Adaptive Online Kernel Sampling for Vertex Classification

Peng Yang and Ping Li
Cognitive Computing Lab
Baidu Research
10900 NE 8th St. Bellevue, WA 98004, USA
{pengyang01, liping11}@baidu.com

Abstract

This paper studies online kernel learning (OKL) for graph classification problem, since the large approximation space provided by reproducing kernel Hilbert spaces often contains an accurate function. Nonetheless, optimizing over this space is computationally expensive. To address this issue, approximate OKL is introduced to reduce the complexity either by limiting the support vector (SV) used by the predictor, or by avoiding the kernelization process altogether using embedding. Nonetheless, as long as the size of the approximation space or the number of SV does not grow over time, an adversarial environment can always exploit the approximation process. In this paper, we introduce an online kernel sampling (OKS) technique, a new second-order OKL method that slightly improve the bound from $\mathcal{O}(d \log T)$ down to $\mathcal{O}(r \log T)$ where r is the rank of the learned data and is usually much smaller than d. To reduce the computational complexity of second-order methods, we introduce a randomized sampling algorithm for sketching kernel matrix $\mathbf{K}_t$ and show that our method is effective to reduce the time and space complexity significantly while maintaining comparable performance. Empirical experimental results demonstrate that the proposed model is highly effective on real-world graph datasets.

1 Introduction

Graph-structured data is becoming more widespread: examples are social networks, knowledge graph, protein or gene regulation networks, or the growing body of research in program flow analysis. For instance, a typical problem of recommendation in social networks: Given a number of users in tweets, the model aims to predict whether users would purchase a product or not. In this problem, one needs data analysis and machine learning methods that can handle large-scale datasets (Bottou et al., 2007; Bottou and Bousquet, 2007; Bottou et al., 2018; Yang and Li, 2020a). This problem has been studied in the setting of online learning (OL) with Laplacian regularization on graphs (Ando and Zhang, 2006). In such setting, the graph structure is provided and a model is learned incrementally in the input from a sequence of T streaming nodes. Pioneer work usually learned a linear predictor in the node-embedded space or graph-embedded space (Herbster and Pontil, 2006; Shrivastava and Li, 2014; Rahmani and Li, 2019). As we can explicitly store and update the d-dimensional linear predictor, total runtime of this algorithm is $\mathcal{O}(Td)$, allowing it to scale to large problems. Unfortunately, it is sometimes the case that no good predictor can be learned starting from linear combination of the input features.

For this reason, online kernel learning (Kivinen et al., 2001) (OKL) first maps the points into a high-dimensional *reproducing kernel Hilbert space* (RKHS) using a nonlinear feature map ϕ, and then runs gradient descent (GD) on the projected points. With the kernel approach, each gradient step does not update a fixed set of weights, but instead introduces feature-mapped points in the predictor as a *support vector* (SV). Zinkevich et al. (Zinkevich, 2003) showed that first-order GD is flexible and data adaptive, but the number of parameters, and therefore the per-trial space and time cost, now scales with $\mathcal{O}(t)$, the number of SVs included after t steps of GD. When the losses posse a certain *directional curvature* properties, kernel-recursive least squares (Zhdanov and Kalnishkan, 2010) or kernelized online Newton step (Calandriello et al., 2017), which exploits second-order (second derivative) information on the losses, can achieve a *logarithmic* regret $\mathcal{O}(d \log T)$. The drawback of second-order methods is that they have to store and invert the $t \times t$ covariance matrix between all SVs included in the predictor. This requires $\mathcal{O}(t^2)$ space and time per-step, resulting in an even more infeasible T^3 runtime, which prevents the standard OKL methods from scaling to

Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108.

large problems.

Contribution We propose an online kernel algorithm, derived from the Laplacian regularized least square. The regret of the objective function is minimized by an adaptive convex-concave optimization framework. The derived algorithm achieves a regret of $\mathcal{O}(r \log T)$, and runs in $\mathcal{O}(t^2)$ time and space. To reduce runtime complexity, we consider a confidence-based sampling technique to approximate the Hessian matrix and predictor with adaptive sketching. By adaptively increasing the size of its sketch, this method provides a favorable accuracy-complexity trade-off, where we can maintain a comparable mistake bound while obtaining a significant improvement in runtime, thereby reducing labeling cost and achieving smaller space and time per iteration. To accelerate the learning process, we further propose an adaptive technique by performing explorative updates over the points of high predicted variance. We empirically evaluate the algorithm on real-world datasets and experimental results demonstrate that this model is highly effective.

2 Related Work

OKL has been proposed, such as functional GD (Kivinen et al., 2001; Zinkevich, 2003) which achieves a $\mathcal{O}(\sqrt{T})$ regret with a $\mathcal{O}(t)$ space and time cost per iteration. For second-order methods, several studies (Cesa-Bianchi et al., 2002; Orabona and Crammer, 2010; Yang and Li, 2020b) for generic curved losses and the others (Zhdanov and Kalnishkan, 2010; Gammerman et al., 2004) for the specific case of ℓ_2 losses provide bounds that scale with the log-determinant of the kernel-matrix. Sketched methods have been proposed to make OKL methods scale to large datasets, which usually take one of two approaches, either performing *Sketched Gradient* updates in the true RKHS (Cesa-Bianchi and Gentile, 2006; Kushnir, 2014), e.g., projectron (Orabona et al., 2008), forgetron (Dekel et al., 2005)), which prevent support vectors (SVs) from entering the predictor, or exact gradient updates in an *sketched RKHS* (Nystrom (Williams and Seeger, 2000; Yang et al., 2012; Lu et al., 2016; Calandriello et al., 2017; Li and Zhang, 2017), random Fourier feature expansion (Rahimi and Recht, 2007; Li, 2017)), where the points are embedded in a finite-dimensional space and the curse of kernelization does not apply. This paper proposes a second-order OKL and exploits the mean and variance of prediction for kernel sketching.

Notation. We use upper-case bold letters $\mathbf{A}$ for matrices, lower-case bold letters $\mathbf{a}$ for vectors, lower-case letters a for scalars. We denote A_{ij} as the (i,j)-th element of $\mathbf{A}$ and a_i as the i-th element of $\mathbf{a}$. With an appropriate size, we denote by $\mathbf{I}$ and $\mathbf{0}$ as the identity matrix and a vector of all zeros, respectively, and by $\mathrm{diag}(\mathbf{a}) \in \mathbb{R}^{d\times d}$ as the diagonal matrix with the vector $\mathbf{a}$ on the diagonal. The transpose of a vector $\mathbf{x}$ is denoted as $\mathbf{x}^\top$, the inverse of a matrix $\mathbf{A}$ as $\mathbf{A}^{-1}$, and the pseudo inverse of $\mathbf{A}$ as $\mathbf{A}^\dagger$. We use $\mathbf{A} \succeq \mathbf{B}$ to indicate that $\mathbf{A} - \mathbf{B}$ is a positive semi-definite (PSD) matrix. With $\|\cdot\|_{op}$ we indicate the *operator* norm.

3 Problem Setting

Given a reproducing kernel Hilbert space $\mathcal{H}$ and an arbitrary input space $\mathcal{X}$ (Schölkopf and Smola, 2002), we can find an associated feature map $\varphi : \mathcal{X} \to \mathcal{H}$ and a positive definite kernel function $\mathcal{K} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, such that $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j)\rangle_{\mathcal{H}}$ is an inner product of feature mapping. For any point $\mathbf{x}_i \in \mathcal{X}$, we shorten $\varphi(\mathbf{x}_i) = \phi_i$ and define feature matrix as $\Phi_n = [\phi_1, \ldots, \phi_n] \in \mathbb{R}^{D\times n}$, where D is very large or potential infinite. For any function $f \in \mathcal{H}$, it is explicitly represented as $\mathbf{u}$ such that $f(\mathbf{x}_i) = \mathbf{u}^\top \phi_i$. With this notation, the empirical kernel matrix is defined as $\mathbf{K}_n = \Phi_n^\top \Phi_n$, the vector with all similarities between old points and a new one as $\mathbf{k}_{[n-1],n} = \Phi_{n-1}^\top \phi_n$, with its element as $k_{i,n} = \phi_i^\top \phi_n$. Throughout the rest of the paper, we assume that $\mathcal{K}$ is normalized and $k_{i,i} = 1$.

In the setting of classification, the real-valued function satisfies: the label prediction should be close to the given labels for that instances, while the model parameter should be generalized to solve ill-posed problem without overfitting. Given a regularized parameter $b > 0$, this work solve a regularized least square function,

$$\min_{\mathbf{u}} \sum_{i=1}^{n} \ell(\mathbf{u}; (\phi_i, y_i)) + b\|\mathbf{u}\|^2. \tag{1}$$

We solve this problem in a setting of online kernel learning.

4 Online Kernel Optimization

The objective of online kernel learning aims to achieve a low regret compared to the best predictor in $\mathcal{H}$. Given an arbitrary node-label sequence $\{(\mathbf{x}_t, y_t)\}_{t=1}^T$ $(T \le n)$, an algorithm receives an input $\mathbf{x}_t \in \mathcal{X}$ at round t, and predicts $\hat{y}_t = \mathrm{sgn}[\mathbf{u}_t^\top \phi_t]$, where the model $\mathbf{u}_t$ is learned from the previous t rounds. Then true label y_t is revealed, the algorithm uses it to update model and then proceeds to next round. For any weight $\mathbf{u} \in \mathcal{H}$, we denote by $\ell_t(\mathbf{u})$ as the instantaneous loss of $\mathbf{x}_t$, and by $L_T(\mathbf{u}) = \sum_{t=1}^T \ell_t(\mathbf{u})$ as the cumulative loss over T rounds. Specifically, we propose an adaptive cumulative loss function, $L_T(\mathbf{u}) = \sum_{t=1}^T a_t(y_t - \mathbf{u}^\top \phi_t)^2$, where $\{a_t\}_{t=1}^T \ge 0$ are the input-dependent weights. Formally, we define the objective function of an algorithm as

$$F(\mathbf{u}) = \min_{\mathbf{u}} \left(b\|\mathbf{u}\|^2 + \sum_{t=1}^{T} a_t(y_t - \mathbf{u}^\top \phi_t)^2 \right),$$

which aim to let the weighted cumulative loss and its regularization term.

4.1 Kernel Learning

Calculating the solution requires performing operations on $\Phi_t \in \mathbb{R}^{D\times t}$ where D is very large and potentially infinite

and vector ϕ_t cannot be explicitly represented. To derive a feasible solution, we show that the prediction f_t can be conveniently computed using only inner products.

To solve these issues above, we derive a closed-form solution which can be computed in practice. Using a rescaled variant of feature vectors ϕ_t, i.e., $\overline{\phi}_t = \sqrt{a_t}\phi_t$ and $\overline{\Phi}_t = [\overline{\phi}_1, \ldots, \overline{\phi}_t]$, we can write empirical kernel matrix $\overline{\mathbf{K}}_t = \overline{\Phi}_t^\top \overline{\Phi}_t$ using a rescaled kernel $\overline{\mathcal{K}}(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{a_i}\sqrt{a_j}\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$ instead of original $\mathcal{K}$, while the rescaled label $\overline{\mathbf{y}}_t = [\sqrt{a_1}y_1, \ldots, \sqrt{a_t}y_t]^\top$ is under the same setting.

We show that, by appropriately setting the adaptive weight a_T, the optimization problem could be convex in $\widehat{y}_T$. The optimal solution of the adaptive optimization problem is summarized as follows.

Theorem 1. *For any $t > 1$, $\overline{\Phi}_t = [\overline{\phi}_1, \ldots, \overline{\phi}_t] \in \mathbb{R}^{D\times t}$, $\overline{\mathbf{K}}_t = \overline{\Phi}_t^\top \overline{\Phi}_t \in \mathbb{R}^{t\times t}$, $\overline{\mathbf{y}}_t = [\overline{y}_1, \ldots, \overline{y}_t]^\top \in \mathbb{R}^t$, and $b > 1$. then the $F(\mathbf{u})$ could be minimized with an optimal solution,*

$$\widehat{y}_t = sgn\left(\mathbf{u}_t^\top \phi_t\right). \tag{2}$$

where the model parameter can be computed in practice

$$\mathbf{u}_t = \overline{\Phi}_{t-1}\left(\overline{\mathbf{K}}_{t-1} + b\mathbf{I}\right)^{-1}\overline{\mathbf{y}}_{t-1},$$

In this way, the model can be updated iteratively with computational complexity $\mathcal{O}(t^2)$.

Remark: Theorem 1 shows that the prediction $\widehat{y}_{t+1}$ can be computed by the kernel matrix $\overline{\mathbf{K}}_t = \overline{\Phi}_t^\top \overline{\Phi}_t \in \mathbb{R}^{t\times t}$ and the similarity vector $\overline{\mathbf{k}}_{[t],t+1} = \overline{\Phi}_t^\top \overline{\phi}_{t+1} \in \mathbb{R}^t$. When its true label y_{t+1} is revealed, the model can be updated efficiently in terms of $(\mathbf{K}_t + b\mathbf{I})^{-1}$ in a recursive way with the complexity $\mathcal{O}(t^2)$ in space and time (Laskov et al., 2006).

Proof.

$$\begin{aligned}
&F(\mathbf{u}) = b\|\mathbf{u}\|^2 + \sum_{t=1}^{T} a_t(y_t - \mathbf{u}^\top\phi_t)^2 \\
=&\sum_{t=1}^{T} a_t\left[y_t^2 - 2\left(\mathbf{u}^\top\phi_t y_t\right) + \mathbf{u}^\top\phi_t\phi_t^\top\mathbf{u}\right] + b\mathbf{u}^2) \\
=&\mathbf{u}^\top\left(b\mathbf{I}_D + \sum_{t=1}^{T}\overline{\phi}_t\overline{\phi}_t^\top\right)\mathbf{u} + \sum_{t=1}^{T}\overline{y}_t^2 - 2\mathbf{u}^\top\left(\sum_{t=1}^{T}\overline{\phi}_t\overline{y}_t\right) \\
=&\sum_{t=1}^{T}\overline{y}_t^2 - 2\mathbf{u}^\top(\overline{\Phi}_T\overline{\mathbf{y}}_T) + \mathbf{u}^\top(b\mathbf{I}_D + \overline{\Phi}_T\overline{\Phi}_T^\top)\mathbf{u},
\end{aligned}$$

it follows that $\nabla F(\mathbf{u}) = 2(b\mathbf{I}_D + \overline{\Phi}_T\overline{\Phi}_T^\top)\mathbf{u} - 2\overline{\Phi}_T\overline{\mathbf{y}}_T$, $\nabla^2 F(\mathbf{u}) = 2(b\mathbf{I}_D + \overline{\Phi}_T\overline{\Phi}_T^\top)$. Thus $F(\mathbf{u})$ is convex and it is minimal if $\nabla F(\mathbf{u}) = (b\mathbf{I}_D + \overline{\Phi}_T\overline{\Phi}_T^\top)\mathbf{u} - \overline{\Phi}_T\overline{\mathbf{y}}_T = 0$ with

$$\mathbf{u} = (b\mathbf{I}_D + \overline{\Phi}_T\overline{\Phi}_T^\top)^{-1}\overline{\Phi}_T\overline{\mathbf{y}}_T.$$

Algorithm 1 OKL: Online Kernel Learning

Input: $\{\mathbf{x}_t, y_t\}_{t=1}^T$, parameters b and h.
Output: $\mathbf{u}_T$
Initialize: $\mathbf{M}_0 = b\mathbf{I}$ with $b > 1$, $\Phi_0 = \emptyset$
for $t = 1, \ldots, T$ **do**
  Receive $\mathbf{x}_t \in R^d$
  Predict $f_t = \overline{\mathbf{y}}_{t-1}^\top \overline{\mathbf{M}}_{t-1}^{-1}\overline{\mathbf{k}}_{t-1}(\phi_t)$
  Query the true label y_t and $\hat{y}_t = \text{sgn}(f_t)$
  if $\hat{y}_t \neq y_t$ **then**
    Update $\overline{\Phi}_t = [\overline{\Phi}_{t-1}, \overline{\phi}_t]$ and $\overline{\mathbf{y}}_t = [\overline{\mathbf{y}}_{t-1}, \overline{y}_t]$
    Update $\overline{\mathbf{M}}_t = b\mathbf{I} + \overline{\Phi}_t^\top\overline{\Phi}_t$
  else
    $\overline{\mathbf{M}}_t = \overline{\mathbf{M}}_{t-1}, \overline{\Phi}_t = \overline{\Phi}_{t-1}, \overline{\mathbf{y}}_t = \overline{\mathbf{y}}_{t-1}$
  end if
end for

For any $\overline{\Phi}_T = [\overline{\phi}_1, \ldots, \overline{\phi}_T] \in \mathbb{R}^{D\times T}$ matrix and $b > 1$,

$$\begin{aligned}
\overline{\Phi}_T\overline{\Phi}_T^\top(\overline{\Phi}_T\overline{\Phi}_T^\top + b\mathbf{I}_D)^{-1} =&\overline{\Phi}_T(\overline{\Phi}_T^\top\overline{\Phi}_T + b\mathbf{I}_T)^{-1}\overline{\Phi}_T^\top \\
=&\overline{\Phi}_T(\overline{\mathbf{K}}_T + b\mathbf{I}_T)^{-1}\overline{\Phi}_T^\top,
\end{aligned}$$

and we compute

$$\begin{aligned}
&(\overline{\Phi}_T\overline{\Phi}_T^\top + b\mathbf{I}_D)^{-1} = \frac{1}{b}b\mathbf{I}(\overline{\Phi}_T\overline{\Phi}_T^\top + b\mathbf{I}_D)^{-1} \\
=&\frac{1}{b}\left(\overline{\Phi}_T\overline{\Phi}_T^\top - \overline{\Phi}_T\overline{\Phi}_T^\top + b\mathbf{I}_D\right)\left(\overline{\Phi}_T\overline{\Phi}_T^\top + b\mathbf{I}_D\right)^{-1} \\
=&\frac{1}{b}\left(\mathbf{I}_D - \overline{\Phi}_T\overline{\Phi}_T^\top\left(\overline{\Phi}_T\overline{\Phi}_T^\top + b\mathbf{I}_D\right)^{-1}\right) \\
=&\frac{1}{b}\left(\mathbf{I}_D - \overline{\Phi}_T\left(\overline{\mathbf{K}}_T + b\mathbf{I}_T\right)^{-1}\overline{\Phi}_T^\top\right).
\end{aligned}$$

So that we can obtain

$$\begin{aligned}
\mathbf{u} &= (\overline{\Phi}_T\overline{\Phi}_T^\top + b\mathbf{I})^{-1}\overline{\Phi}_T\overline{\mathbf{y}}_T \\
&= \frac{1}{b}\left(\mathbf{I}_D - \overline{\Phi}_T\left(\overline{\mathbf{K}}_T + b\mathbf{I}_T\right)^{-1}\overline{\Phi}_T^\top\right)\overline{\Phi}_T\overline{\mathbf{y}}_T \\
&= \frac{1}{b}\overline{\Phi}_T\left(\overline{\mathbf{y}}_T - \left(\overline{\mathbf{K}}_T + b\mathbf{I}_T\right)^{-1}\overline{\mathbf{K}}_T\overline{\mathbf{y}}_T\right) \\
&= \frac{1}{b}\overline{\Phi}_T\left(\mathbf{I}_T - \left(\overline{\mathbf{K}}_T + b\mathbf{I}_T\right)^{-1}\overline{\mathbf{K}}_T\right)\overline{\mathbf{y}}_T \\
&= \frac{1}{b}\overline{\Phi}_T\left(\left(\overline{\mathbf{K}}_T + b\mathbf{I}_T\right)^{-1}\left(\overline{\mathbf{K}}_T + b\mathbf{I}_T - \overline{\mathbf{K}}_T\right)\right)\overline{\mathbf{y}}_T \\
&= \overline{\Phi}_T\left(\overline{\mathbf{K}}_T + b\mathbf{I}_T\right)^{-1}\overline{\mathbf{y}}_T.
\end{aligned}$$

Substituting the solution $\mathbf{u} = \overline{\Phi}_T\left(\overline{\mathbf{K}}_T + b\mathbf{I}_T\right)$ back to $F(\mathbf{u})$, we have

$$F(\mathbf{u}) = \sum_{t=1}^{T}\overline{y}_t^2 - \mathbf{y}_T^\top(\overline{\mathbf{K}}_T + b\mathbf{I}_T)^{-1}\overline{\mathbf{K}}_T\mathbf{y}_T.$$

□

In Theorem 1, we obtain an optimal solution $\mathbf{u}_T$ for regularized online learning. Nonetheless, it is inefficient for online

algorithm to update in each iteration. To address this issue, we make use of a conservative strategy to perform an update when an error $(y_t \neq \widehat{y}_t)$ occurs. We call the algorithm OKL, online kernel learning in Algorithm 1. Note that our update format is different from OLLGC (Gu et al., 2013), SOP (Cesa-Bianchi et al., 2002) and others (Orabona and Crammer, 2010), since we derive a kernelized model that is able to to search for the non-linear hypothesis.

When an error occurs at round t, i.e., $f_t y_t \leq 0$, the model has to update the inverse of kernel matrix $(\mathbf{K}_t + b\mathbf{I})^{-1}$ with a computational complexity of $\mathcal{O}(t^3)$. To address this issue, we provide an efficient solution to update $(\mathbf{K}_t + b\mathbf{I})^{-1}$. We begin with additional annotations:

$$\overline{\mathbf{d}}_{t-1} = (\overline{\mathbf{K}}_{t-1} + b\mathbf{I})^{-1}\overline{\mathbf{k}}_{[t-1],t},$$
$$\gamma_t = (\overline{k}_{t,t} + b - \overline{\mathbf{k}}_{[t-1],t}^\top \overline{\mathbf{d}}_{t-1})^{-1}.$$

According to Woodbury formula, we obtain

$$(\overline{\mathbf{K}}_t + b\mathbf{I})^{-1} = \begin{bmatrix} \overline{\mathbf{K}}_{t-1} + b\mathbf{I} & \overline{\mathbf{k}}_{[t-1],t} \\ \overline{\mathbf{k}}_{[t-1],t}^\top & \overline{k}_{t,t} + b \end{bmatrix}^{-1}$$
$$= \gamma_t \cdot \begin{bmatrix} \gamma_t^{-1}(\overline{\mathbf{K}}_{t-1} + b\mathbf{I})^{-1} + \overline{\mathbf{d}}_{t-1}\overline{\mathbf{d}}_{t-1}^\top & -\overline{\mathbf{d}}_{t-1} \\ -\overline{\mathbf{d}}_{t-1}^\top & 1 \end{bmatrix}$$
$$= \begin{bmatrix} (\overline{\mathbf{K}}_{t-1} + b\mathbf{I})^{-1} & \mathbf{0}_{t-1} \\ \mathbf{0}_{t-1}^\top & 0 \end{bmatrix} + \gamma_t \cdot \begin{bmatrix} \overline{\mathbf{d}}_{t-1} \\ -1 \end{bmatrix} \begin{bmatrix} \overline{\mathbf{d}}_{t-1}^\top & -1 \end{bmatrix}$$

In this way, the kernel model can be updated iteratively with computational complexity $\mathcal{O}(t^2)$.

5 Online Kernel Sampling

For the proposed algorithm, storing and updating the inverse of the $\overline{\mathbf{K}}_t$ matrix requires a complexity $\mathcal{O}(t^2)$ regrading the space and time, which becomes quickly unfeasible as t grows. Moreover, this algorithm assumes that true labels are provided to each instance, which results in expensive labeling cost in practice.

To improve computational efficiency, we adapt the selective sampling algorithm of (Cesa-Bianchi et al., 2006; Rudi et al., 2018) into the kernel setting and propose randomized sampling for kernel selection. Unlike online algorithm that samples all labels, selective sampling has to decide whether to query label or not for each point $\mathbf{x}_t$. If a label y_t is queried of, the algorithm can perform a update with y_t; otherwise, no action is conducted and the learner proceeds next one. Query and update decisions at round t are denoted as binary variable Q_t and Z_t, respectively. When $Q_t = 1$ iff true label y_t is queried of; $Q_t = 0$, no query conducted. The update decision Z_t is under similar setting. Generally, online sampling is a semi-supervised online learning algorithm. Thus, its best predictor could be derived in a form of online learning with query/update choice on each round.

5.1 Adaptive Kernel Sampling

The goal of online kernel sampling is to achieve few mistakes $\sum_t \mathbb{I}[y_t \neq \widehat{y}_t]$ with small sampled number $\sum_t Z_t Q_t$. To achieve this goal, we propose a randomized sampling technique tuned by a confidence score Θ_t: a coin with bias $\frac{h}{h+\max(0,\Theta_t)}$ is flipped; if the coin turns up heads, then actual label y_t is sampled, otherwise $Q_t = 0$ and no action performed.

Definition 1. *At round t, an algorithm predicts with $f_t = \mathbf{u}_t \cdot \overline{\phi}_t$ where $\mathbf{u}_t = \overline{\mathbf{y}}_{t-1}^\top (\overline{\mathbf{K}}_{t-1} + b\mathbf{I})^{-1}\overline{\Phi}_{t-1}$, and computes τ_t, the projection of current instance to the inverse of the learned kernel matrix, as below,*

$$\tau_t = \overline{\phi}_t^\top (\overline{\Phi}_{t-1}\overline{\Phi}_{t-1}^\top + b\mathbf{I}_D)^{-1}\overline{\phi}_t$$
$$= \frac{1}{b}\left(a_t - \overline{\mathbf{k}}_{[t-1],t}^\top (\overline{\mathbf{K}}_{t-1} + b\mathbf{I})^{-1}\overline{\mathbf{k}}_{[t-1],t}\right). \quad (3)$$

Then, the true label is queried by a Bernoulli trial with a probability $\frac{h}{h+\max(0,\Theta_t)}$ $(h > 0)$, where Θ_t is defined as $\Theta_t = |f_t| - \frac{1}{2}a_t\tau_t$, which can be regarded as the lower confidence bound.

Our basic idea is to maintain a tradeoff between exploitation and exploration (Crammer and Gentile, 2011). Intuitively, an algorithm should exploit current hypothesis to decide a query, i.e., sample an input when it is closed to current hypothesis (Gu et al., 2014) ($|f_t| = |\phi_t^\top \mathbf{w}_t| \to 0$). However, the hypothesis is usually biased towards the observed points, especially in a dynamic environment of online learning when model has little knowledge of the latest data. To solve this issue, the algorithm tracks the spectrum structure of observed points and explores whether current learner is uncertain to current prediction (Calandriello et al., 2017), measured by $\overline{\sigma}_t$. Generally, a large $\overline{\sigma}_t$ indicates a high prediction variance. While intensive studies have been done in both directions, few methods leverage two quantities together to make a sampling decision. To fill this gap, we propose an adaptive-margin confidence, Θ_t, acting as the lower confidence bound of the predicted margin. Different from (Hoi et al., 2012), Θ_t is derived from online adversarial setting for kernel sketching. We summarize this algorithm, namely OKSG, in Algorithm 2. We next theoretically analyze the effectiveness of the OKSG.

In the randomized query, the mistake trials can be partitioned into two disjoint sets, set $\mathcal{S} = \{t : \frac{h}{h+\max(0,\Theta_t)} < 1\}$ includes indices on which a stochastic sampling is conduct, while set $\mathcal{I} = \{t : \frac{h}{h+\max(0,\Theta_t)} = 1\}$ includes indices when there is a deterministic sampling. We denote by $\mathcal{M} = \{t : y_t f_t \leq 0\}$ as the set of mistake trials and let $M = |\mathcal{M}|$, and by $\mathcal{Z}_T = \{t \leq T : Q_t Z_t = 1\}$ as the set of sampled kernels and $\ell(x)$ as a hinge loss over x.

For any data sequence $\mathcal{D}_T = \{\mathbf{x}_t\}_{t=1}^T$ and any $b > 1$, the $\sum_t \tau_t$ is closely related to kernel matrix: $\log(\det(\overline{\mathbf{K}}_T/b +$

Algorithm 2 OKSG: online kernel sampling algorithm

Input: $\{\mathbf{x}_t, y_t\}_{t=1}^T$, parameters b and h.
Output: $\mathbf{u}_T$
Initialize: $\mathbf{M}_0 = b\mathbf{I}$, and $\Phi_0 = \emptyset$
for $t = 1, \ldots, T$ **do**
  Receive $\mathbf{x}_t$ and $\mathbf{u}_t = \overline{\Phi}_{t-1}\mathbf{M}_{t-1}^{-1}\overline{\mathbf{y}}_{t-1}$
  Compute $f_t = \mathbf{u}_t^\top \phi_t$ and τ_t in Eq. (3)
  Calculate $\Theta_t = |f_t| - \tau_t$
  Generate $Q_t \sim \frac{h}{h+\max(0,\Theta_t)}$
  if $Q_t = 1$ **then**
    Query the actual label y_t and $\hat{y}_t = \mathrm{sgn}(f_t)$
    if $\hat{y}_t \neq y_t$ **then**
      Set $Z_t = 1$; Otherwise, $Z_t = 0$
    end if
  end if
  Update $\overline{\Phi}_t = [\overline{\Phi}_{t-1}, Z_t\overline{\phi}_t]$ and $\overline{\mathbf{y}}_t = [\overline{\mathbf{y}}_{t-1}, Z_t\overline{y}_t]$
  Update $\mathbf{M}_t = b\mathbf{I} + \overline{\Phi}_t^\top\overline{\Phi}_t$
end for

$\mathbf{I}$)). Using Sylvester's determinant identity,

$$\det(\overline{\Phi}_T\overline{\Phi}_T^\top + b\mathbf{I}) = \det(\overline{\Phi}_T^\top\overline{\Phi}_T + b\mathbf{I}) = \det(\overline{\mathbf{K}}_T + b\mathbf{I}).$$

Then, we have that

$$\begin{aligned}
&\sum_{t=1}^T \overline{\phi}_t^\top \left(\overline{\Phi}_t\overline{\Phi}_t^\top + b\mathbf{I}\right)^{-1}\overline{\phi}^\top \\
=&\sum_{t=1}^T \left(\overline{\phi}_t^\top/\sqrt{b}\right)\left(\overline{\Phi}_t\overline{\Phi}_t^\top/b + \mathbf{I}\right)^{-1}\left(\overline{\phi}_t/\sqrt{b}\right) \\
\leq&\log\left(\frac{\det\left(\overline{\Phi}_T\overline{\Phi}_T^\top + b\mathbf{I}\right)}{\det\left(\overline{\Phi}_0\overline{\Phi}_0^\top + b\mathbf{I}\right)}\right) = \log\det\left(\overline{\mathbf{K}}_T/b + \mathbf{I}\right)
\end{aligned} \tag{4}$$

where the first inequality holds due to

$$\begin{aligned}
&\left(\overline{\Phi}_t\overline{\Phi}_t^\top + b\mathbf{I}\right) = \left(\overline{\Phi}_{t-1}\overline{\Phi}_{t-1}^\top + b\mathbf{I}\right) + \overline{\phi}_t\overline{\phi}_t^\top \Rightarrow \\
\tau_t = \overline{\phi}_t^\top &\left(\overline{\Phi}_t\overline{\Phi}_t^\top + b\mathbf{I}\right)^{-1}\overline{\phi}_t = 1 - \frac{\det\left(\overline{\Phi}_{t-1}\overline{\Phi}_{t-1}^\top + b\mathbf{I}\right)}{\det\left(\overline{\Phi}_t\overline{\Phi}_t^\top + b\mathbf{I}\right)} \\
&\leq -\log\det\left(\frac{\det\left(\overline{\Phi}_{t-1}\overline{\Phi}_{t-1}^\top + b\mathbf{I}\right)}{\det\left(\overline{\Phi}_t\overline{\Phi}_t^\top + b\mathbf{I}\right)}\right).
\end{aligned}$$

We notice that in first inequality we relate τ_t to the log-determinant of the kernel matrix $\overline{\mathbf{K}}_T$. This quantity appears in a large number of works on online linear prediction (Cesa-Bianchi et al., 2006) where they were connected to maximal mutual information gain in Gaussian processes. In general, τ_t has been used to measure the correlation between a point t and the other $t-1$ points, indicating how essential it is in characterizing the dataset. If $\overline{\phi}_i$ is completely orthogonal to the other data, $\tau_t \to 1/(1+b)$ is maximized; while in the case where all the points are identical, $\tau_t \to 1/(t+b)$ is minimized. At time point, we can generate the mistake bounds of online kernel sampling.

Theorem 2. *Assume an arbitrary node-label sequence* $\{(\mathbf{x}_t, y_t)\}_{t=1}^T$, *the OKSG learns on only sampled trails* $\{t : Q_t = 1, s.t.\ Q_t \sim \mathcal{B}\left(\frac{h}{h+\max(0,\Theta_t)}\right)\}$ *with* $h > 0$. *Assume that* $C_{\mathcal{Z}} = |\mathcal{Z}|\max_{t\in\mathcal{Z}} a_t + b$, *the mistake bound holds for any* $\mathbf{u} \in \mathcal{H}$,

$$\begin{aligned}
\mathbb{E}[M] \leq&\mathbb{E}\left[\sum_{t\in\mathcal{Z}_T} a_t\ell(y_t\mathbf{u}^\top\phi_t)\right] + \frac{hC_{\mathcal{Z}_T}}{2}\|\mathbf{u}\|^2 \\
&+ \frac{\max_t a_t}{2h}\log\det(\overline{\mathbf{K}}_{\mathbb{E}[\mathcal{M}\cap\mathcal{I}]}/b + \mathbf{I}).
\end{aligned}$$

The expectation of sampling is upper bound by $\mathbb{E}\left[|\mathcal{I}| + \sum_{t\in\mathcal{S}}\frac{h}{h+\max(0,\Theta_t)}\right]$.

Proof. According to Algorithm 2, if the trails is such that $Q_tZ_t = 0$, then $\mathbf{u}_t = \mathbf{u}_{t-1}$ with no update, which yields $\min_{\mathbf{u}} F_t(\mathbf{u}) = \min_{\mathbf{u}} F_{t-1}(\mathbf{u})$. Hence the equality,

$$\begin{aligned}
&\min_{\mathbf{u}} F_t(\mathbf{u}) - \min_{\mathbf{u}} F_{t-1}(\mathbf{u}) \\
=&Q_tZ_t\left[(y_t - f_t)^2 - (a_t\tau_t - a_t + 1)y_t^2\right],
\end{aligned}$$

holds for all trial t. Summing over $t = 1, \ldots, T$ and expanding the squares in both sides with slightly manipulation, we obtain

$$\begin{aligned}
&\sum_{t=1}^T Q_tZ_t\left(-y_tf_t - \frac{1}{2}a_t\tau_t\right) \leq -\sum_{t=1}^T Q_tZ_t\frac{1}{2}f_t^2 \\
&\frac{1}{2}\mathbf{u}^\top\left(b\mathbf{I} + \sum_t Q_tZ_ta_t\phi_t\phi_t^\top\right)\mathbf{u} - \sum_{t=1}^T Q_tZ_ta_ty_t\mathbf{u}^\top\phi_t.
\end{aligned}$$

holding for any $\mathbf{u}$. We add h on both sides where $h > 0$ and replace $\mathbf{u}$ with $h\mathbf{u}$ since $\mathbf{u}$ is a random variable. Using inequality $1 - x \leq \max(1-x, 0)$ yields

$$\begin{aligned}
&hQ_tZ_t - hQ_tZ_ta_ty_t\mathbf{u}^\top\phi_t \\
\leq&hQ_tZ_ta_t - hQ_tZ_ta_ty_t\mathbf{u}^\top\phi_t \leq hQ_tZ_ta_t\ell(y_t\mathbf{u}^\top\phi_t),
\end{aligned}$$

where $\ell(\cdot)$ is hinge loss. Observing $-y_tf_t = +|f_t|$ whenever an error $Q_tZ_t = 1$ while we omit the terms $-\sum_{t=1}^T f_t^2$ that do not affect the bound, we simplify with the notations $\mathbf{M}_t$ and Θ_t,

$$\begin{aligned}
&\sum_{t=1}^T \mathbb{E}\left[Q_tZ_t(\Theta_t + h)\right] \\
\leq&\frac{1}{2}h^2\mathbf{u}^\top\mathbb{E}[\mathbf{M}_{\mathcal{Z}_T}]\mathbf{u} + h\mathbb{E}\left[\sum_{t\in\mathcal{Z}_T} a_t\ell(y_t\mathbf{u}^\top\phi_t)\right]
\end{aligned}$$

When an error incurs at trial $t \in \mathcal{M}$, the function Θ_t can be positive ($t \in \mathcal{M}\cap\mathcal{S}$) or negative ($t \in \mathcal{M}\cap\mathcal{I}$): In the

former subcase, Q_t is a random variable with expectation $\mathbb{E}[Q_t] = \frac{h}{\Theta_t + h}$ and thus

$$\mathbb{E}\left[Q_t Z_t(\Theta_t + h)\right] = \mathbb{E}[Z_t]\mathbb{E}[Q_t(\Theta_t + h)] = h\mathbb{E}[Z_t];$$

In the later subcase, $\mathbb{E}[Q_t] = 1$. We bound

$$\begin{aligned}&\mathbb{E}\left[Z_t Q_t(h + |f_t| - \frac{1}{2}a_t\tau_t\right]\\ \geq&\mathbb{E}\left[Z_t(h - \frac{1}{2}a_t\tau_t\right] \geq h\mathbb{E}[Z_t] - \mathbb{E}\left[\frac{1}{2}a_t\tau_t\right].\end{aligned}$$

To summarize,

$$\begin{aligned}&\sum_{t=1}^{T}\mathbb{E}\left[Q_t Z_t(\Theta_t + h)\right]\\ \geq&\sum_{t\in\mathcal{M}\cap\mathcal{S}} h\mathbb{E}[Z_t] + \sum_{t\in\mathcal{M}\cap\mathcal{I}}\left(h\mathbb{E}[Z_t] - \mathbb{E}[\frac{1}{2}a_t\tau_t]\right)\\ =&h\mathbb{E}[M] - \mathbb{E}\left[\sum_{t\in\mathcal{M}\cap\mathcal{I}}\frac{1}{2}a_t\tau_t\right]\end{aligned}$$

Summarizing above equations, we complete the proof. Note that labels are selected randomly, so that all expectations occurring are with respect to the randomization. □

Remark: By recalling the regret term in Eq. (4), we notice that there is a deep connection between the mistake bound and cumulative sum of the τ_t. The τ_t captures how much the adversary can increase the mistake bound by picking orthogonal directions that have not been observed before. While in linear model, this can happen at most d times (the rank of matrix $\sum_t \mathbf{x}_t\mathbf{x}_t^\top$ is at most d); In kernel model, it can grow linearly with time, since large $\mathcal{H}$ can have infinite dimensions. Nonetheless, the actual growth rate is directly related to the complexity of sequential points chosen by the adversary and kernel space. While the matrix $\mathbf{K}_t$ captures the capability of the RKHS $\mathcal{H}$ on the points $\mathcal{D}_t$, we see that the bound in Theorem 2 is rather related to $\sum_t \tau_t$, and we show that the two quantities are also related to each other.

5.2 Exploration Learning

Although OKSG is efficient, it uses an exploitation rule to update model when an error occurs (e.g., $f_t y_t \leq 0$). In this section, we propose an adaptive kernel sampling algorithm, namely AKSG in Algorithm 3, to achieve an exploitation-exploration tradeoff. To achieve this, it explores the hypothesis based on the lower confidence bound (LCB) of model prediction. At each round t, the actual label y_t is revealed based on the confidence bound $\Theta_t = |f_t| - \frac{1}{2}a_t\tau_t$, which yields to a stochastic sampling and deterministic sampling. In stochastic sampling, update is driven by mistake. We observe that a deterministic query is issued whenever $\Theta_t \leq 0$. In this case, an aggressive update is performed, i.e., we update model even if no error occurs.

Algorithm 3 AKSG: adaptive kernel sampling algorithm

Input: $\{\mathbf{x}_t, y_t\}_{t=1}^T$, parameters b and h
Output: $\mathbf{u}_T$
Initialize: $\mathbf{M}_0 = b\mathbf{I}$, and $\Phi_0 = \emptyset$
for $t = 1, \ldots, T$ **do**
  Receive $\mathbf{x}_t \in R^d$, predict f_t and τ_t
  Calculate $P_t = \frac{h}{h + \max(0, \Theta_t)}$
  if $P_t < 1$ **then**
    Generate Q_t by a Bernulli trail with P_t
    if $Q_t = 1$ **then**
      Query the actual label y_t and $\hat{y}_t = \text{sgn}(f_t)$
      Set $Z_t = 1$ if $\hat{y}_t \neq y_t$; otherwise $Z_t = 0$
    end if
  else
    Set $Z_t = 1$ for any prediction $\hat{y}_t$
  end if
  Update $\overline{\Phi}_t = [\overline{\Phi}_{t-1}, Z_t\overline{\phi}_t]$ and $\overline{\mathbf{y}}_t = [\overline{\mathbf{y}}_{t-1}, Z_t\overline{y}_t]$
  Update $\mathbf{M}_t = b\mathbf{I} + \overline{\Phi}_t^\top\overline{\Phi}_t$
end for

Remark: To further understand the aggressive method, we compute under what condition a deterministic query will be issued. A query is issued with a probability 1 when $\Theta_t = |f_t| - \frac{1}{2}a_t\tau_t \leq 0$. By solving for $|f_t|$,

$$\Theta_t \leq 0 \Rightarrow |f_t| \leq \theta(\tau_t) = \frac{1}{2}a_t\tau_t.$$

If $|f_t|$ is less than $\theta(\tau_t)$, a deterministic update is conducted, while $|f_t|$ is above $\theta(\tau_t)$, i.e., $\Theta_t > 0$, a update will be issued with a probability strictly less than 1. The upper bound of $\theta(\tau_t)$ increases with τ_t. If all points are identical, i.e., $\tau_t \approx \frac{1}{t+b} \to 0$, a deterministic update is issued only when an input lies on the boundary (i.e. $|f_t| \leq \theta(0) = 0$). However, if current input is completely different from other data, i.e., $\tau_t \approx \frac{1}{b}$, this implies that an deterministic update is issued whenever $|f_t|$ is less than $\theta(\tau_t) = a_t/2b$.

Remark: Given $\{\lambda_t\}$ the eigenvalues of $\overline{\mathbf{K}}_T$, we notice that $\log\det(\overline{\mathbf{K}}_T/b + \mathbf{I}) = \log\left(\prod_{t=1}^T(\lambda_t/b + 1)\right) = \sum_{t=1}^T\log(\lambda_t/b + 1)$. Next, we can decompose this as

$$\begin{aligned}&\sum_{t=1}^{T}\log(\lambda_t/b + 1) = \sum_{t=1}^{T}\log(\lambda_t/b + 1)(\frac{\lambda_t/b + 1}{\lambda_t/b + 1})\\ =&\sum_{t=1}^{T}\log(\lambda_t/b + 1)\frac{\lambda_t/b}{\lambda_t/b + 1} + \sum_{t=1}^{T}\frac{\log(\lambda_t/b + 1)}{\lambda_t/b + 1}\\ \leq&\log(\|\overline{\mathbf{K}}_T\|_{op}/b + 1)\sum_{t=1}^{T}\frac{\lambda_t/b}{\lambda_t/b + 1} + \sum_{t=1}^{T}\frac{\lambda_t/b + 1 - 1}{\lambda_t/b + 1}\\ \leq&(\log(\|\overline{\mathbf{K}}_T\|_{op}/b + 1) + 1)r,\end{aligned}$$

where the first inequality is due to $\|\overline{\mathbf{K}}_T\|_{op} \geq \max_t \lambda_t$ and the second inequality is due to $\log(x) \leq x - 1$. We observe

that it is smaller than the rank r of kernel matrix $\overline{\mathbf{K}}_T$ for any $b > 1$ (This can be seen as $\sum_t \frac{\lambda_t}{\lambda_t + b} \leq \text{rank}(\overline{\mathbf{K}}_T)$). We slightly improve the bound of (Luo et al., 2016) from $\mathcal{O}(d \log T)$ down to $\mathcal{O}(r \log T)$ where r is the rank of the learned points. For the kernel models, simple functional gradient descent (e.g., NORMA (Kivinen et al., 2001)) achieves a $\mathcal{O}(\sqrt{T})$ regret regardless of the loss function.

6 Empirical Experiments

In this section, we introduce empirical results to validate the proposed algorithms. Our experiments shows that he proposed kernel sampling algorithm is effective to reduce the time and space complexity significantly while maintaining comparable performance; Meanwhile the adaptive update strategy achieves a better predictive performance via exploring more informative kernels.

6.1 Data Sets and Evaluation Metrics

We introduce three real-world online datasets to evaluate our algorithms: 1) Coauthor[1] extracted from *DBLP* database is an undirected co-author graph in which 1711 authors are denoted as nodes while the co-authored relationship are regarded as the edges. The authors are classified in four classes in terms of research topic: "data mining", "machine learning", "information retrieval", and "databases". 2) Cora[2] is a citation network including 2485 scientific publications and 5429 citation links. The publications regarded as nodes are related to seven domains: "Case based", "Genetic Algorithms", "Rule Learning", "Probabilistic Methods", "Neural Networks", "Reinforcement Learning", and "Theory". 3) IMDB[3] is a movie organization that presses up-to-date movie information. IMDB connects total 17046 movies with the co-actor associations. The movies as nodes in graph are categorized into four genres: "Action", "Romance", "Animation", and "Thriller".

The graph data is supposed to be undirected and connected. If the edges are directed, we transform them into undirected graphs via $\mathbf{S} \leftarrow \max(\mathbf{S}, \mathbf{S}^\top)$. If the graphs are disconnected, the biggest connected subgraph is chosen for study.

We evaluated the performance of baselines and our algorithms with two measurements: **i)** *cumulative error rate*, reflects the prediction accuracy of online algorithm; **ii)** *number of queried nodes*, reflects the computational efficiency of sampling method. Note that small value of above measures indicates a better performance of a method.

6.2 Baselines and Parameter Setting

We compared the proposed algorithms with state-of-the-art baselines. The algorithms and their parameter settings are summarized as follows.

GPA (Herbster and Pontil, 2006) is a first-order linear learning algorithm on graph. Note that the perceptron algorithm is not affected by the step-size. *FOGD* and *NOGD* (Lu et al., 2016) are first-order kernel learning algorithms with kernel functional approximation techniques. FOGD applies the random Fourier features for approximating kernel function; while NOGD applies the Nystrom method to approximate large kernel matrices. *Pros-KONS* (Calandriello et al., 2017) is a second-order online kernel algorithm with adaptive embedding. It exploits confidence bound to sketch the embedding space of the second-order updates. *OKL*, *OKSG* and *AKSG* are the proposed second-order online kernel sampling algorithms. OKL is an online kernel algorithm without a constraint on kernel size. OKSG is an *exploitive* algorithm that update model whenever an error occurs, while AKSG is an *adaptive* algorithm that performs explorative updates on the uncertain points. For all above methods, the parameter d is set to 100 as the performance of algorithm reaches to a convergence. We tune b with the grid $\{1, \ldots, 10\}$ on a held-out random shuffle. For selective sampling strategy, we set $h = 0.01$ for Cora and Coauthor, $h = 0.001$ for IMDB due to graph structure variable.

In order to compare these algorithms fairly, we randomly shuffled the ordering of samples for each dataset. We repeated each experiment 20 times and calculated the average results. In addition, the above algorithms are designed for binary classification. In order to apply the algorithms to those data sets with multiple classes, we use one-vs-rest scheme to adapt the binary classifiers to the multi-class scenario.

6.3 Comparison Result

The experimental results are presented in Figure 1. The improvement of AKSG and OKSG over NOGD and FOGD is always consistant on the three datasets. This is consistent with previous observations in online learning: the second-order algorithms are generally better than the first-order algorithms (Hoi et al., 2012). The reason is that the covariance matrix $\mathbf{A}_t$ that encodes the confidence of parameters can guide the direction of the parameter update in the learning process. In addition, the kernel-based algorithms, e.g., OKSG and Pros-KONS, always outperform the linear algorithm GPA. This is consistent with existing research in kernel learning: the kernel methods are often better than linear methods (Vapnik, 1995; Hastie et al., 2009). The reason is that the kernel method can generate a high-dimensional and implicit feature vector to learn a nonlinear decision boundary of classification model.

AKSG enjoys fewer queried nodes and lower error rates than Pros-KNOS. The reasons is that the AKSG exploits the adaptive-margin confidence to aggressively sample the points with lower confidence of predicted margin. In this way, the queried number and the error rate can be reduced

[1] https://snap.stanford.edu/data/com-DBLP.html
[2] http://www.cs.umd.edu/ sen/lbc-proj/data/
[3] http://www.imdb.com/

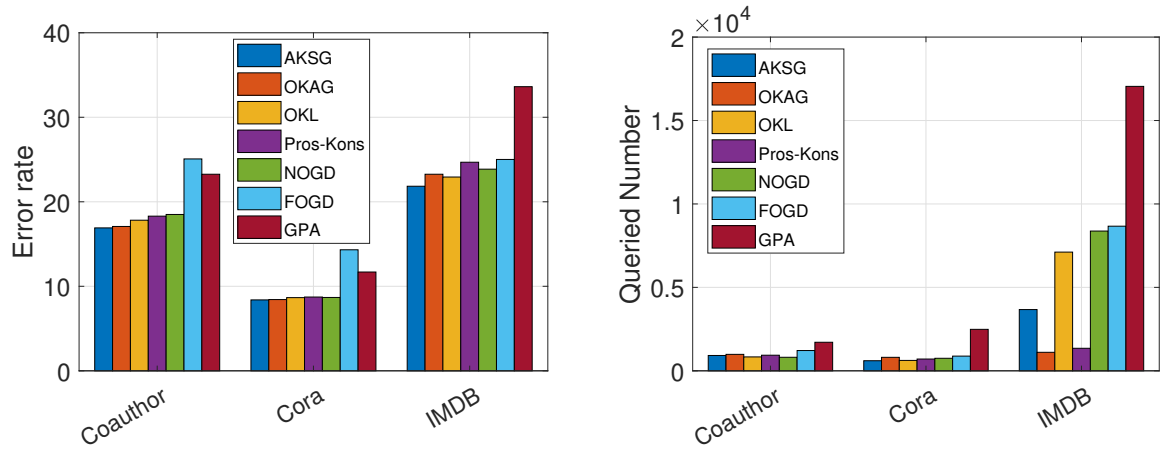

Figure 1: Comparison of the binary-class classification algorithms

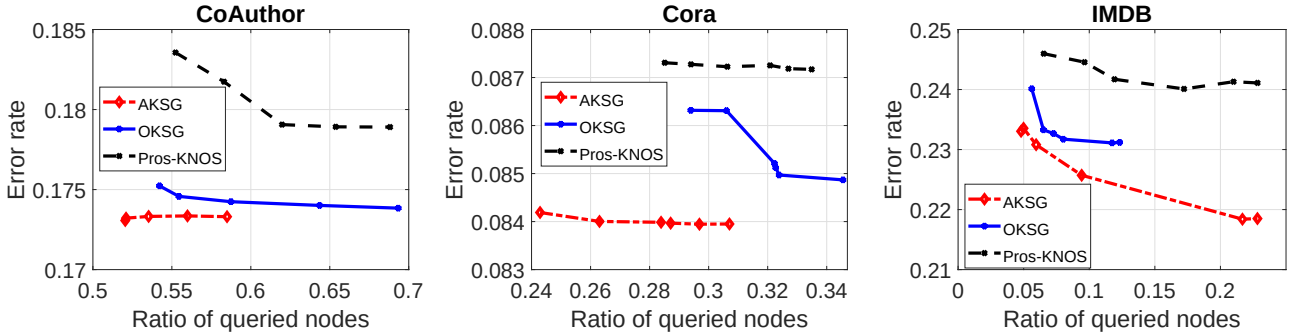

Figure 2: Sensitivity analysis of the update ratio.

further when the model has learned sufficient knowledge of data. It demonstrates the efficacy of the proposed sampling strategy in terms of computation complexity and labeling cost. OKSG achieves a comparable result with OKL. This is reasonable, since OKSG only samples a portion of inputs, whereas OKL can query all inputs. Thus, the performance of the kernel sketched algorithm should be no better than that of the algorithm in fully-supervised setting. However, OKL requires much more queried nodes than OKSG in order to achieve a better accuracy, which is infeasible in many practical scenarios.

6.4 Sensitivity Analysis on Update Ratio

We studied the impact of h with respect to sampling ratio of the OKSG and AKSG. Basically, the smaller the parameter h, the fewer the queried number of nodes. Specifically, we set h to $\{10^{-4}, 10^{-3}, \ldots, 1\}$, and ran algorithm for 20 times under each h. We calculated the average ratio of queried nodes under different values of h. The comparison results in Figure 2 showed that AKSG achieved better performance consistently under different ratios of queried nodes. This validated the computational-efficiency of the proposed confidence score Θ_t that could adaptively prioritize informative kernels to benefit the learning process. We also observed that the AKSG outperformed Pros-KONS significantly over all sketched ratios. The reason was that AKSG maintains a tradeoff between exploitation and exploration for kernel selection. The better results in Figure 2 demonstrated the effectiveness of the proposed sampling strategy.

7 Conclusion

In this paper, we propose a new framework for online kernel learning, leading to a scalable algorithm to tackle binary-class classification on graphs. To reduce the computational complexity, we present a novel randomized kernel sketched technique. Besides, we introduce an aggressive update rule that takes full advantage of the inputs with high predicted variance. The theoretical results demonstrated the efficacy of the proposed algorithms in terms of the expected mistake bound and sampling ratio. The encouraging empirical results on several real-world datasets also indicated that 1) the proposed randomized sampling algorithm is able to achieve comparable or better predictive performance by querying a small amount of kernels; and that 2) the adaptive sampling scheme can further reduce the sampling ratio with the explorative update, showing a superiority of the proposed explorative technique.

References

Rie Kubota Ando and Tong Zhang. Learning on graph with laplacian regularization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 25–32, Vancouver, Canada, 2006.

Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 161–168, Vancouver, Canada, 2007.

Léon Bottou, Olivier Chapelle, Dennis DeCoste, and Jason Weston, editors. *Large-Scale Kernel Machines*. The MIT Press, Cambridge, MA, 2007.

Léon Bottou, Frank E. Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018.

Daniele Calandriello, Alessandro Lazaric, and Michal Valko. Second-order kernel online convex optimization with adaptive sketching. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 645–653, Sydney, Australia, 2017.

Nicolò Cesa-Bianchi and Claudio Gentile. Tracking the best hyperplane with a simple budget perceptron. In *Proceedings of the 19th Annual Conference on Learning Theory, (COLT)*, pages 483–498, Pittsburgh, PA, 2006.

Nicolò Cesa-Bianchi, Alex Conconi, and Claudio Gentile. A second-order perceptron algorithm. In *Proceedings of 15th Annual Conference on Computational Learning Theory (COLT)*, pages 121–137, Sydney, Australia, 2002.

Nicolò Cesa-Bianchi, Claudio Gentile, and Luca Zaniboni. Worst-case analysis of selective sampling for linear classification. *J. Mach. Learn. Res.*, 7:1205–1230, 2006.

Koby Crammer and Claudio Gentile. Multiclass classification with bandit feedback using adaptive regularization. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, pages 273–280, Bellevue, WA, 2011.

Ofer Dekel, Shai Shalev-Shwartz, and Yoram Singer. The forgetron: A kernel-based perceptron on a fixed budget. In *Advances in Neural Information Processing Systems (NIPS)*, pages 259–266, Vancouver, Canada], 2005.

Alexander Gammerman, Yuri Kalnishkan, and Vladimir Vovk. On-line prediction with kernels and the complexity approximation principle. In *Proceedings of the 20th Conference in Uncertainty in Artificial Intelligence (UAI)*, pages 170–176, Banff, Canada, 2004.

Quanquan Gu, Charu C. Aggarwal, Jialu Liu, and Jiawei Han. Selective sampling on graphs for classification. In *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 131–139, Chicago, IL, 2013.

Quanquan Gu, Tong Zhang, and Jiawei Han. Batch-mode active learning via error bound minimization. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 300–309, Quebec City, Canada, 2014.

Trevor J. Hastie, Robert Tibshirani, and Jerome H. Friedman. *The Elements of Statistical Learning:Data Mining, Inference, and Prediction*. Springer, New York, NY, 2nd edition, 2009.

Mark Herbster and Massimiliano Pontil. Prediction on a graph with a perceptron. In *Advances in Neural Information Processing Systems (NIPS)*, pages 577–584, Vancouver, Canada, 2006.

Steven C. H. Hoi, Jialei Wang, and Peilin Zhao. Exact soft confidence-weighted learning. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, Edinburgh, UK, 2012.

Jyrki Kivinen, Alexander J. Smola, and Robert C. Williamson. Online learning with kernels. In *Advances in Neural Information Processing Systems (NIPS)*, pages 785–792, Vancouver, Canada], 2001.

Dan Kushnir. Active-transductive learning with label-adapted kernels. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 462–471, New York, NY, 2014.

Pavel Laskov, Christian Gehl, Stefan Krüger, and Klaus-Robert Müller. Incremental support vector learning: Analysis, implementation and applications. *J. Mach. Learn. Res.*, 7:1909–1936, 2006.

Ping Li. Linearized GMM kernels and normalized random Fourier features. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 315–324, Halifax, NS, Canada, 2017.

Ping Li and Cun-Hui Zhang. Theory of the GMM kernel. In *Proceedings of the 26th International Conference on World Wide Web (WWW)*, pages 1053–1062, Perth, Australia, 2017.

Jing Lu, Steven C. H. Hoi, Jialei Wang, Peilin Zhao, and Zhiyong Liu. Large scale online kernel learning. *J. Mach. Learn. Res.*, 17:47:1–47:43, 2016.

Haipeng Luo, Alekh Agarwal, Nicolò Cesa-Bianchi, and John Langford. Efficient second order online learning by sketching. In *Advances in Neural Information Processing Systems (NIPS)*, pages 902–910, Barcelona, Spain, 2016.

Francesco Orabona and Koby Crammer. New adaptive algorithms for online classification. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1840–1848, Vancouver, Canada, 2010.

Francesco Orabona, Joseph Keshet, and Barbara Caputo. The projectron: a bounded kernel-based perceptron. In *Proceedings of the Twenty-Fifth International Conference on Machine Learning (ICML)*, pages 720–727, Helsinki, Finland, 2008.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1177–1184, Vancouver, Canada, 2007.

Mostafa Rahmani and Ping Li. Graph analysis and graph pooling in the spatial domain. Technical report, 2019. URL arXiv:1910.01589.

Alessandro Rudi, Daniele Calandriello, Luigi Carratino, and Lorenzo Rosasco. On fast leverage score sampling and optimal learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5677–5687, Montréal, Canada, 2018.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels*. The MIT Press, Cambridge, MA, 2002.

Anshumali Shrivastava and Ping Li. A new space for comparing graphs. In *2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 62–71, Beijing, China, 2014.

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag Inc, New York, 1995.

Christopher K. I. Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. In *NIPS*, pages 682–688, Denver, CO, 2000.

Peng Yang and Ping Li. Distributed primal-dual optimization for online multi-task learning. In *Proceedings of the Thirty-Forth AAAI Conference on Artificial Intelligence (AAAI)*, New York, NY, 2020a.

Peng Yang and Ping Li. Efficient online multi-task learning via adaptive kernel selection. In *Proceedings of the 29th International Conference on World Wide Web (WWW)*, Taipei, 2020b.

Tianbao Yang, Yu-feng Li, Mehrdad Mahdavi, Rong Jin, and Zhi-Hua Zhou. Nyström method vs random fourier features: A theoretical and empirical comparison. In *Advances in Neural Information Processing Systems (NIPS)*, pages 476–484. Lake Tahoe, CA, 2012.

Fedor Zhdanov and Yuri Kalnishkan. An identity for kernel ridge regression. In *Proceedings of 21st International Conference on Algorithmic Learning Theory (ALT)*, pages 405–419, Canberra, Australia, 2010.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*, pages 928–936, Washington, DC, 2003.