Online Multiple Kernel Learning: Algorithms and Mistake Bounds

Rong Jin[1], Steven C.H. Hoi[2], and Tianbao Yang[1]

[1]Department of Computer Science and Engineering,
Michigan State University, MI, 48824, USA
[2]School of Computer Engineering,
Nanyang Technological University, 639798, Singapore
[1]{rongjin,yangtia1}@cse.msu.edu, [2]chhoi@ntu.edu.sg

Abstract. *Online learning* and *kernel learning* are two active research topics in machine learning. Although each of them has been studied extensively, there is a limited effort in addressing the intersecting research. In this paper, we introduce a new research problem, termed **Online Multiple Kernel Learning** (OMKL), that aims to learn a kernel based prediction function from a pool of predefined kernels in an online learning fashion. OMKL is generally more challenging than typical online learning because both the kernel classifiers and their linear combination weights must be learned simultaneously. In this work, we consider two setups for OMKL, i.e. combining binary predictions or real-valued outputs from multiple kernel classifiers, and we propose both deterministic and stochastic approaches in the two setups for OMKL. The deterministic approach updates all kernel classifiers for every misclassified example, while the stochastic approach randomly chooses a classifier(s) for updating according to some sampling strategies. Mistake bounds are derived for all the proposed OMKL algorithms.

Keywords: On-line learning and relative loss bounds, Kernels

1 Introduction

In recent years, we have witnessed increasing interests on both *online learning* and *kernel learning*. Online learning refers to the learning process of answering a sequence of questions given the feedback of correct answers to previous questions and possibly some additional prior information [27]; Kernel learning aims to identify an effective kernel for a given learning task [20, 28, 12]. A well-known kernel learning method is Multiple Kernel Learning (MKL) [3, 28], that seeks the combination of multiple kernels in order to optimize the performance of kernel based learning methods (e.g., Support Vector Machines (SVM)).

Although kernel trick has been explored in online learning [10, 7], it is often assumed that kernel function is given apriori. In this work, we address a a new research problem, **Online Multiple Kernel Learning** (OMKL), which aims to simultaneously learn multiple kernel classifiers and their linear combinations from a pool of given kernels in an online fashion. Compared to the exiting methods for multiple kernel learning (see [17] and reference therein), online multiple kernel learning is computationally advantageous in that it only requires going through training examples once. We emphasize

that online multiple kernel learning is significantly more challenging than typical online learning because both the optimal kernel classifiers and their linear combinations have to be learned simultaneously in an online fashion.

In this paper, we consider two different setups for online multiple kernel learning. In the first setup, termed as *Online Multiple Kernel Learning by Predictions* or OMKL-P, its objective is to combine the *binary predictions* from multiple kernel classifiers. The second setup, termed as *Online Multiple Kernel Learning by Outputs* or OMKL-O, improves OMKL-P by combining the *real-valued outputs* from multiple kernel classifiers. Our online learning framework for multiple kernel learning is based on the combination of two types of online learning techniques: the *Perceptron* algorithm [25] that learns a classifier for a given kernel, and the *Hedge* algorithm [9] that linearly combines multiple classifiers. Based on the proposed framework, we present two types of approaches for each setup of OMKL , i.e., *deterministic* and *stochastic* approaches. The deterministic approach updates each kernel classifier for every misclassified example, while the stochastic approach chooses a subset of classifiers for updating based on certain sampling strategies. Mistake bounds are derived for all the proposed algorithms for online kernel learning.

The rest of this paper is organized as follows. Section 2 reviews the related work on both online learning and kernel learning. Section 3 overviews the problem of online multiple kernel learning. Section 4 presents the algorithms for Online Multiple Kernel Learning by Predictions and their mistake bounds; Section 5 presents algorithms for Online Multiple Kernel Learning by Outputs and their mistake bounds. Section 6 concludes this study with future work.

2 Related Work

Our work is closely related to both *online learning* and *kernel learning*. Below we briefly review the important work in both areas.

Extensive studies have been devoted to online learning for classification. Starting from Perceptron algorithm [1, 25, 23], a number of online classification algorithms have been proposed including the ROMMA algorithm [21], the ALMA algorithm [11], the MIRA Algorithm [8], the NORMA algorithm [16, 15], and the online Passive-Aggressive algorithms [7]. Several studies extended the Perceptron algorithm into a nonlinear classifier by the introduction of kernel functions [16, 10]. Although these algorithms are effective for nonlinear classification, they usually assume that appropriate kernel functions are given apriori, which limits their applications. Besides online classification, our work is also related to online prediction with expert advices [9, 22, 30]. The most well-known work is probably the Hedge algorithm [9], which was a direct generalization of Littlestone and Warmuth's Weighted Majority (WM) algorithm [22]. We refer readers to the book [4] for the in-depth discussion of this subject.

Kernel learning has been actively studied thanks to the great successes of kernel methods, such as support vector machines (SVM) [29, 26]. Recent studies of kernel learning focus on learning an effective kernel automatically from training data. Various algorithms have been proposed to learn parametric or semi-parametric kernels from labeled and/or unlabeled data. Example techniques include cluster kernels [5],

diffusion kernels [18], marginalized kernels [14], graph-based spectral kernel learning approaches [32, 13], non-parameric kernel learning [12, 6], and lower-rank kernel learning[19]. Among various approaches for kernel learning, Multiple Kernel Learning(MKL) [20], whose goal is to learn an optimal combination of multiple kernels, has emerged as a promising technique. A number of approaches have been proposed to solve the optimization problem related to MKL, including the conic combination approach via regular convex optimization [20], the semi-infinite linear program (SILP) approach [28], the Subgradient Descent approach [24], and the recent level method [31].

We emphasize that although both online learning and kernel learning have been extensively studied, little work has been done to address online kernel learning, especially online multiple kernel learning. To the best of our knowledge, this is the first theoretic study that addresses the OMKL problem.

3 Online Multiple Kernel Learning

Before presenting the algorithms for online multiple kernel learning, we first briefly describe the Multiple Kernel Learning (MKL) problem. Given a set of training examples $\mathcal{D}_T = \{(x_t, y_t), t = 1, \ldots, T\}$ where $y_t \in \{-1, +1\}, t = 1, \ldots, T$, and a collection of kernel functions $\mathcal{K}_m = \{\kappa_i : \mathcal{X} \times \mathcal{X} \to \mathbb{R}, i = 1, \ldots, m\}$, our goal is to identify the optimal combination of kernel functions, denoted by $\mathbf{u} = (u_1, \cdots, u_m)^\top$, that minimizes the margin classification error. It is cast as the following optimization problem:

$$\min_{\mathbf{u} \in \Delta} \min_{f \in \mathcal{H}_{\kappa_\mathbf{u}}} \frac{1}{2}\|f\|^2_{\mathcal{H}_{\kappa_\mathbf{u}}} + C \sum_{t=1}^{T} \ell(f(x_t), y_t) \tag{1}$$

where $\mathcal{H}_\kappa$ denotes the reproducing kernel Hilbert space defined by kernel κ, Δ denotes a simplex, i.e. $\Delta = \{\theta \in \mathbb{R}^m_+ | \sum_{i=1}^m \theta_i = 1\}$, and

$$\kappa_\mathbf{u}(\cdot, \cdot) = \sum_{j=1}^{m} u_j \kappa_j(\cdot, \cdot),\ \ell(f(x_t), y_t) = \max(0, 1 - y_t f(x_t))$$

It can also be cast into the following minimax problem:

$$\min_{\mathbf{u} \in \Delta} \max_{\alpha \in [0, C]^T} \left\{ \sum_{t=1}^{T} \alpha_t - \frac{1}{2}(\alpha \circ \mathbf{y})^\top \left(\sum_{i=1}^{m} u_i K^i \right) (\alpha \circ \mathbf{y}) \right\} \tag{2}$$

where $K^i \in \mathbb{R}^{n \times n}$ with $K^i_{j,l} = \kappa_i(x_j, x_l)$, $\mathbf{y} = (y_1, \cdots, y_T)^\top$, and $\circ$ is the element-wise product between two vectors.

The formulation for batch mode multiple kernel learning in (1) aims to learn a single function in the space of $\mathcal{H}_{\kappa_\mathbf{u}}$. It is well recognized that solving the optimization problem in (1) is in general computationally expensive. In this work, we aim to alleviate the computational difficulty of multiple kernel learning by online learning that only needs to scan through training examples once.

The following theorem allows us to simplify this problem by decomposing it into two separate tasks, i.e., learning (i) a classifier for each individual kernel, and (ii) weights that combine the outputs of individual kernel classifier to form the final prediction.

Theorem 1 *The optimization problem in (1) is equivalent to*

$$\min_{\mathbf{u}\in\Delta,\{f_i\in\mathcal{H}_{\kappa_i}\}_{i=1}^m} \sum_{i=1}^m \frac{u_i}{2}\|f_i\|_{\mathcal{H}_{\kappa_i}}^2 + C\sum_{t=1}^T \ell\left(\sum_{i=1}^m u_i f_i(x_t), y_t\right) \tag{3}$$

Proof. It is important to note that problem in (3) is non-convex, and therefore we can not directly deploy the standard approach to convert it into its dual form. In order to transform (3) into (1), we rewrite $\ell(z,y) = \max_{\alpha\in[0,1]} \alpha(1-yz)$, and rewrite (3) as follows

$$\min_{\mathbf{u}\in\Delta} \min_{\{f_i\in\mathcal{H}_{\kappa_i}\}_{i=1}^m} \max_{\alpha\in[0,C]^T} \sum_{i=1}^m \frac{u_i}{2}\|f_i\|_{\mathcal{H}_{\kappa_i}}^2 + \sum_{t=1}^T \alpha_t\left(1 - y_t\sum_{i=1}^m u_i f_i(x_t)\right)$$

Since the problem is convex in f_i and concave in α, we can switch the minimization of f_i with the maximization of α. By taking the minimization of f_i, we have

$$f_i(\cdot) = \sum_{t=1}^T \alpha_t y_t \kappa_i(x_t,\cdot), i = 1,\ldots,m$$

and the resulting optimization problem becomes

$$\min_{\mathbf{u}\in\Delta} \max_{\alpha\in[0,C]^T} \sum_{t=1}^T \alpha_t - \sum_{i=1}^m \frac{u_i}{2}(\alpha\circ\mathbf{y})^\top K^i(\alpha\circ\mathbf{y}),$$

which is identical to the optimization problem of batch mode multiple kernel learning in (2).

Based on the results of the above theorem, our strategy toward online kernel learning is to simultaneously learn a set of kernel classifiers $f_i, i = 1,\ldots,m$ and their combination weighs $\mathbf{u}$. We consider two setups for Online Multiple Kernel Learning (OMKL). In the first setup, termed Online Multiple Kernel Learning by Predictions (OMKL-P), we simplify the problem by only considering combining the binary predictions from multiple kernel classifiers, i.e., $\hat{y} = \sum_{i=1}^m u_i \text{sign}(f_i(x))$. In the second setup, termed Online Multiple Kernel Learning by Outputs (OMKL-O), we learn to combine the real-valued outputs from multiple kernel classifiers, i.e. $\hat{y} = \sum_{i=1}^m u_i f_i(x)$. In the next two sections, we will discuss algorithms and theoretic properties for OMKL-P and OMKL-O, respectively.

For the convenience of analysis, throughout the paper, we assume $\kappa_i(x,x) \leq 1$ for all the kernel functions $\kappa_i(\cdot,\cdot)$ and for any example x. Below we summarize the notations that are used throughout this paper:

- $\mathcal{D}_T = \{(x_t, y_t), t = 1,\ldots,T\}$ denotes a sequence of T training examples. $\mathcal{K}_m = \{\kappa_i : \mathcal{X}\times\mathcal{X}\to\mathbb{R}, i = 1,\ldots,m\}$ denotes a collection of m kernel functions.
- $\mathbf{f}_t(\cdot) = (f_1^t(\cdot),\cdots,f_m^t(\cdot))^\top$ denotes the collection of m classifiers in round t, where $f_i^t(\cdot)$ represents the classifier learned using the kernel $\kappa_i(\cdot,\cdot)$. For the purpose of presentation, we use $\mathbf{f}_t, f_i^t$ for short. $\mathbf{f}_t(x) = (f_1^t(x),\cdots,f_m^t(x))^\top$ denotes the real-valued outputs on example x by the m classifiers learned in round t, and $\text{sign}(\mathbf{f}_t(x)) = (\text{sign}(f_1^t(x)),\cdots,\text{sign}(f_m^t(x)))^\top$ denotes the binary predictions by the corresponding classifiers on example x.

- $\mathbf{w}_t = (w_1^t, \cdots, w_m^t)^\top$ denotes the weight vector for the m classifiers in round t; $W_t = \sum_{i=1}^m w_i^t$ represents the sum of weights in round t; $\mathbf{q}_t = (q_1^t, \ldots, q_m^t)^\top$ denotes the normalized weight vector, i.e. $q_i^t = w_i^t / W_t$.
- $\mathbf{z}_t = (z_1^t, \cdots, z_m^t)^\top$ denotes the indicator vector, where $z_i^t = I(y_t f_i^t(x_t) \leq 0)$ indicates if the ith kernel classifier makes a mistake on example x_t, where $I(C)$ outputs 1 when C is true and zero otherwise.
- $\mathbf{m}_t = (m_1^t, \cdots, m_m^t)^\top$ denotes the 0-1 random variable vector, where $m_i^t \in \{0, 1\}$ indicates if the ith kernel classifier is chosen for updating in round t.
- $\mathbf{p}_t = (p_1^t, \cdots, p_m^t)^\top$ denotes a probability vector, i.e. $p_i^t \in [0, 1]$.
- $\mathbf{a} \cdot \mathbf{b}$ denotes the dot-product between vector $\mathbf{a}$ and vector $\mathbf{b}$, $\mathbf{1}$ denotes a vector with all elements equal to 1, and $\mathbf{0}$ denotes a vector with all elements equal to 0.
- Multi_Sample($\mathbf{p}_t$) denotes a multinomial sampling process following the probability distribution $\mathbf{p}_t \in \Delta$ that outputs $i_t \in \{1, \ldots, m\}$. Bern_Sample(p_i^t) denotes a Bernoulli sampling process following the probability p_i^t that outputs a binary variable $m_i^t \in \{1, 0\}$.

4 Algorithms for Online Kernel Learning by Predictions (OMKL-P)

4.1 Deterministic Approaches(DA)

As already pointed out, the main challenge of OMKL is that both the kernel classifiers and their combination are unknown. The most straightforward approach is to learn a classifier for each individual kernel function and decide its combination weight based on the number of mistakes made by the kernel classifier. To this end, we combine the Perceptron algorithm and the Hedge algorithm together. In particular, for each kernel, the Perceptron algorithm is employed to learn a classifier, and the Hedge algorithm is used to update its weight. Algorithm 1 shows the deterministic algorithm for OMKL-P.

The theorem below shows the mistake bound for Algorithm 1. For the convenience of presentation, we define the optimal margin error for kernel $\kappa_i(\cdot, \cdot)$ with respect to a collection of training examples $\mathcal{D}_T$ as:

$$g(\kappa_i, \ell) = \min_{f \in \mathcal{H}_{\kappa_i}} \left(\|f\|_{\mathcal{H}_{\kappa_i}}^2 + 2 \sum_{t=1}^T \ell(f(x_t), y_t) \right)$$

Theorem 2 *After receiving a sequence of T training examples $\mathcal{D}_T$, the number of mistakes made by running Algorithm 1 is bounded as follows*

$$M = \sum_{t=1}^T I\left(\mathbf{q}_t \cdot \mathbf{z}_t \geq 0.5\right) \leq \frac{2 \ln(1/\beta)}{1 - \beta} \min_{1 \leq i \leq m} g(\kappa_i, \ell) + \frac{2 \ln m}{1 - \beta} \tag{4}$$

The proof for the theorem as well as the following theorems is sketched in the Appendix.

Note that in Algorithm 1 the weight for each individual kernel classifier is based on whether they classify the training example correctly. An alternative approach for updating the weights is to take into account the output values of $\{f_i^t\}_{i=1}^m$ by penalizing a

Algorithm 1 DA for OMKL-P (1)

INPUT:
- Kernels: $\mathcal{K}_m$
- Discount weight: $\beta \in (0, 1)$

Initialization: $\mathbf{f}_1 = \mathbf{0}$, $\mathbf{w}_1 = \mathbf{1}$
for $t = 1, 2, \ldots$ **do**
  Receive an instance x_t
  Predict: $\hat{y}_t = \text{sign}\left(\mathbf{q}_t \cdot \text{sign}(\mathbf{f}_t(x_t))\right)$
  Receive the class label y_t
  for $i = 1, 2, \ldots, m$ **do**
    Set $z_i^t = I(y_t f_i^t(x_t) \leq 0)$
    Update $w_i^{t+1} = w_i^t \beta^{z_i^t}$
    Update $f_i^{t+1} = f_i^t + z_i^t y_t \kappa_i(x_t, \cdot)$
  end for
end for

Algorithm 2 DA for OMKL-P (2)

INPUT:
- Kernels: $\mathcal{K}_m$
- Discount weight: $\beta \in (0, 1)$
- Max-misclassification level: $\gamma > 0$

Initialization: $\mathbf{f}_1 = \mathbf{0}, \mathbf{w}_1 = \mathbf{1}$
for $t = 1, 2, \ldots$ **do**
  Receive an instance x_t
  Predict: $\hat{y}_t = \text{sign}\left(\mathbf{q}_t \cdot \text{sign}(\mathbf{f}_t(x_t))\right)$
  Receive the class label y_t
  for $i = 1, 2, \ldots, m$ **do**
    Set $z_i^t = I(y_t f_i^t(x_t) \leq 0)$, $\nu_i^t = z_i^t(1/2 + \min(\gamma, -y_t f_i^t(x_t)))$
    Update $w_i^{t+1} = w_i^t \beta^{\nu_i(t)}$
    Update $f_i^{t+1} = f_i^t + z_i^t y_t \kappa_i(x_t, \cdot)$
  end for
end for

kernel classifier more if its degree of misclassification, measured by $-y_t f_i^t(x_t)$, is large. To this end, we present the second version of the deterministic approach for OMKL-P in Algorithm 2 that takes into account the value of $\{f_i^t\}_{i=1}^m$ when updating weights $\{w_i\}_{i=1}^m$. In this alternate algorithm, we introduce the parameter γ, which can be interpreted as the maximum level of misclassification. The key quantity introduced in Algorithm 2 is ν_i^t that measures the degree of misclassification by $1/2 + \min(\gamma, -y_t f_i^t(x))$. Note that we did not directly use $-y_t f_i^t(x)$ for updating weights $\{w_i\}_{i=1}^m$ because it is unbounded.

Theorem 3 *After receiving a sequence of T training examples $\mathcal{D}_T$, the number of mistakes made by Algorithm 2 is bounded as follows*

$$M = \sum_{t=1}^{T} I\left(\mathbf{q}_t \cdot \mathbf{z}_t \geq 0.5\right) \leq \frac{2(1/2+\gamma)\ln(1/\beta)}{1-\beta^{1/2+\gamma}} \min_{1 \leq i \leq m} g(\kappa_i, \ell) + \frac{4(1/2+\gamma)\ln m}{1-\beta^{1/2+\gamma}}$$

The proof is essentially similar to that of Theorem 2 with the modification that addresses variable $\nu_i(t)$ introduced in Algorithm 2.

One problem with Algorithm 1 is how to decide an appropriate value for β. A straightforward approach is to choose β that minimizes the mistake bound, leading to the following corollary.

Corollary 4 *By choosing $\beta = \frac{\sqrt{\min_{1 \leq i \leq m} \sum_{t=1}^{T} z_i^t}}{\sqrt{\min_{1 \leq i \leq m} \sum_{t=1}^{T} z_i^t} + \sqrt{\ln m}}$, we have the following mistake bound*

$$M \leq 2\left(\min_{1 \leq i \leq m} g(\kappa_i, \ell) + \ln m + 2\sqrt{\min_{1 \leq i \leq m} g(\kappa_i, \ell) \ln m}\right)$$

Proof. Followed by inequality in (4), we have

$$M \leq \frac{2}{\beta} \min_{1 \leq i \leq m} \sum_{t=1}^{T} z_i^t + \frac{2 \ln m}{1-\beta}$$

where we use $\ln(1/\beta) \leq 1/\beta - 1$. By setting the derivative of the above upper bound with respect to β to zero, and using the inequality $\sum_{t=1}^{T} z_i^t \leq g(\kappa_i, \ell)$ as shown in the appendix, we have the result.

Directly using the result in Corollary 4 is unpractical because it requires foreseeing the future to compute the quantity $\min_{1 \leq i \leq m} \sum_{t=1}^{T} z_i^t$. We resolve this problem by exploiting the doubling trick [4]. In particular, we divide the sequence $1, 2, \ldots, T$ into s segments:

$$[T_0 + 1 = 1, T_1], [T_1 + 1, T_2], \ldots, [T_{s-1} + 1, T_s = T]$$

such that (a) $\min_{1 \leq i \leq m} \sum_{t=T_k+1}^{T_{k+1}} z_i^t = 2^k$ for $k = 0, \ldots, s-2$, and (b) $\min_{1 \leq i \leq m} \sum_{t=T_{s-1}+1}^{T_s} z_i^t \leq 2^{s-1}$. Now, for each segment $[T_k + 1, T_{k+1}]$, we introduce a different β, denote by β_k, and set its value as

$$\beta_k = \frac{2^{k/2}}{\sqrt{\ln m} + 2^{k/2}}, \quad k = 0, \ldots, s-1 \tag{5}$$

The following theorem shows the mistake bound of Algorithm 1 with such β.

Theorem 5 *By running Algorithm 1 with β_k specified in (5), we have the following mistake bound*

$$\begin{aligned} M \leq & 2 \left(\min_{1 \leq i \leq m} g(\kappa_i, \ell) + \ln m + \frac{2}{\sqrt{2}-1} \sqrt{\min_{1 \leq i \leq m} g(\kappa_i, \ell) \ln m} \right) \\ & + 2 \left\lceil \log_2 \left(\min_{1 \leq i \leq m} g(\kappa_i, \ell) \right) \right\rceil \ln m \end{aligned}$$

where $\lceil x \rceil$ computes the smallest integer that is larger than or equal to x.

4.2 Stochastic Approaches

The analysis in the previous section allows us to bound the mistakes when classifying examples with a mixture of kernels. The main shortcoming with the deterministic approach is that in each round, all the kernel classifiers have to be checked and potentially updated if the training example is classified incorrectly. This could lead to a high computational cost when the number of kernels is large. In this section, we present stochastic approaches for online multiple kernel learning that explicitly address this challenge.

Single Update Approach(SUA) Algorithm 3 shows a stochastic algorithm for OMKL-P. In each round, instead of checking every kernel classifier, we sample a single kernel classifier according to the weights that are computed based on the number of mistakes made by individual kernel classifiers. However, it is important to note that rather than using q_i^t directly to sample one classifier to update, we add a smoothing term δ/m to the sampling probability p_i^t, This smoothing term guarantees a low bound for p_i^t, which ensures that each kernel classifier will be explored with at least certain amount of probability, which is similar to methods for the multi-arm bandit problem [2] to ensure the tradeoff between exploration and exploitation. The theorem below shows the mistake bound of Algorithm 3.

Theorem 6 *After receiving a sequence of T training examples $\mathcal{D}_T$, the expected number of mistakes made by Algorithm 3 is bounded as follows*

$$\mathrm{E}[M] = \mathrm{E}\left[\sum_{t=1}^{T} I\left(\mathbf{q}_t \cdot \mathbf{z}_t \geq 0.5\right)\right] \leq \frac{2m\ln(1/\beta)}{\delta(1-\beta)} \min_{1\leq i\leq m} g(\kappa_i, \ell) + \frac{2m\ln m}{\delta(1-\beta)}$$

Remark Comparing to the mistake bound in Theorem 2 by Algorithm 1, the mistake bound by Algorithm 3 is amplified by a factor of m/δ due to the stochastic procedure of updating one out of m kernel classifiers. The smoothing parameter δ essentially controls the tradeoff between efficacy and efficiency. To see this, we note that the bound for the expected number of mistakes is inversely proportional to δ; in contrast, the bound for the expected number of updates $\mathrm{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{m} m_i^t z_i^t\right] = \mathrm{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{m} p_i^t z_i^t\right] \leq (1-\delta)\mathrm{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{m} q_i^t z_i^t\right] + \delta T$ has a leading term δT when δ is large, which is proportional to δ.

Multiple Updates Approach(MUA) Compared with the deterministic approaches, the stochastic approach, i.e. the single update algorithm, does significantly improve the computational efficiency. However, one major problem with the single update algorithm is that in any round, only one single kernel classifier is selected for updating. As a result, the unselected but possibly effective kernel classifiers lose their opportunity for updating. This issue is particularly critical at the beginning of an online multiple kernel learning task where most individual kernel classifiers could perform poorly.

In order to make a better tradeoff between efficacy and efficiency, we develop another stochastic algorithm for online multiple kernel learning. The main idea of this new algorithm is to randomly choose multiple kernel classifiers for updating and predictions. Instead of choosing a kernel classifier from a multinomial distribution, the updating of each individual kernel classifier is determined by a separate Bernoulli distribution governed by p_i^t for each classifier. The detailed procedure is shown in Algorithm 4. The theorem below shows the mistake bound of the multiple updates algorithm.

Theorem 7 *After receiving a sequence of T training examples $\mathcal{D}_T$, the expected number of mistakes made by Algorithm 4 is bounded as follows*

$$\mathrm{E}[M] = \mathrm{E}\left[\sum_{t=1}^{T} I\left(\mathbf{q}_t \cdot \mathbf{z}_t \geq 0.5\right)\right] \leq \frac{2\ln(1/\beta)}{\delta(1-\beta)} \min_{1\leq i\leq m} g(\kappa_i, \ell) + \frac{2\ln m}{\delta(1-\beta)}$$

Algorithm 3 SUA for OMKL-P

INPUT:
- Kernels: $\mathcal{K}_m$
- Discount weight: $\beta \in (0,1)$
- Smoothing parameter: $\delta \in (0,1)$

Initialization: $\mathbf{f}_1 = 0, \mathbf{w}_1 = \mathbf{1}, \mathbf{p}_1 = \mathbf{1}/m$
for $t = 1, 2, \ldots$ **do**
  Receive an instance x_t
  Predict: $\hat{y}_t = \text{sign}\Big(\mathbf{q}_t \cdot \text{sign}(\mathbf{f}_t(x_t))\Big)$
  Receive the class label y_t
  i_t=Multi_Sample($\mathbf{p}_t$)
  for $i = 1, 2, \ldots, m$ **do**
    Set $m_i^t = I(i = i_t)$
    Set $z_i^t = I(y_t f_i^t(x_t) \leq 0)$
    Update $w_i^{t+1} = w_i^t \beta^{m_i^t z_i^t}$
    Update $f_i^{t+1} = f_i^t + m_i^t z_i^t y_t \kappa_i(x_t, \cdot)$
  end for
  Update $\mathbf{p}_{t+1} = (1-\delta)\mathbf{q}_{t+1} + \delta\mathbf{1}/m$
end for

Algorithm 4 MUA for OMKL-P

INPUT:
- Kernels: $\mathcal{K}_m$
- Discount weight: $\beta \in (0,1)$
- Smoothing parameter: $\delta \in (0,1)$

Initialization: $\mathbf{f}_1 = \mathbf{0}, \mathbf{w}_1 = \mathbf{1}, \mathbf{p}_1 = \mathbf{1}$
for $t = 1, 2, \ldots$ **do**
  Receive an instance x_t
  Predict: $\hat{y}_t = \text{sign}\Big(\mathbf{q}_t \cdot \text{sign}(\mathbf{f}_t(x_t))\Big)$
  Receive the class label y_t
  for $i = 1, 2, \ldots, m$ **do**
    Sample m_i^t = Bern_Sample(p_i^t)
    Set $z_i^t = I(y_t f_i^t(x_t) \leq 0)$
    Update $w_i^{t+1} = w_i^t \beta^{m_i^t z_i^t}$
    Update $f_i^{t+1} = f_i^t + m_i^t z_i^t y_t \kappa_i(x_t, \cdot)$
  end for
  Update $\mathbf{p}_{t+1} = (1-\delta)\mathbf{q}_{t+1} + \delta\mathbf{1}$
end for

Remark Compared to the mistake bound in Theorem 2 by Algorithm 1, the mistake bound by Algorithm 4 is amplified by a factor of $1/\delta$ due to the stochastic procedure. On the other hand, compared to the mistake bound of single update in Theorem 6, the mistake bound by Algorithm 4 is improved by a factor of m, mainly due to simultaneously updating multiple kernel classifiers in each round. The expected number of updates for multiple updates approach is bounded by $\mathrm{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{m} m_i^t z_i^t\right] = \mathrm{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{m} p_i^t z_i^t\right] \leq (1-\delta)\mathrm{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{m} q_i^t z_i^t\right] + \delta m T$, where the first term is discounted by a factor of m and the second term is amplified by a factor of m compared to that of single update approach.

5 Algorithms for Online Multiple Kernel Learning by Outputs (OMKL-O)

5.1 A Deterministic Approach

In the following analysis, we assume the functional norm of any classifier $f_i(\cdot)$ is bounded by R, i.e., $\|f_i\|_{\mathcal{H}_{\kappa_i}} \leq R$. We define domain Ω_{κ_i} as $\Omega_{\kappa_i} = \{f \in \mathcal{H}_{\kappa_i} : \|f\|_{\mathcal{H}_{\kappa_i}} \leq R\}$. Algorithm 5 shows the deterministic algorithm for OMKL-O. Compared to Algorithm 1, there are three key features of Algorithm 5. First, in step 11, the updated kernel classifier $f_i(\cdot)$ is projected into domain Ω_{κ_i} to ensure its norm is no more than R. This projection step is important for the proof of the mistake bound that will be shown later. Second, each individual kernel classifier is updated only when

the prediction of the combined classifier is incorrect i.e., $y_t\hat{y}_t \leq 0$. This is in contrast to the Algorithm 1, where each kernel classifier $f_i^t(\cdot)$ is updated when it misclassifies the training example x_t. This feature will make the proposed algorithm significantly more efficient than Algorithm 1. Finally, in step 9 of Algorithm 5, we update weights w_i^{t+1} based on the output $f_i^t(x_t)$. This is in contrast to Algorithm 1 where weights are updated only based on if the individual classifiers classify the example correctly.

Theorem 8 *After receiving a sequence of T training examples $\mathcal{D}_T$, the number of mistakes made by Algorithm 5 is bounded as follows if $R^2\ln(1/\beta) < 1$*

$$M \leq \frac{1}{1-R^2\ln(1/\beta)} \min_{\mathbf{u}\in\Delta,\{f_i\in\Omega_{\kappa_i}\}_{i=1}^m} g\left(\mathbf{u},\{f_i\}_{i=1}^m\right) + \frac{2\ln m}{(1-R^2\ln(1/\beta))\ln(1/\beta)}$$

where $g\left(\mathbf{u},\{f_i\}_{i=1}^m\right) = \sum_{i=1}^m u_i\|f_i\|^2_{\mathcal{H}_{\kappa_i}} + 2\sum_{t=1}^T \ell\left(\mathbf{u}\cdot\mathbf{f}(x_t), y_t\right)$.

Using the result in Theorem 1, we have the following corollary that bounds the number of mistakes of online kernel learning by the objective function used in the batch mode multiple kernel learning.

Corollary 9 *We have the following mistake bound for running Algorithm 5 if $R^2\ln(1/\beta) < 1$*

$$M \leq \frac{1}{1-R^2\ln(1/\beta)} \min_{\mathbf{u}\in\Delta, f\in\Omega_{\kappa(\mathbf{u})}} g\left(\kappa(\mathbf{u}),\ell\right) + \frac{2\ln m}{(1-R^2\ln(1/\beta))\ln(1/\beta)}$$

where $g\left(\kappa(\mathbf{u}),\ell\right) = \|f\|^2_{\mathcal{H}_{\kappa(\mathbf{u})}} + 2\sum_{t=1}^T \ell\left(f(x_t), y_t\right)$.

5.2 A Stochastic Approach

Finally, we present a stochastic strategy in Algorithm 6 for OMKL-O. In each round, we randomly sample one classifier to update by following the probability distribution $\mathbf{p}_t$. Similar to Algorithm 3, the probability distribution $\mathbf{p}_t$ is a mixture of the normalized weights for classifiers and a smoothing term δ/m. Different from Algorithm 5, the updating rule for Algorithm 6 has two additional factors, i.e. $\widetilde{m}_i^t$ which is non-zero for the chosen classifier and has expectation equal to 1, and the step size η which is essentially introduced to ensure a good mistake bound as shown in the following theorem.

Theorem 10 *After receiving a sequence of T training examples $\mathcal{D}_T$, the expected number of mistakes made by Algorithm 6 is bounded as follows*

$$\mathrm{E}[M] \leq \min_{\mathbf{u}\in\Delta,\{f_i\in\Omega_{\kappa_i}\}_{i=1}^m} \sum_{t=1}^T \ell\left(\mathbf{u}\cdot\mathbf{f}(x_t), y_t\right) + 2\sqrt{a(R,\beta,\delta)b(R,\beta,\delta)T}$$

where $a(R,\beta,\delta) = \dfrac{R^2}{2} + \dfrac{\ln m}{\ln(1/\beta)}$, $b(R,\beta,\delta) = \dfrac{\ln(1/\beta)R^2m^2}{2\delta^2} + \dfrac{m}{2\delta}$, and η is set to $\eta = \sqrt{\dfrac{a(R,\beta,\delta)}{b(R,\beta,\delta)}}$

Algorithm 5 DA for OMKL-O

INPUT:
- Kernels: $\mathcal{K}_m$
- Discount weight: $\beta \in (0,1)$
- Maximum functional norm: R

Initilization: $\mathbf{f}_1 = \mathbf{0}, \mathbf{w}_1 = \mathbf{1}$
for $t = 1, 2, \ldots$ **do**
  Receive an instance x_t
  Predict: $\hat{y}_t = \mathrm{sign}\Big(\mathbf{q}_t \cdot \mathbf{f}_t(x_t)\Big)$
  Receive the class label y_t
  if $\hat{y}_t y_t \leq 0$ **then**
    for $i = 1, 2, \ldots, m$ **do**
      Update $w_i^{t+1} = w_i^t \beta^{-y_t f_i^t(x_t)}$
      Update $\tilde{f}_i^{t+1} = f_i^t + y_t \kappa_i(x_t, \cdot)$
      Project $\tilde{f}_i^{t+1}$ into Ω_{κ_i} by

$$f_i^{t+1} = \tilde{f}_i^{t+1} / \max(1, \|\tilde{f}_i^{t+1}\|_{\mathcal{H}_{\kappa_i}} / R)$$

    end for
  end if
end for

Algorithm 6 SUA for OMKL-O

INPUT:
- $\mathcal{K}_m, \beta, R$ as in Algoirthm 5
- Smoothing parameter $\delta \in (0,1)$, and Step size: $\eta > 0$

Initialization: $\mathbf{f}_1 = \mathbf{0}, \mathbf{w}_1 = \mathbf{1}, \mathbf{p}_1 = \mathbf{1}/m$
for $t = 1, 2, \ldots$ **do**
  Receive an instance x_t
  Predict: $\hat{y}_t = \mathrm{sign}\Big(\mathbf{q}_t \cdot \mathbf{f}_t(x_t)\Big)$
  Receive the class label y_t
  if $\hat{y}_t y_t \leq 0$ **then**
    i_t=Multi_Sample($\mathbf{p}_t$)
    for $i = 1, 2, \ldots, m$ **do**
      Set $m_i^t = I(i = i_t)$, $\widetilde{m}_i^t = m_i^t / p_i^t$
      Update $w_i^{t+1} = w_i^t \beta^{-\eta \widetilde{m}_i^t y_t f_i^t(x_t)}$
      Update $\tilde{f}_i^{t+1} = f_i^t + \eta \widetilde{m}_i^t y_t \kappa_i(x_t, \cdot)$
      Project $\tilde{f}_i^{t+1}(x)$ into Ω_{κ_i}.
    end for
    Update $\mathbf{p}_t = (1-\delta)\mathbf{q}_t + \delta \mathbf{1}/m$
  end if
end for

6 Conclusions

This paper investigates a new research problem, online multiple kernel learning (OMKL), which aims to attack an online learning task by learning a kernel based prediction function from a pool of predefined kernels. We consider two setups for online kernel learning, online kernel learning by predictions that combines the binary predictions from multiple kernel classifiers and online kernel learning by outputs that combines the real-valued outputs from kernel classifiers. We proposed a framework for OMKL by learning a combination of multiple kernel classifiers from a pool of given kernel functions. We emphasize that OMKL is generally more challenging than typical online learning because both the kernel classifiers and their linear combination are unknown. To solve this challenge, we propose to combine two online learning algorithms, i.e., the Perceptron algorithm that learns a classifier for a given kernel, and the Hedge algorithm that combines classifiers by linear weighting. Based on this idea, we present two types of algorithms for OMKL, i.e., deterministic approaches and stochastic approaches. Theoretical bounds were derived for the proposed OMKL algorithms.

Acknowledgement

The work was supported in part by National Science Foundation (IIS-0643494), Office of Naval Research (N00014-09-1-0663), and Army Research Office (W911NF-09-1-0421). Any opinions, findings, and conclusions or recommendations expressed in this

material are those of the authors and do not necessarily reflect the views of NSF, ONR and ARO.

Appendix

Proof of Theorem 2

Proof. The proof essentially combines the proofs of the Perceptron algorithm [25] and the Hedge algorithm [9]. First, following the analysis in [9], we can easily have

$$\sum_{t=1}^{T} \mathbf{q}_t \cdot \mathbf{z}_t \leq \frac{\ln(1/\beta)}{1-\beta} \min_{1\leq i\leq m} \sum_{t=1}^{T} z_i^t + \frac{\ln m}{1-\beta}$$

Second, due to the convexity of ℓ and the updating rule for f_i, when $z_i^t = 1$ we have

$$\begin{aligned}\ell(f_i^t(x_t), y_t) - \ell(f(x_t), y_t) &\leq -y_t \langle f_i^t - f, z_i^t \kappa_i(x_t, \cdot)\rangle_{\mathcal{H}_{\kappa_i}} = -\langle f_i^t - f, f_i^{t+1} - f_i^t\rangle_{\mathcal{H}_{\kappa_i}} \\ &\leq \frac{1}{2}\left(\|f_i^t - f\|^2_{\mathcal{H}_{\kappa_i}} - \|f_i^{t+1} - f\|^2_{\mathcal{H}_{\kappa_i}} + z_i^t\right)\end{aligned}$$

Since $\ell(f_i^t(x_t), y_t) \geq z_i^t$, then $z_i^t \leq \|f_i^t - f\|^2_{\mathcal{H}_{\kappa_i}} - \|f_i^{t+1} - f\|^2_{\mathcal{H}_{\kappa_i}} + 2\ell(f(x_t), y_t)$. Taking summation on both sides, we have

$$\sum_{t=1}^{T} z_i^t \leq \min_{f\in\mathcal{H}_{\kappa_i}} \sum_{t=1}^{T}\left(\|f_i^t - f\|^2_{\mathcal{H}_{\kappa_i}} - \|f_i^{t+1} - f\|^2_{\mathcal{H}_{k_i}}\right) + 2\sum_{t=1}^{T}\ell(f(x_t), y_t) \leq g(\kappa_i, \ell)$$

Using the above inequality and noting that $M = \sum_{t=1}^{T} I\left(\mathbf{q}_t \cdot \mathbf{z}_t \geq 0.5\right) \leq 2\sum_{t=1}^{T} \mathbf{q}_t \cdot \mathbf{z}_t$, we have the result in the theorem.

Proof of Theorem 3

Proof. The proof can be constructed similarly to the proof for Theorem 2 by noting the following three differences. First, the updating rule for the weights can be written as $w_i^{t+1} = w_i^t(\beta^{1/2+\gamma})^{\nu_i^t/(1/2+\gamma)}$, where $\nu_i^t \leq 1/2 + \gamma$. Second, $\sum_i q_i^t \nu_i^t \geq \sum_i q_i^t z_i^t/2$. Third, we have $\ell(f_i^t(x_t), y_t) \geq \nu_i(t) + z_i(t)/2$, and therefore $\sum_{t=1}^{T} \nu_i^t \leq g(\kappa_i, \ell)/2$.

Proof of Theorem 5

Proof. We denote by M_k the number of mistakes made during the segment $[T_k + 1, T_{k+1}]$. It can be shown that the following equality

$$M_k \leq 2\left(\min_{1\leq i\leq m} \sum_{t=T_k+1}^{T_{k+1}} z_i^t + \ln m + 2\sqrt{\ln m}\, 2^{k/2}\right),$$

holds for any $k = 0, \ldots, s-1$. Taking summation over all M_k

$$M = \sum_{k=0}^{s-1} M_k \leq 2 \left(\sum_{k=0}^{s-1} \left[\min_{1 \leq i \leq m} \sum_{t=T_k+1}^{T_{k+1}} z_i^t + 22^{k/2} \sqrt{\ln m} \right] \right) + 2s \ln m$$

we can obtain the bound in the theorem by noting that $\sum_{k=0}^{s-1} \min_{1 \leq i \leq m} \sum_{t=T_k+1}^{T_k} z_i^t \leq \min_{1 \leq i \leq m} \sum_{t=1}^{T} z_i^t$, and $2^{s-1} - 1 \leq \min_{1 \leq i \leq m} \sum_{t=1}^{T} z_i^t \leq \min_{1 \leq i \leq m} g(\kappa_i, \ell)$.

Proof of Theorem 6

Proof. Similar to the proof for Theorem 2, we can prove

$$\sum_{t=1}^{T} \sum_{i=1}^{m} q_i^t m_i^t z_i^t \leq \frac{\ln(1/\beta)}{1-\beta} \sum_{t=1}^{T} m_i^t z_i^t + \frac{\ln m}{1-\beta}, \text{ and } \sum_{t=1}^{T} m_i^t z_i^t \leq g(\kappa_i, \ell)$$

Taking expectation on both sides, and noting that $\mathrm{E}[m_i^t] = p_i^t \geq \delta/m$, we have

$$\mathrm{E}\left(\sum_{t=1}^{T} \mathbf{q}_t \cdot \mathbf{z}_t \right) \leq \frac{m \ln(1/\beta)}{\delta(1-\beta)} \min_{1 \leq i \leq m} g(\kappa_i, \ell) + \frac{m \ln m}{\delta(1-\beta)}$$

Since $M \leq 2 \sum_{t=1}^{T} \mathbf{q}_t \cdot \mathbf{z}_t$, we have the result stated in the theorem.

Proof of Theorem 7

Proof. The proof can be duplicated similarly to the proof for Theorem 6, except for $p_i^t \geq \delta, i = 1, \ldots, m$ in this case.

Proof of Theorem 8

Proof. In the following analysis, we only consider the subset of iterations where the algorithm makes a mistake. By slightly abusing the notation, we denote by $1, 2, \ldots, M$ the trials where the examples are misclassified by Algorithm 5. For any combination weight $\mathbf{u} = (u_1, \cdots, u_m)^\top \in \Delta$ and any kernel classification function $f_i \in \Omega_{\kappa_i}, i = 1, \ldots, m$, we have

$$\begin{aligned} \ell\big(\mathbf{q}_t \cdot \mathbf{f}_t(x_t), y_t\big) - \ell\big(\mathbf{u} \cdot \mathbf{f}(x_t), y_t\big) &\leq g_t y_t \left(\mathbf{q}_t \cdot \mathbf{f}_t(x_t) - \mathbf{u} \cdot \mathbf{f}(x_t) \right) \\ &= g_t y_t \left(\mathbf{q}_t \cdot \mathbf{f}_t(x_t) - \mathbf{u} \cdot \mathbf{f}_t(x_t) \right) + g_t y_t \left(\mathbf{u} \cdot \mathbf{f}_t(x_t) - \mathbf{u} \cdot \mathbf{f}(x_t) \right) \end{aligned}$$

where $g_t = \left. \frac{\partial \max(0, 1-z)}{\partial z} \right|_{z = y_t \mathbf{q}_t \cdot \mathbf{f}_t(x_t)} = -1$ because examples are misclassified in these trials. For the first term, following the proof for Theorem 11.3 in [4], we can have

$$g_t y_t (\mathbf{q}_t - \mathbf{u}) \cdot \mathbf{f}_t(x_t) \leq \frac{\ln(1/\beta)}{2} R^2 + \frac{1}{\ln(1/\beta)} \left\{ \mathrm{KL}(\mathbf{u} \| \mathbf{q}_t) - \mathrm{KL}(\mathbf{u} \| \mathbf{q}_{t+1}) \right\}$$

For the second term, following the analysis in the proof for Theorem 2, we can have

$$g_t y_t \left(\mathbf{u} \cdot \mathbf{f}_t(x_t) - \mathbf{u} \cdot \mathbf{f}(x_t)\right) = \sum_{i=1}^{m} u_i \langle f_t^i - f^i, g_t y_t \kappa_i(x_t, \cdot)\rangle_{\mathcal{H}_{\kappa_i}}$$
$$\leq \frac{1}{2} + \sum_{i=1}^{m} \frac{u_i}{2} \left(\|f_i^t - f_i\|_{\mathcal{H}_{\kappa_i}}^2 - \|f_i^{t+1} - f_i\|_{\mathcal{H}_{\kappa_i}}^2 \right)$$

Combining the above result together, we arrive at

$$\sum_{t=1}^{M} \ell\left(\mathbf{q}_t \cdot \mathbf{f}_t(x_t), y_t\right) - \sum_{t=1}^{M} \ell\left(\mathbf{u} \cdot \mathbf{f}(x_t), y_t\right) \leq \frac{1}{2} \sum_{i=1}^{m} u_i \|f_i\|_{\mathcal{H}_{\kappa_i}}^2 + \frac{\ln m}{\ln(1/\beta)} + \frac{1}{2}(\ln(1/\beta)R^2 + 1)M$$

Since $\ell\left(\mathbf{q}_t \cdot \mathbf{f}_t(x_t), y_t\right) \geq 1$, we can have the result in the theorem.

Proof of Theorem 10

Proof. Similar to the proof for the Theorem 8, we have the following bound for the two terms

$$\mathrm{E}\left[g_t y_t \left(\mathbf{q}_t \cdot \mathbf{f}_t(x_t) - \mathbf{u} \cdot \mathbf{f}_t(x_t)\right)\right] = \frac{1}{\eta} \mathrm{E}\left[g_t y_t (\mathbf{q}_t - \mathbf{u}_t) \cdot \left(\eta \widetilde{m}_t \circ \mathbf{f}_t(x_t)\right)\right]$$
$$\leq \frac{1}{\eta \ln(1/\beta)} \mathrm{E}\left[\mathrm{KL}(\mathbf{u}\|\mathbf{q}(t)) - \mathrm{KL}(\mathbf{u}\|\mathbf{q}(t+1))\right] + \frac{\ln(1/\beta) R^2 \eta^2 m^2}{2\delta^2}$$

$$\mathrm{E}\left[g_t y_t \mathbf{u} \cdot \left(\mathbf{f}_t(x_t) - \mathbf{f}(x_t)\right)\right] = \mathrm{E} \sum_{i=1}^{m} \frac{u_i}{\eta} \left\langle f_i^t - f_i, \eta \widetilde{m}_i^t g_t y_t \kappa_i(x_t, \cdot)\right\rangle_{\mathcal{H}_{\kappa_i}}$$
$$\leq \frac{m\eta}{2\delta} + \mathrm{E}\left[\sum_{i=1}^{m} \frac{u_i}{2\eta} (\|f_i^t - f_i\|_{\mathcal{H}_{\kappa_i}}^2 - \|f_i^{t+1} - f_i\|_{\mathcal{H}_{\kappa_i}}^2)\right]$$

Combining the above results together, and setting η as in the theorem, we have the result in the theorem.

References

1. S. Agmon. The relaxation method for linear inequalities. *CJM*, 6(3):382–392, 1954.
2. P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. *SICOMP.*, 32(1), 2003.
3. F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In *ICML*, 2004.
4. N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. 2006.
5. O. Chapelle, J. Weston, and B. Schölkopf. Cluster kernels for semi-supervised learning. In *NIPS*, pages 585–592, 2002.
6. Y. Chen, M. R. Gupta, and B. Recht. Learning kernels from indefinite similarities. In *ICML*, pages 145–152, 2009.

7. K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *JMLR*, 7, 2006.
8. K. Crammer and Y. Singer. Ultraconservative online algorithms for multiclass problems. *JMLR*, 3, 2003.
9. Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *JCSS*, 55(1), 1997.
10. Y. Freund and R. E. Schapire. Large margin classification using the perceptron algorithm. *ML*, 37(3), 1999.
11. C. Gentile. A new approximate maximal margin classification algorithm. *JMLR*, 2, 2001.
12. S. C. Hoi, R. Jin, and M. R. Lyu. Learning non-parametric kernel matrices from pairwise constraints. In *ICML*, pages 361–368, 2007.
13. S. C. H. Hoi, M. R. Lyu, and E. Y. Chang. Learning the unified kernel machines for classification. In *KDD*, pages 187–196, 2006.
14. H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs. In *ICML*, pages 321–328, 2003.
15. J. Kivinen, A. Smola, and R. Williamson. Online learning with kernels. *IEEE Trans. on Sig. Proc.*, 52(8), 2004.
16. J. Kivinen, A. J. Smola, and R. C. Williamson. Online learning with kernels. In *NIPS*, pages 785–792, 2001.
17. M. Kloft, U. Rückert, and P. L. Bartlett. A unifying view of multiple kernel learning. *CoRR*, abs/1005.0437, 2010.
18. R. I. Kondor and J. D. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *ICML*, pages 315–322, 2002.
19. B. Kulis, M. Sustik, and I. Dhillon. Learning low-rank kernel matrices. In *ICML*, pages 505–512, 2006.
20. G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan. Learning the kernel matrix with semidefinite programming. *JMLR*, 5, 2004.
21. Y. Li and P. M. Long. The relaxed online maximum margin algorithm. *ML*, 46(1-3), 2002.
22. N. Littlestone and M. K. Warmuth. The weighted majority algorithm. In *FOCS*, 1989.
23. A. Novikoff. On convergence proofs on perceptrons. In *Proceedings of the Symposium on the Mathematical Theory of Automata*, volume XII, 1962.
24. A. Rakotomamonjy, F. R. Bach, S. Canu, and Y. Grandvalet. Simplemkl. *JMLR*, 11, 2008.
25. F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65, 1958.
26. B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. 2001.
27. S. Shalev-Shwartz. Online learning: Theory, algorithms, and applications. In *Ph.D thesis*, 2007.
28. S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large scale multiple kernel learning. *JMLR*, 7, 2006.
29. V. N. Vapnik. *Statistical Learning Theory*. Wiley, 1998.
30. V. Vovk. A game of prediction with expert advice. *J. Comput. Syst. Sci.*, 56(2), 1998.
31. Z. Xu, R. Jin, I. King, and M. R. Lyu. An extended level method for efficient multiple kernel learning. In *NIPS*, pages 1825–1832, 2008.
32. X. Zhu, J. S. Kandola, Z. Ghahramani, and J. D. Lafferty. Nonparametric transforms of graph kernels for semi-supervised learning. In *NIPS*, pages 1641–1648, 2004.