UNIVERSITY AT ALBANY
State University of New York

Fast Optimization Algorithms for AUC Maximization

Doctoral Dissertation Defense

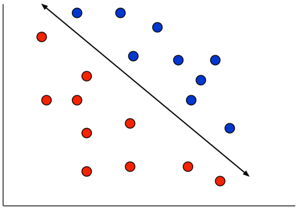
Michael Natole, Jr.

Committee: Prof. Yiming Ying
Prof. Yunlong Feng
Prof. Boris Goldfarb
Prof. Karen Reinhold

March 31, 2020

Overview

Background: Classification



- Given data $\{z_i = (\mathrm{x}_i, y_i) \in \mathcal{Z} : i = 1...T\}$, where $\mathrm{x}_i \in \mathcal{X} \subseteq \mathbb{R}^d$, $y_i \in \mathcal{Y} = \{\pm 1\}$, and $\mathcal{X} \times \mathcal{Y} = \mathcal{Z}$, we wish to learn the following function

$$f(\mathrm{x}_i) = \mathrm{sign}(\mathrm{w}^T \mathrm{x}_i), \tag{1}$$

  where $\mathrm{w} \in \mathbb{R}^d$ is the parameter to be learned.
- We want w that gives the best performance, or accuracy.

Confusion Matrix

- The decision made by a binary classifier can be made into a structure called a **confusion matrix** (C) having four categories: true positive (TP), false positive (FP), false negative (FN), and true negative (TN).

	Actual Positive	Actual Negative
Predicted Positive	TP	FP
Predicted Negative	FN	TN

- Let P denote the number of samples that of the positive class and N denote the number of samples that of the negative class.
- The most obvious performance metric to consider is **accuracy**:

$$Accuracy = \frac{TP + TN}{P + N}. \tag{2}$$

Background: Setup

- In machine learning, we want to optimize the empirical risk:

$$\min_{\mathrm{w}} R_{emp}(\mathrm{w}). \tag{3}$$

- So in the case of accuracy, we have

$$R_{emp}(\mathrm{w}) = \frac{1}{T} \sum_{i=1}^{T} \mathbb{I}[y_i \mathrm{w}^T \mathrm{x}_i < 0]. \tag{4}$$

- Other loss functions include square loss and logistic loss.
  - Hinge Loss: $\phi(s) = \max\{0, 1 - s\}$
  - Logistic Loss: $\phi(s) = \frac{1}{\ln 2} \ln(1 + e^{-s})$
  - Square Loss: $\phi(s) = (1 - s)^2$
- Empirical Risk Minimization (ERM)

$$\mathrm{w}^* = \arg\min_{\mathrm{w}} \frac{1}{T} \sum_{i=1}^{T} \phi(y_i \mathrm{w}^T \mathrm{x}_i) \tag{5}$$

Stochastic Gradient Descent

Stochastic Gradient Descent

Initialize w_1, and for any $t \geq 1$, draw sample $z_t = (\mathrm{x}_t, y_t)$ at random, and then

$$\mathrm{w}_{t+1} = \mathrm{w}_t - \eta_t \nabla_{\mathrm{w}} \phi(y_t \mathrm{w}^T \mathrm{x}_t). \tag{6}$$

- The idea of SGD dates back to [Robbins and Monro, 1951].
- The literature on SGD is extensive [Bottou and Cun, 2004, Srebro and Tewari, 2010, Moulines and Bach, 2011, Ying and Pontil, 2008]
- Most of the literature focuses on the misclassification error or accuracy, but is it always a good performance measure?

Imbalanced Data

- In many application domains (cancer diagnosis, wildfire prediction, fraud detection, etc.), the ratio of class observations are disproportionate resulting in the data being imbalanced.
- Consider the case for a sample of 1000 instances with 990 "true" negative instances and 10 "true" positive instances. Suppose we obtain the following results:

	True +1	True -1
Predicted +1	1	11
Predicted -1	9	979

- The misclassification error (or classification accuracy) could be misleading for real world applications.
- This classifier has 98% accuracy, but told us very little.

Precision and Recall

- **Precision** and **Recall** using the confusion matrix are defined as follows:

$$\mathrm{Prec}(C) = \frac{TP}{TP + FP} \qquad \mathrm{Rec}(C) = \frac{TP}{P}$$

- Precision is defined as the fraction of relevant instances among the retrieved instances.
- Recall is the fraction of relevant instances that have been retrieved over the total amount of relevant instances.
- These two values together can be used to measure the performance of a classifier.
- This method is popular to use for web search engines since user typically only scan the first few results that are presented.

Example

- Consider the previous confusion matrix:

	True +1	True -1
Predicted +1	1	11
Predicted -1	9	979

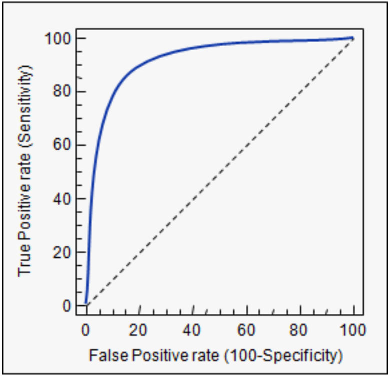
- Calculating precision and recall gives:

$$\mathrm{Prec}(C) = \frac{TP}{TP + FP} = \frac{1}{1 + 11} = 0.04762$$

$$\mathrm{Rec}(C) = \frac{TP}{P} = \frac{1}{10} = 0.1$$

- Even though the accuracy is high, the classifier has poor performance. Consider the following examples:
  - Determining if a transaction is fraudulent
  - Diagnosing if a patient has cancer

Receiver Operating Characteristic (ROC) Curve



- Given a confusion matrix, a ROC curve is a plot of the False Positive Rate (FPR) on the x-axis and the True Positive Rate (TPR) on the y-axis.

$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{FP + TN}$$

[Hanley and McNeil, 1982, Bradley, 1997, Fawcett, 2006]

Probabilistic Definition of AUC

Definition

For a linear scoring function $f(\mathrm{x}) = \mathrm{w}^T\mathrm{x}$, AUC is

$$\begin{aligned} AUC(\mathrm{w}) &= \Pr(\mathrm{w}^T\mathrm{x} \geq \mathrm{w}^T\mathrm{x}' | y = 1, y' = -1) \\ &= 1 - \mathbb{E}[\mathbb{I}_{[\mathrm{w}^T(\mathrm{x}-\mathrm{x}')<0]} | y = 1, y' = -1], \end{aligned}$$

where (x, y), $(\mathrm{x}', y') \in \mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ are independent.

AUC Maximization

- AUC maximization can be easily modified to:

$$\min_{\mathrm{w}} \mathbb{E}[\mathbb{I}_{[\mathrm{w}^T(\mathrm{x}-\mathrm{x}')<0]}|y=1, y'=-1] + \Omega(\mathrm{w}), \tag{7}$$

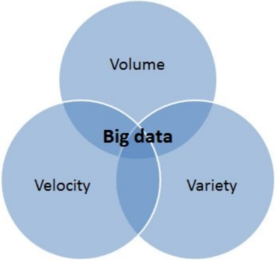
  where $\Omega(\cdot)$ is a penalty function.
- Replacing the indicator function by the least square loss, AUC optimization can be formulated as:

$$\min_{\mathrm{w}} \mathbb{E}[(1-\mathrm{w}^T(\mathrm{x}-\mathrm{x}'))^2|y=1, y'=-1] + \Omega(\mathrm{w}). \tag{8}$$

- When ρ is a uniform distribution over the finite data $\{z_i = (\mathrm{x}_i, y_i) \in \mathcal{Z} : i = 1 \dots T\}$, AUC maximization reduces to

$$\min_{\mathrm{w}} \frac{1}{n_+ n_-} \sum_{i=1}^{n_+} \sum_{j=1}^{n_-} (1-\mathrm{w}^T(\mathrm{x}_i - \mathrm{x}_j))^2 \mathbb{I}_{y_i=1 \wedge y_j=-1} + \Omega(\mathrm{w}).$$

Research Challenges



- How can we design learning algorithms that optimize the AUC score instead?
- Key Challenges:
  - What happens if the dataset is very large?
  - How do we handle streaming data?
- AUC is expressed as a sum of pairwise losses between instances from different classes.
  - Computing AUC is quadratic in the number of received training samples.
  - In a real world scenario, data arrives sequentially, not in pairs of different classes.

Existing Methods

$$L(\mathrm{w}) = \frac{\lambda}{2}\|\mathrm{w}\|^2 + \sum_{i=1}^{n_+}\sum_{j=1}^{n_-} \max\{0, 1 - \mathrm{w}^T(\mathrm{x}_i^+ - \mathrm{x}_j^-)\} \tag{9}$$

- The authors rewrote the loss function as a sum of losses for individual instances, i.e. $L(\mathrm{w}) = \sum_{t=1}^{T} L_t(\mathrm{w})$ where:

$$L_t(\mathrm{w}) = \mathbb{I}_{y_t=1} h_+^t(\mathrm{w}) + \mathbb{I}_{y_t=-1} h_-^t(\mathrm{w}) \tag{10}$$

- In the above, $h_\pm^t(\mathrm{w})$ are defined as:

$$h_+^t(\mathrm{w}) = \sum_{t'=1}^{t-1} \mathbb{I}_{y_{t'}=-1} \ell(\mathrm{w}, \mathrm{x}_t - \mathrm{x}_{t'}),\ h_-^t(\mathrm{w}) = \sum_{t'=1}^{t-1} \mathbb{I}_{y_{t'}=+1} \ell(\mathrm{w}, \mathrm{x}_{t'} - \mathrm{x}_t) \tag{11}$$

[Zhao et al., 2011]

Existing Methods

- The authors then apply gradient descent as in [Zinkevich, 2003]. This however, requires all previously stored samples to be used to compute the gradient.
- To overcome this, the authors used reservoir sampling by [Vitter, 1985] which is widely used for streaming data.
    - A new instance will randomly replace one instance inside the buffer.
- Although this reduces the storage costs, the buffer size needs to be set sufficiently large.

Existing Methods

$$L(\mathrm{w}) = \frac{\lambda}{2}\|\mathrm{w}\|^2 + \sum_{i=1}^{n_+}\sum_{j=1}^{n_-}\frac{(1-\mathrm{w}^T(\mathrm{x}_i^+ - \mathrm{x}_j^-))^2}{2n^+n^-} \qquad (12)$$

- The authors modified the loss function to a sum of losses over individual samples. They then observed that by taking the gradient, that you could easily update the mean and covariance matrix by $c_t^- = \sum_{i:y_i=-1}\mathrm{x}_i/T_t^-$ and $S_t^- = \sum_{i:y_i=-1}(\mathrm{x}_i\mathrm{x}_i^T - c_t^-[c_t^-]^T)/T_t^-$, respectively.
- The same approach can be applied for when $y_t = -1$. Then the gradient solution of w_{t+1} is updated by $\mathrm{w}_t = \mathrm{w}_t - \eta_t \nabla L(\mathrm{w}_t)$.
- A significant drawback of this solution is the storage of the covariance matrix.

[Gao et al., 2013]

Existing Methods

- The previous two methods only represent a small sample of approaches to developing methods for AUC optimization.
- Work on optimizing AUC has been done using a variety of methods and continues to be an active area of research.
  - Online Learning [Ding et al., 2016, Ding et al., 2017, Liu et al., 2018, Lei and Ying, 2019]
  - Deep Learning [Liu et al., 2019]
  - Variance Reduction [Dan and Sahoo, 2019]

Stochastic Online AUC Maximization

Theorem

AUC optimization (7) in the linear case is equivalent to the following saddle point problem:

$$\min_{\mathrm{w},a,b} \max_{\alpha \in \mathbb{R}} \Big\{ \mathbb{E}[F(\mathrm{w}, a, b, \alpha; z)] + \Omega(\mathrm{w}) \Big\}, \tag{13}$$

where the expectation is with respect to $z = (\mathrm{x}, y)$, $p = Pr(y = 1)$, and

$$\begin{aligned} F(\mathrm{w}, a, b, \alpha; z) = {} & (1 - \hat{p})(\mathrm{w}^\top \mathrm{x} - a)^2 \mathbb{I}_{[y=1]} + \hat{p}(\mathrm{w}^\top \mathrm{x} - b)^2 \mathbb{I}_{[y=-1]} \\ & + 2(1+\alpha)\mathrm{w}^\top \mathrm{x}(\hat{p}\mathbb{I}_{[y=-1]} - (1-\hat{p})\mathbb{I}_{[y=1]}) \\ & - \hat{p}(1-\hat{p})\alpha^2. \end{aligned}$$

Summary of Existing Work

Algorithm	Loss	Penalty	Storage	Iteration	Rate
OAM	Hinge	L^2	$\mathcal{O}(td)$	$\mathcal{O}(td)$	$\mathcal{O}(1/\sqrt{T})$
OPAUC	Least-Square	L^2	$\mathcal{O}(d^2)$	$\mathcal{O}(d^2)$	$\mathcal{O}(1/\sqrt{T})$
SOLAM	Least-Square	L^2	$\mathcal{O}(d)$	$\mathcal{O}(d)$	$\mathcal{O}(1/\sqrt{T})$
SPAM	Least-Square	General	$\mathcal{O}(d)$	$\mathcal{O}(d)$	$\mathcal{O}(1/T)$
SPDAM	Least-Square	L^2	$\mathcal{O}(md)$	$\mathcal{O}(md)$	$\mathcal{O}(\theta^t)$

- Can we improve the current convergence rate for AUC maximization?
- Can we include other penalty terms besides L^2?

Completed Work #1

- First, we begin with Theorem 1.
- For fixed w, it is easy to see that the optima for a, b, and α are respectively achieved at

$$a(\mathrm{w}) = \mathrm{w}^\top \mathbb{E}[\mathrm{x}|y = 1], \quad b(\mathrm{w}) = \mathrm{w}^\top \mathbb{E}[\mathrm{x}|y = -1], \tag{14}$$

$$\alpha(\mathrm{w}) = \mathrm{w}^\top (\mathbb{E}[\mathrm{x}|y' = -1] - \mathbb{E}[\mathrm{x}|y = 1]). \tag{15}$$

- Using the updates for a, b, and α, our new AUC optimization formulations becomes

$$\min_{\mathrm{w}} \mathbb{E}[F(\mathrm{w}, a(\mathrm{w}), b(\mathrm{w}), \alpha(\mathrm{w}); z_t)] + \Omega(\mathrm{w}) \tag{16}$$

[Natole et al., 2018]

Proximal Step

- The proximal step is given by

$$\text{prox}_{\eta_t\Omega}(\hat{\mathrm{w}}_{t+1}) = \arg\min \left\{ \frac{1}{2}\|\hat{\mathrm{w}}_{t+1} - \mathrm{w}\|_2^2 + \eta_t\Omega(\mathrm{w}) \right\}$$

- The main idea is to find a point w that is close to $\hat{\mathrm{w}}_{t+1}$, the solution from the gradient step.
- The proximal operator reduces to Euclidean projection when $\Omega(\mathrm{w})$ is an indicator function.
- Because of the use of the proximal operator, SPAM can accommodate for a non-smooth penalty term $\Omega(\cdot)$.
- We will consider for $\Omega(\mathrm{w})$ the following:
  - L^2, i.e. $\Omega(\mathrm{w}) = \frac{\beta}{2}\|\mathrm{w}\|_2^2$
  - Elastic Net, i.e. $\Omega(\mathrm{w}) = \frac{\beta}{2}\|\mathrm{w}\|_2^2 + \beta_1\|\mathrm{w}\|_1$

[Parikh et al., 2014, Zou and Hastie, 2005]

Stochastic Proximal AUC Maximization

SPAM

Input: Step sizes $\{\eta_t > 0 : t \in \mathbb{N}\}$
Initialize $\mathrm{w}_1 \in \mathbb{R}^d$.
for $t = 1$ to T **do**
  Receive sample $z_t = (\mathrm{x}_t, y_t)$
  Compute $a(\mathrm{w}_t)$, $b(\mathrm{w}_t)$, and $\alpha(\mathrm{w}_t)$ according to (14) and (15).
  $\hat{\mathrm{w}}_{t+1} = \mathrm{w}_t - \eta_t \partial_1 F(\mathrm{w}_t, a(w_t), b(\mathrm{w}_t), \alpha(\mathrm{w}_t); z_t)$
  $\mathrm{w}_{t+1} = \mathrm{prox}_{\eta_t \Omega}(\hat{\mathrm{w}}_{t+1})$
end for

- **Note:** $\partial_1 F$ denotes the partial derivative of F with respect to the first argument.

Key Lemma

Lemma

Let w_t be given by SPAM as described and let $f(\mathrm{w}) = p(1-p)\mathbb{E}_{z_t}[(1-\mathrm{w}^T(\mathrm{x}-\mathrm{x}'))^2|y=1, y'=-1]$. Then, we have that

$$\partial f(\mathrm{w}) = \mathbb{E}_{z_t}[\partial_1 F(\mathrm{w}_t, a(\mathrm{w}_t), b(\mathrm{w}_t), \alpha(\mathrm{w}_t); z_t)] \tag{17}$$

where ∂_1 denotes the partial derivative with respect to the first argument.

- The above lemma implies, conditioned on $\{z_1, \ldots, z_{t-1}\}$, that $\partial_1 F(\mathrm{w}_t, a(\mathrm{w}_t), b(\mathrm{w}_t), \alpha(\mathrm{w}_t); z_t)$ is an unbiased estimator of the true gradient $\partial_{\mathrm{w}} f(\mathrm{w}_t)$.

Convergence Analysis: Assumptions

- The convergence results are established based on the following two assumptions:
  - (A1) Assume that $\Omega(\cdot)$ is β-strongly convex.
  - (A2) There exists an $M > 0$ such that $\|\mathrm{x}\| \leq M$ for any $x \in \mathcal{X}$.
- Furthermore, we define the following constants:

$$C_{\beta,M} := \frac{\beta}{128M^4} \qquad \widetilde{C}_{\beta,M} = \frac{\beta}{(1 + \frac{\beta^2}{128M^4})^2}$$

- We use the conventional notation that for any $T \in \mathbb{N}$, $\mathbb{N}_T = \{1, \dots, T\}$.
- Let w^* denote the optimal solution of formulation (16), i.e.,

$$\mathrm{w}^* = \arg\min_{\mathrm{w}\in\mathbb{R}^d} \{\mathbb{E}[F(\mathrm{w}, a(\mathrm{w}), b(\mathrm{w}), \alpha(\mathrm{w}); z_t)] + \Omega(\mathrm{w}))\}.$$

Convergence Analysis: Summary

Theorem

Under the assumptions (A1), (A2), and choosing step sizes with some $\theta \in (0,1)$ in the form of $\left\{\eta_t = \frac{C_{\beta,M}}{t^\theta} : t \in \mathbb{N}\right\}$, the algorithm SPAM achieves a convergence rate of $\mathcal{O}(T^{-\theta})$.

Theorem

Under the assumptions of (A1), (A2), and choosing step sizes $\{\eta_t = [\widetilde{C}_{\beta,M}(t+1)]^{-1} : t \in \mathbb{N}\}$, the algorithm SPAM achieves a convergence rate of $\mathcal{O}\left(\log T/T\right)$.

- The convergence of SPAM can achieve $\mathcal{O}(1/T)$ up to a logarithmic term, which matches the optimal rate of standard stochastic gradient descent

Experimental Setup

- For training, 80% of the data was used for training while the remaining data was used for testing.
- The results are based on 20 runs for each dataset to compute the average and standard deviation.
- All experiments were conducted with MATLAB and the codes the compared methods where obtained from the authors.

datasets	♯inst	♯feat	datasets	♯inst	♯feat
diabetes	768	8	fourclass	862	2
german	1,000	24	splice	3,175	60
usps	9,298	256	a9a	32,561	123
mnist	60,000	780	acoustic	78,823	50
ijcnn1	141,691	22	covtype	581,012	54
sector	9,619	55,197	news20	15,935	62,061

Table: *Basic information about the benchmark datasets used in the experiments.*

Experimental Setup

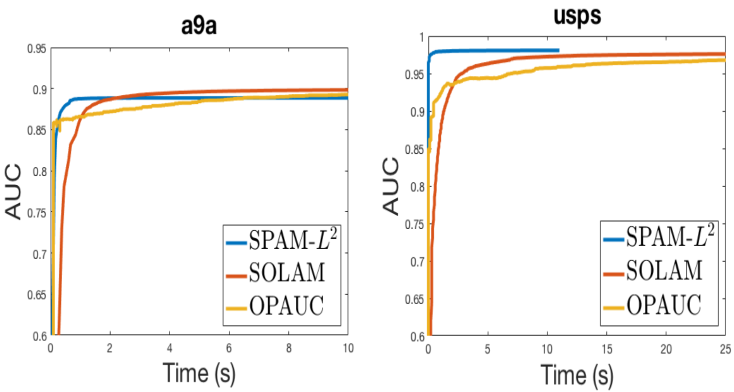
- **SPAM-L^2**: The proposed stochastic proximal algorithm for AUC maximization with Frobenious norm.
- **SPAM-NET**: The proposed stochastic proximal algorithm for AUC maximization with elastic net.
- **SOLAM**: The online projected gradient descent algorithm for AUC maximization.
- **OPAUC**: The one-pass AUC optimization algorithm with square loss function.
- **OAMseq**: The OAM algorithm with reservoir sampling and sequential updating method.
- **OAMgra**: The OAM algorithm with reservoir sampling and online gradient updating method.
- **B-LS-SVM**: A batch learning algorithm which optimizes the pairwise square loss [Joachims, 2006] .

Evaluation on Test Data

Data	SPAM-L^2	SPAM-NET	SOLAM	OPAUC	OAM_{seq}	OAM_{gra}	B-LS-SVM
diabetes	.8272$\pm$.0277	.8085$\pm$.0431	.8128$\pm$.0304	.8309$\pm$.0350	.8264$\pm$.0367	.8262$\pm$.0338	.8325$\pm$.0329
fourclass	.8210$\pm$.0203	.8211$\pm$.0205	.8213$\pm$.0209	.8310$\pm$.0251	.8306$\pm$.0247	.8295$\pm$.0251	.8309$\pm$.0309
german	.7942$\pm$.0388	.7937$\pm$.0386	.7778$\pm$.0373	.7978$\pm$.0347	.7747$\pm$.0411	.7723$\pm$.0358	.7994$\pm$.0343
splice	.9263$\pm$.0091	.9267$\pm$.0090	.9246$\pm$.0087	.9232$\pm$.0099	.8594$\pm$.0194	.8864$\pm$.0166	.9245$\pm$.0092
usps	.9868$\pm$.0032	.9855$\pm$.0029	.9822$\pm$.0036	.9620$\pm$.0040	.9310$\pm$.0159	.9348$\pm$.0122	.9634$\pm$.0045
a9a	.8998$\pm$.0046	.8980$\pm$.0047	.8966$\pm$.0043	.9002$\pm$.0047	.8420$\pm$.0174	.8571$\pm$.0173	.8982$\pm$.0028
mnist	.9254$\pm$.0025	.9132$\pm$.0026	.9118$\pm$.0029	.9242$\pm$.0021	.8615$\pm$.0087	.8643$\pm$.0112	.9336$\pm$.0025
acoustic	.8120$\pm$.0030	.8109$\pm$.0028	.8099$\pm$.0036	.8192$\pm$.0032	.7113$\pm$.0590	.7711$\pm$.0217	.8210$\pm$.0033
ijcnn1	.9174$\pm$.0024	.9155$\pm$.0024	.9129$\pm$.0030	.9269$\pm$.0021	.9209$\pm$.0079	.9100$\pm$.0092	.9320$\pm$.0037
covtype	.9504$\pm$.0011	.9508$\pm$.0011	.9503$\pm$.0012	.8244$\pm$.0014	.7361$\pm$.0317	.7403$\pm$.0289	.8222$\pm$.0014
sector	.8768$\pm$.0126	.9077$\pm$.0104	.8767$\pm$.0129	.9292$\pm$.0081	.9163$\pm$.0087	.9043$\pm$.0100	-
news20	.8708$\pm$.0069	.8704$\pm$ .0070	.8712$\pm$.0073	.8871$\pm$.0083	.8543$\pm$.0099	.8346$\pm$.0094	-

- Comparison of AUC values (mean$\pm$std) on the evaluated datasets.

Running Time Comparison



Completed Work #2

- In this work, we determine if we can achieve a faster rate of convergence by compromising on the per-iteration cost.
- Recall the empirical risk minimization problem for AUC:

$$\operatorname{argmin}_{\mathrm{w}} \frac{1}{n^+ n^-} \sum_{i=1}^{n_+} \sum_{j=1}^{n_-} (1 - \mathrm{w}^\top (\mathrm{x}_i - \mathrm{x}_j))^2 \mathbb{I}_{[y_i = 1 \wedge y_j = -1]} + \frac{\lambda}{2} \|\mathrm{w}\|^2.$$

- Denote by $\mathbb{N}_T = \{1, 2, \ldots, T\}$ for any $T \in \mathbb{N}$. Now, when ρ is a uniform distribution over finite data $\{(x_i, y_i) : i \in \mathbb{N}_T\}$, we can reformulate Theorem 1 as a:

$$\min_{\substack{\mathrm{w} \in \mathbb{R}^d \\ (a,b) \in \mathbb{R}^2}} \max_{\alpha \in \mathbb{R}} \frac{1}{T} \sum_{i \in \mathbb{N}_T} F(\mathrm{w}, a, b, \alpha, z_i) \tag{18}$$

New Formulation

- Using the definition of $F(\mathrm{w}, a, b, \alpha, z_i)$, we now consider the following general saddle point problem for AUC maximization

$$\min_{\mathrm{w},a,b} \max_{\alpha} \Big\{ \frac{1}{n_+} \sum_{i \in \mathbb{N}_n} (\mathrm{w}^\top \mathrm{x}_i - a)^2 \mathbb{I}_{y_i=1} + \frac{1}{n_-} \sum_{i \in \mathbb{N}_n} (\mathrm{w}^\top \mathrm{x}_i - b)^2 \mathbb{I}_{y_i=-1}$$
$$+ 2(1+\alpha)\mathrm{w}^\top \Big[\frac{1}{n_-} \sum_{i \in \mathbb{N}_n} \mathrm{x}_i \mathbb{I}_{y_i=-1} - \frac{1}{n_+} \sum_{i \in \mathbb{N}_n} \mathrm{x}_i \mathbb{I}_{y_i=1} \Big] - \alpha^2$$
$$+ \Omega(\mathrm{w}) \Big\} \qquad (19)$$

where $\Omega(\mathrm{w})$ is a penalty term. If $\Omega(\mathrm{w}) = \mathbb{I}_{\|\mathrm{w}\| \le R}(\mathrm{w})$, the above formulation is equivalent to the saddle point formulation (18).

Key Ideas

- Before we apply the motivation ideas, the following notations will be useful: let $p = \frac{n_+}{n}$ and $\mathrm{b} = \mathrm{m}_- - \mathrm{m}_+$ where m_+ and m_- are the means of the positive and negative classes, respectively, i.e.

$$\mathrm{m}_+ = \frac{1}{n_+} \sum_{i \in \mathbb{N}_n} \mathrm{x}_i \mathbb{I}_{y_i = 1} \text{ and } \mathrm{m}_- = \frac{1}{n_-} \sum_{i \in \mathbb{N}_n} \mathrm{x}_i \mathbb{I}_{y_i = -1}.$$

- For any $i \in \mathbb{N}_n$, denote

$$\bar{x}_i = \frac{\mathrm{x}_i - \mathrm{m}_+}{\sqrt{2p}} \quad \text{if } y_i = 1, \qquad \bar{x}_i = \frac{\mathrm{x}_i - \mathrm{m}_-}{\sqrt{2(1-p)}} \quad \text{if } y_i = -1. \tag{20}$$

- Let $g(\mathrm{w}) = \frac{|\mathrm{b}^\top \mathrm{w}|^2}{2} + \mathrm{b}^\top \mathrm{w} + \Omega(\mathrm{w})$. To satisfy the hypothesis that g is a λ strong convex function, we will let $\Omega(\mathrm{w}) = \frac{\lambda}{2}\|\mathrm{w}\|^2$.

Algorithm Formulation

- By minimizing out a, b, and α in (19) and using (20), we obtain the following new formulation:

$$\min_{\mathrm{w}} \max_{\beta} \left\{ \frac{1}{n} \sum_{i \in \mathbb{N}_n} \beta_i \mathrm{w}^\top \bar{x}_i - \frac{\|\beta\|^2}{2} + g(\mathrm{w}) \right\}$$

where $g : \mathbb{R}^d \to \mathbb{R}$ is defined, for any $\mathrm{w} \in \mathbb{R}^d$, by

$$g(\mathrm{w}) = \frac{|\mathrm{b}^\top \mathrm{w}|^2}{2} + \mathrm{b}^\top \mathrm{w} + \frac{\lambda}{2} \|\mathrm{w}\|^2.$$

Algorithm Formulation

- The following algorithm is inspired by the stochastic primal-dual algorithm as in [Zhang and Xiao, 2017, Yu et al., 2015]
- As from before, we uniformly and randomly select a mini-batch of size m.
- The critical ideas are as follows:
  - First, we solve the dual variable,
  - Second, solve for the primal variable w.
  - The auxiliary variables (u^t and $\bar{u}^{t+1}$) are similar to Nesterov's acceleration technique [Nesterov, 2013] to help the algorithm yield a faster rate of convergence.

Solution to Dual Variable

- The first step is to solve for the dual variable, $\beta_i^{(t+1)}$.

$$\beta_i^{(t+1)} = \begin{cases} \text{argmax}_{\beta_i \in \mathbb{R}} \left\{ \beta_i \langle \bar{\mathrm{w}}^{(t)}, \mathrm{x}_i \rangle - \frac{|\beta_i|^2}{2} - \frac{|\beta_i - \beta_i^{(t)}|^2}{2\sigma} \right\} & \text{if } i \in I \\ \beta_i^{(t)} & \text{otherwise.} \end{cases}$$

$$u^{(t+1)} = u^{(t)} + \frac{1}{n} \sum_{i \in I} (\beta_i^{(t+1)} - \beta_i^{(t)}) \mathrm{x}_i.$$

$$\bar{u}^{(t+1)} = u^{(t)} + \frac{n}{m} (u^{(t+1)} - u^{(t)})$$

- The additional steps are based on Nesterov's formulation to increase the rate of convergence.

Solution to the Primal Variable

- Second, solve for the primal variable w.

$$\mathrm{w}^{(t+1)} = \operatorname{argmin}_{\mathrm{w}\in\mathbb{R}^d}\Big\{\langle\bar{u}^{(t+1)},\mathrm{w}\rangle + g(\mathrm{w}) + \frac{\|\mathrm{w}-\mathrm{w}^{(t)}\|^2}{2\tau}\Big\}.$$

$$\bar{\mathrm{w}}^{(t+1)} = \mathrm{w}^{(t+1)} + \theta(\mathrm{w}^{(t+1)} - \mathrm{w}^{(t)}).$$

Theorem

Assume that g is λ-strongly convex. Let (w^*, β^*) be the saddle point of (13). If the parameters σ, τ and θ are chosen in a specific manner, then

$$\mathbb{E}[\|\mathrm{w}^{t+1} - \mathrm{w}^*\|] = \mathcal{O}(\theta^t)$$

.

Experimental Setup

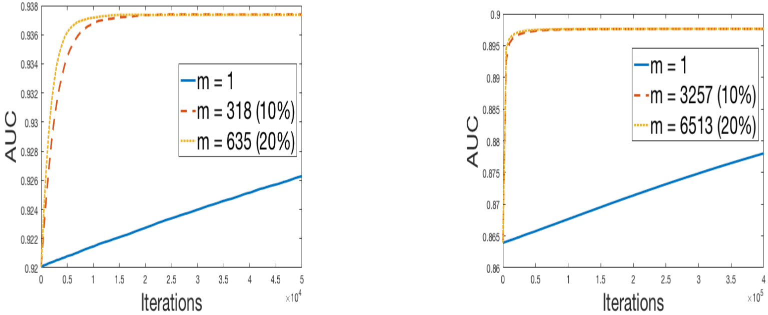
- Same setup as for the previous algorithm.
- Comparison Algorithms
  - **SPDAM**: The proposed stochastic primal-dual algorithm for AUC maximization.
  - **regSOLAM**: The proposed regularized online projected gradient descent algorithm for AUC maximization.
  - **Online Uni-Exp**: Online learning algorithm which optimizes the (weighted) univariate exponential loss [Kotlowski et al., 2011].
  - **B-SVM-OR**: A batch learning algorithm which optimizes the pairwise hinge loss [Joachims, 2006]
  - **B-LS-SVM**: A batch learning algorithm which optimizes the pairwise square loss.
  - Previous Algorithms
    - OPAUC
    - OAMgra

Experiments

Datasets	SPDAM	regSOLAM	OPAUC	$\mathrm{OAM_{gra}}$	online Uni-Exp	B-SVM-OR	B-LS-SVM
diabetes	.8275±.0302	.8140±.0330	.8309±.0350	.8262±.0338	.8215±.0309	.8326±.0328	.8325±.0329
fourclass	.8223±.0275	.8222±.0276	.8310±.0251	.8295±.0251	.8281±.0305	.8305±.0311	.8309±.0309
german	.7959±.0265	.7830±.0247	.7978±.0347	.7723±.0358	.7908±.0367	.7935±.0348	.7994±.0343
splice	.9227±.0128	.9237±.0090	.9232±.0099	.8864±.0166	.8931±.0213	.9239±.0089	.9245±.0092
usps	.9854±.0019	.9848±.0021	.9620±.0040	.9348±.0122	.9538±.0045	.9630±.0047	.9634±.0045
a9a	.8967±.0032	.8970±.0048	.9002±.0047	.8571±.0173	.9005±.0024	.9009±.0036	.8982±.0028
mnist	.9552±.0011	.9599±.0014	.9242±.0021	.8643±.0112	.7932±.0245	.9340±.0020	.9336±.0025
acoustic	.8119±.0039	.8114±.0035	.8192±.0032	.7711±.0217	.8171±.0034	.8262±.0032	.8210±.0033
ijcnn1	.9132±.0016	.9108±.0030	.9269±.0021	.9100±.0092	.9264±.0035	.9337±.0024	.9320±.0037
covtype	.9409±.0011	.9332±.0020	.8244±.0014	.7403±.0289	.8236±.0017	.8248±.0013	.8222±.0014
sector	.9406±.0062	.9734±.0036	.9292±.0081	.9043±.0100	.9215±.0034	-	-

- Comparison of AUC values (mean±std) on the evaluated datasets.

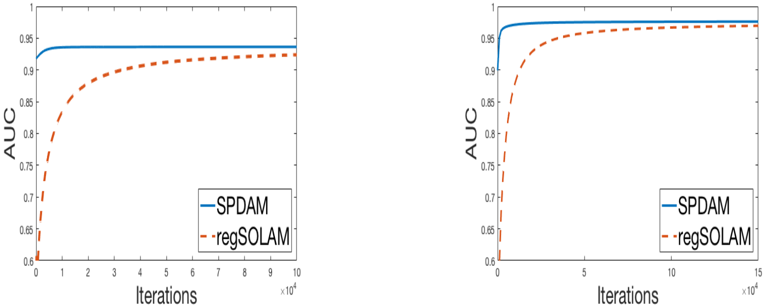
Convergence Rate: Batch Sizes



(a) splice (b) a9a

Figure: *AUC vs. Iteration curves of SPDAM algorithm for various batch sizes. The batch size is a percentage of the number of samples.*

Convergence Rate: SPDAM vs. regSPAM



(a) splice (b) usps

Figure: *AUC vs. Iteration curves of SPDAM against regSOLAM. For SPDAM, 10% of the data was chosen for a batch size. The optimal value of the parameter λ from SPDAM was used in regSOLAM.*

Benchmark Datasets

Dataset	♯inst	♯feat	Dataset	♯inst	♯feat
a9a	32,561	123	ijcnn1	141,691	22
acoustic	78,823	50	ionosphere	351	34
alpha	500,000	500	mnist	60,000	780
beta	500,000	500	news20	15,935	62,061
covtype	581,012	54	sector	9,619	55,197
diabetes	768	8	splice	3,175	60
fourclass	862	2	svmguide3	1243	21
german	1,000	24	usps	9,298	256

Table: *Summary of standard benchmark datasets used in the experiments.*

Anomaly Detection Tasks

- **Malicious Websites**. We can apply the algorithms to determine if a website is malicious or not using the *webspam* dataset.
- **Bioinformatics** Detecting noncoding RNAs from sequenced genomes will be done using the *cod-rna* dataset.
- **Credit Card Fraud**. The *australian* dataset is used for predicting credit card fraud detection.
- **Medical Diagnosis** The datasets *arrhythmia*, *breast-cancer*, *mammography*, and *thyroid* are used for detecting various illnesses.
- **Spam Filter** The *spambase* dataset is used for determining whether an email is considered legitimate or not.

Anomaly Detection Datasets

datasets	♯inst	♯feat	datasets	♯inst	♯feat
arrhythmia	452	274	mammography	11183	6
australian	690	14	spambase	4601	57
bio	145,751	73	thyroid	3772	6
breastw	683	9	webspam	350,000	254

Table: *Summary of datasets used for anomaly detection.*

Benchmark Dataset Results

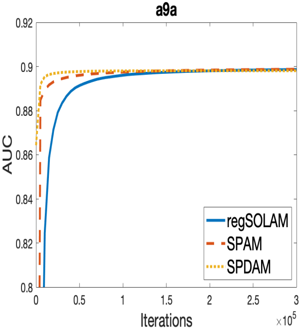
Datasets	regSOLAM	SPAM	SPDAM
a9a	$.8951 \pm .0046$	$.8995 \pm .0041$	$.8969 \pm .0048$
acoustic	$.7926 \pm .0040$	$.8055 \pm .0084$	$.8153 \pm .0032$
alpha	$.8152 \pm .0025$	$.8525 \pm .0027$	$.8152 \pm .0012$
beta	$.5011 \pm .0019$	$.5037 \pm .0011$	$.5033 \pm .0006$
covtype	$.7658 \pm .0156$	$.7990 \pm .0001$	$.8197 \pm .0013$
diabetes	$.8178 \pm .0309$	$.8269 \pm .0339$	$.8287 \pm .0311$
fourclass	$.8212 \pm .0209$	$.8214 \pm .0214$	$.8217 \pm .0205$
german	$.7765 \pm .0360$	$.7899 \pm .0313$	$.7913 \pm .0302$
ijcnn1	$.9161 \pm .0024$	$.9285 \pm .0019$	$.9145 \pm .0019$
ionosphere	$.8821 \pm .0400$	$.9064 \pm .0376$	$.9292 \pm .0364$
mnist	$.9267 \pm .0093$	$.9467 \pm .0067$	$.9356 \pm .0028$
news20	$.9399 \pm .0038$	$.8708 \pm .0069$	$.8655 \pm .0028$
sector	$.9734 \pm .0036$	$.8768 \pm .0126$	$.9406 \pm .0062$
splice	$.9100 \pm .0155$	$.9173 \pm .0143$	$.9243 \pm .0125$
svmguide3	$.6488 \pm .0328$	$.6073 \pm .0490$	$.7227 \pm .0408$
usps	$.9690 \pm .0033$	$.9775 \pm .0032$	$.9791 \pm .0033$

Results

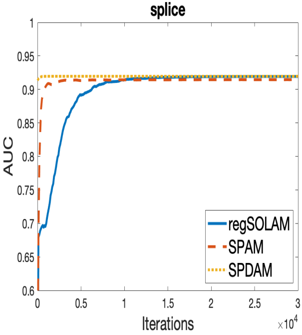
Datasets	regSOLAM	SPAM	SPDAM
arrhythmia	.8284±.0775	.8523±.0672	.8738±.0576
australian	.7178±.0462	.7178±.0466	.7656±.0406
breastw	.9308±.0208	.9352±.0168	.9315±.0204
cod-rna	.9930±.0001	.9062±.0025	.9931 ±.0001
mammography	.7815±.2305	.9178±.0205	.9152±.0181
spambase	.6491±.0673	.7232±.0204	.7716±.0277
thyroid	.9972±.0023	.9976±.0014	.9976±.0012
webspam	.9609±.0022	.9660±.0005	.9527±.0006

Table: *Comparison of the testing AUC values (mean±std.) on anomaly detection datasets.*

Convergence Rate: Iterations



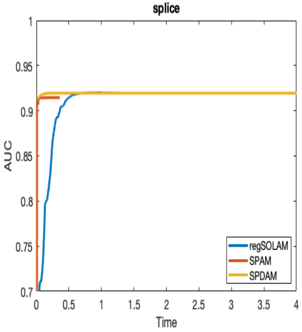
(a) splice



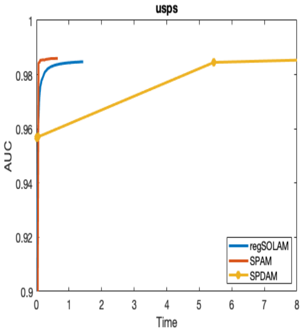
(b) usps

Figure: *AUC vs. Iteration curves of SPDAM against regSOLAM. For SPDAM, 10% of the data was chosen for a batch size. The optimal value of the parameter λ from SPDAM was used in regSOLAM.*

Convergence Rate: Time



(a) splice



(b) usps

Figure: *AUC vs. Iteration curves of SPDAM against regSOLAM. For SPDAM, 10% of the data was chosen for a batch size. The optimal value of the parameter λ from SPDAM was used in regSOLAM.*

Conclusion

Algorithm	Loss	Penalty	Storage	Iteration	Rate
OAM	Hinge	L^2	$\mathcal{O}(td)$	$\mathcal{O}(td)$	$\mathcal{O}(1/\sqrt{T})$
OPAUC	Least-Square	L^2	$\mathcal{O}(d^2)$	$\mathcal{O}(d^2)$	$\mathcal{O}(1/\sqrt{T})$
regSOLAM	Least-Square	L^2	$\mathcal{O}(d)$	$\mathcal{O}(d)$	$\mathcal{O}(1/\sqrt{T})$
SPAM	Least-Square	General	$\mathcal{O}(d)$	$\mathcal{O}(d)$	$\mathcal{O}(1/T)$
SPDAM	Least-Square	L^2	$\mathcal{O}(md)$	$\mathcal{O}(md)$	$\mathcal{O}(\theta^t)$

- Developed a stochastic proximal algorithm for AUC maximization with a convergence rate of $\mathcal{O}(1/T)$
- Developed a stochastic proximal algorithm with a linear convergence rate.
- Demonstrated the proposed methods on anomaly detection tasks.

Possible Future Work

- Variance Reduction methods for AUC optimization using [Johnson and Zhang, 2013, Johnson and Zhang, 2013] and stochastic primal-dual algorithms [Zhang and Xiao, 2017]
- Extend the work presented here to include kernel functions [Dai et al., 2014]
- Many of the ideas can also be extended to optimizing the area under a lift chart [Ling and Li, 1998, Shen et al., 2007]

Publications

- **Natole, Jr., M.**, Ying, Y., and Lyu, S. (2018)
  Stochastic Proximal Algorithms for AUC Maximization
  In *International Conference on Machine Learning*, pages 3707-3716, 2018.
- **Natole, Jr., M.**, Ying, Y., and Lyu, S. (2019)
  Stochastic AUC Optimization Algorithms with Linear Convergence.
  In *Frontiers in Applied Mathematics and Statistics*, 5:30, 2019.
- **Natole, Jr., M.**, Ying, Y., Buyantuev, A., Stessin, M., Buyantuev, V., and Lapenas, A.
  Climate Warming as Principle Control of Forest Mega-Fires in East Siberia
  TBD.

Thank you!

References I

Bottou, L. and Cun, Y. L. (2004).
Large scale online learning.
In *Advances in neural information processing systems*.

Bradley, A. P. (1997).
The use of the area under the roc curve in the evaluation of machine learning algorithms.
Pattern recognition, 30(7):1145–1159.

Dai, B., Xie, B., He, N., Liang, Y., Raj, A., Balcan, M.-F. F., and Song, L. (2014).
Scalable kernel methods via doubly stochastic gradients.
In *Advances in Neural Information Processing Systems*, pages 3041–3049.

References II

Dan, S. and Sahoo, D. (2019).
Variance reduced stochastic proximal algorithm for auc maximization.
arXiv preprint arXiv:1911.03548.

Ding, Y., Liu, C., Zhao, P., and Hoi, S. C. (2017).
Large scale kernel methods for online auc maximization.
In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 91–100. IEEE.

Ding, Y., Zhao, P., Hoi, S. C., and Ong, Y.-S. (2016).
Adaptive subgradient methods for online auc maximization.
arXiv preprint arXiv:1602.00351.

References III

Fawcett, T. (2006).
An introduction to roc analysis.
Pattern recognition letters, 27(8):861–874.

Gao, W., Jin, R., Zhu, S., and Zhou, Z.-H. (2013).
One-pass auc optimization.
In *International Conference on Machine Learning*, pages 906–914.

Hanley, J. A. and McNeil, B. J. (1982).
The meaning and use of the area under a receiver operating characteristic (roc) curve.
Radiology, 143(1):29–36.

References IV

Joachims, T. (2006).
Training linear svms in linear time.
In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 217–226, New York, NY, USA. ACM.

Johnson, R. and Zhang, T. (2013).
Accelerating stochastic gradient descent using predictive variance reduction.
In *Advances in neural information processing systems*, pages 315–323.

Kotlowski, W., Dembczynski, K., and Hüllermeier, E. (2011).
Bipartite ranking through minimization of univariate loss.
In *ICML*.

References V

Lei, Y. and Ying, Y. (2019).
Stochastic proximal auc maximization.
arXiv preprint arXiv:1906.06053.

Ling, C. X. and Li, C. (1998).
Data mining for direct marketing: Problems and solutions.
In *Kdd*, volume 98, pages 73–79.

Liu, M., Yuan, Z., Ying, Y., and Yang, T. (2019).
Stochastic auc maximization with deep neural networks.
arXiv preprint arXiv:1908.10831.

Liu, M., Zhang, X., Chen, Z., Wang, X., and Yang, T. (2018).
Fast stochastic auc maximization with $o(1/n)$-convergence rate.
In *International Conference on Machine Learning*, pages 3189–3197.

References VI

Moulines, E. and Bach, F. R. (2011).
Non-asymptotic analysis of stochastic approximation algorithms for machine learning.
In *Advances in Neural Information Processing Systems*, pages 451–459.

Natole, Jr., M., Ying, Y., and Lyu, S. (2018).
Stochastic proximal algorithms for AUC maximization.
In Dy, J. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3710–3719, Stockholmsmässan, Stockholm Sweden. PMLR.

Nesterov, Y. (2013).
Introductory lectures on convex optimization: A basic course, volume 87.
Springer Science & Business Media.

References VII

Parikh, N., Boyd, S., et al. (2014).
Proximal algorithms.
Foundations and Trends® in Optimization, 1(3):127–239.

Robbins, H. and Monro, S. (1951).
A stochastic approximation method.
The annals of mathematical statistics, pages 400–407.

Shen, A., Tong, R., and Deng, Y. (2007).
Application of classification models on credit card fraud detection.
In *2007 International conference on service systems and service management*, pages 1–4. IEEE.

References VIII

Srebro, N. and Tewari, A. (2010).
Stochastic optimization for machine learning.
ICML Tutorial.

Vitter, J. S. (1985).
Random sampling with a reservoir.
ACM Transactions on Mathematical Software (TOMS), 11(1):37–57.

Ying, Y. and Pontil, M. (2008).
Online gradient descent learning algorithms.
Foundations of Computational Mathematics, 8(5):561–596.

References IX

Yu, W., Lin, Q., and Yang, T. (2015).
Doubly stochastic primal-dual coordinate method for regularized empirical risk minimization with factorized data.
CoRR, abs/1508.03390.

Zhang, Y. and Xiao, L. (2017).
Stochastic primal-dual coordinate method for regularized empirical risk minimization.
The Journal of Machine Learning Research, 18(1):2939–2980.

Zhao, P., Jin, R., Yang, T., and Hoi, S. C. (2011).
Online auc maximization.
In *Proceedings of the 28th international conference on machine learning (ICML-11)*.

References X

Zinkevich, M. (2003).

Online convex programming and generalized infinitesimal gradient ascent.

In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 928–936.

Zou, H. and Hastie, T. (2005).

Regularization and variable selection via the elastic net.

Journal of the Royal Statistical Society: Series B (Statistical Methodology), 67(2):301–320.