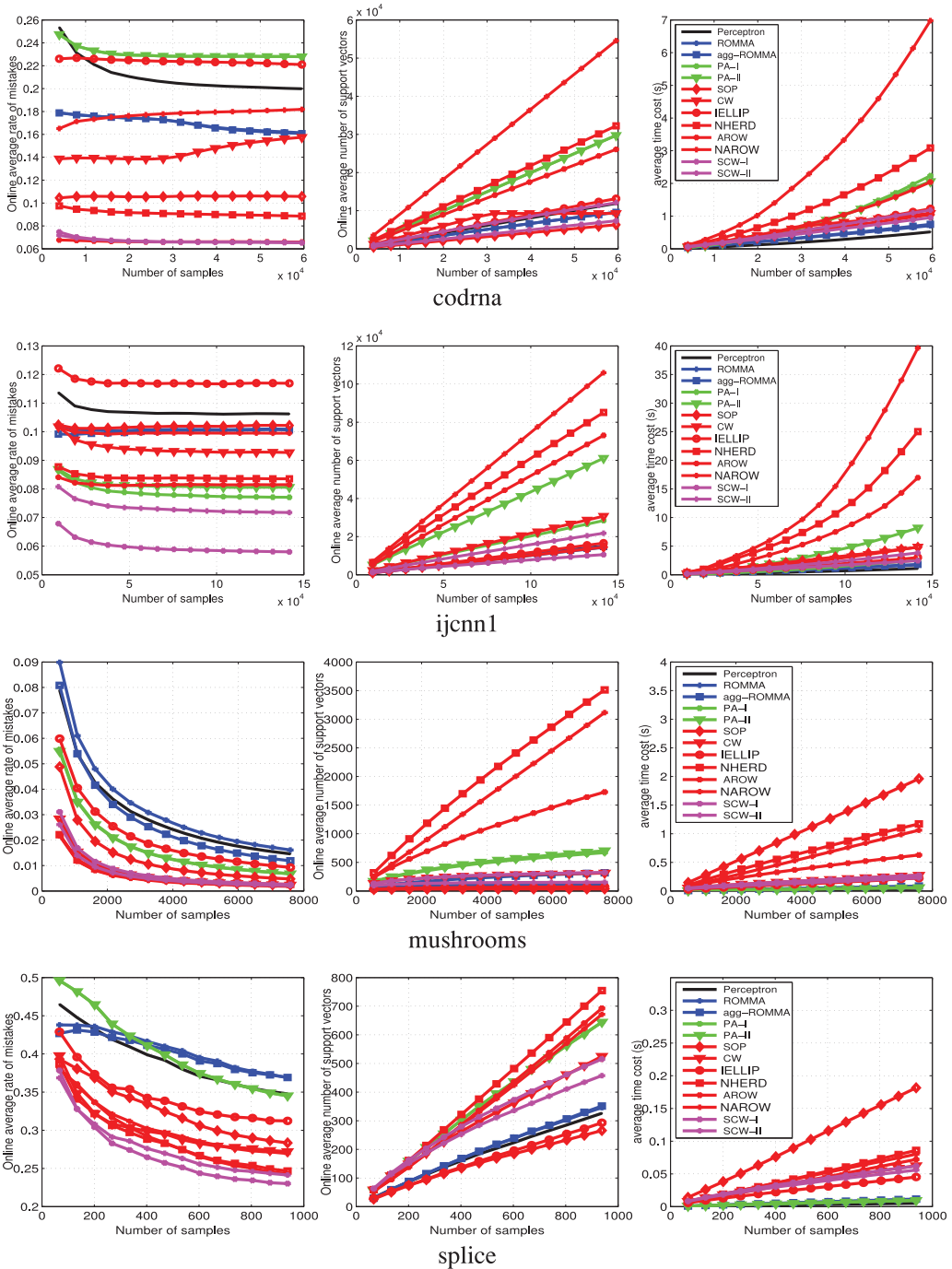
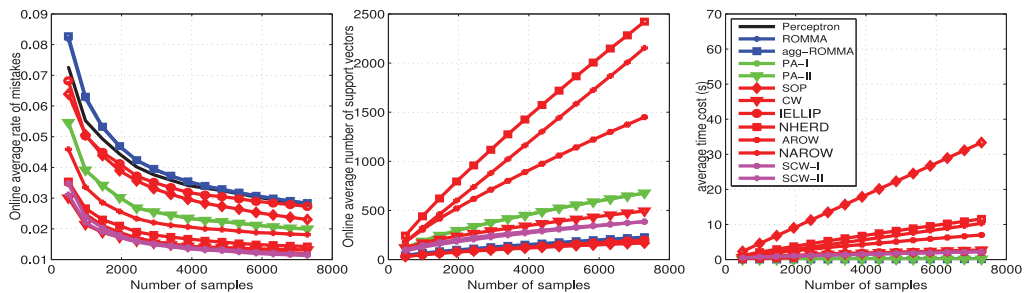
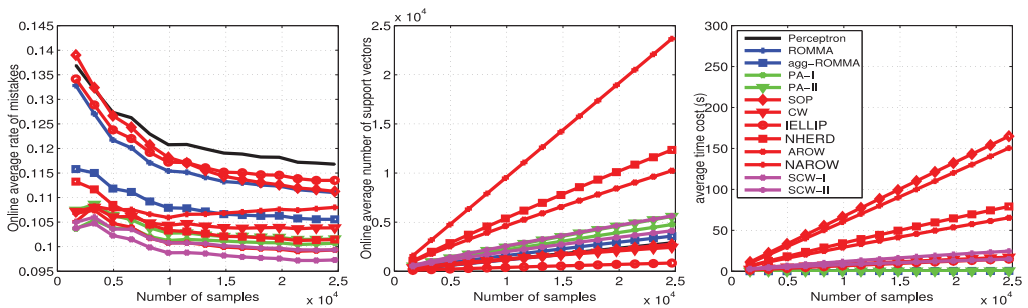


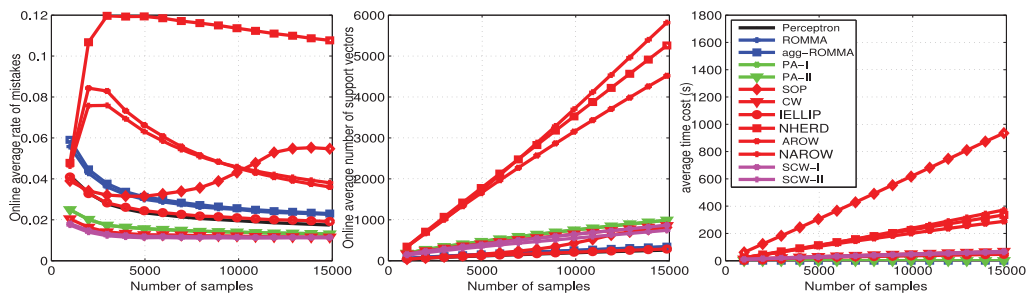
Fig. 1. Evaluation of SCW.



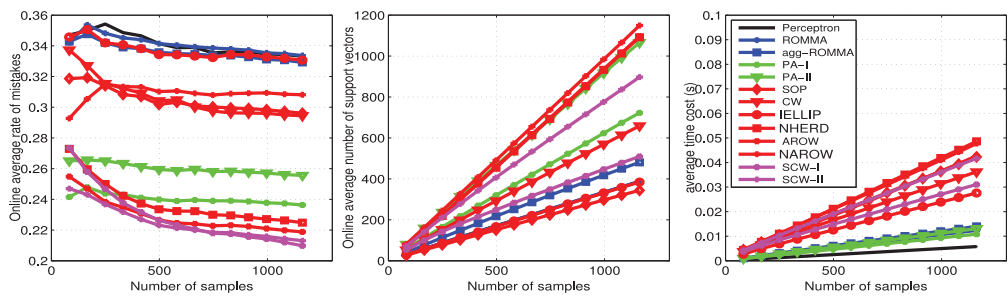
usps "1" vs. "all"



w7a



mnist "1" vs. "2"



svmguide3

Fig. 2. Evaluation of SCW.

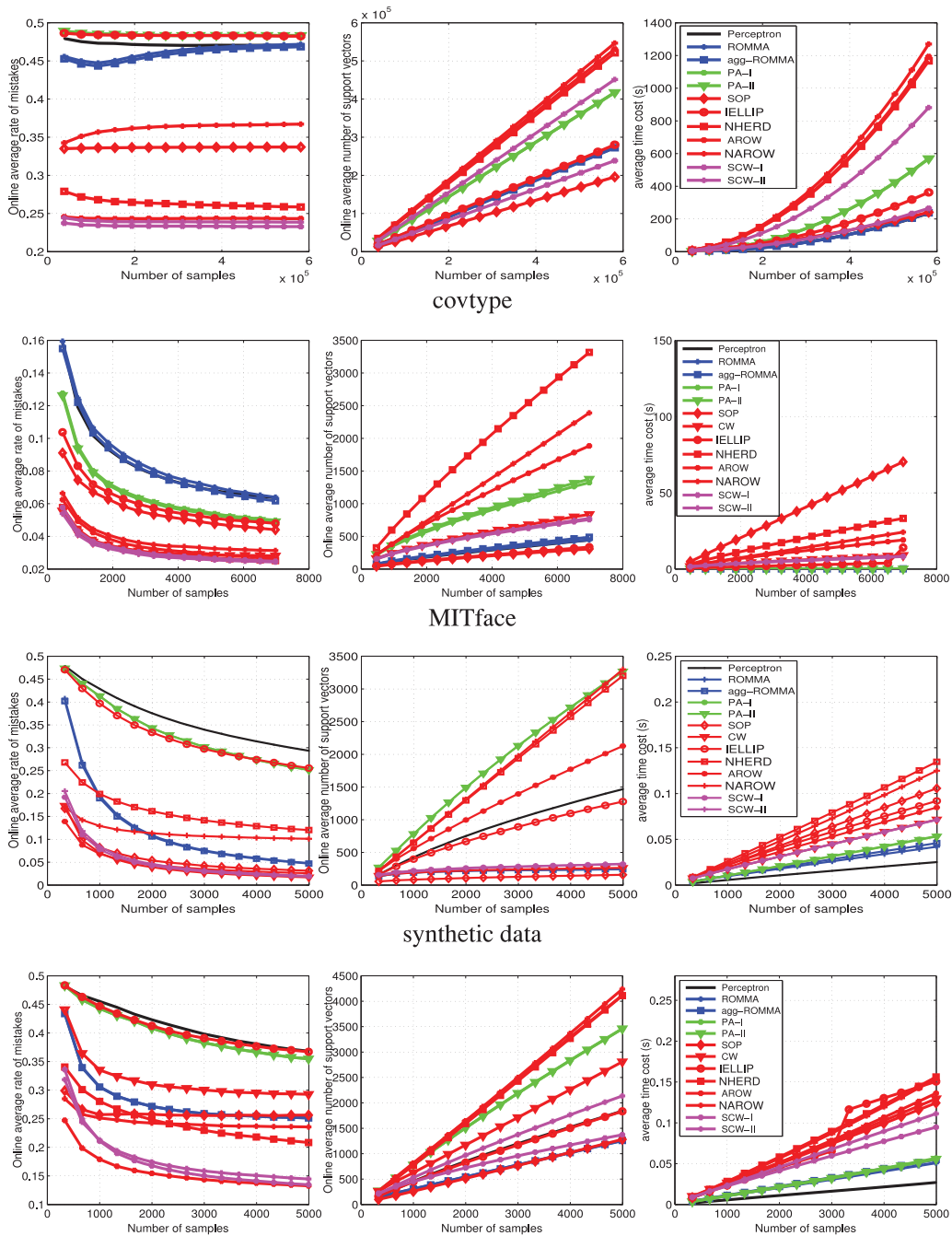


Fig. 3. Evaluation of SCW.

Table V. List of Datasets for Online Multiclass Learning Used in the Experiments

Dataset	# Training Examples	# Classes	# Features
glass	214	6	9
svmguide2	391	3	20
vowel	528	11	10
ucidigits	1,797	10	64
dna	2,000	3	180
segment	2,130	7	19
satimage	4,335	6	36
usps	7,291	10	256
letter	15,000	26	16
protein	17,766	3	357
poker	25,010	10	10
shuttle	43,500	7	9

## 5.2. Empirical Evaluation on Online Multiclass Learning

5.2.1. *Datasets and Compared Algorithms.* We adopt a variety of datasets from different domains:

- Digital recognition: We use two benchmarks: “USPS”<sup>5</sup> and “UCI digits.”<sup>6</sup>
- Machine-learning datasets: We randomly choose several public machine learning datasets from LIBSVM.<sup>7</sup>

Table V shows the statistics of the list of datasets used.

We compare the empirical performance of the following online multiclass learning algorithms:

- PerceptronM**: The Perceptron method based on the max-score multiclass update [Crammer and Singer 2003]
- PerceptronU**: The Perceptron method based on the uniform multiclass update [Crammer and Singer 2003]
- PerceptronS**: The Perceptron method based on the similarity-score multiclass update [Crammer and Singer 2003]
- MROMMA, MaROMMA**: The Multiclass ROMMA algorithm and its aggressive version [Li and Long 2002]
- MOGD**: The Multiclass Online Gradient Descent algorithm [Zinkevich 2003]
- PAM, PAM1, PAM2**: The Multiclass Passive Aggressive algorithms [Crammer et al. 2006]
- MCW**: The Multiprototype modification version of the Multiclass Confidence Weight Learning algorithm [Crammer et al. 2009]
- MAROW**: The Multiclass Adaptive Regularization of Weight algorithm [Crammer et al. 2013]
- MSCW1, MSCW2**: The proposed Multiprototype modification version of the Multiclass Soft Confidence Weight Learning algorithms

The parameters  $C$  in PAM1, PAM2, MSCW1, and MSCW2 are all determined by cross-validation to select the best one from  $\{2^{-4}, 2^{-3}, \dots, 2^3, 2^4\}$ ; the parameters  $\eta$  in

<sup>5</sup><http://www-i6.informatik.rwth-aachen.de/~keyzers/usps.html>.

<sup>6</sup><http://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits>.

<sup>7</sup><http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/>.

MSCW1 and MSCW2 are determined by cross-validation to select the best one from  $\{0.5, 0.55, \dots, 0.9, 0.95\}$ . After the best parameters were determined, all the experiments were conducted over 20 random permutations for each dataset. All the results were reported by averaging over these 20 runs, and the source codes of all the algorithms are available at Hoi et al. [2014]. We evaluate the performance by three metrics: (1) online cumulative mistake rate, (2) number of updates (which would be closely related to the potential number of support vectors in kernel extension), and (3) running time cost.

*5.2.2. Experimental Results.* Tables VI and VII summarize the results of our empirical evaluation for various kinds of online multiclass learning algorithms. The **bold** elements indicate the best performance with paired t-test at the 95% significance level. We can draw several observations as follows.

First of all, by examining the overall mistakes, we found that second-order algorithms usually outperform first-order algorithms (e.g., MCW, MSCW vs. PAM, and Perceptron-based algorithms), and margin-based algorithms usually outperform non-margin-based methods (e.g., PAM vs. Perceptron-based algorithms). This again shows the efficacy of “Large Margin” and “Confidence” properties for learning better classifiers in the multiclass setting.

Second, by examining the original CW algorithm, we found that it significantly outperforms the first-order algorithms (e.g., PAM and Perceptron-based algorithms) on the datasets without noise (e.g., segment) but fails to outperform the first-order algorithms on some real-world datasets that often have noisy data (e.g., letter and poker). This empirical result verifies the importance of the “Handling Nonseparable” property in producing robust classifiers when dealing with noisy data.

Moreover, MSCW often achieves significantly better accuracy performance than the other algorithms, and the number of updates and running time cost of MSCW are comparable with the MCW algorithm.

Finally, Figures 4, 5, and 6 show the online results of eight algorithms with respect to varied numbers of samples in the online learning process. The results again validate the advantages of MSCW in both efficacy and efficiency among all of the state-of-the-art algorithms. We also observe an interesting phenomenon of the MCW algorithm on the letter dataset: the online mistake rate first decreases and then increases during the online learning process, which might be due to the letter dataset being hard to separate and the hard constraint on the MCW algorithm making it update too aggressively so that the classification accuracy drops during part of the online learning process. This again verifies the necessity of developing the soft confidence-weighted learning algorithms.

## 6. CONCLUSION

This article proposed Soft Confidence-Weighted (SCW) learning, a new second-order online learning method with state-of-the-art empirical performance. Unlike the existing second-order algorithms, SCW enjoys the following four properties: (1) large margin training, (2) confidence weighting, (3) adaptive margin, and (4) capability of handling nonseparable data. Empirically, we found that the proposed SCW algorithms perform significantly better than the original CW algorithm and outperform the state-of-the-art AROW algorithm for most cases in terms of both accuracy and efficiency. Future work will conduct more in-depth analysis of the mistake bounds and its extension to other problems, such as multidomain learning [Dredze et al. 2010].

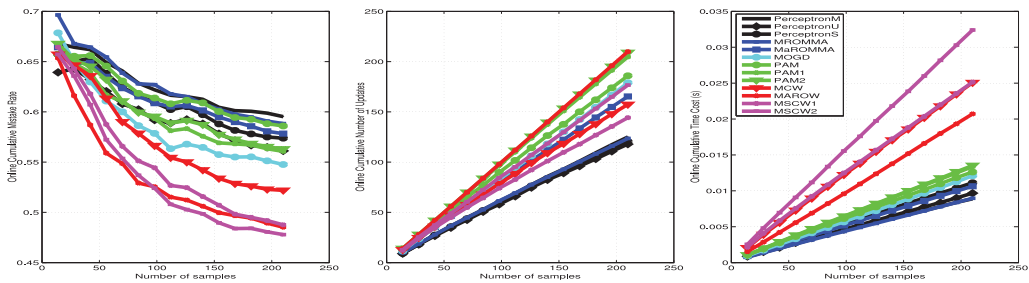
Table VI. Evaluation of the Classification Performance of Multiclass SCW

Algorithm	glass			svmguide2		
	Mistake Rate	#Updates	Time(s)	Mistake Rate	#Updates	Time(s)
PerceptronM	0.595 ± 0.025	127.3 ± 5.3	0.009 ± 0.001	0.401 ± 0.022	156.8 ± 8.6	0.017 ± 0.005
PerceptronU	0.561 ± 0.025	120.1 ± 5.3	0.010 ± 0.001	0.406 ± 0.017	158.7 ± 6.5	0.019 ± 0.005
PerceptronS	0.573 ± 0.027	122.7 ± 5.7	0.011 ± 0.001	0.417 ± 0.023	163.2 ± 8.9	0.019 ± 0.004
MROMMA	0.588 ± 0.027	125.9 ± 5.9	0.009 ± 0.001	0.373 ± 0.021	146.0 ± 8.1	0.016 ± 0.004
MaROMMA	0.578 ± 0.035	168.2 ± 11.0	0.011 ± 0.001	0.354 ± 0.016	258.3 ± 8.6	0.019 ± 0.003
MOGD	0.547 ± 0.027	182.4 ± 4.3	0.012 ± 0.001	0.440 ± 0.005	391.0 ± 0.0	0.024 ± 0.004
PAM	0.586 ± 0.027	189.3 ± 4.6	0.013 ± 0.001	0.402 ± 0.016	295.4 ± 8.1	0.023 ± 0.005
PAM1	0.561 ± 0.016	208.5 ± 1.7	0.014 ± 0.001	0.376 ± 0.020	311.5 ± 7.8	0.022 ± 0.000
PAM2	0.563 ± 0.034	212.9 ± 1.1	0.014 ± 0.001	0.375 ± 0.021	361.3 ± 4.1	0.023 ± 0.001
MCW	0.520 ± 0.023	160.2 ± 5.6	0.026 ± 0.002	0.308 ± 0.015	207.7 ± 7.2	0.038 ± 0.001
MAROW	0.486 ± 0.019	213.9 ± 0.3	0.021 ± 0.002	0.230 ± 0.011	369.8 ± 4.1	0.038 ± 0.001
MSCW1	0.487 ± 0.025	146.7 ± 5.7	0.026 ± 0.002	<b>0.221 ± 0.014</b>	172.7 ± 3.6	0.036 ± 0.001
MSCW2	<b>0.477 ± 0.024</b>	180.0 ± 4.5	0.033 ± 0.003	0.222 ± 0.013	257.7 ± 10.9	0.051 ± 0.001
Algorithm	vowel			ucidigits		
	Mistake Rate	#Updates	Time(s)	Mistake Rate	#Updates	Time(s)
PerceptronM	0.781 ± 0.016	412.3 ± 8.4	0.023 ± 0.005	0.203 ± 0.007	364.8 ± 11.8	0.065 ± 0.007
PerceptronU	0.768 ± 0.012	405.4 ± 6.5	0.031 ± 0.009	0.196 ± 0.005	353.1 ± 9.5	0.070 ± 0.008
PerceptronS	0.769 ± 0.013	406.1 ± 6.7	0.033 ± 0.005	0.198 ± 0.006	356.2 ± 10.7	0.075 ± 0.007
MROMMA	0.844 ± 0.053	445.5 ± 28.0	0.026 ± 0.005	0.248 ± 0.019	445.8 ± 33.3	0.068 ± 0.006
MaROMMA	0.740 ± 0.021	487.0 ± 6.8	0.029 ± 0.005	0.247 ± 0.019	448.8 ± 33.3	0.082 ± 0.004
MOGD	0.725 ± 0.018	468.4 ± 6.1	0.031 ± 0.005	0.145 ± 0.009	262.9 ± 16.0	0.076 ± 0.006
PAM	0.767 ± 0.014	516.5 ± 3.3	0.034 ± 0.004	0.144 ± 0.007	1,112.5 ± 15.6	0.093 ± 0.009
PAM1	0.765 ± 0.014	516.8 ± 2.9	0.033 ± 0.002	0.144 ± 0.007	1,112.5 ± 15.6	0.093 ± 0.004
PAM2	0.766 ± 0.015	521.1 ± 2.3	0.033 ± 0.002	0.144 ± 0.007	1,112.5 ± 15.6	0.095 ± 0.006
MCW	0.630 ± 0.016	422.7 ± 8.1	0.063 ± 0.003	0.077 ± 0.003	439.9 ± 9.4	0.168 ± 0.004
MAROW	0.612 ± 0.015	527.0 ± 0.0	0.050 ± 0.003	0.127 ± 0.008	806.9 ± 42.8	0.200 ± 0.007
MSCW1	0.592 ± 0.020	466.3 ± 5.2	0.070 ± 0.004	0.077 ± 0.003	439.9 ± 9.4	0.174 ± 0.008
MSCW2	<b>0.589 ± 0.018</b>	471.4 ± 7.2	0.083 ± 0.004	<b>0.075 ± 0.004</b>	438.4 ± 7.8	0.182 ± 0.010
Algorithm	dna			segment		
	Mistake Rate	#Updates	Time(s)	Mistake Rate	#Updates	Time(s)
PerceptronM	0.159 ± 0.003	318.3 ± 6.6	0.077 ± 0.012	0.248 ± 0.006	572.0 ± 14.0	0.089 ± 0.005
PerceptronU	0.157 ± 0.005	314.9 ± 10.6	0.085 ± 0.006	0.239 ± 0.006	552.8 ± 13.3	0.093 ± 0.004
PerceptronS	0.159 ± 0.004	318.6 ± 8.8	0.088 ± 0.005	0.242 ± 0.007	558.7 ± 17.1	0.099 ± 0.005
MROMMA	0.202 ± 0.009	404.3 ± 17.6	0.085 ± 0.005	0.221 ± 0.009	510.0 ± 21.1	0.085 ± 0.007
MaROMMA	0.198 ± 0.009	525.6 ± 16.5	0.101 ± 0.003	0.196 ± 0.010	1,147.8 ± 86.7	0.105 ± 0.002
MOGD	0.151 ± 0.007	388.0 ± 13.9	0.094 ± 0.006	0.161 ± 0.007	951.8 ± 39.4	0.110 ± 0.006
PAM	0.114 ± 0.005	981.3 ± 11.8	0.109 ± 0.004	0.211 ± 0.007	1,362.7 ± 18.3	0.119 ± 0.006
PAM1	0.114 ± 0.005	981.3 ± 11.8	0.109 ± 0.005	0.199 ± 0.006	1,415.7 ± 11.1	0.120 ± 0.001
PAM2	0.111 ± 0.004	1,027.0 ± 11.6	0.110 ± 0.002	0.203 ± 0.005	1,611.4 ± 12.6	0.123 ± 0.004
MCW	0.105 ± 0.004	702.5 ± 10.9	0.651 ± 0.023	0.119 ± 0.004	536.2 ± 12.4	0.157 ± 0.002
MAROW	0.099 ± 0.005	1,106.3 ± 27.7	0.845 ± 0.029	0.109 ± 0.010	1,482.5 ± 78.2	0.185 ± 0.005
MSCW1	<b>0.093 ± 0.005</b>	694.8 ± 7.9	0.620 ± 0.012	<b>0.091 ± 0.005</b>	452.6 ± 10.6	0.156 ± 0.004
MSCW2	<b>0.093 ± 0.005</b>	748.3 ± 11.8	0.682 ± 0.016	0.094 ± 0.004	584.5 ± 19.7	0.187 ± 0.003

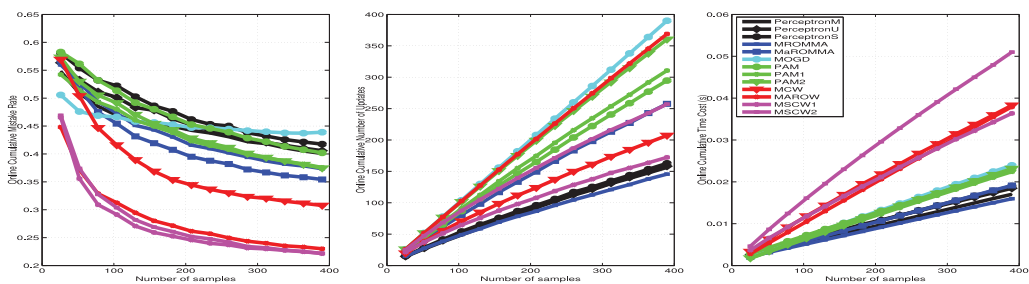
Table VII. Evaluation of the Classification Performance of Multiclass SCW

Algorithm	satimage			usps		
	Mistake Rate	#Updates	Time(s)	Mistake Rate	#Updates	Time(s)
PerceptronM	0.221 ± 0.004	978.4 ± 15.8	0.157 ± 0.009	0.152 ± 0.002	1,105.2 ± 15.3	0.306 ± 0.009
PerceptronU	0.213 ± 0.004	943.5 ± 17.4	0.178 ± 0.010	0.143 ± 0.002	1,040.8 ± 12.7	0.342 ± 0.018
PerceptronS	0.216 ± 0.004	958.3 ± 18.6	0.187 ± 0.008	0.145 ± 0.002	1,059.8 ± 16.5	0.355 ± 0.017
MROMMA	0.232 ± 0.004	1,030.0 ± 16.9	0.166 ± 0.009	0.162 ± 0.009	1,178.8 ± 62.6	0.344 ± 0.019
MaROMMA	0.217 ± 0.004	1,803.1 ± 28.1	0.203 ± 0.010	0.154 ± 0.008	1,538.0 ± 88.5	0.501 ± 0.008
MOGD	0.166 ± 0.002	1,580.1 ± 43.0	0.209 ± 0.007	0.106 ± 0.003	897.3 ± 28.6	0.368 ± 0.009
PAM	0.229 ± 0.003	2,059.4 ± 19.1	0.215 ± 0.009	0.119 ± 0.002	3,128.9 ± 26.2	0.428 ± 0.011
PAM1	0.175 ± 0.003	2,016.8 ± 18.2	0.216 ± 0.008	0.119 ± 0.002	3,128.9 ± 26.2	0.426 ± 0.010
PAM2	0.188 ± 0.003	2,626.2 ± 23.4	0.229 ± 0.007	0.119 ± 0.003	3,130.7 ± 26.6	0.427 ± 0.010
MCW	0.195 ± 0.004	1,391.6 ± 20.0	0.428 ± 0.016	0.083 ± 0.002	1,463.6 ± 15.9	2.790 ± 0.047
MAROW	0.173 ± 0.005	3,264.8 ± 79.9	0.542 ± 0.019	0.075 ± 0.002	4,303.3 ± 119.7	6.734 ± 0.188
MSCW1	<b>0.155 ± 0.002</b>	1,168.8 ± 25.5	0.406 ± 0.011	<b>0.065 ± 0.001</b>	1,181.5 ± 14.2	2.364 ± 0.057
MSCW2	0.157 ± 0.003	1,726.8 ± 48.0	0.539 ± 0.019	0.068 ± 0.002	1,315.7 ± 20.1	2.622 ± 0.056
Algorithm	letter			protein		
	Mistake Rate	#Updates	Time(s)	Mistake Rate	#Updates	Time(s)
PerceptronM	0.489 ± 0.003	7,331.0 ± 40.9	0.668 ± 0.032	0.429 ± 0.002	7,622.3 ± 41.9	0.907 ± 0.025
PerceptronU	0.453 ± 0.002	6,792.7 ± 35.9	0.749 ± 0.027	0.415 ± 0.002	7,381.4 ± 38.4	0.993 ± 0.031
PerceptronS	0.463 ± 0.003	6,950.4 ± 49.2	0.808 ± 0.021	0.423 ± 0.002	7,518.8 ± 42.0	1.081 ± 0.036
MROMMA	0.549 ± 0.003	8,232.5 ± 44.0	0.672 ± 0.022	0.430 ± 0.003	7,647.9 ± 50.9	1.039 ± 0.021
MaROMMA	0.518 ± 0.004	12,784.4 ± 46.6	0.836 ± 0.024	0.424 ± 0.002	9,931.0 ± 96.4	1.251 ± 0.032
MOGD	0.405 ± 0.003	14,422.3 ± 39.2	0.897 ± 0.039	0.346 ± 0.003	12,157.2 ± 67.1	1.198 ± 0.03
PAM	0.530 ± 0.004	13,053.6 ± 35.0	0.924 ± 0.027	0.429 ± 0.002	13,078.0 ± 42.9	1.288 ± 0.024
PAM1	0.410 ± 0.002	13,267.5 ± 29.0	0.925 ± 0.022	0.382 ± 0.003	13,104.2 ± 35.2	1.294 ± 0.033
PAM2	0.434 ± 0.002	14,599.2 ± 19.1	0.967 ± 0.024	0.404 ± 0.002	14,743.4 ± 34.6	1.360 ± 0.036
MCW	0.514 ± 0.012	9,705.5 ± 157.3	1.693 ± 0.069	0.432 ± 0.002	11,751.5 ± 51.2	38.753 ± 0.725
MAROW	0.355 ± 0.003	15,000.0 ± 0.0	1.510 ± 0.032	0.343 ± 0.002	17,017.1 ± 28.9	54.043 ± 1.147
MSCW1	<b>0.286 ± 0.002</b>	8,205.0 ± 43.6	1.604 ± 0.053	<b>0.338 ± 0.002</b>	12,136.5 ± 58.8	39.897 ± 0.850
MSCW2	0.304 ± 0.003	11,506.7 ± 45.1	2.234 ± 0.091	0.349 ± 0.002	11,850.2 ± 103.1	39.185 ± 1.164
Algorithm	poker			shuttle		
	Mistake Rate	#Updates	Time(s)	Mistake Rate	#Updates	Time(s)
PerceptronM	0.568 ± 0.003	14,198.0 ± 73.2	0.991 ± 0.030	0.068 ± 0.001	2,977.6 ± 23.9	1.503 ± 0.033
PerceptronU	0.544 ± 0.001	13,613.4 ± 34.5	1.138 ± 0.019	0.065 ± 0.001	2,810.2 ± 35.3	1.558 ± 0.048
PerceptronS	0.554 ± 0.002	13,856.2 ± 49.2	1.283 ± 0.025	0.067 ± 0.001	2,908.6 ± 29.9	1.588 ± 0.031
MROMMA	0.591 ± 0.005	14,769.3 ± 118.9	1.063 ± 0.026	0.070 ± 0.001	3,037.2 ± 38.6	1.429 ± 0.032
MaROMMA	0.586 ± 0.007	19,749.2 ± 735.3	1.252 ± 0.025	0.066 ± 0.001	5,320.7 ± 299.0	1.646 ± 0.072
MOGD	0.509 ± 0.002	24,666.5 ± 40.9	1.476 ± 0.023	0.086 ± 0.002	12,370.6 ± 142.6	1.878 ± 0.110
PAM	0.569 ± 0.002	20,967.4 ± 52.5	1.448 ± 0.026	0.068 ± 0.001	7,971.4 ± 65.5	1.691 ± 0.072
PAM1	0.527 ± 0.003	24,124.3 ± 38.4	1.537 ± 0.028	0.043 ± 0.001	7,742.9 ± 32.2	1.699 ± 0.025
PAM2	0.544 ± 0.003	24,946.5 ± 12.3	1.564 ± 0.023	0.049 ± 0.000	14,367.8 ± 97.7	1.883 ± 0.036
MCW	0.639 ± 0.106	16,025.7 ± 2642.9	2.811 ± 0.304	0.054 ± 0.001	4,733.7 ± 91.1	2.348 ± 0.033
MAROW	0.508 ± 0.001	25,010.0 ± 0.0	2.421 ± 0.048	0.058 ± 0.007	20,276.8 ± 1732.5	2.850 ± 0.108
MSCW1	<b>0.503 ± 0.004</b>	17,890.8 ± 1521.9	2.961 ± 0.146	<b>0.027 ± 0.002</b>	1,506.2 ± 139.6	2.160 ± 0.026
MSCW2	0.509 ± 0.004	25,010.0 ± 0.0	4.295 ± 0.060	0.052 ± 0.009	2,822.7 ± 398.4	2.413 ± 0.071

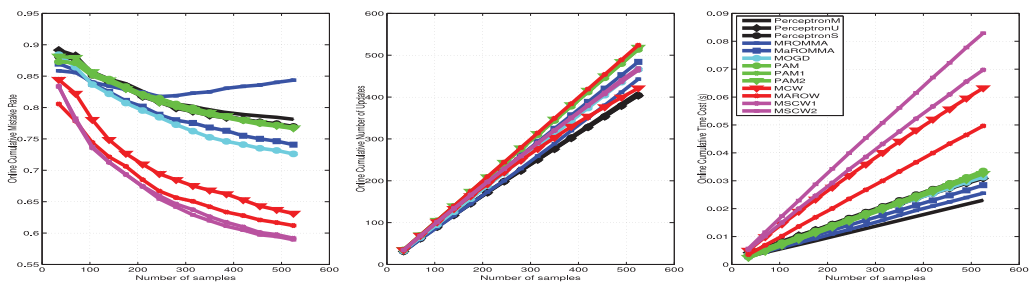




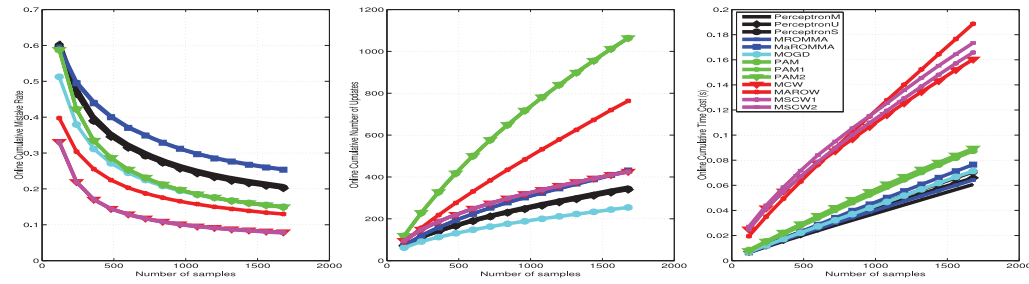
glass



svmguide2

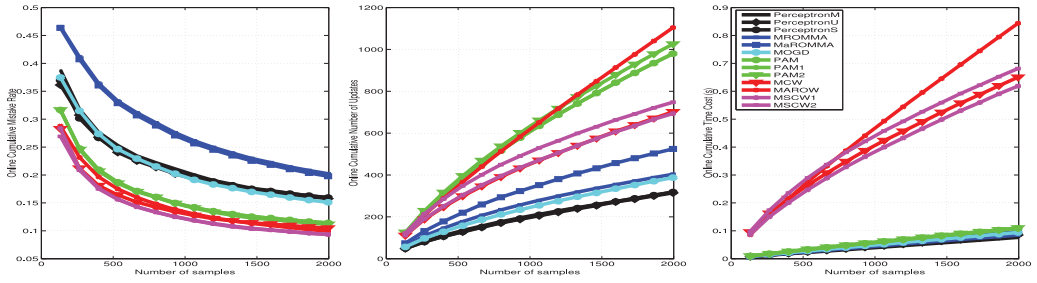


vowel

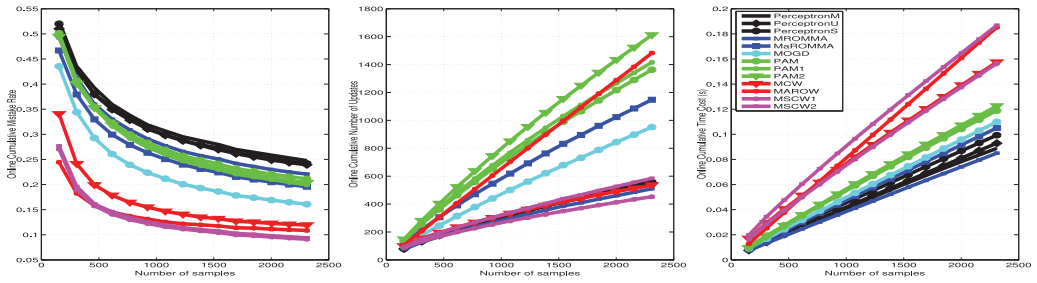


ucidigits

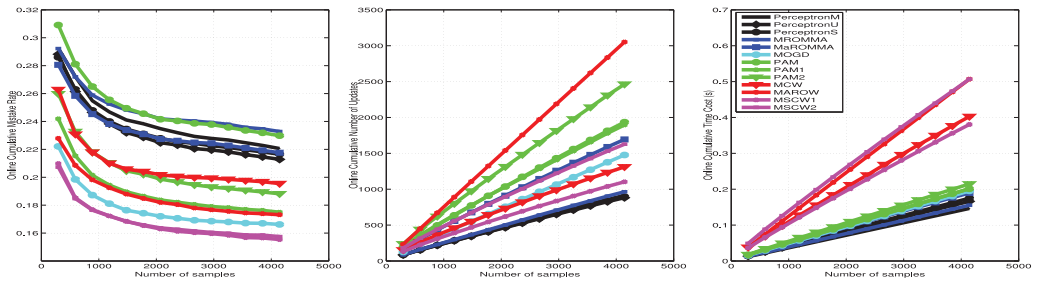
Fig. 4. Evaluation of Multiclass SCW.



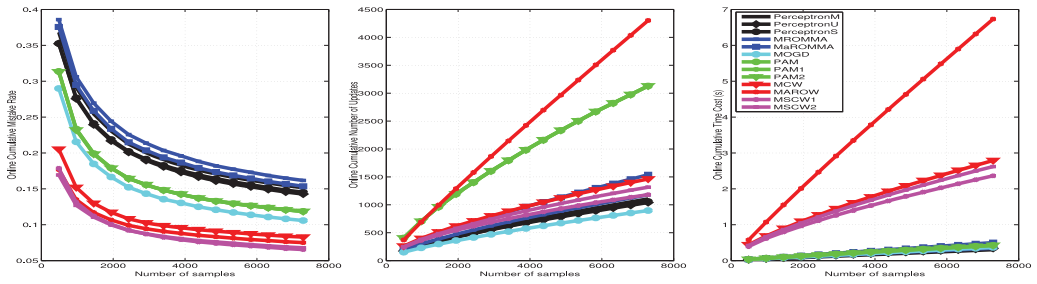
dna



segment

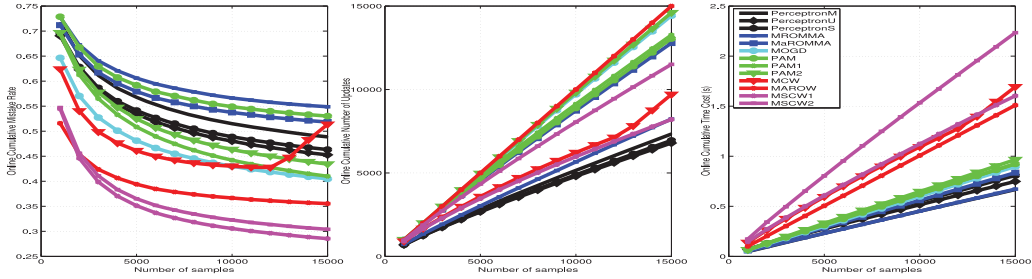


satimage

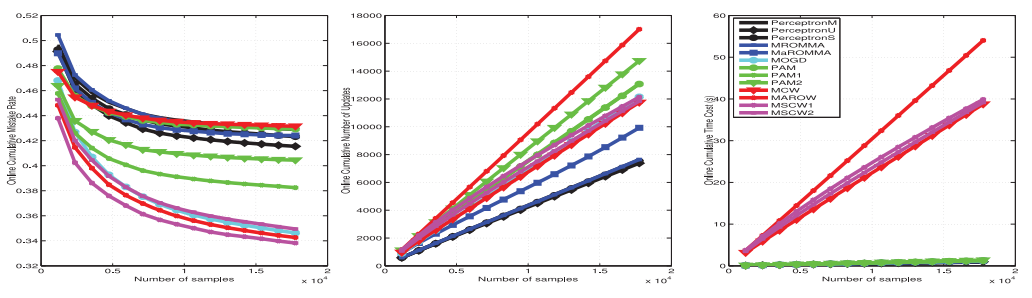


usps

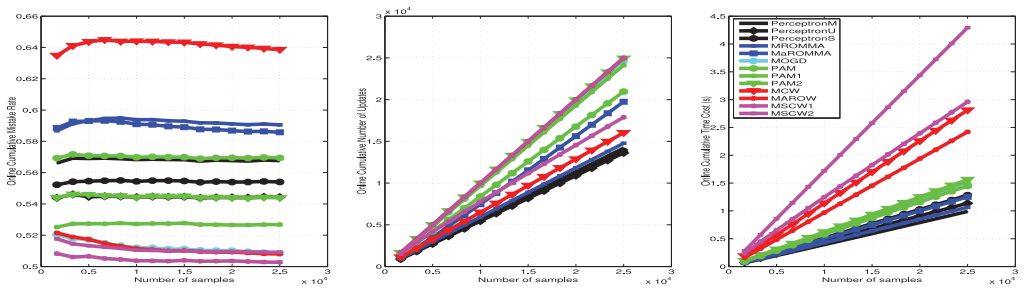
Fig. 5. Evaluation of Multiclass SCW.



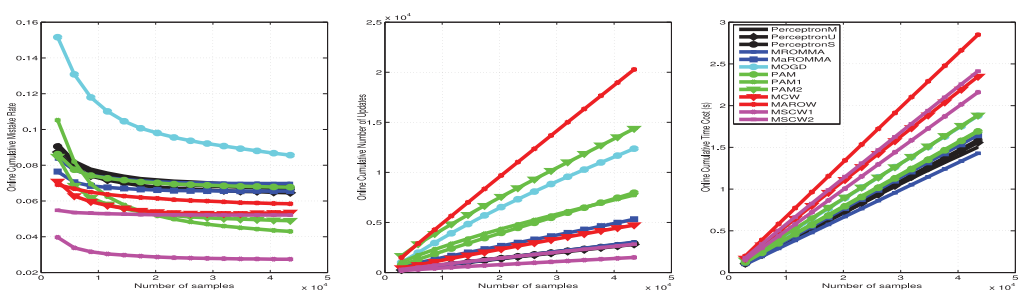
letter



protein



poker



shuttle

Fig. 6. Evaluation of Multiclass SCW.

## APPENDIX: PROOFS OF PROPOSITIONS

### Proof of Proposition 1

PROOF. First, when  $\ell^\phi(\mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t); (\mathbf{x}_t, y_t)) = 0$ , it is easy to see the solution is valid. When  $\ell^\phi(\mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t); (\mathbf{x}_t, y_t)) > 0$ , it is easy to see the optimization problem is equivalent to

$$D_{KL}(\mathcal{N}(\boldsymbol{\mu}, \Sigma) \parallel \mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t)) + C\xi, \\ \text{s.t. } \ell^\phi(\mathcal{N}(\boldsymbol{\mu}, \Sigma); (\mathbf{x}_t, y_t)) \leq \xi, \quad \text{and } \xi \geq 0.$$

Though the problem is only quasi-convex, we are able to transform the original problem to a convex problem by the method introduced in Crammer et al. [2008]. Since  $\Sigma$  is positive semidefinite (PSD), let  $\Sigma = Q \text{diag}(\lambda_1, \dots, \lambda_d) Q^T$  be the eigendecomposition of  $\Sigma$ , and then  $\Sigma$  can be written as  $\Sigma = \Upsilon^2$ , where  $\Upsilon = Q \text{diag}(\sqrt{\lambda_1}, \dots, \sqrt{\lambda_d}) Q^T$ . Now the problem is convex in  $\boldsymbol{\mu}$  and  $\Upsilon$  jointly. But for convenience, we will still use  $\Sigma$  instead of  $\Upsilon^2$  in the following analysis. The Lagrangian of the previous optimization is

$$\begin{aligned} \mathcal{L}(\boldsymbol{\mu}, \Sigma, \xi, \tau, \lambda) &= D_{KL}(\mathcal{N}(\boldsymbol{\mu}, \Sigma) \parallel \mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t)) + C\xi + \tau \left( \phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - y_t \boldsymbol{\mu} \cdot \mathbf{x}_t - \xi \right) - \lambda \xi \\ &= D_{KL}(\mathcal{N}(\boldsymbol{\mu}, \Sigma) \parallel \mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t)) + \xi(C - \tau - \lambda) + \tau \left( \phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - y_t \boldsymbol{\mu} \cdot \mathbf{x}_t \right) \\ &= \frac{1}{2} \log \left( \frac{\det \Sigma_t}{\det \Sigma} \right) + \frac{1}{2} \text{Tr}(\Sigma_t^{-1} \Sigma) + \frac{1}{2} (\boldsymbol{\mu}_t - \boldsymbol{\mu})^\top \Sigma_t^{-1} (\boldsymbol{\mu}_t - \boldsymbol{\mu}) - \frac{d}{2} + \xi(C - \tau - \lambda) \\ &\quad + \tau \left( \phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - y_t \boldsymbol{\mu} \cdot \mathbf{x}_t \right), \end{aligned}$$

where  $\tau \geq 0$  and  $\lambda \geq 0$  are Lagrange multipliers. We now find the minimum of the Lagrangian with respect to the primal variables  $\boldsymbol{\mu}_r$ ,  $\boldsymbol{\mu}_s$ ,  $\Sigma$ , and  $\xi$ :

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_{t+1}} &= \Sigma_t^{-1} (\boldsymbol{\mu}_{t+1} - \boldsymbol{\mu}_t) + \tau (-y_t \mathbf{x}_t) = 0 \Rightarrow \boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t + \tau y_t \Sigma_t \mathbf{x}_t, \\ \frac{\partial \mathcal{L}}{\partial \Sigma_{t+1}} &= 0 \Rightarrow \Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \tau \phi \frac{\mathbf{x}_t \mathbf{x}_t^\top}{\sqrt{\mathbf{x}_t^\top \Sigma_{t+1} \mathbf{x}_t}}, \end{aligned}$$

and  $C - \tau - \lambda = 0$ , so  $\tau = C - \lambda \leq C$ , and thus,  $\tau \in [0, C]$ . The KKT conditions for the optimization are

$$\phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - y_t \boldsymbol{\mu} \cdot \mathbf{x}_t - \xi \leq 0, \quad -\xi \leq 0, \quad \tau, \lambda \geq 0, \quad \tau \left( \phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - y_t \boldsymbol{\mu} \cdot \mathbf{x}_t - \xi \right) = 0, \quad \lambda \xi = 0.$$

#### Case 1. $\tau \neq 0$

As  $\tau (\phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - y_t \boldsymbol{\mu} \cdot \mathbf{x}_t - \xi) = 0$  implies  $(\phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - y_t \boldsymbol{\mu} \cdot \mathbf{x}_t - \xi) = 0$ , the KKT conditions are simplified:

$$-\xi \leq 0, \quad \tau > 0, \quad \lambda \geq 0 \quad \phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - y_t \boldsymbol{\mu} \cdot \mathbf{x}_t - \xi = 0, \quad \lambda \xi = 0.$$

#### Subcase 1.1. $\lambda \neq 0$

When  $\lambda \neq 0$ ,  $\lambda \xi = 0$ , which implies  $\xi = 0$ . The KKT conditions are simplified as

$$\tau > 0, \quad \lambda > 0, \quad \xi = 0, \quad \phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - y_t \boldsymbol{\mu} \cdot \mathbf{x}_t = 0.$$

Finally, by the Sherman-Morrison formula, we have the following:

$$\Sigma_{t+1} = \left( \Sigma_t^{-1} + \tau \phi \frac{\mathbf{x}_t \mathbf{x}_t^\top}{\sqrt{\mathbf{x}_t^\top \Sigma_{t+1} \mathbf{x}_t}} \right)^{-1} = \Sigma_t - \Sigma_t \mathbf{x}_t \left( \frac{\tau \phi}{\sqrt{\mathbf{x}_t^\top \Sigma_{t+1} \mathbf{x}_t} + \tau \phi \mathbf{x}_t^\top \Sigma_t \mathbf{x}_t} \right) \mathbf{x}_t^\top \Sigma_t.$$

Let  $u_t = \mathbf{x}_t^\top \Sigma_{t+1} \mathbf{x}_t$ ,  $v_t = \mathbf{x}_t^\top \Sigma_t \mathbf{x}_t$ ,  $m_t = y_t(\boldsymbol{\mu}_t \cdot \mathbf{x}_t)$ ; multiplying by  $\mathbf{x}_t^\top$  (left) and  $\mathbf{x}_t$  (right), we get  $u_t = v_t - v_t \left( \frac{\tau \phi}{\sqrt{u_t} + \tau \phi v_t} \right) v_t$ , which can be used to solve  $u_t$ :

$$\sqrt{u_t} = \frac{-\tau \phi v_t + \sqrt{\tau^2 \phi^2 v_t^2 + 4v_t}}{2}.$$

And  $\phi \sqrt{\mathbf{x}_t^\top \Sigma_{t+1} \mathbf{x}_t} - y_t \boldsymbol{\mu}_t \cdot \mathbf{x}_t = 0$  implies  $\phi \sqrt{u_t} - m_t - \tau v_t = 0$ . Thus,  $\phi \frac{-\tau \phi v_t + \sqrt{\tau^2 \phi^2 v_t^2 + 4v_t}}{2} - m_t - \tau v_t = 0$ , which can be rearranged as  $v_t^2(1 + \phi^2)\tau^2 + 2m_t v_t(1 + \frac{\phi^2}{2})\tau + (m_t^2 - \phi^2 v_t)$ . The larger root is then

$$\tau = \frac{-m_t v_t(1 + \frac{\phi^2}{2}) + \sqrt{m_t^2 v_t^2(1 + \frac{\phi^2}{2})^2 - v_t^2(1 + \phi^2)(m_t^2 - \phi^2 v_t)}}{v_t^2(1 + \phi^2)}.$$

If  $\tau \in (0, C)$ , then  $\lambda = C - \tau \in (0, C)$ .

**Subcase 1.2.**  $\lambda = 0$

$C - \tau - \lambda = 0$  implies  $\tau = C$ . The KKT conditions can be simplified as

$$-\xi \leq 0, \quad \tau = C, \quad \lambda = 0, \quad \phi \sqrt{\mathbf{x}_t^\top \Sigma_{t+1} \mathbf{x}_t} - y_t \boldsymbol{\mu}_t \cdot \mathbf{x}_t - \xi = 0.$$

We thus have

$$\phi \sqrt{\mathbf{x}_t^\top \Sigma_{t+1} \mathbf{x}_t} - y_t \boldsymbol{\mu}_t \cdot \mathbf{x}_t = \left[ \phi \frac{-\tau \phi v_t + \sqrt{\tau^2 \phi^2 v_t^2 + 4v_t}}{2} - m_t - \tau v_t \right] \Big|_{\tau=C} = \xi \geq 0.$$

It is easy to verify that

$$f'(\tau) = \frac{-\phi^2 v_t}{2} + \frac{\phi^3 v_t^2 \tau}{2\sqrt{\tau^2 \phi^2 v_t^2 + 4v_t}} - v_t = 0$$

has no solution on  $[0, +\infty)$  and  $f'(0) = \frac{-\phi^2 v_t}{2} - v_t < 0$ . As a result,  $f'(\tau) < 0$ ,  $\tau \in [0, +\infty)$ , which implies  $f(\tau)$  is decreasing on  $[0, +\infty)$ :

$$f(C) \geq 0 = f(\theta),$$

where  $\theta = \frac{-m_t v_t(1 + \frac{\phi^2}{2}) + \sqrt{m_t^2 v_t^2(1 + \frac{\phi^2}{2})^2 - v_t^2(1 + \phi^2)(m_t^2 - \phi^2 v_t)}}{v_t^2(1 + \phi^2)}$ , which thus implies  $C \leq \theta$ .

**Case 2.**  $\tau = 0$

When  $\tau = 0$ , since  $C - \tau - \lambda = 0$ ,  $\lambda = C$ , the KKT conditions are simplified as

$$\phi \sqrt{\mathbf{x}_t^\top \Sigma_{t+1} \mathbf{x}_t} - y_t \boldsymbol{\mu}_t \cdot \mathbf{x}_t \leq 0, \quad \tau = 0, \quad \lambda = C, \quad \xi = 0.$$

Thus,  $\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t$  and  $\Sigma_{t+1} = \Sigma_t$ ; as a result,  $\phi \sqrt{\mathbf{x}_t^\top \Sigma_{t+1} \mathbf{x}_t} - y_t \boldsymbol{\mu}_t \cdot \mathbf{x}_t \leq 0$ , which contradicts  $\ell^\phi(\mathcal{N}(\boldsymbol{\mu}, \Sigma); (\mathbf{x}_t, y_t)) > 0$   $\square$ .

### Proof of Proposition 2

PROOF. For SCW-II, the Lagrangian of the optimization is

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\mu}, \Sigma, \xi, \tau, \lambda) &= D_{KL}(\mathcal{N}(\boldsymbol{\mu}, \Sigma) \parallel \mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t)) + C\xi^2 + \tau \left( \phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - y_t \boldsymbol{\mu} \cdot \mathbf{x}_t - \xi \right) - \lambda \xi \\
&= D_{KL}(\mathcal{N}(\boldsymbol{\mu}, \Sigma) \parallel \mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t)) + \xi(C\xi - \tau - \lambda) + \tau \left( \phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - y_t \boldsymbol{\mu} \cdot \mathbf{x}_t \right) \\
&= \frac{1}{2} \log \left( \frac{\det \Sigma_t}{\det \Sigma} \right) + \frac{1}{2} \text{Tr}(\Sigma_t^{-1} \Sigma) + \frac{1}{2} (\boldsymbol{\mu}_t - \boldsymbol{\mu})^\top \Sigma_t^{-1} (\boldsymbol{\mu}_t - \boldsymbol{\mu}) - \frac{d}{2} + \xi(C\xi - \tau - \lambda) \\
&\quad + \tau (\phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - y_t \boldsymbol{\mu} \cdot \mathbf{x}_t),
\end{aligned}$$

where  $\tau \geq 0$  and  $\lambda \geq 0$  are Lagrange multipliers. We now find the minimum of the Lagrangian with respect to the primal variables  $\boldsymbol{\mu}$ ,  $\Sigma$ , and  $\xi$ :

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_{t+1}} &= \Sigma_t^{-1} (\boldsymbol{\mu}_{t+1} - \boldsymbol{\mu}_t) + \tau (-y_t \mathbf{x}_t) = 0 \Rightarrow \boldsymbol{\mu} = \boldsymbol{\mu}_t + \tau y_t \Sigma_t \mathbf{x}_t, \\
\frac{\partial \mathcal{L}}{\partial \Sigma_{t+1}} &= 0 \Rightarrow \Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \tau \phi \frac{\mathbf{x}_t \mathbf{x}_t^\top}{\sqrt{\mathbf{x}_t^\top \Sigma_{t+1} \mathbf{x}_t}},
\end{aligned}$$

and  $2C\xi - \tau - \lambda = 0$ , so  $\xi = \frac{\tau + \lambda}{2C}$ . The KKT conditions for the optimization are

$$\begin{aligned}
\phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - y_t \boldsymbol{\mu} \cdot \mathbf{x}_t - \xi &\leq 0, \quad \xi \geq 0, \quad \tau \geq 0, \quad \lambda \geq 0, \\
\tau \left( \phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - y_t \boldsymbol{\mu} \cdot \mathbf{x}_t - \xi \right) &= 0, \quad \lambda \xi = 0.
\end{aligned}$$

The rest of the proof is similar to that of SCW-I.  $\square$

### Proof of Proposition 3

PROOF. First, when  $\ell^\phi(\mathcal{N}(\boldsymbol{\mu}_{t,r_t}, \boldsymbol{\mu}_{t,s_t}, \Sigma_t); (\mathbf{x}_t, y_t)) = 0$ , it is easy to see the solution is valid. When  $\ell^\phi(\mathcal{N}(\boldsymbol{\mu}_{t,r_t}, \boldsymbol{\mu}_{t,s_t}, \Sigma_t); (\mathbf{x}_t, y_t)) > 0$ , it is easy to see the optimization problem is equivalent to

$$\begin{aligned}
&(\boldsymbol{\mu}_{t+1,r_t}, \boldsymbol{\mu}_{t+1,s_t}, \Sigma_{t+1}) \\
&= \arg \min_{\boldsymbol{\mu}_r, \boldsymbol{\mu}_s, \Sigma} D_{KL}(\mathcal{N}(\boldsymbol{\mu}_r, \Sigma) \parallel \mathcal{N}(\boldsymbol{\mu}_{t,r_t}, \Sigma_t)) + D_{KL}(\mathcal{N}(\boldsymbol{\mu}_s, \Sigma) \parallel \mathcal{N}(\boldsymbol{\mu}_{t,s_t}, \Sigma_t)) + C\xi, \\
&\text{s.t. } \ell^\phi(\mathcal{N}(\boldsymbol{\mu}_r, \boldsymbol{\mu}_s, \Sigma_t); (\mathbf{x}_t, y_t)) \leq \xi \quad \text{and} \quad \xi \geq 0.
\end{aligned}$$

Since  $\Sigma$  is PSD, it can be written as  $\Sigma = \Upsilon^2$  to make the optimization with a convex constraint in  $\boldsymbol{\mu}$  and  $\Upsilon$  simultaneously. But for convenience, we will still use  $\Sigma$  instead

of  $\Upsilon^2$  in the following analysis. The Lagrangian of the previous optimization is

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\mu}_r, \boldsymbol{\mu}_s, \Sigma, \xi, \tau, \lambda) &= D_{KL}(\mathcal{N}(\boldsymbol{\mu}_r, \Sigma) \| \mathcal{N}(\boldsymbol{\mu}_{t,r_t}, \Sigma_t)) + D_{KL}(\mathcal{N}(\boldsymbol{\mu}_s, \Sigma) \| \mathcal{N}(\boldsymbol{\mu}_{t,s_t}, \Sigma_t)) + C\xi \\
&\quad + \tau \left( \phi \sqrt{2\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) - \xi \right) - \lambda \xi \\
&= D_{KL}(\mathcal{N}(\boldsymbol{\mu}_r, \Sigma) \| \mathcal{N}(\boldsymbol{\mu}_{t,r_t}, \Sigma_t)) + D_{KL}(\mathcal{N}(\boldsymbol{\mu}_s, \Sigma) \| \mathcal{N}(\boldsymbol{\mu}_{t,s_t}, \Sigma_t)) + \xi(C - \tau - \lambda) \\
&\quad + \tau \left( \phi \sqrt{2\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) \right) \\
&= \frac{1}{2} \log \left( \frac{\det \Sigma_t}{\det \Sigma} \right) + \frac{1}{2} \text{Tr}(\Sigma_t^{-1} \Sigma) + \frac{1}{2} (\boldsymbol{\mu}_{t,r_t} - \boldsymbol{\mu}_r)^\top \Sigma_t^{-1} (\boldsymbol{\mu}_{t,r_t} - \boldsymbol{\mu}_r) \\
&\quad + \frac{1}{2} (\boldsymbol{\mu}_{t,s_t} - \boldsymbol{\mu}_s)^\top \Sigma_t^{-1} (\boldsymbol{\mu}_{t,s_t} - \boldsymbol{\mu}_s) - d + \xi(C - \tau - \lambda) \\
&\quad + \tau (\phi \sqrt{2\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t)),
\end{aligned}$$

where  $\tau \geq 0$  and  $\lambda \geq 0$  are Lagrange multipliers. We now find the minimum of the Lagrangian with respect to the primal variables  $\boldsymbol{\mu}_r$ ,  $\boldsymbol{\mu}_s$ ,  $\Sigma$ , and  $\xi$ :

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_r} &= \Sigma_t^{-1} (\boldsymbol{\mu}_r - \boldsymbol{\mu}_{t,r_t}) + \tau (-\mathbf{x}_t) = 0 \Rightarrow \boldsymbol{\mu}_r = \boldsymbol{\mu}_{t,r_t} + \tau \Sigma_t \mathbf{x}_t \\
\frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_s} &= \Sigma_t^{-1} (\boldsymbol{\mu}_s - \boldsymbol{\mu}_{t,s_t}) + \tau (\mathbf{x}_t) = 0 \Rightarrow \boldsymbol{\mu}_s = \boldsymbol{\mu}_{t,s_t} - \tau \Sigma_t \mathbf{x}_t \\
\frac{\partial \mathcal{L}}{\partial \Sigma_{t+1}} &= 0 \Rightarrow \Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \frac{\sqrt{2}}{2} \tau \phi \frac{\mathbf{x}_t \mathbf{x}_t^\top}{\sqrt{\mathbf{x}_t^\top \Sigma_{t+1} \mathbf{x}_t}},
\end{aligned}$$

and  $C - \tau - \lambda = 0$ , so  $\tau = C - \lambda \leq C$ , and thus,  $\tau \in [0, C]$ . The KKT conditions for the optimization are

$$\begin{aligned}
\phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) - \xi &\leq 0, \quad -\xi \leq 0, \quad \tau, \lambda \geq 0, \\
\tau \left( \phi \sqrt{\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) - \xi \right) &= 0, \quad \lambda \xi = 0.
\end{aligned}$$

**Case 1.**  $\tau \neq 0$

As  $\tau (\phi \sqrt{2\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) - \xi) = 0$  implies  $(\phi \sqrt{2\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) - \xi) = 0$ , the KKT conditions are simplified:

$$-\xi \leq 0, \quad \tau > 0, \quad \lambda \geq 0, \quad \phi \sqrt{2\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) - \xi = 0, \quad \lambda \xi = 0.$$

**Subcase 1.1.**  $\lambda \neq 0$

When  $\lambda \neq 0$ ,  $\lambda \xi = 0$ , which implies  $\xi = 0$ . The KKT conditions are simplified as

$$\tau > 0, \quad \lambda > 0, \quad \xi = 0, \quad \phi \sqrt{2\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) = 0.$$

Finally, we have the following:

$$\Sigma_{t+1} = \left( \Sigma_t^{-1} + \frac{\sqrt{2}}{2} \tau \phi \frac{\mathbf{x}_t \mathbf{x}_t^\top}{\sqrt{\mathbf{x}_t^\top \Sigma_{t+1} \mathbf{x}_t}} \right)^{-1} = \Sigma_t - \Sigma_t \mathbf{x}_t \left( \frac{\tau \phi}{\sqrt{2\mathbf{x}_t^\top \Sigma_{t+1} \mathbf{x}_t} + \tau \phi \mathbf{x}_t^\top \Sigma_t \mathbf{x}_t} \right) \mathbf{x}_t^\top \Sigma_t.$$



Let  $u_t = \mathbf{x}_t^\top \Sigma_{t+1} \mathbf{x}_t$ ,  $v_t = \mathbf{x}_t^\top \Sigma_t \mathbf{x}_t$ ,  $m_t = \boldsymbol{\mu}_{t,r_t} \cdot \mathbf{x}_t - \boldsymbol{\mu}_{t,s_t} \cdot \mathbf{x}_t$ ; multiplying by  $\mathbf{x}_t^\top$  (left) and  $\mathbf{x}_t$  (right), we get  $u_t = v_t - v_t \left( \frac{\tau \phi}{\sqrt{2u_t + \tau \phi v_t}} \right) v_t$ , which can be used to solve  $u_t$ :

$$\sqrt{u_t} = \frac{-\tau \phi v_t + \sqrt{\tau^2 \phi^2 v_t^2 + 8v_t}}{2\sqrt{2}}.$$

And  $\phi \sqrt{2\mathbf{x}_t^\top \Sigma_t \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) = 0$  implies  $\phi \sqrt{2u_t} - m_t - 2\tau v_t = 0$ . Thus,  $\phi \frac{-\tau \phi v_t + \sqrt{\tau^2 \phi^2 v_t^2 + 8v_t}}{2} - m_t - 2\tau v_t = 0$ , which can be rearranged as  $v_t^2(4 + 2\phi^2)\tau^2 + m_t v_t(4 + 2\phi^2)\tau + (m_t^2 - \phi^2 v_t)$ . The larger root is then

$$\tau = \frac{-m_t(1 + \frac{\phi^2}{2}) + \sqrt{m_t^2(1 + \frac{\phi^2}{2})^2 - m_t^2(1 + \frac{\phi^2}{2}) + 2v_t\phi^2(1 + \frac{\phi^2}{2})}}{2v_t(1 + \frac{\phi^2}{2})}.$$

If  $\tau \in (0, C)$ , then  $\lambda = C - \tau \in (0, C)$ .

**Subcase 1.2.**  $\lambda = 0$

$C - \tau - \lambda = 0$  implies  $\tau = C$ . The KKT conditions can be simplified as

$$-\xi \leq 0, \quad \tau = C, \quad \lambda = 0, \quad \phi \sqrt{2\mathbf{x}_t^\top \Sigma_t \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) - \xi = 0.$$

We thus have

$$\phi \sqrt{2\mathbf{x}_t^\top \Sigma_t \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) = \left[ \phi \frac{-\tau \phi v_t + \sqrt{\tau^2 \phi^2 v_t^2 + 8v_t}}{2} - m_t - 2\tau v_t \right] \Big|_{\tau=C} = \xi \geq 0.$$

It is easy to verify that

$$f'(\tau) = \frac{-\phi^2 v_t}{2} + \frac{\phi^3 v_t^2 \tau}{2\sqrt{\tau^2 \phi^2 v_t^2 + 8v_t}} - 2v_t = 0$$

has no solution on  $[0, +\infty)$  and  $f'(0) = \frac{-\phi^2 v_t}{2} - 2v_t < 0$ . As a result,  $f'(\tau) < 0$ ,  $\tau \in [0, +\infty)$ , which implies  $f(\tau)$  is decreasing on  $[0, +\infty)$ :

$$f(C) \geq 0 = f(\theta),$$

where  $\theta = \frac{-m_t(1 + \frac{\phi^2}{2}) + \sqrt{m_t^2(1 + \frac{\phi^2}{2})^2 - m_t^2(1 + \frac{\phi^2}{2}) + 2v_t\phi^2(1 + \frac{\phi^2}{2})}}{2v_t(1 + \frac{\phi^2}{2})}$ , which thus implies  $C \leq \theta$ .

**Case 2.**  $\tau = 0$

When  $\tau = 0$ , since  $C - \tau - \lambda = 0$ ,  $\lambda = C$ , and the KKT conditions are simplified as

$$\phi \sqrt{2\mathbf{x}_t^\top \Sigma_t \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) \leq 0, \quad \tau = 0, \quad \lambda = C, \quad \xi = 0.$$

Thus,  $\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t$  and  $\Sigma_{t+1} = \Sigma_t$ ; as a result,  $\phi \sqrt{2\mathbf{x}_t^\top \Sigma_t \mathbf{x}_t} - (\boldsymbol{\mu}_{t,r_t} \cdot \mathbf{x}_t - \boldsymbol{\mu}_{t,s_t} \cdot \mathbf{x}_t) \leq 0$ , which contradicts with  $\ell^\phi(\mathcal{N}(\boldsymbol{\mu}_{t,r_t}, \boldsymbol{\mu}_{t,s_t}, \Sigma_t); (\mathbf{x}_t, y_t)) > 0$ .  $\square$

#### Proof of Proposition 4

PROOF. For SCW-II, the Lagrangian of the optimization is

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\mu}_r, \boldsymbol{\mu}_s, \Sigma, \xi, \tau, \lambda) &= D_{KL}(\mathcal{N}(\boldsymbol{\mu}_r, \Sigma) \parallel \mathcal{N}(\boldsymbol{\mu}_{t,r_t}, \Sigma_t)) + D_{KL}(\mathcal{N}(\boldsymbol{\mu}_s, \Sigma) \parallel \mathcal{N}(\boldsymbol{\mu}_{t,s_t}, \Sigma_t)) + C\xi \\
&\quad + \tau \left( \phi \sqrt{2\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) - \xi \right) - \lambda \xi \\
&= D_{KL}(\mathcal{N}(\boldsymbol{\mu}_r, \Sigma) \parallel \mathcal{N}(\boldsymbol{\mu}_{t,r_t}, \Sigma_t)) + D_{KL}(\mathcal{N}(\boldsymbol{\mu}_s, \Sigma) \parallel \mathcal{N}(\boldsymbol{\mu}_{t,s_t}, \Sigma_t)) + \xi(C - \tau - \lambda) \\
&\quad + \tau \left( \phi \sqrt{2\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) \right) \\
&= \frac{1}{2} \log \left( \frac{\det \Sigma_t}{\det \Sigma} \right) + \frac{1}{2} \text{Tr}(\Sigma_t^{-1} \Sigma) + \frac{1}{2} (\boldsymbol{\mu}_{t,r_t} - \boldsymbol{\mu}_r)^\top \Sigma_t^{-1} (\boldsymbol{\mu}_{t,r_t} - \boldsymbol{\mu}_r) \\
&\quad + \frac{1}{2} (\boldsymbol{\mu}_{t,s_t} - \boldsymbol{\mu}_s)^\top \Sigma_t^{-1} (\boldsymbol{\mu}_{t,s_t} - \boldsymbol{\mu}_s) - d + \xi(C\xi - \tau - \lambda) \\
&\quad + \tau \left( \phi \sqrt{2\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) \right),
\end{aligned}$$

where  $\tau \geq 0$  and  $\lambda \geq 0$  are Lagrange multipliers. We now find the minimum of the Lagrangian with respect to the primal variables  $\boldsymbol{\mu}$ ,  $\Sigma$ , and  $\xi$ :

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_r} &= \Sigma_t^{-1} (\boldsymbol{\mu}_r - \boldsymbol{\mu}_{t,r_t}) + \tau (-\mathbf{x}_t) = 0 \Rightarrow \boldsymbol{\mu}_r = \boldsymbol{\mu}_{t,r_t} + \tau \Sigma_t \mathbf{x}_t, \\
\frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}_s} &= \Sigma_t^{-1} (\boldsymbol{\mu}_s - \boldsymbol{\mu}_{t,s_t}) + \tau (\mathbf{x}_t) = 0 \Rightarrow \boldsymbol{\mu}_s = \boldsymbol{\mu}_{t,s_t} - \tau \Sigma_t \mathbf{x}_t, \\
\frac{\partial \mathcal{L}}{\partial \Sigma_{t+1}} &= 0 \Rightarrow \Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \frac{\sqrt{2}}{2} \tau \phi \frac{\mathbf{x}_t \mathbf{x}_t^\top}{\sqrt{\mathbf{x}_t^\top \Sigma_{t+1} \mathbf{x}_t}},
\end{aligned}$$

and  $2C\xi - \tau - \lambda = 0$ , so  $\xi = \frac{\tau + \lambda}{2C}$ . The KKT conditions for the optimization are

$$\begin{aligned}
\phi \sqrt{2\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) - \xi &\leq 0, \quad \xi \geq 0, \quad \tau \geq 0, \quad \lambda \geq 0 \\
\tau \left( \phi \sqrt{2\mathbf{x}_t^\top \Sigma \mathbf{x}_t} - (\boldsymbol{\mu}_r \cdot \mathbf{x}_t - \boldsymbol{\mu}_s \cdot \mathbf{x}_t) - \xi \right) &= 0, \quad \lambda \xi = 0.
\end{aligned}$$

The rest of the proof is similar to that of MSCW1.  $\square$

#### ACKNOWLEDGMENTS

This work was supported in part by the Singapore Ministry of Education (MOE) Academic Research Fund (AcRF) Tier 1 Research Grant (C220/MSS14C003) and Microsoft Research Grant.

#### REFERENCES

- Nicolò Cesa-Bianchi, Alex Conconi, and Claudio Gentile. 2004. On the generalization ability of on-line learning algorithms. *IEEE Trans. Inf. Theory* 50, 9 (2004), 2050–2057.
- Nicolò Cesa-Bianchi, Alex Conconi, and Claudio Gentile. 2005. A second-order Perceptron algorithm. *SIAM J. Comput.* 34, 3 (2005), 640–668.
- Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *The Conference on Empirical Methods on Natural Language Processing (EMNLP)*.
- Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *J. Mach. Learn. Res.* 7 (2006), 551–585.

- Koby Crammer and Daniel D. Lee. 2010. Learning via Gaussian herding. In *Annual Conference on Neural Information Processing Systems (NIPS)*. 345–352.
- Koby Crammer, Mark Dredze, and Alex Kulesza. 2009. Multi-class confidence weighted algorithms. In *EMNLP*. 496–504.
- Koby Crammer, Mark Dredze, and Fernando Pereira. 2008. Exact convex confidence-weighted learning. In *Annual Conference on Neural Information Processing Systems (NIPS)*. 345–352.
- Koby Crammer, Alex Kulesza, and Mark Dredze. 2009. Adaptive regularization of weight vectors. In *Annual Conference on Neural Information Processing Systems (NIPS)*. 345–352.
- Koby Crammer, Alex Kulesza, and Mark Dredze. 2013. Adaptive regularization of weight vectors. *Mach. Learn.* 91 (2013), 155–187.
- Koby Crammer and Yoram Singer. 2001. On the algorithmic implementation of multiclass kernel-based vector machines. *J. Mach. Learn. Res.* 2 (2001), 265–292.
- Koby Crammer and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *J. Mach. Learn. Res.* 3 (2003), 951–991.
- Koby Crammer and Yoram Singer. 2005. Loss bounds for online category ranking. In *The Annual Conference on Learning Theory (COLT)*. 48–62.
- Mark Dredze, Koby Crammer, and Fernando Pereira. 2008. Confidence-weighted linear classification. In *The International Conference on Machine Learning (ICML)*. 264–271.
- Mark Dredze, Alex Kulesza, and Koby Crammer. 2010. Multi-domain learning by confidence-weighted parameter combination. *Mach. Learn.* 79, 1–2 (2010), 123–149.
- John C. Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.* 12 (2011), 2121–2159.
- Michael Fink, Shai Shalev-Shwartz, Yoram Singer, and Shimon Ullman. 2006. Online multiclass learning by interclass hypothesis sharing. In *The International Conference on Machine Learning (ICML)*. 313–320.
- Yoav Freund and Robert E. Schapire. 1999. Large margin classification using the Perceptron algorithm. *Mach. Learn.* 37, 3 (1999), 277–296.
- Claudio Gentile. 2001. A new approximate maximal margin classification algorithm. *J. Mach. Learn. Res.* 2 (2001), 213–242.
- Steven C. H. Hoi, Rong Jin, Peilin Zhao, and Tianbao Yang. 2013. Online multiple kernel classification. *Mach. Learn.* 90, 2 (2013), 289–316.
- Steven C. H. Hoi, Jialei Wang, and Peilin Zhao. 2014. Libol: A library for online learning algorithms. *J. Mach. Learn. Res.* 15, 1 (2014), 495–499.
- Jyrki Kivinen, Alex J. Smola, and Robert C. Williamson. 2001. Online learning with kernels. In *Advances in Neural Information Processing Systems (NIPS'01)*. 785–792.
- Bin Li and Steven C. H. Hoi. 2014. Online portfolio selection: A survey. *ACM Comput. Surv. (CSUR)* 46, 3 (2014), 35.
- Bin Li, Peilin Zhao, Steven C. H. Hoi, and Vivekanand Gopalkrishnan. 2012. PAMR: Passive aggressive mean reversion strategy for portfolio selection. *Mach. Learn.* 87, 2 (2012), 221–258.
- Yi Li and Philip M. Long. 1999. The relaxed online maximum margin algorithm. In *Advances in Neural Information Processing Systems (NIPS'99)*. 498–504.
- Yi Li and Philip M. Long. 2002. The relaxed online maximum margin algorithm. *Mach. Learn.* 46, 1–3 (2002), 361–387.
- Jing Lu, Steven Hoi, and Jialei Wang. 2013. Second order online collaborative filtering. In *Asian Conference on Machine Learning (ACML'13)*. 325–340.
- Francesco Orabona and Koby Crammer. 2010. New adaptive algorithms for online classification. In *Annual Conference on Neural Information Processing Systems (NIPS)*. 1840–1848.
- Frank Rosenblatt. 1958a. The Perceptron: A probabilistic model for information storage and organization in the brain. *Psych. Rev.* 7 (1958), 551–585.
- F. Rosenblatt. 1958b. The Perceptron: A probabilistic model for information storage and organization in the brain. *Psych. Rev.* 65 (1958), 386–407.
- Shai Shalev-Shwartz and Yoram Singer. 2007. A unified algorithmic approach for efficient online label ranking. *J. Mach. Learn. Res. - Proc. Track* 2 (2007), 452–459.
- Jialei Wang, Steven C. H. Hoi, Peilin Zhao, and Zhiyong Liu. 2013. Online multi-task collaborative filtering for on-the-fly recommender systems. In *7th ACM Conference on Recommender Systems (RecSys'13)*. 237–244. DOI: <http://dx.doi.org/10.1145/2507157.2507176>
- Jialei Wang, Ji Wan, Yongdong Zhang, and Steven C. H. Hoi. 2015. SOLAR: Scalable online learning algorithms for ranking. In *The Conference of the Association for Computational Linguistics (ACL)*. 1692–1701.

- Jialei Wang, Peilin Zhao, and Steven C. H. Hoi. 2014a. Cost-sensitive online classification. *IEEE Trans. Knowl. Data Eng.* 26, 10 (2014), 2425–2438.
- Jialei Wang, Peilin Zhao, Steven C. H. Hoi, and Rong Jin. 2014b. Online feature selection and its applications. *Knowledge and Data Engineering, IEEE Transactions on* 26, 3 (2014), 698–710.
- Jialei Wang, Peilin Zhao, and Steven C. H. Hoi. 2012. Exact soft confidence-weighted learning. In *Proceedings of the 29th International Conference on Machine Learning (ICML'12)*.
- Pengcheng Wu, Steven C. H. Hoi, Hao Xia, Peilin Zhao, Dayong Wang, and Chunyan Miao. 2013. Online multimodal deep similarity learning with application to image retrieval. In *ACM Multimedia Conference (MM'13)*. 153–162. DOI: <http://dx.doi.org/10.1145/2502081.2502112>
- Hao Xia, Steven C. H. Hoi, Rong Jin, and Peilin Zhao. 2014. Online multiple kernel similarity learning for visual search. *IEEE Pattern Anal. Mach. Intell.* 36, 3 (2014), 536–549.
- Liu Yang, Rong Jin, and Jieping Ye. 2009. Online learning by ellipsoid method. In *The International Conference on Machine Learning (ICML)*. 145.
- Peilin Zhao, Steven C. H. Hoi, Jialei Wang, and Bin Li. 2014. Online transfer learning. *Artific. Intell.* 216 (2014), 76–102.
- Peilin Zhao and Steven C. H. Hoi. 2013. Cost-sensitive online active learning with application to malicious URL detection. In *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'13)*. 919–927. DOI: <http://dx.doi.org/10.1145/2487575.2487647>
- Peilin Zhao, Steven C. H. Hoi, and Rong Jin. 2011a. Double updating online learning. *J. Mach. Learn. Res.* 12 (2011), 1587–1615.
- Peilin Zhao, Steven C. H. Hoi, Rong Jin, and Tianbao Yang. 2011b. Online AUC maximization. In *The International Conference on Machine Learning (ICML)*. 233–240.
- Martin Zinkevich. 2003. Online convex programming and generalized infinitesimal gradient ascent. In *The International Conference on Machine Learning (ICML)*. 928–936.