Online Bayesian Multiple Kernel Bipartite Ranking

Changying Du[1,2], **Changde Du**[3,4], **Guoping Long**[1], **Qing He**[2], **Yucheng Li**[1]
[1]Laboratory of Parallel Software and Computational Science, Institute of Software
Chinese Academy of Sciences (CAS), Beijing 100190, China
[2]Key Lab of Intelligent Information Processing of CAS, Institute of Computing Technology, CAS, Beijing 100190, China
[3]Key Laboratory of Optical Astronomy, National Astronomical Observatories, CAS, Beijing 100012, China
[4]University of Chinese Academy of Sciences, Beijing 100049, China
changying@iscas.ac.cn, cddu@nao.cas.cn, heq@ics.ict.ac.cn

Abstract

Bipartite ranking aims to maximize the area under the ROC curve (AUC) of a decision function. To tackle this problem when the data appears sequentially, existing online AUC maximization methods focus on seeking a point estimate of the decision function in a linear or predefined single kernel space, and cannot learn effective kernels automatically from the streaming data. In this paper, we first develop a Bayesian multiple kernel bipartite ranking model, which circumvents the kernel selection problem by estimating a posterior distribution over the model weights. To make our model applicable to streaming data, we then present a kernelized online Bayesian passive-aggressive learning framework by maintaining a variational approximation to the posterior based on data augmentation. Furthermore, to efficiently deal with large-scale data, we design a fixed budget strategy which can effectively control online model complexity. Extensive experimental studies confirm the superiority of our Bayesian multi-kernel approach.

1 INTRODUCTION

Aiming to learn a ranking decision function that is likely to place a positive instance before most negative ones, bipartite ranking [Agarwal and Roth, 2005; Clémençon *et al.*, 2008; Kotlowski *et al.*, 2011] is especially useful for imbalanced data sets, where the area under the ROC curve (AUC) [Hanley and McNeil, 1982; Bradley, 1997; Cortes and Mohri, 2004] is a commonly used evaluation metric. Recently, several online AUC maximization algorithms with pairwise loss functions were proposed to tackle this problem when the data appears sequentially [Zhao *et al.*, 2011a,b; Zhao and Hoi, 2012; Yang *et al.*, 2013; Gao *et al.*, 2013; Hu *et al.*, 2015; Ding *et al.*, 2015]. Furthermore, the generalization performance of online learning algorithms with pairwise loss functions have been investigated in [Wang *et al.*, 2012; Kar *et al.*, 2013].

Nevertheless, existing work only focus on seeking a point estimate of the decision function in a linear or predefined single kernel space, and cannot learn effective kernels automatically from the streaming data. As well known, choosing an appropriate kernel for real-world situations usually is not easy for users without enough domain knowledge. Multiple Kernel Learning (MKL) [Rakotomamonjy *et al.*, 2008; Gönen and Alpaydın, 2011] can circumvent such a problem by learning an optimal linear combination of a set of predefined kernels. However, conventional online MKL algorithms [Luo *et al.*, 2010; Hoi *et al.*, 2013; Sahoo *et al.*, 2014] are unsuitable for a direct use since AUC is a metric represented by the sum of pairwise losses between instances from different classes. More importantly, existing online MKL methods typically maintain a point estimate of their model weights, which can be affected seriously by online outliers and usually needs a tough tuning process for their penalty parameters.

Focusing on these problems, we first develop a Bayesian Bipartite Ranking model with Multiple Kernels (B^2RMK), which circumvents the kernel selection problem by estimating a posterior distribution over the model weights. Specifically, B^2RMK imposes global-local sparsity shrinkage priors on the weights of kernels and support vectors and adopts a margin-based ranking pseudo-likelihood function that mimics the pairwise hinge loss and can be expressed as a location-scale mixture of normals. Within the Bayesian formalism, the new model can automatically infer the weight penalty parameters and naturally alleviate overfitting to small training sets via model averaging over the posterior.

To make our model applicable to streaming data, we then present a kernelized online Bayesian Passive-Aggressive (PA) [Crammer *et al.*, 2006] learning framework. As far as we know, this is the first effort to perform online Bayesian learning with multiple kernels. The key of Bayesian PA

is to maintain a posterior distribution at each time step. Unfortunately, exact posterior inference of B^{2}RMK is intractable, so we devise an efficient data augmentation [Tanner and Wong, 1987] based variational method to approximate the posterior with mean-field assumption.

Unlike most existing online MKL methods [Jin *et al.*, 2010; Luo *et al.*, 2010; Hoi *et al.*, 2013], which don't bound their model complexity and require more and more memory and computation when the data arrives sequentially, online B^{2}RMK only maintains two fixed-size buffers (one for positive instances, and another for negative ones) to store the learned support vectors, thus are much more efficient for large-scale data sets. When a buffer is full, we propose a principled strategy to update it, which can guarantee that the most important support vectors won't be discarded. As in [Hu *et al.*, 2015], smooth updating and compensation schemes are employed to further boost performance when updating the buffer. Extensive experimental studies confirm the superiority of our online Bayesian multi-kernel approach.

2 PRELIMINARIES

To better motivate our work, we first introduce some preliminaries, including bipartite ranking, AUC optimization and MKL. Let $\mathcal{X} = \{x \in \mathbb{R}^d\}$ be the instance space, $\mathcal{Y} = \{+1, -1\}$ be the label set, and $\mathbf{S} = \mathbf{S}_+ \cup \mathbf{S}_-$ be a set of training instances, where $\mathbf{S}_+$ and $\mathbf{S}_-$ include N_+ positive instances and N_- negative instances, respectively. The goal of bipartite ranking is to learn a ranking decision function that is likely to place a positive instance before most negative ones. AUC is a commonly used evaluation metric for bipartite ranking. For a ranking decision function $f : \mathcal{X} \to \mathbb{R}$, its AUC measure on $\mathbf{S}$ is defined as:

$$\mathrm{AUC}(f) = 1 - \frac{\sum_{i=1}^{N_+}\sum_{j=1}^{N_-}\mathbb{I}\big(f(x_i^+)\le f(x_j^-)\big)}{N_+N_-}$$

where $\mathbb{I}(\pi)$ is the indicator function that equals 1 when π is true and 0 otherwise. Since directly maximizing $\mathrm{AUC}(f)$ leads to a difficult combinatorial optimization problem, in practice, the indicator function is usually replaced by its convex surrogate, e.g., pairwise hinge loss [Hu *et al.*, 2015; Zhao *et al.*, 2011a] and squared loss [Gao *et al.*, 2013], which leads to minimizing the following regularized pairwise learning task [Christmann and Zhou, 2015]:

$$\mathcal{L}(f) = \frac{c}{2}\|f\|_{\mathcal{H}}^2 + \sum_{i=1}^{N_+}\sum_{j=1}^{N_-}\ell\left(f; x_i^+, x_j^-\right)$$

where $\|\cdot\|_{\mathcal{H}}$ denotes the norm in Reproducing Kernel Hilbert Space (RKHS), $\ell(\cdot)$ is the convex surrogate loss function and c is the regularization parameter balancing the model complexity and training errors.

A kernelized ranking decision function $f : \mathcal{X} \to \mathbb{R}$ that is used to predict the ranking score of a test instance x_i can be written as

$$f(x_i) = \boldsymbol{a}^\top \boldsymbol{k}_i + b$$

where $\boldsymbol{a} = [a_1, \ldots, a_N]^\top$ denotes the vector of weights assigned to each training instance ($N = N_+ + N_-$); b is the bias; and $\boldsymbol{k}_i = [\mathfrak{K}(x_1, x_i), \ldots, \mathfrak{K}(x_N, x_i)]^\top$, where $\mathfrak{K} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a kernel function that measures the similarities between x_i and x_j, $j = 1, 2, \ldots, N$.

An important problem in single kernel learning is to prespecify the kernel parameters, which is often done in an empirical way. This problem can be circumvented by using the MKL framework [Rakotomamonjy *et al.*, 2008], which is a popular technique for learning an optimal linear combination of a set of predefined kernels. MKL algorithms basically use a weighted sum of P kernels $\{\mathfrak{K}_m : \mathcal{X}\times\mathcal{X} \to \mathbb{R}\}_{m=1}^P$ to get the following multi-kernel ranking decision function:

$$f(x_i) = \boldsymbol{a}^\top\left(\sum_{m=1}^{P} e_m \boldsymbol{k}_{m,i}\right) + b = \sum_{m=1}^{P} e_m \boldsymbol{a}^\top \boldsymbol{k}_{m,i} + b$$

where $\boldsymbol{k}_{m,i} = [\mathfrak{K}_m(x_1, x_i), \ldots, \mathfrak{K}_m(x_N, x_i)]^\top$, and e_m denotes the weight assigned to the m-th kernel.

In the sequel, the m-th $N \times N$ kernel matrix will be denoted by $\mathbf{K}_m$, and the vector of ranking scores of positive instances and negative instances will be denoted by $\mathbf{f}_+$ and $\mathbf{f}_-$, respectively, and $\mathbf{f} = \left[\mathbf{f}_+^\top \;\; \mathbf{f}_-^\top\right]^\top$.

3 BAYESIAN MULTIPLE KERNEL BIPARTITE RANKING

In this section, we formulate a probabilistic model for multi-kernel bipartite ranking. We impose a fairly general global-local shrinkage prior on the sample weights and kernel combination weights to keep the sparsity of them. Since pairwise hinge loss makes traditional Bayesian analysis hard, we then define a margin-based ranking pseudo-likelihood function to overcome this situation. After introducing a set of augmented variables and making the mean-field assumption, the intractable inference problem of our model can be transformed into a tractable one.

3.1 BAYESIAN SPARSITY SHRINKAGE PRIOR

Similar as in kernelized SVM, the sample weights $\boldsymbol{a}$ is often expected to be sparse for selecting support vectors actually needed in decision function. From the Bayesian perspective, a fairly general global-local shrinkage prior is the Three Parameter Beta Normal ($\mathcal{TPBN}$) [Armagan *et al.*, 2011], which favors strong shrinkage of small signals while having heavy tails to avoid over-shrinkage of the larger signals. Thus, concentrated at zero, $\mathcal{TPBN}$ can yield sparse model representations by suppressing unnecessary support vectors. In this paper, we select $\mathcal{TPBN}$ as the sparsity shrinkage prior due to its better mixing properties than

priors such as spike-slab and Laplace. Besides, $\mathcal{TPBN}$ works well for high-dimensional settings since it can specify global and local properties independently. Assuming $\boldsymbol{a} \sim \prod_{i=1}^{N} \mathcal{TPBN}(a_i|\alpha_a, \beta_a, \phi)$, we have the following hierarchical prior:

$$\begin{aligned} a_i|\lambda_i &\sim \mathcal{N}(a_i|0, \lambda_i), \\ \lambda_i|\xi_i &\sim \Gamma(\lambda_i|\alpha_a, \xi_i), \qquad \xi_i|\phi \sim \Gamma(\xi_i|\beta_a, \phi), \\ \phi|\eta &\sim \Gamma(\phi|1/2, \eta), \qquad \eta \sim \Gamma(\eta|1/2, 1), \end{aligned}$$

where $\Gamma(\cdot)$ is the Gamma distribution (with shape-rate parameterization). Note that sample-level sparsity can be tuned by assigning suitable values to the hyper-parameters (α_a, β_a). Setting $\alpha_a = \beta_a = 0.5$, a special case of $\mathcal{TPBN}$ corresponds to the horseshoe prior.

Similarly, sparsity on kernel combination weights $\boldsymbol{e} = [e_1, \ldots, e_P]^\top$ is also desirable for better interpretability. So, we assume $\boldsymbol{e} \sim \prod_{m=1}^{P} \mathcal{TPBN}(e_m|\alpha_e, \beta_e, \rho)$, involving latent variables ω_m, φ_m and τ. Kernel-level sparsity can also be tuned by changing (α_e, β_e).

With the above priors on model weights, we assume the ranking scores of our B^{2}RMK model have the following distributions,

$$\begin{aligned} \mathbf{G}|\boldsymbol{a}, \{\mathbf{K}_m\}_{m=1}^{P}, \upsilon &\sim \prod_{m=1}^{P}\prod_{i=1}^{N} \mathcal{N}\left(g_{mi}|\boldsymbol{a}^\top \boldsymbol{k}_{m,i}, \upsilon^{-1}\right), \\ \mathbf{f}_+|\boldsymbol{e}, \mathbf{G}, b, c &\sim \prod_{i=1}^{N_+} \mathcal{N}\left(f_i|\boldsymbol{e}^\top \boldsymbol{g}_{\cdot i} + b, c^{-1}\right), \\ \mathbf{f}_-|\boldsymbol{e}, \mathbf{G}, b, c &\sim \prod_{j=1}^{N_-} \mathcal{N}\left(f_j|\boldsymbol{e}^\top \boldsymbol{g}_{\cdot j} + b, c^{-1}\right), \end{aligned}$$

where $\upsilon \sim \Gamma(\upsilon|\alpha_\upsilon, \beta_\upsilon)$, $b|\gamma \sim \mathcal{N}\left(b|0, \gamma^{-1}\right)$, $\gamma \sim \Gamma(\gamma|\alpha_\gamma, \beta_\gamma)$, $c \sim \Gamma(c|\alpha_c, \beta_c)$. The intermediate variables g_{mi} $(m = 1, \ldots, P,\ i = 1, \ldots, N)$ are introduced to make the inference procedures efficient. The $P \times N$ matrix of intermediate variables is denoted by $\mathbf{G}$, and the m-th row and i-th column of $\mathbf{G}$ by $\boldsymbol{g}_{m\cdot}$ and $\boldsymbol{g}_{\cdot i}$, respectively. In the sequel, all hyper-parameters will be denoted by $\Upsilon = \{\alpha_a, \beta_a, \alpha_e, \beta_e, \alpha_\upsilon, \beta_\upsilon, \alpha_\gamma, \beta_\gamma, \alpha_c, \beta_c\}$, while the priors by $\Psi = \{\boldsymbol{\lambda}, \boldsymbol{\xi}, \phi, \eta, \boldsymbol{\omega}, \boldsymbol{\varphi}, \rho, \tau, \upsilon, \gamma, c\}$ and the remaining variables by $\Omega = \{\boldsymbol{a}, b, \boldsymbol{e}, \mathbf{G}, \mathbf{f}_+, \mathbf{f}_-\}$.

3.2 MARGIN-BASED RANKING LIKELIHOOD

Several loss functions are available for bipartite ranking. Among them, pairwise hinge loss is the tightest convex upper bound on the rank loss, which is beneficial for better performance and faster convergence. Specifically, the pairwise hinge loss on data set $\mathbf{S}$ can be written as

$$\ell(f; \mathbf{S}) = \sum_{i=1}^{N_+}\sum_{j=1}^{N_-} \max\left(0, 1 - f(x_i^+) + f(x_j^-)\right),$$

which does not lend itself to a convenient description of a likelihood function. To overcome this situation, we propose to define a margin-based ranking pseudo-likelihood

$$L(\mathbf{y}|\mathbf{f}) = \prod_{i=1}^{N_+}\prod_{j=1}^{N_-} \exp\left(-2\max(0, 1 - f_i + f_j)\right),$$

where $\mathbf{y}$ is the label vector and has been divided out. With $\mathcal{TPBN}$ priors and the above margin-based ranking pseudo-likelihood, we can get the following pseudo posterior distribution using Bayes' rule

$$p(\Psi, \Omega|\mathbf{y}) = L(\mathbf{y}|\mathbf{f})p(\Omega|\Psi)p(\Psi)/p(\mathbf{y}|\Upsilon),$$

where $p(\mathbf{y}|\Upsilon)$ is the normalization constant. Note that it is hard to compute $p(\Psi, \Omega|\mathbf{y})$ analytically due to the max function in $L(\mathbf{y}|\mathbf{f})$. Fortunately, we can re-express $L(\mathbf{y}|\mathbf{f})$ as the product of $N_+ \cdot N_-$ location-scale mixtures of normals (see Supplement Section A) based on data augmentation idea [Tanner and Wong, 1987]

$$L(\mathbf{y}|\mathbf{f}) = \prod_{i=1}^{N_+}\prod_{j=1}^{N_-} \int_0^\infty \frac{\exp\left(\frac{(\theta_{ij}+1-f_i+f_j)^2}{-2\theta_{ij}}\right)}{\sqrt{2\pi\theta_{ij}}}\, d\theta_{ij}, \quad (1)$$

where θ_{ij} is the augmented variable. In the following, the matrix of augmented variables will be denoted by $\boldsymbol{\theta}$, and the i-th row and j-th column of $\boldsymbol{\theta}$ by $\boldsymbol{\theta}_{i\cdot}$ and $\boldsymbol{\theta}_{\cdot j}$, respectively. (1) indicates that the posterior distribution $p(\Psi, \Omega|\mathbf{y})$ can be expressed as the marginal of a higher-dimensional distribution that includes the augmented variables. Therefore, the augmented posterior distribution has the form

$$p(\Psi, \Omega, \boldsymbol{\theta}|\mathbf{y}) = \frac{L(\mathbf{y}, \boldsymbol{\theta}|\mathbf{f})p(\Omega|\Psi)p(\Psi)}{p(\mathbf{y}|\Upsilon)}, \quad (2)$$

where the unnormalized joint distribution of $\mathbf{y}$ and $\boldsymbol{\theta}$ conditioned on $\mathbf{f}$ is

$$L(\mathbf{y}, \boldsymbol{\theta}|\mathbf{f}) = \prod_{i=1}^{N_+}\prod_{j=1}^{N_-} \frac{1}{\sqrt{2\pi\theta_{ij}}} \exp\left(\frac{(\theta_{ij} + 1 - f_i + f_j)^2}{-2\theta_{ij}}\right).$$

3.3 VARIATIONAL APPROXIMATE INFERENCE

Directly solving for the augmented posterior is intractable due to the normalization constant $p(\mathbf{y}|\Upsilon)$, thus we appeal to the mean-field variational approximate Bayesian inference method, which is generally much more efficient than the Markov Chain Monte Calo (MCMC) based sampling methods. Specifically, we assume there are a family of factorable and free-form variational distributions

$$\begin{aligned} q(\Psi, \Omega, \boldsymbol{\theta}) = q(\boldsymbol{\lambda})q(\boldsymbol{\xi})q(\phi)q(\eta)q(\boldsymbol{\omega})q(\boldsymbol{\varphi})q(\rho)q(\tau)q(\upsilon) \\ \cdot q(\gamma)q(c)q(\boldsymbol{a})q(b, \boldsymbol{e})q(\mathbf{G})q(\mathbf{f}_+)q(\mathbf{f}_-)q(\boldsymbol{\theta}), \end{aligned}$$

and the objective is to get the optimal one which minimizes the Kullback-Leibler (KL) divergence between the approximating distribution and the target posterior, i.e.,

$$\min_{q(\Psi,\Omega,\boldsymbol{\theta})\in\mathcal{P}} \mathrm{KL}\left(q(\Psi, \Omega, \boldsymbol{\theta})\|p(\Psi, \Omega, \boldsymbol{\theta}|\mathbf{y})\right),$$

where $\mathcal{P}$ is the space of probability distributions. To this end, we first initialize the moments of all factor distributions of $q(\Psi, \Omega, \boldsymbol{\theta})$ appropriately and then iteratively optimize each of the factors in turn using the current estimates for all of the other factors. It can be shown that when keeping all other factors fixed the optimal distribution $q^*(\boldsymbol{a})$ satisfies

$$q^*(\boldsymbol{a}) \propto \exp\{\mathbb{E}_{-\boldsymbol{a}}[\log p(\Psi, \Omega, \boldsymbol{\theta}, \{\mathbf{K}_m\}_{m=1}^P, \mathbf{y})]\}, \quad (3)$$

where $\mathbb{E}_{-\boldsymbol{a}}$ denotes the expectation with respect to $p(\Psi, \Omega, \boldsymbol{\theta})$ over all variables except for $\boldsymbol{a}$, and $p(\Psi, \Omega, \boldsymbol{\theta}, \{\mathbf{K}_m\}_{m=1}^P, \mathbf{y})$ is the joint distribution of data and all variables, which has the following form

$$\begin{aligned} &p(\Psi, \Omega, \boldsymbol{\theta}, \{\mathbf{K}_m\}_{m=1}^P, \mathbf{y}) = \\ &\quad L(\mathbf{y}, \boldsymbol{\theta}|\mathbf{f}) p(\mathbf{f}|\boldsymbol{e}, \mathbf{G}, b, c) p(\mathbf{G}|\boldsymbol{a}, \{\mathbf{K}_m\}_{m=1}^P, \upsilon) \\ &\quad \cdot p(\boldsymbol{a}|\boldsymbol{\lambda}) p(\boldsymbol{e}|\boldsymbol{\omega}) p(b|\gamma) p(\boldsymbol{\lambda}|\boldsymbol{\xi}) p(\boldsymbol{\xi}|\phi) p(\phi|\eta) p(\eta) \\ &\quad \cdot p(\boldsymbol{\omega}|\boldsymbol{\varphi}) p(\boldsymbol{\varphi}|\rho) p(\rho|\tau) p(\tau) p(\upsilon) p(\gamma) p(c). \end{aligned}$$

Expanding the right side of (3) and ignoring the terms unrelated to $\boldsymbol{a}$, we can further get[1]

$$\begin{aligned} q^*(\boldsymbol{a}) &\propto \exp\{\mathbb{E}_{-\boldsymbol{a}}[\log p(\mathbf{G}|\boldsymbol{a}, \{\mathbf{K}_m\}_{m=1}^P, \upsilon) + \log p(\boldsymbol{a}|\boldsymbol{\lambda})]\} \\ &= \mathcal{N}\left(\boldsymbol{\Sigma}_{\boldsymbol{a}} \left(\langle\upsilon\rangle \sum_{m=1}^P \mathbf{K}_m \langle \boldsymbol{g}_{m\cdot}^\top \rangle\right), \boldsymbol{\Sigma}_{\boldsymbol{a}}\right), \end{aligned}$$

where $\boldsymbol{\Sigma}_{\boldsymbol{a}} = \left(\langle\upsilon\rangle \sum_{m=1}^P \mathbf{K}_m \mathbf{K}_m^\top + \langle \Lambda_{\boldsymbol{\lambda}}^{-1} \rangle\right)^{-1}$ and $\Lambda_{\boldsymbol{\lambda}}^{-1} = \mathrm{diag}(\boldsymbol{\lambda})^{-1}$. Similarly, we can also get

$$q^*(b, \boldsymbol{e}) = \mathcal{N}\left(\Sigma_{(b,\boldsymbol{e})} \langle c \rangle \left[\mathbf{1}, \langle \mathbf{G} \rangle^\top\right]^\top \langle \mathbf{f} \rangle, \Sigma_{(b,\boldsymbol{e})}\right),$$

$$q^*(\mathbf{G}) = \prod_{i=1}^N \mathcal{N}\left(\mu_{\boldsymbol{g}_{\cdot i}}, \left(\langle c \rangle \langle \boldsymbol{e}\boldsymbol{e}^\top \rangle + \langle \upsilon \rangle \mathbf{I}\right)^{-1}\right),$$

$$q^*(\mathbf{f}_+) = \mathcal{N}\Big(\mu_{\mathbf{f}_+}, \Big(\sum_{j=1}^{N_-} \langle \Lambda_{\boldsymbol{\theta}_{\cdot j}}^{-1} \rangle + \langle c \rangle \mathbf{I}\Big)^{-1}\Big),$$

$$q^*(\mathbf{f}_-) = \mathcal{N}\Big(\mu_{\mathbf{f}_-}, \Big(\sum_{i=1}^{N_+} \langle \Lambda_{\boldsymbol{\theta}_{i\cdot}}^{-1} \rangle + \langle c \rangle \mathbf{I}\Big)^{-1}\Big),$$

$$q^*(\boldsymbol{\theta}) = \prod_{i=1}^{N_+} \prod_{j=1}^{N_-} \mathcal{GIG}\Big(\frac{1}{2}, 1, \langle (1 - f_i + f_j)^2 \rangle\Big),$$

where

$$\Sigma_{(b,\boldsymbol{e})} = \begin{bmatrix} \langle c \rangle N + \langle \gamma \rangle, & \langle c \rangle \mathbf{1}^\top \langle \mathbf{G}^\top \rangle \\ \langle c \rangle \langle \mathbf{G} \rangle \mathbf{1}, & \langle c \rangle \langle \mathbf{G}\mathbf{G}^\top \rangle + \langle \Lambda_{\boldsymbol{\omega}}^{-1} \rangle \end{bmatrix}^{-1},$$

$$\mu_{\boldsymbol{g}_{\cdot i}} = \Sigma_{\boldsymbol{g}_{\cdot i}} (\langle c \rangle (\langle f_i \rangle \langle \boldsymbol{e} \rangle - \langle b\boldsymbol{e} \rangle) + \langle \upsilon \rangle \mathbf{K}_i^\top \langle \boldsymbol{a} \rangle),$$

$$\mu_{\mathbf{f}_+} = \Sigma_{\mathbf{f}_+} (N_- \mathbf{1} + \langle \boldsymbol{\theta}^{-1} \rangle (\langle \mathbf{f}_- \rangle + \mathbf{1}) + \langle c \rangle (\langle \mathbf{G}_+^\top \rangle \langle \boldsymbol{e} \rangle + \langle b \rangle \mathbf{1})),$$

$$\mu_{\mathbf{f}_-} = \Sigma_{\mathbf{f}_-} (\langle \boldsymbol{\theta}^{-1} \rangle^\top (\langle \mathbf{f}_+ \rangle - \mathbf{1}) - N_+ \mathbf{1} + \langle c \rangle (\langle \mathbf{G}_-^\top \rangle \langle \boldsymbol{e} \rangle + \langle b \rangle \mathbf{1})).$$

[1]Here $\langle \cdot \rangle$ means the expectation operator, e.g., $\langle \upsilon \rangle$ means the expectation of υ over its current optimal variational distribution.

Note that, $\boldsymbol{\theta}^{-1}$ is the element-wise inversion of $\boldsymbol{\theta}$, and $\mathbf{G}_+$ ($\mathbf{G}_-$) denotes the part of $\mathbf{G}$ corresponding to positive (negative) instances. The detailed derivations of the above equations together with the equations for all other variables can be found in the Supplement Section B.

4 ONLINE B^{2}RMK LEARNING

The above batch B^{2}RMK model still suffers from efficiency problems, e.g., huge memory consumption in dealing with large data sets and time-consuming re-training when the data appears sequentially. Online Passive-Aggressive (PA) [Crammer *et al.*, 2006] learning is a principled way to solve such problems. However, it is less explored under the Bayesian framework. Here, we present a fixed budget strategy to conduct online multiple kernel PA learning in Bayesian manner. Our method can be seen as an extension of the linear framework in [Shi and Zhu, 2014], which studied online max-margin topic models.

In the online setting, what we truly care about is how to update the current model, on the arrival of a new data. So, assume we already have an existing B^{2}RMK model at the t-th trial. In the updating of the kernel-based ranking decision function, the main problem is to compute the Gram matrix of pairwise kernel evaluations between the historical instances and new arriving instance, and we have to store all the received historical instances, making it impractical for large-scale online learning tasks. We address this challenge by maintaining only a small number of received historical instances. Specifically, we define two buffers B_t^+ and B_t^- of size $|B_t^+|$ and $|B_t^-|$, for storing the learned important positive and negative support vectors at the t-th trial, respectively. These support vectors in B_t^+ and B_t^- are essential for constructing the ranking decision function at the $(t+1)$-th trial, thus are expected to keep track of the global information of the decision boundary. To this end, existing online MKL methods [Luo *et al.*, 2010; Hoi *et al.*, 2013] typically assume an infinite buffer. However, their naive treatment of storing all received historical instances, requires more and more memory and computation when the data arrives sequentially. On the contrary, we control the computational complexity of our Online B^{2}RMK (OB^{2}RMK) model by fixing the buffer size to an appropriate value, assuming that the performance of OB^{2}RMK increases gradually with the increase of the buffer and it will be saturated when the buffer is large enough. As will be seen in the experimental section, such assumptions are well validated. Overall, the proposed OB^{2}RMK consists of two key modules, i.e., buffer update and model update.

4.1 UPDATE BUFFER

How to maintain the buffers with the most informative support vectors to get better generalization performance is a

key challenge. Traditionally, First-In-First-Out (FIFO) and Reservoir Sampling (RS) are two typical stream oblivious policies to update the buffers and have demonstrated their effectiveness in online linear AUC maximization [Zhao *et al.*, 2011a]. However, FIFO and RS will cast off the important support vectors, thus will degrade the performance of the online learning algorithms. Compared with linear algorithms, this adverse impact will be more intense in the kernel-based algorithms which work with the Gram matrix of pairwise kernel evaluations [Yang *et al.*, 2013; Hu *et al.*, 2015]. In the following, we will propose a novel strategy to update the buffers for OB^{2}RMK, aiming to preserve the most important support vectors.

For the incoming new instance $(x_\star, y_\star)$, instead of counting its pairwise losses with all support vectors in the opposite buffer which is easily affected by outliers, we only count the pairwise losses with its k-nearest opposite support vectors X_k. This allows us to utilize the local information around $x_\star$ and improve the robustness of the ranking decision function [Hu *et al.*, 2015].

When the buffers B_t^+ and B_t^- are not full, the incoming new instance will be put into one of the buffers directly, thus the pairwise hinge loss at the $(t+1)$-th trial is

$$\ell(f; x_\star, X_k) = \begin{cases} \sum_{j=1}^{k} \max\left(0, 1 - f(x_\star) + f(x_j^-)\right), & y_\star = 1, \\ \sum_{i=1}^{k} \max\left(0, 1 - f(x_i^+) + f(x_\star)\right), & y_\star = -1. \end{cases}$$

When either buffer is full, one of the instances in this buffer should be discarded, to accommodate the incoming new instance. Recall that the $\mathcal{TPBN}$ prior concentrates at zero, thus a positive (negative) support vector x_i is likely to have a positive (negative) weight a_i to attain a discriminative decision function. Supposing B_t^+ (B_t^-) is full, we first search for a support vector x_r which has the smallest (largest) weight in the buffer. Instead of discarding x_r directly, we put x_r and $x_\star$ together to count the pairwise losses between them and X_k. This compensation scheme guarantees the useful information of x_r can be fully utilized before it is discarded. Hence, the pairwise hinge loss at the $(t+1)$-th trial is

$$\ell(f; x_\star, x_r, X_k) = \begin{cases} \sum_{j=1}^{k} \{\max(0, 1 - f(x_\star) + f(x_j^-)) \\ \qquad + \max(0, 1 - f(x_r) + f(x_j^-))\}, & y_\star = 1, \\ \sum_{i=1}^{k} \{\max(0, 1 - f(x_i^+) + f(x_\star)) \\ \qquad + \max(0, 1 - f(x_i^+) + f(x_r))\}, & y_\star = -1. \end{cases}$$

Concisely, the pairwise hinge loss at the $(t+1)$-th trial can be uniformly written as

$$\ell(f; x_\star, B_t) = \sum_{i=1}^{\tilde{N}_+} \sum_{j=1}^{\tilde{N}_-} \max\left(0, 1 - f(x_i^+) + f(x_j^-)\right),$$

where $B_t = B_t^+ \cup B_t^-$, and $\tilde{N}_+$ and $\tilde{N}_-$ are the number of positive and negative instances, respectively, in counting the pairwise hinge loss at the $(t+1)$-th trial. For example, we have $\tilde{N}_+ = 1, \tilde{N}_- = k$ when $y_\star = 1$ and B_t^+ was not full (i.e., without x_r), and $\tilde{N}_+ = 2, \tilde{N}_- = k$ when $y_\star = 1$ and B_t^+ was full (i.e., with x_r).

4.2 UPDATE MODEL

In online B^{2}RMK, the model parameters $\boldsymbol{a}$, b and $\boldsymbol{e}$ should vary with t, to update the ranking decision function. Note that the dimensionality of $\boldsymbol{a}$ will increase until the buffers are full, whereas b and $\boldsymbol{e}$ are always fixed-size.

4.2.1 Priors

Let $p_t(\boldsymbol{a}_t, b, \boldsymbol{e})$ denote the posterior distribution of $(\boldsymbol{a}, b, \boldsymbol{e})$ at the t-th trial, which will become a prior distribution at the $(t+1)$-th trial. Assuming $a_\star$ is the corresponding weight of $x_\star$ at the $(t+1)$-th trial. After imposing a $\mathcal{TPBN}$ prior on $a_\star$, i.e., $a_\star|\alpha_{a_\star}, \beta_{a_\star}, \phi_\star \sim \mathcal{TPBN}(a_\star|\alpha_{a_\star}, \beta_{a_\star}, \phi_\star)$, we have the hierarchical priors: $a_\star|\lambda_\star \sim \mathcal{N}(a_\star|0, \lambda_\star)$, $\lambda_\star|\xi_\star \sim \Gamma(\lambda_\star|\alpha_{a_\star}, \xi_\star), \xi_\star|\phi_\star \sim \Gamma(\xi_\star|\beta_{a_\star}, \phi_\star)$, $\phi_\star|\eta_\star \sim \Gamma(\phi_\star|1/2, \eta_\star)$ and $\eta_\star \sim \Gamma(\eta_\star|1/2, 1)$.

The prior distributions of $\mathbf{G}$, $\mathbf{f}_+$, $\mathbf{f}_-$, υ and c are omitted here as they are similar as in B^{2}RMK. Now let $\tilde{\Upsilon} = \{\alpha_{a_\star}, \beta_{a_\star}\}$, $\tilde{\Psi} = \{\lambda_\star, \xi_\star, \phi_\star, \eta_\star\}$ and $\tilde{\Omega} = \{\mathbf{f}_+, \mathbf{f}_-, \mathbf{G}, \upsilon, c\}$ for notational simplicity.

4.2.2 Posterior Updating

Since $\boldsymbol{a}, b, \boldsymbol{e}, \tilde{\Psi}$ and $\tilde{\Omega}$ actually are random variables, we have to average the loss $\ell(f; x_\star, B_t)$ over their joint distribution. Let $\mathbb{E}_{p(\boldsymbol{a}, b, \boldsymbol{e}, \tilde{\Psi}, \tilde{\Omega})}$ denote the expectation over $p(\boldsymbol{a}, b, \boldsymbol{e}, \tilde{\Psi}, \tilde{\Omega})$, then we define the following expected pairwise hinge loss at the $(t+1)$-th trial:

$$\mathcal{R}(p(\boldsymbol{a}, b, \boldsymbol{e}, \tilde{\Psi}, \tilde{\Omega})) = \sum_{i=1}^{\tilde{N}_+} \sum_{j=1}^{\tilde{N}_-} \mathbb{E}_{p(\boldsymbol{a}, b, \boldsymbol{e}, \tilde{\Psi}, \tilde{\Omega})}[\max(0, 1 - f_i + f_j)].$$

Note that, expected pairwise hinge loss is an upper bound of the pairwise hinge loss of the expected bipartite ranking model by Jensen's inequality, and is more convenient for our inference as shown below.

Now we can infer the new posterior distribution $p_{t+1}(\boldsymbol{a}, b, \boldsymbol{e})$ on the arrival of the new data $(x_\star, y_\star)$ by solving the following optimization problem:

$$\min_{p(\boldsymbol{a}, b, \boldsymbol{e}) \in \mathcal{P}} \mathrm{KL}\left(p(\boldsymbol{a}, b, \boldsymbol{e}) \| p_t(\boldsymbol{a}_t, b, \boldsymbol{e}) p(a_\star)\right) + 2C \cdot \mathcal{R}(p(\boldsymbol{a}, b, \boldsymbol{e}, \tilde{\Psi}, \tilde{\Omega})), \tag{4}$$

where C is a positive regularization parameter, the constant 2 is just for convenience, and $\mathcal{R}(p(\boldsymbol{a}, b, \boldsymbol{e}, \tilde{\Psi}, \tilde{\Omega}))$ is the expected pairwise hinge loss at the $(t+1)$-th trial. Intuitively,

we find a posterior distribution $p_{t+1}(\boldsymbol{a}, b, \boldsymbol{e})$ in the feasible zone that is not only close to $p_t(\boldsymbol{a}_t, b, \boldsymbol{e})p(a_\star)$, but also has small loss at the new trial. Note that $\boldsymbol{a} = \left[\boldsymbol{a}_t^\top \ a_\star\right]^\top$ at the $(t+1)$-th trial, but if B_t^+ (B_t^-) is already full and $y_\star = 1$ ($y_\star = -1$), the weight vector $\boldsymbol{a}_t$ has been truncated one dimension before the optimization (4) to accommodate $a_\star$, which corresponds to an updating of the buffer.

Directly optimizing (4) with $\mathcal{R}$ is difficult and inefficient. Here we regard

$$L_{t+1}(\mathbf{y}|\boldsymbol{a}, b, \boldsymbol{e}) = \prod_{i=1}^{\tilde{N}_+}\prod_{j=1}^{\tilde{N}_-} \exp\left\{-2C \cdot \mathbb{E}_{p(\tilde{\Psi},\tilde{\Omega})}\left[\max(0, 1 - f_i + f_j)\right]\right\},$$

as the unnormalized pseudo-likelihood of label vector $\mathbf{y}$, then (4) can be rewritten as

$$\min_{p(\boldsymbol{a},b,\boldsymbol{e})\in\mathcal{P}} \mathrm{KL}\left(p(\boldsymbol{a}, b, \boldsymbol{e})\|p_t(\boldsymbol{a}_t, b, \boldsymbol{e})p(a_\star)\right) - \mathbb{E}_{p(\boldsymbol{a},b,\boldsymbol{e})}[\log(L_{t+1}(\mathbf{y}|\boldsymbol{a}, b, \boldsymbol{e}))].$$

It is straightforward to verify that the solution of this information theoretical optimization problem is

$$p(\boldsymbol{a}, b, \boldsymbol{e}) = \frac{p_t(\boldsymbol{a}_t, b, \boldsymbol{e})p(a_\star)L_{t+1}(\mathbf{y}|\boldsymbol{a}, b, \boldsymbol{e})}{p(\mathbf{y}|\tilde{\Upsilon})},$$

where $p(\mathbf{y}|\tilde{\Upsilon})$ is the normalization constant, and the posterior at the t-th trail $p_t(\boldsymbol{a}_t, b, \boldsymbol{e})$ becomes a prior. Since it is hard to compute the expectation with respect to $p(\tilde{\Psi}, \tilde{\Omega})$ in $L_{t+1}(\mathbf{y}|\boldsymbol{a}, b, \boldsymbol{e})$, we can regard the posterior distribution $p(\boldsymbol{a}, b, \boldsymbol{e})$ as the marginal of a higher-dimensional distribution that includes variables $\tilde{\Psi}$ and $\tilde{\Omega}$. Note that $\tilde{\Psi}$ and $\tilde{\Omega}$ will play only an auxiliary role in the inference procedure.

Therefore, the higher-dimensional distribution of $\boldsymbol{a}, b, \boldsymbol{e}, \tilde{\Psi}$ and $\tilde{\Omega}$ at the $(t+1)$-th trial satisfies the following form:

$$p_{t+1}(\boldsymbol{a}, b, \boldsymbol{e}, \tilde{\Psi}, \tilde{\Omega}) = \frac{p_t(\boldsymbol{a}_t, b, \boldsymbol{e})p(a_\star)p(\tilde{\Psi}, \tilde{\Omega})L_{t+1}(\mathbf{y}|\mathbf{f})}{p(\mathbf{y}|\tilde{\Upsilon})},$$

$$L_{t+1}(\mathbf{y}|\mathbf{f}) = \prod_{i=1}^{\tilde{N}_+}\prod_{j=1}^{\tilde{N}_-} \exp\left(-2C \cdot \max(0, 1 - f_i + f_j)\right),$$

which is intractable to compute analytically due to the max function in it. Similar as in B^{2}RMK, where data augmentation presents an elegant way to deal with the challenging posterior inference problem, we transform $L_{t+1}(\mathbf{y}|\mathbf{f})$ into the product of $\tilde{N}_+ \cdot \tilde{N}_-$ location-scale mixtures of normals:

$$L_{t+1}(\mathbf{y}|\mathbf{f}) = \prod_{i=1}^{\tilde{N}_+}\prod_{j=1}^{\tilde{N}_-} \int_0^\infty \frac{\exp\left(\frac{(\theta_{ij}+C(1-f_i+f_j))^2}{-2\theta_{ij}}\right)}{\sqrt{2\pi\theta_{ij}}}\, d\theta_{ij}.$$

Now $L_{t+1}(\mathbf{y}|\mathbf{f})$ can also be seen as the marginal of the higher-dimensional distribution

$$L_{t+1}(\mathbf{y}, \boldsymbol{\theta}|\mathbf{f}) = \prod_{i=1}^{\tilde{N}_+}\prod_{j=1}^{\tilde{N}_-} \frac{\exp\left(\frac{(\theta_{ij}+C(1-f_i+f_j))^2}{-2\theta_{ij}}\right)}{\sqrt{2\pi\theta_{ij}}},$$

and the complete posterior distribution at the $(t+1)$-th trial can be expressed as

$$p_{t+1}(\boldsymbol{a}, b, \boldsymbol{e}, \tilde{\Psi}, \tilde{\Omega}, \boldsymbol{\theta}) = \frac{p_t(\boldsymbol{a}_t, b, \boldsymbol{e})p(a_\star)p(\tilde{\Psi}, \tilde{\Omega})L_{t+1}(\mathbf{y}, \boldsymbol{\theta}|\mathbf{f})}{p(\mathbf{y}|\tilde{\Upsilon})}.$$

4.2.3 Approximate Inference

We then make the variational approximate inference for $p_{t+1}(\boldsymbol{a}, b, \boldsymbol{e}, \tilde{\Psi}, \tilde{\Omega}, \boldsymbol{\theta})$. Specifically, assume there are a family of factorable and free-form variational distributions

$$q_{t+1}(\boldsymbol{a}, b, \boldsymbol{e}, \tilde{\Psi}, \tilde{\Omega}, \boldsymbol{\theta}) = q_{t+1}(\boldsymbol{a})q_{t+1}(b, \boldsymbol{e})q_{t+1}(\lambda_\star)q_{t+1}(\xi_\star)q_{t+1}(\phi_\star)q_{t+1}(\eta_\star) \cdot q_{t+1}(\upsilon)q_{t+1}(c)q_{t+1}(\mathbf{G})q_{t+1}(\mathbf{f}_+)q_{t+1}(\mathbf{f}_-)q_{t+1}(\boldsymbol{\theta}),$$

and the goal is to get the optimal one which minimizes $\mathrm{KL}(q_{t+1}(\boldsymbol{a}, b, \boldsymbol{e}, \tilde{\Psi}, \tilde{\Omega}, \boldsymbol{\theta})\|p_{t+1}(\boldsymbol{a}, b, \boldsymbol{e}, \tilde{\Psi}, \tilde{\Omega}, \boldsymbol{\theta}))$ between the approximating distribution and the target posterior. Following similar derivations as in B^{2}RMK model, we can infer the optimal distributions for all involved variables at the $(t+1)$-th trial, and the detailed descriptions are shown in Supplement Section C.

5 CONVERGENCE, COMPLEXITY AND PREDICTION

In both B^{2}RMK and OB^{2}RMK, the inference mechanism sequentially updates the approximate posterior distributions of the model parameters and the augmented parameters until convergence, which is guaranteed because the KL divergence is convex with respect to each of the factors.

Rather than addressing the high-dimensionality problem directly, our kernel-based approach always transforms the original data into kernel matrices whose sizes only depend on the number of instances. Meanwhile, our online algorithm can naturally overcome the memory consumption challenge posed by large training sets.

For B^{2}RMK, the computational complexity for transforming original data into kernel matrices $\{\mathbf{K}_m\}_{m=1}^P$ is $\mathrm{O}(N^2Pd)$, where d is the dimensionality of the original data. Note that $\sum_{m=1}^P \mathbf{K}_m\mathbf{K}_m^\top$ should be cached before starting inference to reduce the computation. The computational complexity for each iteration of the variational inference on training data is $\mathrm{O}(N^3 + P^3)$, where the matrix inversions for computing the covariances $\boldsymbol{\Sigma}_{\boldsymbol{a}}$ and $\boldsymbol{\Sigma}_{b,\boldsymbol{e}}$ consume $\mathrm{O}(N^3)$ and $\mathrm{O}(P^3)$ computation, respectively. Note that we can explore the symmetric positive semi-definite property to accelerate these matrix inversions.

For OB^{2}RMK, the computational complexity for the construction of P kernel matrices $\{\mathbf{K}_m\}_{m=1}^P$ at the t-th trial is $\mathrm{O}(|B_t|\tilde{N}Pd)$, where $|B_t| = |B_t^+| + |B_t^-|$ and $\tilde{N} = \tilde{N}_+ + \tilde{N}_-$. The computational complexity for each iteration of the variational inference during training is $\mathrm{O}(|B_t|^3 + P^3)$, which indicates that the computational complexity of OB^{2}RMK will be limited by the buffer size $|B_t|$ when P is given.

After obtaining the approximate posterior distributions $q^*(\boldsymbol{a})$ and $q^*(b, \boldsymbol{e})$, the ranking score for a new instance x_{new} can be calculated by:

$$f_{new} = \langle \boldsymbol{a} \rangle^\top \left(\sum_{m=1}^P \langle e_m \rangle \boldsymbol{k}_{m,new} \right) + \langle b \rangle.$$

6 EXPERIMENTS

In this section, we present extensive experimental results on real data sets to demonstrate the effectiveness of the proposed B^{2}RMK and OB^{2}RMK with fixed budgets. Specifically, we compare them with the following algorithms:

- Three batch learning algorithms, including SVM^{perf} for ordinal regression (SVM-OR) [Joachims, 2006], a bipartite ranking model with univariate logistic loss (Uni-Log) [Kotlowski *et al.*, 2011] and least square SVM (LS-SVM);
- Several state-of-the-art online AUC maximization algorithms, including one-pass AUC optimization (OPAUC) [Gao *et al.*, 2013], kernelized online imbalanced learning (KOIL) algorithm with fixed budgets [Hu *et al.*, 2015] and a bounded kernel-based online learning algorithm (Projectron++) [Orabona *et al.*, 2009];
- A latest online multiple kernel classification (OMKC) algorithm with infinite buffer size, which doesn't bound its model complexity, thus requires more and more computational resources when the data arrives sequentially [Hoi *et al.*, 2013].

6.1 EXPERIMENTAL TESTBED AND SETUP

We conduct experiments on a variety of data sets obtained from the UCI and the LIBSVM websites, as summarized in Table 1. To be consistent with previous studies [Gao *et al.*, 2013; Hu *et al.*, 2015], the features have been scaled to $[-1, 1]$ for all data sets, and multi-class data sets have been transformed into class-imbalanced binary ones. On each data set, we conduct four independent trials of 5-fold cross validation for all the algorithms, where four folds of the data are used for training while the rest for test in each trial. The averaged AUC value over these 20 runs is reported. For kernelized methods, we predefine a pool of 18 kernel functions on all features, including Gaussian kernels with 13 different widths $\{2^{-6}, 2^{-5}, \ldots, 2^6\}$ and polynomial kernels with 5 different degrees $\{1, 2, \ldots, 5\}$. Following [Gönen, 2012], all kernel matrices are normalized to have unit diagonal entries, i.e., spherical normalization, which can be done online. Note that our Bayesian multi-kernel approach infers model parameters automatically via posterior inference rather than time-consuming grid search. Though we still have to specify the hyper-parameters, they are fixed for all 15 data sets without re-adjustment on each data set.

Table 1: Details of the data sets used in our experiments.

Datasets	#instances	Datasets	#instances	Datasets	#instances
sonar	208	svmguide2	391	svmguide3	1243
glass	214	diabetes	768	segment	2310
heart	270	fourclass	862	satimage	4435
bupaliver	345	german	1000	spambase	4601
ionosphere	351	splice	1000	usps	9298

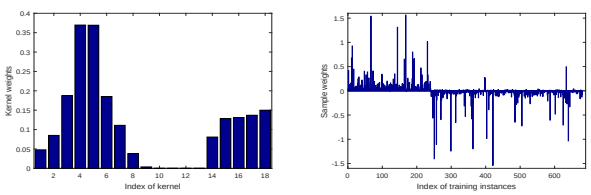
6.2 PERFORMANCE EVALUATION

6.2.1 Batch Learning

Though we mainly focus on online learning, we also briefly compare the proposed B^{2}RMK with three batch methods. We force sparsity at both sample-level and kernel-level by imposing sparsity-inducing priors ($\mathcal{TPBN}$) on the sample weights and kernel weights, respectively. Specifically, the hyper-parameters of the proposed B^{2}RMK are set to $(\alpha_a, \beta_a) = (\alpha_e, \beta_e) = (0.5, 0.5)$, $(\alpha_v, \beta_v) = (\alpha_c, \beta_c) = (10^{+2}, 10^{-2})$ and $(\alpha_\gamma, \beta_\gamma) = (10^{-2}, 10^{-2})$ for all data sets, while 5-fold cross validation is conducted on training sets to choose a better regularization parameter from $2^{[-10:10]}$ for other methods. The averaged AUC values of these batch algorithms are listed in Table 2, which show that B^{2}RMK performs significantly better than the competitors on 3 out of 5 data sets. Figure 1 illustrates the kernel weights and sample weights obtained by B^{2}RMK on 'fourclass' data set, which show that our model can effectively identify the key kernels and support vectors that are actually needed.

Table 2: Average AUC of batch learning methods.

Methods	diabetes	fourclass	german	splice	usps
SVM-OR	.832±.032	.831±.031	.793±.035	.924±.009	.963±.005
Uni-Log	**.833**±.032	.829±.031	**.799**±.034	.921±.011	.964±.004
LS-SVM	.832±.033	.831±.031	**.799**±.034	.925±.009	.963±.005
B^{2}RMK	.832±.028	**1.000**±.000	.795±.027	**.936**±.012	**.999**±.001



(a) (b)

Figure 1: (a) Kernel weights and (b) sample weights obtained by B^{2}RMK on 'fourclass' data set.

6.2.2 Online Learning

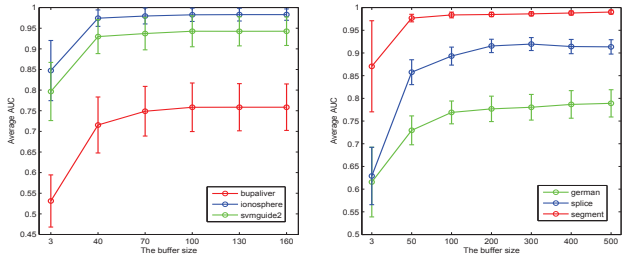
In real world online learning problems, there is usually very few training data at the beginning, and it is required to maintain a dynamic model with the sequentially arriving data. Since revisiting historical data is expensive for online algorithms, we cannot do cross validation on the entire training set. As such, we assume a half of one fold in the training data is available in a batch manner, and all algorithms tune their parameters on them.

For KOIL updated by RS++ or FIFO++, we set the learning rate $\eta = 0.01$ and select the penalty parameter C from $2^{[-10:10]}$ as suggested in their paper. For OPAUC, we select the learning rate parameter η and the regularization parameter λ from $2^{[-12:10]}$ and $2^{[-10:2]}$, respectively. For Projectron++, we select the parameter of projection difference η from $2^{[-10:10]}$. For OMKC, we adopt the deterministic updating and combination strategy, and the discount weight parameter β and smoothing parameter δ are fixed to 0.8 and 0.01, respectively. For OB^{2}RMK, we initially conduct B^{2}RMK on the half fold of training data to get the posterior distribution $p(\boldsymbol{a}, b, \boldsymbol{e})$. Then the hyper-parameters of OB^{2}RMK are set to $(\alpha_{a_\star}, \beta_{a_\star}) = (\alpha_v, \beta_v) = (\alpha_c, \beta_c) = (1, 1)$. The parameter k is fixed to 3, and the regularization parameter C is chosen from $\{0.1, 1, 3\}$. Furthermore, to provide KOIL and Projectron++ with multiple kernel information, we also run them with an unweighted combination of all the kernels as well as the best kernel among the pool of kernels. The symbols u and $*$ marked on the upper right corner are used for denoting them, respectively. All buffer sizes for KOIL and OB^{2}RMK are set to 100.

Table 3 summarizes the average AUC values of the compared online algorithms. We also list the performance of the proposed batch B^{2}RMK in the last column for reference. Several observations can be drawn as follows. First, by comparing the proposed OB^{2}RMK algorithm against the other online algorithms, we can find that the OB^{2}RMK performs considerably better on most data sets. In particular, the AUC values of OB^{2}RMK significantly surpass the baseline algorithms on some data sets. For example, on 'bupaliver', the AUC values for the baseline algorithms are lower than 71%, while OB^{2}RMK is able to achieve 75.8%. Secondly, online multiple kernel learning algorithms show better AUC performance than the single kernel and linear online learning algorithms on most data sets. This demonstrates the power of multiple kernel learning methods in classifying real-world data sets. Thirdly, OPAUC achieves fairly comparable or even better results than kernel-based algorithms on the data sets of 'svmguide2' and 'german'. We attribute this to the fact that a linear algorithm is enough to achieve good performance on some data sets, while the kernel-based algorithms may be easily affected by outliers. Finally, by examining the proposed OB^{2}RMK algorithm against the online multiple kernel learning algorithm OMKC$_{(D,D)}$ with infinite buffers, we can find that the OB^{2}RMK algorithm tends to outperform the OMKC$_{(D,D)}$ algorithm on most data sets. This encouraging result shows that the OB^{2}RMK algorithm with fixed buffer size is able to maintain an accurate sketch of historical training examples by exploring the compensation scheme.

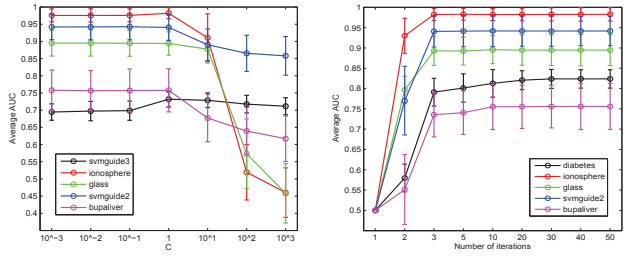
6.3 SENSITIVITY ANALYSIS

The default values of parameter C, the buffer size and the number of iterations for each online updating are 1, 100 and 30 respectively. When we study one of them through varying its values, the other two are fixed to their default values. First, from Figure 2 we observe that the performance of OB^{2}RMK increases gradually with the increase of the buffer size and it is saturated when the size is relatively large, which is consistent with the observations in [Zhao *et al.*, 2011a; Hu *et al.*, 2015]. Then, we can conclude from Figure 3 (a) that the regularization parameter C should not be too large, whereas there is a relatively broad range between $[10^{-3},\ 1]$ where OB^{2}RMK achieves good results. Finally, Figure 3 (b) shows that OB^{2}RMK converges rapidly, and typically stable results can be attained within 30 iterations.



(a) Small data sets (b) Large data sets

Figure 2: Average AUC of OB^{2}RMK vs. buffer size.



(a) (b)

Figure 3: Average AUC of OB^{2}RMK when different (a) parameter C and (b) number of iterations are used.

7 RELATED WORK

Online pairwise learning has gained increasing attention recently. In [Zhao *et al.*, 2011a], a buffer sampling based linear Online AUC maximization (OAM) model was proposed. In [Hu *et al.*, 2015], the Kernelized Online Imbal-

Table 3: Average AUC values (mean±std) on a variety of data sets. •/○ indicates that OB^{2}RMK is significantly better/worse than the corresponding method (pairwise t-tests at 95% significance level).

Datasets	OB^{2}RMK	KOIL$^u_{\mathrm{RS++}}$	KOIL$^*_{\mathrm{RS++}}$	KOIL$^u_{\mathrm{FIFO++}}$	KOIL$^*_{\mathrm{FIFO++}}$	OMKC$_{(D,D)}$	Projectron++^{u}	Projectron++*	OPAUC	B^{2}RMK
sonar	.915±.044	.865±.050•	.847±.051•	.865±.050•	.850±.049•	.916±.036	.713±.082•	.825±.071•	.813±.082•	**.942**±.034○
glass	.896±.035	.842±.050•	.784±.053•	.839±.052•	.785±.054•	.856±.071•	.582±.098•	.763±.066•	.822±.059•	**.898**±.034
heart	.895±.050	.893±.057	.895±.056	.894±.057	.895±.056	.854±.075•	.773±.056•	.806±.055•	.893±.051	**.905**±.053○
bupaliver	**.758**±.058	.703±.056•	.689±.062•	.707±.054•	.690±.059•	.654±.074•	.511±.020•	.587±.053•	.685±.068•	.755±.061
ionosphere	**.982**±.012	**.982**±.012	.971±.015•	**.982**±.012	.974±.018•	.969±.021•	.850±.041•	.898±.040•	.826±.075•	.980±.012
svmguide2	**.943**±.040	.939±.033	.924±.032•	.942±.032	.919±.038•	.921±.043•	.790±.065•	.836±.040•	.922±.035•	.942±.035
diabetes	.824±.028	.808±.039•	.789±.045•	.817±.036•	.798±.038•	.789±.022•	.574±.047•	.681±.041•	.742±.067•	**.832**±.028○
fourclass	**1.000**±.000	.987±.039•	.999±.001	.987±.039•	.999±.001	.999±.001	.815±.045•	.998±.002•	.830±.021•	**1.000**±.000
german	.768±.029	.760±.039•	.724±.046•	.761±.042•	.733±.041•	.723±.042•	.544±.030•	.625±.031•	.749±.043•	**.795**±.027○
splice	.894±.019	.886±.021•	.869±.018•	.887±.019•	.872±.024•	.906±.022○	.701±.030•	.797±.024•	.865±.020•	**.936**±.012○
svmguide3	.732±.023	.686±.039•	.620±.052•	.685±.051•	.625±.045•	.725±.034•	.503±.005•	.626±.031•	.715±.042•	**.807**±.026○
segment	.984±.005	.975±.006•	.959±.011•	.975±.006•	.957±.009•	.970±.006•	.789±.036•	.962±.010•	.895±.016•	**.998**±.001○
satimage	.978±.006	.958±.006•	.973±.007•	.956±.008•	.973±.007•	.974±.003	.903±.016•	.937±.006•	.848±.028•	**.991**±.002○
spambase	.944±.010	.938±.016•	.922±.017•	.938±.009•	.927±.015•	.958±.007○	.866±.021•	.893±.019•	.941±.009	**.983**±.004○
usps	.990±.003	.985±.004•	.988±.004	.984±.005•	.988±.003	.986±.002	.867±.030•	.976±.003•	.957±.003•	**.999**±.001○
win/tie/loss		12/3/0	12/3/0	12/3/0	12/3/0	9/4/2	15/0/0	15/0/0	13/2/0	0/5/10

anced Learning (KOIL) algorithm extended OAM to the nonlinear case with a predefined single kernel. Both of them provided regret bound based on pairwise hinge loss. [Ding *et al.*, 2015] extended OAM with adaptive gradient method which can exploit the knowledge of historical gradients. Besides, [Gao *et al.*, 2013] proposed a linear one-pass AUC optimization model which scans through the training data only once owing to the use of squared loss. A more recent squared loss based algorithm, named OPERA, was proposed in [Ying and Zhou, 2015], where a non-strongly convex objective was formulated in an unconstrained reproducing kernel Hilbert space. All the aforementioned methods seek point estimates of the decision function in a linear or single kernel space.

On the other hand, recent years witnessed the efforts on online extensions of multiple kernel learning due to its efficiency constrictions. Luo et al. [Luo *et al.*, 2010] first introduced an Online Multiclass Multi-kernel (OM2) classification algorithm with hinge loss. In [Hoi *et al.*, 2013], Hoi et al. proposed several Online Multiple Kernel Classification (OMKC) algorithms that aim to learn multiple kernelized classifiers and their linear combination simultaneously. In [Sahoo *et al.*, 2014], a family of Online Multiple Kernel Regression (OMKR) algorithms were proposed with sliding windows to address non-stationary times-series data. [Xia *et al.*, 2014] proposed an Online Multiple Kernel Similarity (OMKS) learning method, which learns a flexible nonlinear proximity function for visual search. These online MKL methods are effective for their specific applications, but generally involves several regularization parameters that are difficult to tune in real online scenario.

Bayesian learning is a principled way to infer the entire posterior distribution of various model parameters (e.g., kernel weights) from data automatically, providing the user a more simple way to accurately model data. In [Girolami and Rogers, 2005] and [Gönen, 2012], the authors studied Dirichlet and Gaussian priors for the kernel weights respectively. While both of them yield promising results, the former is difficult for inference. [Christoudias *et al.*, 2009] studied Bayesian localized MKL with Gaussian processes. So far, there is not only no online Bayesian MKL model but also no Bayesian multiple kernel AUC optimization model. Our margin-based model is the first combination of pairwise learning/AUC optimization with Bayesian MKL in the online setting.

8 CONCLUSION AND FUTURE WORK

We developed a Bayesian multi-kernel bipartite ranking model, which can circumvent the kernel selection problem by estimating a posterior distribution over the model weights. To make our model applicable to streaming data, we presented a kernelized online Bayesian PA learning framework by maintaining a variational approximation to the posterior. Furthermore, to efficiently deal with large-scale data, we maintained two fixed size buffers to control the number of support vectors while keeping track of the global information of the decision boundary. Extensive experimental studies confirmed the superiority of our Bayesian multi-kernel approach. In future, we plan to extend our model to deal with multi-source data with multi-task and transfer learning [Evgeniou *et al.*, 2005; Gönen and Margolin, 2014].

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61303059, 61573335, 91546122), Guangdong provincial science and technology plan projects (No. 2015B010109005), and the Science and Technology Funds of Guiyang (No. 201410012).

References

Shivani Agarwal and Dan Roth. Learnability of bipartite ranking functions. In *COLT*, pages 16–31, 2005.

Artin Armagan, Merlise Clyde, and David B Dunson. Generalized beta mixtures of gaussians. In *NIPS*, pages 523–531, 2011.

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

Andreas Christmann and Ding-Xuan Zhou. On the robustness of regularized pairwise learning methods based on kernels. *arXiv preprint arXiv:1510.03267*, 2015.

Mario Christoudias, Raquel Urtasun, and Trevor Darrell. Bayesian localized multiple kernel learning. *Univ. California Berkeley, Berkeley, CA*, 2009.

Stéphan Clémençon, Gabor Lugosi, and Nicolas Vayatis. Ranking and empirical minimization of u-statistics. *The Annals of Statistics*, pages 844–874, 2008.

Corinna Cortes and Mehryar Mohri. Auc optimization vs. error rate minimization. In *NIPS*, 2004.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *JMLR*, 7:551–585, 2006.

Yi Ding, Peilin Zhao, Steven CH Hoi, and Yew-Soon Ong. An adaptive gradient method for online auc maximization. In *AAAI*, 2015.

Theodoros Evgeniou, Charles A Micchelli, and Massimiliano Pontil. Learning multiple tasks with kernel methods. In *JMLR*, pages 615–637, 2005.

Wei Gao, Rong Jin, Shenghuo Zhu, and Zhi-Hua Zhou. One-pass AUC optimization. In *ICML*, 2013.

Mark Girolami and Simon Rogers. Hierarchic bayesian models for kernel learning. In *ICML*, 2005.

Mehmet Gönen and Ethem Alpaydın. Multiple kernel learning algorithms. *JMLR*, 12:2211–2268, 2011.

Mehmet Gönen and Adam A Margolin. Kernelized bayesian transfer learning. In *AAAI*, 2014.

Mehmet Gönen. Bayesian efficient multiple kernel learning. In *ICML*, pages 1–8, 2012.

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

Steven C. H. Hoi, Rong Jin, Peilin Zhao, and Tianbao Yang. Online multiple kernel classification. *Machine Learning*, 90(2):289–316, 2013.

Junjie Hu, Haiqin Yang, Irwin King, Michael R. Lyu, and Anthony Man-Cho So. Kernelized online imbalanced learning with fixed budgets. In *AAAI*, 2015.

Rong Jin, Steven CH Hoi, and Tianbao Yang. Online multiple kernel learning: Algorithms and mistake bounds. In *Algorithmic Learning Theory*, pages 390–404, 2010.

Thorsten Joachims. Training linear svms in linear time. In *SIGKDD*, pages 217–226, 2006.

Purushottam Kar, Bharath K Sriperumbudur, Prateek Jain, and Harish C Karnick. On the generalization ability of online learning algorithms for pairwise loss functions. In *ICML*, 2013.

Wojciech Kotlowski, Krzysztof J Dembczynski, and Eyke Huellermeier. Bipartite ranking through minimization of univariate loss. In *ICML*, 2011.

Jie Luo, Francesco Orabona, Marco Fornoni, Barbara Caputo, and Nicolo Cesa-Bianchi. Om-2: An online multi-class multi-kernel learning algorithm. In *CVPR Workshop on Online Learning for Computer Vision*, 2010.

Francesco Orabona, Joseph Keshet, and Barbara Caputo. Bounded kernel-based online learning. *JMLR*, 10:2643–2666, 2009.

Alain Rakotomamonjy, Francis Bach, Stéphane Canu, and Yves Grandvalet. Simplemkl. *JMLR*, 9:2491–2521, 2008.

Doyen Sahoo, Steven CH Hoi, and Bin Li. Online multiple kernel regression. In *SIGKDD*, pages 293–302, 2014.

Tianlin Shi and Jun Zhu. Online bayesian passive-aggressive learning. In *ICML*, 2014.

Martin A Tanner and Wing Hung Wong. The calculation of posterior distributions by data augmentation. *Journal of the American Statistical Association*, 82(398):528–540, 1987.

Yuyang Wang, Roni Khardon, Dmitry Pechyony, and Rosie Jones. Generalization bounds for online learning algorithms with pairwise loss functions. In *COLT*, 2012.

Hao Xia, Steven CH Hoi, Rong Jin, and Peilin Zhao. Online multiple kernel similarity learning for visual search. *IEEE Transactions on PAMI*, 36(3):536–549, 2014.

Haiqin Yang, Junjie Hu, Michael R Lyu, and Irwin King. Online imbalanced learning with kernels. In *NIPS Workshop on Big Learning*, 2013.

Yiming Ying and Ding-Xuan Zhou. Online pairwise learning algorithms with kernels. *arXiv preprint arXiv:1502.07229*, 2015.

Peilin Zhao and Steven CH Hoi. Bduol: Double updating online learning on a fixed budget. In *Machine Learning and Knowledge Discovery in Databases*, pages 810–826. Springer, 2012.

Peilin Zhao, Steven C. H. Hoi, Rong Jin, and Tianbao Yang. Online AUC maximization. In *ICML*, pages 233–240, 2011.

Peilin Zhao, Steven CH Hoi, and Rong Jin. Double updating online learning. *JMLR*, 12:1587–1615, 2011.