

# Confidence-Weighted Sparse Online Learning

Yue Wu

December 10, 2013

## 1 Introduction

The main idea of Confidence-Weighted algorithms is to assume a Gaussian distribution of the linear classifier  $\omega \sim N(\mu, \Sigma)$ . Each update tries to stay close to the previous distribution and ensure the probability of the current precision for  $x_i$  is larger than  $\eta$ .

$$Pr[y_i(\mu \cdot x_i) \geq 0] \geq \eta \quad \text{or} \quad y_i(\mu \cdot x_i) \geq \phi \sqrt{x_i^T \Sigma x_i} \quad (1)$$

## 2 Problem Formulation

In Sparse online learning, we set the weight vector equal to the average  $\omega = \mu$ . Each iteration step is a minimization problem as follows:

$$\mu_t = \arg \min_{\mu} f(\mu) + \tilde{\lambda} r(\mu) \quad (2)$$

where  $f(\mu)$  is often the loss function. In this paper, we follow the setting of AROW and use the squared hinge loss:

$$f(\mu) = \max(0, 1 - y_t(\mu \cdot x_t))^2 \quad (3)$$

To the regularization term, previous learning algorithms treat all the coordinates the same by adding an regularization term  $\lambda|\omega|$ . We extend the confidence to apply different regularization intensities to different coordinates. A high variance value corresponds a low confidence. Accordingly, we apply a strong shrinkage to less confident coordinates and weak shrinkage to confident ones.

### 2.1 Smooth Regularization

Smooth regularization is the  $L1$  norm. We call it smooth sparse regularization, as it shrink the weight values by some amount once a iteration.

$$r(\mu) = |\Sigma \mu| \quad (4)$$

### 2.2 Aggressive Regularization

Aggressive regularization is a strict condition on the number of non-zero coordinates, as Equ 5 shows. This setting is useful in problems like feature selection.

$$r(\mu) = \begin{cases} 0, & |\Sigma \mu|_0 \leq B \\ \infty, & |\Sigma \mu|_0 > B \end{cases} \quad (5)$$

## 3 Solution

Common approaches such as subgradient methods to Equ. 1 will rarely lead to non-differential points of  $f(\omega)$  or  $r(\omega)$ . While these non-differential points are the true minima in cases like  $L1$  regularization. Instead, we adopt a *forward-backward splitting* approach to alleviate the problems of non-differentiability.

In the first step, we adopt AROW on  $f(\mu)$  to obtain one step iteration. The iteration is as Equ 6 shows.

$$\begin{aligned} \mu_{t-\frac{1}{2}} &= \mu_{t-1} + \alpha_t \Sigma_{t-1} y_t x_t & \Sigma_t &= \Sigma_{t-1} - \beta_t \Sigma_{t-1} x_t x_t^T \Sigma_{t-1} \\ \beta_t &= \frac{1}{x_t^T \Sigma_{t-1} x_t + r} & \alpha_t &= \max(0, 1 - y_t x_t^T \mu_{t-1}) \beta_t \end{aligned} \quad (6)$$

We re-write the above iteration to be second order sub-gradient update as Equ. 7.

$$\begin{aligned}\boldsymbol{\mu}_{t-\frac{1}{2}} &= \boldsymbol{\mu}_{t-1} - \frac{\beta_t}{2} \Sigma_{t-1} g_t^f \\ g_t^f &= \partial f(\boldsymbol{\mu})\end{aligned}\tag{7}$$

In the above update equation,  $\frac{\beta_t}{2}$  is the common learning rate.  $\Sigma_{t-1}$  is the matrix to apply different learning rates to different coordinates.

The second step is a projection step with the smooth regularization penalty.

$$\begin{aligned}\tilde{\lambda} &= \frac{\beta_t}{2} \lambda \\ \boldsymbol{\mu}_t &= \arg \min_{\boldsymbol{\mu}} \frac{1}{2} \|\boldsymbol{\mu} - \boldsymbol{\mu}_{t-\frac{1}{2}}\|^2 + \frac{\beta_t}{2} \lambda |\Sigma_{t-1} \boldsymbol{\mu}|\end{aligned}\tag{8}$$

The final updating rule goes to:

$$\mu_{t,j} = \text{sign}(\mu_{t-1,j} - \frac{\beta_t}{2} \Sigma_{t-1,jj} g_{t-1,j}^f) [|\mu_{t-1,j} - \frac{\beta_t}{2} \Sigma_{t-1,jj} g_{t-1,j}^f| - \frac{\beta_t}{2} \lambda \Sigma_{jj}] \tag{9}$$

## 4 Experimental results

### 4.1 Experiment on real datasets

Table 1: Real Datasets

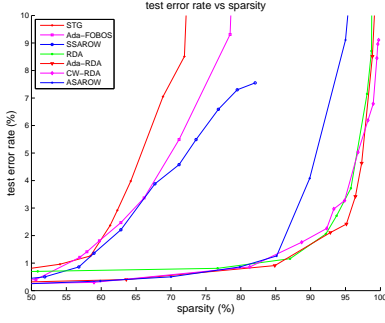
DataSet	Feat Dim	Train Data No.	Test Data No.	Feat No.	sparsity
MNIST67	780	12183	1987	1,751,945	0
news	732209	9960	9994	5,513,533	0
rcv1	47,152	781,265	23,149	59,155,144	0
url_combined	3,231,961	2,000,000	396,130	231,259,917	0
webspam	16,609,143	300,000	50,000	1, 118,443,083	0

### 4.2 test error rate vs sparsity

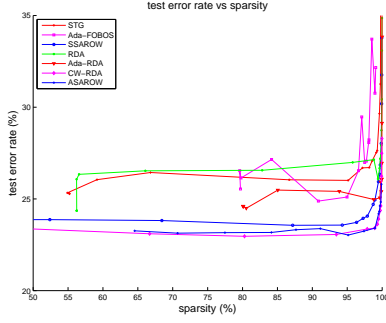
#### 4.2.1 TG-based algorithms

#### 4.2.2 DA-based algorithms

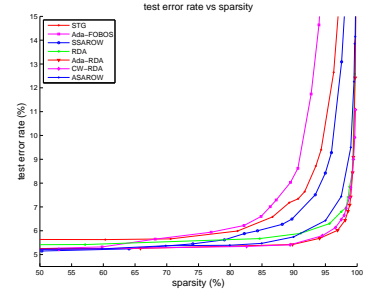
## 5 Reference



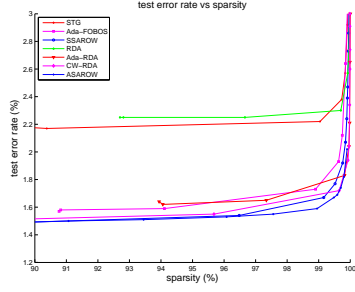
(a) MNIST



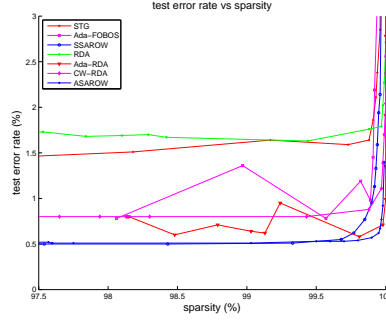
(b) news



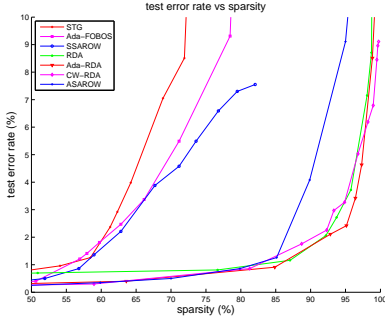
(c) rcv1



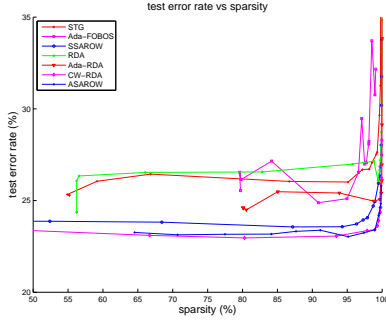
(d) url



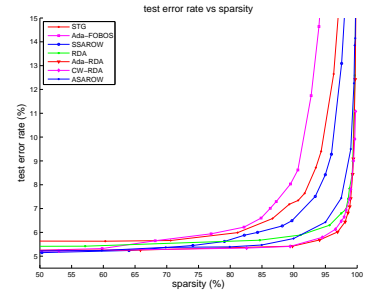
(e) webspam



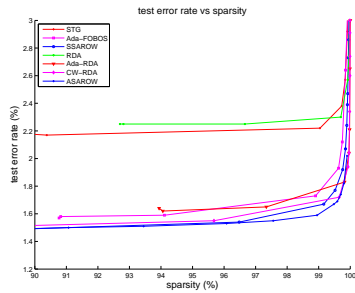
(f) MNIST



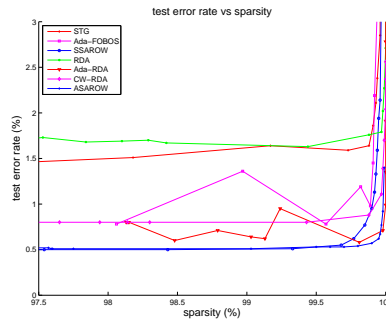
(g) news



(h) rcv1



(i) url



(j) webspam