Kernelized Online Imbalanced Learning with Fixed Budgets

Junjie Hu[1,2], Haiqin Yang[1,2], Irwin King[1,2], Michael R. Lyu[1,2], and Anthony Man-Cho So[3]
[1]Shenzhen Key Laboratory of Rich Media Big Data Analytics and Applications
Shenzhen Research Institute, The Chinese University of Hong Kong
[2]Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong
[3]Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Hong Kong
{jjhu, hqyang, king, lyu}@cse.cuhk.edu.hk, manchoso@se.cuhk.edu.hk

Abstract

Online learning from imbalanced streaming data to capture the nonlinearity and heterogeneity of the data is significant in machine learning and data mining. To tackle this problem, we propose a kernelized online imbalanced learning (KOIL) algorithm to directly maximize the area under the ROC curve (AUC). We address two more challenges: 1) How to control the number of support vectors without sacrificing model performance; and 2) how to restrict the fluctuation of the learned decision function to attain smooth updating. To this end, we introduce two buffers with fixed budgets (buffer sizes) for positive class and negative class, respectively, to store the learned support vectors, which can allow us to capture the global information of the decision boundary. When determining the weight of a new support vector, we confine its influence only to its k-nearest opposite support vectors. This can restrict the effect of new instances and prevent the harm of outliers. More importantly, we design a sophisticated scheme to compensate the model after replacement is conducted when either buffer is full. With this compensation, the learned model approaches the one learned with infinite budgets. We present both theoretical analysis and extensive experimental comparison to demonstrate the effectiveness of our proposed KOIL.

Introduction

Learning binary classification models from imbalanced data, where the number of instances from one class is significantly larger than that from the other class, is an important topic in both machine learning and data mining (Liang and Cohn 2013; Wu et al. 2008; Yang and King 2009). In the literature, area under the ROC curve (AUC) is an effective metric to measure the performance of classifiers learned from imbalanced data (Bradley 1997; Brefeld and Scheffer 2005; Hanley and McNeil 1982; Joachims 2005). Recently, online AUC maximization is proposed to tackle this problem when the data appears sequentially (Gao et al. 2013; Yang et al. 2010; Zhao et al. 2011). However, these algorithms only focus on seeking the decision function in a linear form. They cannot capture nonlinearity and heterogeneity of the data and miss the learning power of kernel methods.

Here, we aim at developing kernelized online imbalanced learning techniques, which are less explored, but important in both theory and applications. We address two more challenges. First, the learned kernel-based estimator becomes more complex as the number of observations increases (Kivinen, Smola, and Williamson 2004; Yang et al. 2011; 2012). Without suitable stream oblivious strategy, in the extreme case, the number of learned support vectors can be scaled to infinity. This is undesirable for large-scale applications. Although in the literature, refinement techniques, e.g., Projectron (Orabona, Keshet, and Caputo 2009), and online learning with fixed budgets algorithms, such as randomized budget perceptron (Cavallanti, Cesa-Bianchi, and Gentile 2007) and Forgetron (Dekel, Shalev-Shwartz, and Singer 2008), have been proposed, it is non-trival to tackle online imbalanced learning. Second, fluctuation by outliers is unavoidable in online learning (Cesa-Bianchi and Lugosi 2006; Ross, Mineiro, and Langford 2013; Karampatziakis and Langford 2011). How to attain smooth updating requires additional effort.

To this end, we propose a Kernelized Online Imbalanced Learning algorithm with fixed budgets, namely, **KOIL**, for online nonlinear AUC maximization. Our contributions are as follows:

- First, our proposed KOIL directly maximizes the AUC metric representing by the kernel expansion with fixed budgets, i.e., maintaining two buffers with the same buffer sizes to store the most informative data from each class as learned support vectors. This is an effective way to handle imbalanced streaming data while capturing their nonlinearity and heterogeneity (Kivinen, Smola, and Williamson 2004; Yang et al. 2013) and also sets it apart from Projectron (Orabona, Keshet, and Caputo 2009), which may include too many support vectors. Another important and novel feature is that we update the weights of the new and old support vectors by confining the influence of a new instance to its k-nearest opposite support vectors. This leads to smooth updating and makes KOIL different from previously proposed online AUC maximization algorithms (Gao et al. 2013; Zhao et al. 2011), which update the weight of a new instance based on all information stored in the buffers.
- Second, other than the standard stream oblivious policies, such as First-In-First-Out (FIFO) and reservoir

sampling (RS), which replace a support vector when either buffer is full, we design a sophisticated scheme to compensate the loss when a support vector is removed. We show that after the compensation, the learned decision function approaches the one learned with infinite budgets, i.e., sufficiently exploiting all the information along the training. This compensation scheme also makes our KOIL different from currently proposed online learning with fixed budgets algorithms (Cavallanti, Cesa-Bianchi, and Gentile 2007; Dekel, Shalev-Shwartz, and Singer 2008), which cast off information along training.

- Third, we demonstrate the effectiveness of confined updating and compensation in KOIL via both theoretical analysis and extensive experimental comparison.

KOIL for AUC Maximization

Notation and Problem Definition

We focus on learning a nonlinear decision function $f : \mathbb{R}^d \to \mathbb{R}$ from a sequence of imbalanced feature-labeled pair instances in binary classification, $\{\mathbf{z}_t = (\mathbf{x}_t, y_t) \in \mathcal{Z}, t \in [T]\}$, where $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, $\mathbf{x}_t \in \mathcal{X} \subseteq \mathbb{R}^d$, $y_t \in \mathcal{Y} = \{-1, +1\}$ and $[T] = \{1, \ldots, T\}$. Without loss of generality, we assume the positive class is the minority class while the negative class is the majority class. We denote $N_{t,k}^{\tilde{y}}(\mathbf{z})$ as the set of feature-labeled pair instances which are the k-nearest neighbors of $\mathbf{z}$ and have the label of $\tilde{y}$ at the t-th trial. Here, the neighborhood is defined by the distance or the similarity between two instances, i.e., the smaller the distance (or the more similarity) of instances, the close the neighbors. Besides, we define the index sets, I_t^+ and I_t^-, to record the indexes of positive support vectors and negative support vectors at the t-th trial. Moreover, for simplicity, we define two buffers, $\mathcal{K}_t^+$ and $\mathcal{K}_t^-$, to store the learned information for two classes at the t-th trial, respectively:

$$\mathcal{K}_t^{\mathfrak{s}}.\mathcal{A} = \{\alpha_{i,t}^{\mathfrak{s}} \,|\, \alpha_{i,t}^{\mathfrak{s}} \neq 0, i \in I_t^{\mathfrak{s}}\},$$
$$\mathcal{K}_t^{\mathfrak{s}}.\mathcal{B} = \{\mathbf{z}_i \,|\, y_i = \mathfrak{s}1, i \in I_t^{\mathfrak{s}}\},$$

where for simplicity, $\mathfrak{s}$ denotes $+$ or $-$, respectively, and $\alpha_{i,t}$ denotes the weight of the support vector firstly occurred at the i-th trial and updated at the t-th trial. We fix the budgets, i.e., the buffer sizes, to N. That is, $|I^+| = |I^-| = N$.

The objective of KOIL is to seek a decision function at the t-th trial expressed as follows:

$$f_t(\mathbf{x}) = \sum\nolimits_{i \in I_t^+} \alpha_{i,t}^+ k(\mathbf{x}_i, \mathbf{x}) + \sum\nolimits_{j \in I_t^-} \alpha_{j,t}^- k(\mathbf{x}_j, \mathbf{x}), \quad (1)$$

where the information of support vectors is stored at $\mathcal{K}_t^+$ and $\mathcal{K}_t^-$, respectively, and the prediction of a new sample $\mathbf{x}$ can be made by $\text{sgn}(f_t(\mathbf{x}))$. More generally, $f_t(\mathbf{x})$ is an element of a Reproducing Kernel Hilbert Space (RKHS) and can be expressed as $f_t(\mathbf{x}) = \langle f_t(\cdot), k(\mathbf{x}, \cdot)\rangle_{\mathcal{H}}$ to capture the nonlinearity and heterogeneity of the data (Schölkopf and Smola 2002).

Learning with Kernels for AUC Maximization

Given the positive dataset $\mathcal{D}^+ = \{\mathbf{z}_i | y_i = +1, i \in I^+\}$ and the negative dataset $\mathcal{D}^- = \{\mathbf{z}_j | y_j = -1, j \in I^-\}$, the AUC metric for a kernel representation function f is calculated by:

$$\begin{aligned} AUC(f) &= \frac{\sum_{i \in I^+} \sum_{j \in I^-} \mathbb{I}[f(\mathbf{x}_i) - f(\mathbf{x}_j) > 0]}{|I^+||I^-|} \\ &= 1 - \frac{\sum_{i \in I^+} \sum_{j \in I^-} \mathbb{I}[f(\mathbf{x}_i) - f(\mathbf{x}_j) \leq 0]}{|I^+||I^-|}, \end{aligned} \quad (2)$$

where $\mathbb{I}[\pi]$ is the indicator function that equals 1 when π is true and 0 otherwise. Hence, maximizing $AUC(f)$ is equivalent to minimizing $\sum_{i \in I^+} \sum_{j \in I^-} \mathbb{I}[f(\mathbf{x}_i^+) - f(\mathbf{x}_j^-) \leq 0]$. Since directly maximizing AUC score yields a combinatorial optimization problem, which is NP-hard (Cortes and Mohri 2003), the indicator function is usually replaced by its convex surrogate, e.g., the following pairwise hinge loss function (Gao et al. 2013; Zhao et al. 2011):

$$\ell_h(f, \mathbf{z}, \mathbf{z}') = \frac{|y - y'|}{2} \left[1 - \frac{1}{2}(y - y')(f(\mathbf{x}) - f(\mathbf{x}'))\right]_+, \quad (3)$$

where $[v]_+ = \max\{0, v\}$.

Hence, finding the decision function in Eq. (1) for AUC maximization is equivalent to minimizing the following objective function:

$$\mathcal{L}(f) = \frac{1}{2}\|f\|_{\mathcal{H}}^2 + C \sum\nolimits_{i \in I^+} \sum\nolimits_{j \in I^-} \ell_h(f, \mathbf{z}_i, \mathbf{z}_j), \quad (4)$$

where $\frac{1}{2}\|f\|_{\mathcal{H}}^2$ is the regularization term controlling the functional complexity and $C > 0$ is a penalty parameter balancing the functional complexity and training errors.

Online AUC Maximization by KOIL

In the online setting, our goal is to update the kernel decision function based on the arrival of a new instance $\mathbf{z}_t$. Considering the AUC approximation by its convex surrogate in Eq. (3), we can transform the objective of Eq. (4) to the following *instantaneous regularized risk of AUC* on $\mathbf{z}_t$:

$$\mathcal{L}_t(f) = \frac{1}{2}\|f\|_{\mathcal{H}}^2 + C \sum\nolimits_{i=1}^{t-1} \ell_h(f, \mathbf{z}_t, \mathbf{z}_i). \quad (5)$$

Different from NORMA, only measuring the predictive error of a new instance, the risk defined in Eq. (5) involves computing pairwise losses between $\mathbf{z}_t$ and all previous instances with the opposite label. Obviously, storing all previous instances is undesirable for large-scale applications.

To tackle the scalability issue, we introduce two buffers for each class with fixed budgets as in (Yang et al. 2013; Zhao et al. 2011) to keep track of the most informative positive and negative instances, which are also used as support vectors to construct the kernel decision function in Eq. (1). However, if the new instance counts all errors in the opposite buffer as in (Yang et al. 2013), the decision function is easily affected by outliers, which will cause much more errors and may yield decision functions fluctuated. To increase the robustness of the model, we propose a novel setting in the objective function, i.e., we only count the losses between the new instance $\mathbf{z}_t$ and its k-nearest opposite support vectors, i.e., the k-nearest support vectors

Algorithm 1 Kernelized Online Imbalanced Learning (KOIL) with Fixed Budgets

Input:
- the penalty parameter C and the learning rate η
- the maximum budget size N^+ and N^-
- the number of nearest neighbors k

Initialize $\mathcal{K}^+.\mathcal{A} = \mathcal{K}^-.\mathcal{A} = \emptyset, \mathcal{K}^+.\mathcal{B} = \mathcal{K}^-.\mathcal{B} = \emptyset, N_p = N_n = 0$
for $t = 1$ **to** T **do**
  Receive a training sample $\mathbf{z}_t = (\mathbf{x}_t, y_t)$
  if $y_t = +1$ **then**
    $N_p = N_p + 1$
    $[\mathcal{K}^-, \mathcal{K}^+, \alpha_t^+] = \text{UpdateKernel}(\mathbf{z}_t, \mathcal{K}^-, \mathcal{K}^+, C, \eta, k)$
    $\mathcal{K}^+ = \text{UpdateBuffer}(\alpha_t^+, \mathbf{z}_t, \mathcal{K}^+, k, N^+, N_p)$
  else
    $N_n = N_n + 1$
    $[\mathcal{K}^+, \mathcal{K}^-, \alpha_t^-] = \text{UpdateKernel}(\mathbf{z}_t, \mathcal{K}^+, \mathcal{K}^-, C, \eta, k)$
    $\mathcal{K}^- = \text{UpdateBuffer}(\alpha_t^-, \mathbf{z}_t, \mathcal{K}^-, k, N^-, N_n)$
  end if
end for

with the opposite label of the new instance. We then seek the decision function by minimizing the following *localized instantaneous regularized risk of AUC* on $\mathbf{z}_t$:

$$\hat{\mathcal{L}}_t(f) := \hat{\mathcal{L}}(f, \mathbf{z}_t) = \frac{1}{2}\|f\|_{\mathcal{H}}^2 + C \sum_{\mathbf{z}_i \in N_{t,k}^{-y_t}(\mathbf{z}_t)} \ell_h(f, \mathbf{z}_t, \mathbf{z}_i). \tag{6}$$

Note that k is usually set as a small value, e.g., around 10% of the budget. More importantly, this new setting contains the following advantages: 1) Maintaining two buffers with relatively larger budgets can keep track of the global information of the decision boundary; 2) only considering k-nearest opposite support vectors of the new instance allows us to utilize the local information around the new instance and avoid the fluctuation of the decision function.

We show the framework of Kernelized Online Imbalanced Learning with fixed budgets in Algorithm 1, which consists of two key components, **UpdateKernel** and **UpdateBuffer**, elaborated in the following. More detailed descriptions are shown in the Appendix.

Update Kernels We apply the classical stochastic gradient descent method to update the decision function at the t-th trial as follows:

$$f_{t+1} := f_t - \eta \partial_f \hat{\mathcal{L}}_t(f)|_{f=f_t}, \tag{7}$$

where ∂_f is shorthand for $\partial/\partial f$ (the gradient with respect to f), and $\eta > 0$ is the learning rate which can be a constant or decreases with the number of trials.

To compute the gradient of $\hat{\mathcal{L}}_t(f)$ with respect to f, we first calculate the gradient of ℓ_h with respect to f, i.e., $\partial_f \ell_h(f, \mathbf{z}_t, \mathbf{z}_i)$, by

$$\partial_f \ell_h(\cdot) = \begin{cases} 0, & \ell_h(f, \mathbf{z}_t, \mathbf{z}_i) \le 0, \\ -\varphi(\mathbf{z}_t, \mathbf{z}_i), & \ell_h(f, \mathbf{z}_t, \mathbf{z}_i) > 0 \end{cases}, \tag{8}$$

where $\varphi(\mathbf{z}_t, \mathbf{z}_i) = y_t(k(\mathbf{x}_t, \cdot) - k(\mathbf{x}_i, \cdot))$.

Hence, by substituting f_t and Eq. (8) for the gradient of Eq. (6), we obtain:

$$\partial_f \hat{\mathcal{L}}_t(f_t) = f_t - C \sum_{\mathbf{z}_i \in N_{t,k}^{-y_t}(\mathbf{z}_t)} \mathbb{I}[\ell_h(f, \mathbf{z}_t, \mathbf{z}_i) > 0] \varphi(\mathbf{z}_t, \mathbf{z}_i). \tag{9}$$

Practically, we initialize the first hypothesis to zero, i.e., $f_1 = 0$ and express the decision function at the t-th trial as a kernel expansion defined in Eq.(1) while updating the $(t+1)$-th trial in an incremental mode,

$$f_{t+1}(\mathbf{x}) = f_t(\mathbf{x}) + \alpha_{t,t} k(\mathbf{x}_t, \mathbf{x}). \tag{10}$$

For simplicity, we define the valid set V_t satisfying the indication function in Eq. (9) and its complementary set $\overline{V_t}$ at the t-th trial as follows:

$$V_t := \{i \in I_t^{-y_t} \mid \mathbf{z}_i \in N_{t,k}^{-y_t}(\mathbf{z}_t) \wedge \ell_h(f, \mathbf{z}_t, \mathbf{z}_i) > 0\},$$
$$\overline{V_t} := I_t^{-y_t} \setminus V_t. \tag{11}$$

Hence, the corresponding updating rule for the kernel weights at the t-th trial is derived as follows:

$$\alpha_{i,t} = \begin{cases} \eta C y_t |V_t|, & i = t \\ (1-\eta)\alpha_{i,t-1} - \eta C y_t, & \forall i \in V_t \\ (1-\eta)\alpha_{i,t-1}, & \forall i \in I_t^{y_t} \cup \overline{V_t} \end{cases} \tag{12}$$

The updating rule in Eq. (12) divides the data into three cases and leads to a tight regret bound; see the Theoretical Analysis section:

- For a new instance, we only count at most its k opposite pairwise losses. This is a key to prevent the fluctuation of the decision function. Otherwise, if we count all pairwise losses as (Yang et al. 2013), a new instance may yield undesired updating on those remote support vectors, which yield outlier effect.
- For the k-nearest opposite support vectors to the new instance $\mathbf{z}_t$, i.e., the support vectors in $N_{t,k}^{-y_t}(\mathbf{z}_t)$, we add their weights by $|\eta C y_t|$; see the second case in Eq. (12). This will keep a balanced updating, which is in favor of imbalanced data.
- When the new instance does not incur errors or the label of previously learned support vectors is the same as that of the new instance, the updating rule is the same as NORMA (Kivinen, Smola, and Williamson 2004), i.e., just decaying the weight by a constant factor, $1 - \eta$.

Update Buffers The setting of fixed budgets raises the problem of updating the buffer when it is full. How to maintain the buffers with the most informative support vectors to achieve stable model performance is a key challenge. Traditionally, First-In-First-Out (FIFO) and Reservoir Sampling (RS) are two typical stream oblivious policies (Vitter 1985) to update the buffers and have demonstrated their effectiveness in online linear AUC maximization (Zhao et al. 2011). However, they will degrade the performance of the kernel-based online learning algorithms as they will cast off support vectors (Yang et al. 2013).

To avoid information loss, we need to design a compensation scheme. Let the removed support vector be $\mathbf{z}_r = (\mathbf{x}_r, y_r)$, we find the most similar support vector $\mathbf{z}_c = (\mathbf{x}_c, y_c)$ with $y_c = y_r$ in $\mathcal{K}_t^{y_r}$ and update its corresponding

weight. By considering the updating rule in Eq. (10), we obtain the new decision function as follows:

$$\hat{f}_{t+1}(\mathbf{x}) = f_{t+1}(\mathbf{x}) - \alpha_{r,t} k(\mathbf{x}_r, \mathbf{x}).$$

We determine the updated weight $\Delta\alpha_{c,t}$ of the compensated support vector $\mathbf{z}_c$ by keeping track of all information with changing the value of the decision function. That it,

$$\begin{aligned} f_{t+1}(\mathbf{x}) &= \hat{f}_{t+1}(\mathbf{x}) + \Delta\alpha_{c,t} \cdot k(\mathbf{x}_c, \mathbf{x}) \\ &= f_{t+1}(\mathbf{x}) - \alpha_{r,t} k(\mathbf{x}_r, \mathbf{x}) + \Delta\alpha_{c,t} \cdot k(\mathbf{x}_c, \mathbf{x}). \quad (13) \end{aligned}$$

Hence, we set $\Delta\alpha_{c,t} = \alpha_{r,t} \frac{k(\mathbf{x}_r, \mathbf{x})}{k(\mathbf{x}_c, \mathbf{x})} \approx \alpha_{r,t}$ due to the similarity of the removed support vector, $\mathbf{x}_r$ and the compensated support vector, $\mathbf{x}_c$. We then obtain the updating rule of f_{t+1} with its compensation, f_{t+1}^{++}:

$$f_{t+1}^{++} = (1-\eta) f_t^{++} + \eta \partial_f \hat{\mathcal{L}}_t(f)|_{f=f_t^{++}} + \alpha_{r,t}(k(\mathbf{x}_c, \cdot) - k(\mathbf{x}_r, \cdot)), \quad (14)$$

where f_t^{++} is the previously compensated decision function. When either buffer is not full, $f_t^{++} = f_t$, updated by Eq. (7). Ideally, if $k(\mathbf{x}_c, \mathbf{x})$ equals $k(\mathbf{x}_r, \mathbf{x})$, f_t^{++} reserves all the support vectors and corresponds to the one learned with infinite budgets. Hence, we call the replacement with the compensation scheme as the extended updating policy. For the Reservoir Sampling policy, it is named **RS++**, while for the FIFO policy, it is named **FIFO++**. Significantly, the extended updating policies are not heuristic because a regret bound can be derived accordingly.

Theoretical Analysis

We define the regret bound as the difference between the objective value up to the T-th step and the smallest objective value from hindsight, i.e.,

$$R_T = \sum_{t=1}^{T} \hat{\mathcal{L}}_t(f_t) - \hat{\mathcal{L}}_t(f^*), R_T^{++} = \sum_{t=1}^{T} \hat{\mathcal{L}}_t(f_t^{++}) - \hat{\mathcal{L}}_t(f^*), \quad (15)$$

where f^* is the optimal decision function obtained from hindsight, and f_t and f_t^{++} correspond to the updating in Eq. (7) and Eq. (14), respectively.

We can derive the regret bounds in the following theorem:

Theorem 1. *Suppose for all $\mathbf{x} \in \mathbb{R}^d$, $k(\mathbf{x}, \mathbf{x}) \leq X^2$, where $X > 0$. Let ξ_1 be in $[0, X]$, such that $k(\mathbf{x}_t, \mathbf{x}_i) \geq \xi_1^2$, $\forall\, \mathbf{z}_i = (\mathbf{x}_i, y_i) \in N_t^{-y_t}(\mathbf{z}_t)$. Given $k > 0, C > 0, \eta > 0$ and a bounded convex loss function $\ell : \mathcal{H} \times \mathcal{Z} \times \mathcal{Z} \to [0, U]$ for f_t updated by Eq. (7), with $f_1 = 0$, we have*

$$R_T \leq \frac{\|f^*\|_{\mathcal{H}}^2}{2\eta} + \eta C k \sum_{t=1}^{T} \left((U-1) + (k+1) C (X^2 - \xi_1^2) \right). \quad (16)$$

Moreover, assume that $\forall i \in I_t^+ \cup I_t^-$, $|\alpha_{i,t}| \in [0, \gamma\eta]$ and $k(\mathbf{x}_r, \mathbf{x}_c) \geq \xi_2^2$ with $0 < \xi_2 \leq X$ for any replaced support vector $\mathbf{x}_r$ and compensated support vector $\mathbf{x}_c$ at any trial. With $f_1^{++} = 0$ and f_t^{++} updated by Eq. (14), we have

$$R_T^{++} \leq R_T + T\left(4\gamma C k \sqrt{(X^2 - \xi_2^2)(X^2 - \xi_1^2)} + 2\gamma^2 (X^2 - \xi_2^2) \right). \quad (17)$$

Details of the proof and more results are given in the Appendix. Several remarks include:

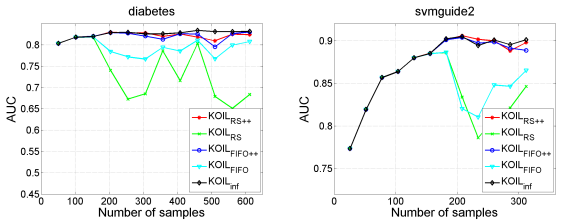
- The assumption on the bound of the loss function is valid. For the pair-wise hinge loss function in Eq. (3), we can derive the bound $U = 1 + 2Ck(X^2 - \xi_1^2)$. The assumption on the bound of the weight α also makes sense. Otherwise the sought optimal decision function is worse than the initial one, i.e., $f_1^{++} = 0$.
- By setting η to $O(1/\sqrt{T})$, we can derive the regret bounds, $R_T \sim O(\sqrt{T})$, which is equivalent to the $O(1/\sqrt{T})$ convergence rate for KOIL. The bounds we derived are the same as standard online learning algorithms. By exploiting the smoothness of loss functions, we can derive a fast convergence rate as in (Gao et al. 2013), i.e., $O(1/T)$ when $\hat{\mathcal{L}}(f^*) = 0$. It should be noted that our derived regret bounds are also different from the mistake bounds derived in (Cavallanti, Cesa-Bianchi, and Gentile 2007; Dekel, Shalev-Shwartz, and Singer 2008; Orabona, Keshet, and Caputo 2009), which aims at maximizing classification accuracy.
- The regret bound R_T^{++} is larger than R_T with an undesired term related to T. However, we argue that it is meaningful as γ can be restricted to be proportional to $O(1/\sqrt{T})$, which yields a regret bound in $O(\sqrt{T})$. For better theoretical result, we leave it as future work. Moreover, when $\xi_2^2 = X^2$, i.e., the compensated support vector is the same as the replaced support vector, we can obtain $R_T^{++} = R_T$. This result implies that the decision function learned by the replacement with compensation updating strategy can approach the decision function learned with infinite budgets. The experimental results also verify this observation.
- The derived regret bounds are proportional to k, not N (the budget). This is different from the regret bound derived in online AUC maximization (Zhao et al. 2011). Although the result implies that $k = 1$ can attain the smallest theoretical regret bound, we observe the best k is around 10% of the budget; see more details in the experimental section. We conclude that the bound derived here is based on an approximate surrogate of the AUC metric, i.e., the pair-wise hinge loss function. More precise theoretical results require more accurate measure of the AUC metric.

Experiments

In this section, we present extensive experimental results on real-world datasets to demonstrate the effectiveness of our proposed KOIL [1] algorithm with fixed budgets.

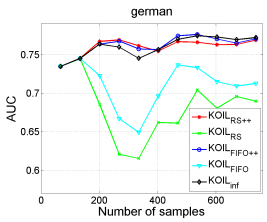
Compared Algorithms. Since we only focus on online imbalanced learning, for fair comparison, we do not compare with existing batch-trained imbalanced learning algorithms. Specifically, we compare our proposed KOIL with the state-of-the-art online learning algorithms, including online linear algorithms and kernel-based online learning algorithms with a finite or infinite buffer size:

[1] Demo codes in both C++ and Matlab can be downloaded in `https://www.dropbox.com/sh/nuepinmqzepx54r/AAAKuL4NSZe0IRpGuNIsuxQxa?dl=0`.

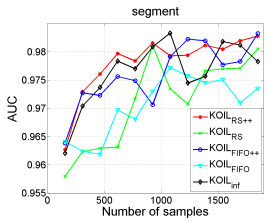


(a) diabetes

(b) svmguide2

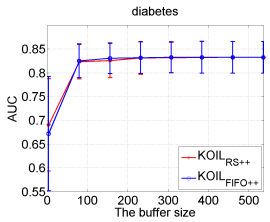


(c) german

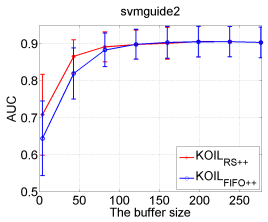


(d) segment

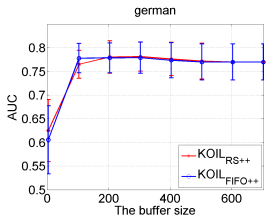
Figure 1: Average AUC performance on four benchmark datasets obtained via different updating policies of KOIL.



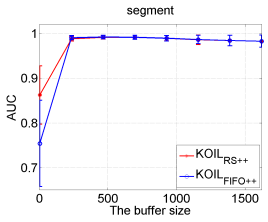
(a)



(b)



(c)



(d)

Figure 2: Average AUC of KOIL with different buffer sizes.

- "**Perceptron**": the classical perceptron algorithm (Rosenblatt 1958);
- "**OAM$_{\text{seq}}$**": an online linear AUC maximization algorithm (Zhao et al. 2011);
- "**OPAUC**": One-pass AUC maximization (Gao et al. 2013);
- "**NORMA**": online learning with kernels (Kivinen, Smola, and Williamson 2004);
- "**RBP**": Randomized budget perceptron (Cavallanti, Cesa-Bianchi, and Gentile 2007);
- "**Forgetron**": a kernel-based perceptron on a fixed budget (Dekel, Shalev-Shwartz, and Singer 2008);
- "**Projectron/Projectron++**": a bounded kernel-based perceptron (Orabona, Keshet, and Caputo 2009);
- "**KOIL$_{\text{RS++}}$/KOIL$_{\text{FIFO++}}$**": our proposed kernelized online imbalanced learning algorithm with fixed budgets updated by RS++ and FIFO++, respectively.

Experimental Setup. To make fair comparisons, all algorithms adopt the same setup. We set the learning rate to a small constant $\eta = 0.01$ and apply a 5-fold cross validation to find the penalty cost $C \in 2^{[-10:10]}$. For kernel-based methods, we use the Gaussian kernel and tune its parameter $\sigma \in 2^{[-10:10]}$ by a 5-fold cross validation. For NORMA, we apply a 5-fold cross validation to select λ and $\nu \in 2^{[-10:10]}$. For Projectron, we apply a similar 5-fold cross validation to select the parameter of projection difference $\eta \in 2^{[-10:10]}$.

Experiments on Benchmark Datasets. We conduct experiments on 14 benchmark datasets obtained from the UCI and the LIBSVM websites. The imbalanced ratio ranges from 1.144 to 328.546. For each dataset, we conduct 5-fold cross validation for all the algorithms, where four folds of the data are used for training while the rest for test. The 5-fold cross validation is independently repeated four times. We set the buffer size to 100 for each class for all related algorithms, including OAM$_{\text{seq}}$, RBP, and Forgetron. We then average the AUC performance of 20 runs and report the results in Table 1. Several observations can be drawn as follows:

- Our KOIL with RS++ and FIFO++ updating policies perform better than online linear AUC maximization algorithms in most datasets. By examining the results of OAM$_{\text{seq}}$ on the datasets of australian, heart, diabetes, german, and shuttle and those of OPAUC on australian and german, we speculate that for these datasets, a linear classifier is enough to achieve good performance, while a nonlinear classifier may be easily affected by outliers.
- Kernel-based online learning algorithms, especially Projectron/Projectron++, show better AUC performance than the linear online learning algorithms in most datasets. This again demonstrates the power of kernel methods in classifying real-world datasets.
- Our proposed KOIL significantly outperforms all competing kernel-based online learning algorithms in nearly all datasets. The results demonstrate the effectiveness of our KOIL in online nonlinear imbalanced learning.

Evaluation on Updating Policies. We compare the extended updating policies, RS++ and FIFO++, with the original updating policies, RS and FIFO. We show in Figure 1 for the average AUC performance of 20 runs on four typical datasets. The results of KOIL$_{\text{inf}}$, i.e., learning with infinite budgets, are provided for reference. We have the following observations:

- KOIL$_{\text{RS++}}$ and KOIL$_{\text{FIFO++}}$ attain nearly the same performance as KOIL$_{\text{inf}}$. The results confirm that the extended updating policies maintain all available revealed information during the training.
- Our KOIL with compensation updating policy significant outperforms the corresponding with original stream oblivious policy when either buffer is full. Without compensation, the performance fluctuates and is easily affected by noisy instances. On the other hand, with compensation, KOIL can attain stable performance.

Table 1: Average AUC performance (mean±std) on the benchmark datasets, •/◦ (-) indicates that both/one of $\text{KOIL}_{\text{RS++}}$ and $\text{KOIL}_{\text{FIFO++}}$ are/is significantly better (worse) than the corresponding method (pairwise t-tests at 95% significance level).

Data	$\text{KOIL}_{\text{RS++}}$	$\text{KOIL}_{\text{FIFO++}}$	Perceptron	OAM_{seq}	OPAUC	NORMA	RBP	Forgetron	Projectron	Projectron++
sonar	**.955**±.028	**.955**±.028	.803±.083•	.843±.056•	.844±.077•	.925±.044•	.913±.032•	.896±.054•	.896±.049•	.896±.049•
australian	.923±.023	.922±.026	.869±.035•	**.925**±.024	.923±.025	.919±.023	.911±.017•	.912±.026•	.923±.024	.923±.024
heart	.908±.040	.910±.040	.876±.066•	**.912**±.040	.901±.043◦	.890±.051•	.865±.043•	.900±.053	.902±.038	.905±.042
ionosphere	**.985**±.015	**.985**±.015	.851±.056•	.905±.041•	.888±.046•	.961±.016•	.960±.030•	.945±.031•	.964±.025•	.963±.027•
diabetes	.826±.036	.830±.030	.726±.059•	.827±.033	.805±.035•	.792±.032•	.828±.034	.820±.027◦	.832±.033	**.833**±.033
glass	**.887**±.053	.884±.054	.810±.065•	.827±.064•	.800±.074•	.811±.077•	.811±.071•	.813±.075•	.811±.070•	.781±.076•
german	.769±.032	.778±.031	.748±.033•	.777±.027	**.787**±.026 -	.766±.032◦	.699±.038•	.712±.054•	.769±.028◦	.770±.024
svmguide2	**.897**±.040	.885±.043	.860±.037•	.886±.045◦	.859±.050•	.865±.046•	.890±.038	.864±.045•	.886±.044◦	.886±.045◦
segment	.983±.008	**.985**±.012	.875±.020•	.919±.020•	.882±.019•	.910±.042•	.969±.017•	.943±.038•	.979±.013•	.978±.016•
satimage	**.924**±.012	.923±.015	.700±.015•	.755±.018•	.724±.016•	.914±.025•	.899±.018•	.892±.032•	.910±.015•	.904±.011•
vowel	**1.000**±.000	**1.000**±.001	.848±.070•	.905±.024•	.885±.034•	.996±.005•	.968±.017•	.987±.027•	.982±.013•	.994±.019•
letter	.933±.021	**.942**±.017	.767±.029•	.827±.021•	.823±.018•	.910±.027•	.928±.011◦	.815±.102•	.926±.016•	.926±.015•
poker	.681±.031	**.693**±.032	.514±.030•	.503±.024•	.509±.031•	.577±.040•	.501±.031•	.572±.029•	.675±.027•	.675±.027•
shuttle	.950±.040	.956±.021	.520±.134•	**.999**±.000 -	.754±.043•	.725±.053•	.844±.041•	.839±.060•	.873±.063•	.795±.063•
win/tie/loss			14/0/0	9/4/1	12/1/1	13/1/0	12/2/0	13/1/0	11/3/0	10/4/0

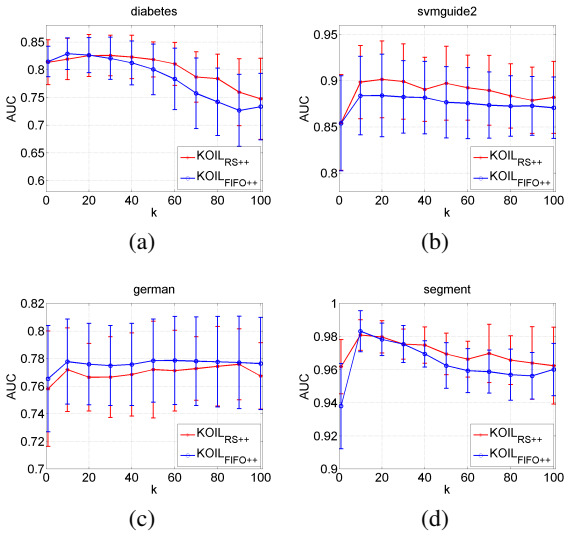


Figure 3: Average AUC of KOIL with different k. Here k = [1, 10:10:100] and the budget is 100 for each buffer.

Sensitivity Analysis of KOIL. We first test the performance of KOIL with different buffer sizes. From Figure 2, we observe that the performance increases gradually with the increase of the buffer size and it is saturated when the size is relatively large. This is similar to the observations in (Yang et al. 2013; Zhao et al. 2011). Next, we test the performance of KOIL with different k, which determines the number of localized support vectors. From Figure 3, we have the following observations:

- When k is extremely small, say $k = 1$, KOIL only considers the pairwise loss yielded by the nearest opposite support vector of the new instance and can not fully utilize the localized information. In this case, the updating weight is a constant, $|\eta C y_t|$, which is the same value of the initial weight for the misclassified instance in NORMA.
- KOIL usually attains the best performance when k equals 10% of the buffer size. The performance decreases when k increases. The results consistently show that by only utilizing the local information of new instances indeed prevents the effect of outliers.
- For some datasets, e.g., svmguide2 and german, the performance is not so sensitive to k. The reason may be that the learned support vectors in these datasets are well-separated when the buffers are full. Hence, new instances play little influence on seeking the decision function.

Conclusion

We proposed a kernel-based online learning algorithm to tackle the imbalanced binary classification problem. We maintain two buffers with fixed budgets to control the number of support vectors, which keep track of the global information of the decision boundary. We update the weight of a new support vector by confining its influence only on its k-nearest opposite support vectors. More importantly, we design a sophisticated compensation scheme to avoid information loss by transferring the weight of the removed support vector to its most similar one, when either buffer is full. We show that this compensation can make our learning decision function approach the one learned with infinite budgets. We provide theoretical analysis on the regret bounds and conduct extensive experiments to demonstrate the effectiveness of our proposed KOIL.

Several challenging and promising directions can be considered in the future: First, the current regret bounds only provide standard results. How to seek more advance mathematical tools to derive better regret bounds for imbalanced learning deserves investigation. Second, our current KOIL mainly presents the results of the pair-wise hinge loss function. Exploring more accurate surrogate functions for the AUC metric contain both theoretical and practical merits. Third, how to select a good kernel or similarly, how to borrow the idea of multiple kernel learning for online imbalanced learning is a potential way to improve the model performance.

Acknowledgments

The work described in this paper was fully supported by the National Grand Fundamental Research 973 Program of China (No. 2014CB340401 and No. 2014CB340405), the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK 413212 and CUHK 415113), Microsoft Research Asia Regional Seed Fund in Big Data Research (Grant No. FY13-RES-SPONSOR-036), and a Microsoft Research Asia gift grant.

References

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.

Brefeld, U., and Scheffer, T. 2005. AUC maximizing support vector learning. In *Proceedings of the ICML 2005 workshop on ROC Analysis in Machine Learning*.

Cavallanti, G.; Cesa-Bianchi, N.; and Gentile, C. 2007. Tracking the best hyperplane with a simple budget perceptron. *Machine Learning* 69(2-3):143–167.

Cesa-Bianchi, N., and Lugosi, G. 2006. *Prediction, Learning, and Games*. New York, NY, USA: Cambridge University Press.

Cortes, C., and Mohri, M. 2003. AUC optimization vs. error rate minimization. In *NIPS*. MIT Press.

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

Dekel, O.; Shalev-Shwartz, S.; and Singer, Y. 2008. The Forgetron: A kernel-based perceptron on a budget. *SIAM Journal on Computing* 37(5):1342–1372.

Gao, W.; Jin, R.; Zhu, S.; and Zhou, Z.-H. 2013. One-pass AUC optimization. In *ICML*, 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.

Herschtal, A., and Raskutti, B. 2004. Optimising area under the ROC curve using gradient descent. In *ICML*.

Hoi, S. C. H.; Wang, J.; Zhao, P.; Zhuang, J.; and Liu, Z. 2013. Large scale online kernel classification. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*.

Joachims, T. 2005. A support vector method for multivariate performance measures. In *ICML*, 377–384.

Karampatziakis, N., and Langford, J. 2011. Online importance weight aware updates. In *UAI 2011, Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence, Barcelona, Spain, July 14-17, 2011*, 392–399.

Kivinen, J.; Smola, A. J.; and Williamson, R. C. 2004. Online learning with kernels. *IEEE Transactions on Signal Processing* 52(8):2165–2176.

Liang, G., and Cohn, A. G. 2013. An effective approach for imbalanced classification: Unevenly balanced bagging. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14-18, 2013, Bellevue, Washington, USA*.

Orabona, F.; Keshet, J.; and Caputo, B. 2009. Bounded kernel-based online learning. *Journal of Machine Learning Research* 10:2643–2666.

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

Ross, S.; Mineiro, P.; and Langford, J. 2013. Normalized online learning. *CoRR* abs/1305.6646.

Schölkopf, B., and Smola, A. 2002. *Learning with Kernels*. Cambridge, MA: MIT Press.

Vitter, J. S. 1985. Random sampling with a reservoir. *ACM Trans. Math. Softw.* 11(1):37–57.

Wu, X.; Kumar, V.; Quinlan, J. R.; Ghosh, J.; Yang, Q.; Motoda, H.; McLachlan, G. J.; Ng, A. F. M.; Liu, B.; Yu, P. S.; Zhou, Z.; Steinbach, M.; Hand, D. J.; and Steinberg, D. 2008. Top 10 algorithms in data mining. *Knowl. Inf. Syst.* 14(1):1–37.

Yang, H., and King, I. 2009. Ensemble learning for imbalanced e-commerce transaction anomaly classification. In *ICONIP*, 866–874.

Yang, H.; Xu, Z.; King, I.; and Lyu, M. R. 2010. Online learning for group lasso. In *ICML*, 1191–1198.

Yang, H.; Xu, Z.; Ye, J.; King, I.; and Lyu, M. R. 2011. Efficient sparse generalized multiple kernel learning. *IEEE Transactions on Neural Networks* 22(3):433–446.

Yang, T.; Mahdavi, M.; Jin, R.; Yi, J.; and Hoi, S. C. H. 2012. Online kernel selection: Algorithms and evaluations. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada.*

Yang, H.; Hu, J.; Lyu, M. R.; and King, I. 2013. Online imbalanced learning with kernels. In *NIPS Workshop on Big Learning*.

Yang, H.; King, I.; and Lyu, M. R. 2011. *Sparse Learning Under Regularization Framework*. LAP Lambert Academic Publishing.

Yang, H.; Lyu, M. R.; and King, I. 2013. Efficient online learning for multi-task feature selection. *ACM Transactions on Knowledge Discovery from Data* 7(2):1–27.

Yang, H.; Lyu, M. R.; and King, I. 2014. Big data oriented online learning algorithms. *Communications of the CCF* 10(11):36–40.

Zhang, L.; Jin, R.; Chen, C.; Bu, J.; and He, X. 2012. Efficient online learning for large-scale sparse kernel logistic regression. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada.*

Zhao, P.; Hoi, S. C. H.; Jin, R.; and Yang, T. 2011. Online AUC maximization. In *ICML*, 233–240.

Zhao, P.; Hoi, S. C.; and Jin, R. 2011. Double updating online learning. *Journal of Machine Learning Research* 12:1587–1615.