# ONLINE MULTI-KERNEL LEARNING WITH ORTHOGONAL RANDOM FEATURES

*Yanning Shen, Tianyi Chen, and Georgios B. Giannakis*

Dept. of ECE and DTC, University of Minnesota, Minneapolis, USA

## ABSTRACT

Kernel-based methods have well-appreciated performance in various nonlinear learning tasks. Most of them rely on a preselected kernel, whose prudent choice presumes task-specific prior information. To cope with this limitation, multi-kernel learning has gained popularity thanks to its flexibility in choosing kernels from a prescribed kernel dictionary. Leveraging the random feature approximation and its recent orthogonality-promoting variant, the present contribution develops an online multi-kernel learning scheme to infer the intended nonlinear function 'on the fly.' Performance analysis shows that the novel algorithm can afford sublinear regret. Numerical tests on real datasets are carried out to showcase the effectiveness of the proposed algorithms.

***Index Terms*—** Multi-kernel learning, random features, online learning, online regression.

## 1. INTRODUCTION

Function approximation emerges in various learning tasks such as regression, classification, and reinforcement learning [1, 2]. Kernel-based methods are powerful tools for nonlinear function approximation with strong theoretical guarantees. While most kernel methods utilize a pre-selected kernel, multi-kernel learning (MKL) approaches have attracted attention, thanks to their flexibility in selecting the task-specific kernel based on a prescribed kernel dictionary [3, 4].

In addition to the attractive generalization capability that kernel methods employ, several learning tasks are also expected to be performed in an *online* fashion. Such a need naturally arises when the data arrive sequentially, as in online spam detection [5], and time series prediction [6]; or, when the sheer volume of data makes it impossible to carry out data analytics in batch form [7]. This motivates well online kernel-based learning methods that inherit the merits of their batch counterparts, while at the same time allowing efficient online implementation. Taking a step further, the optimal function may itself change over time in *non-stationary* environments. This is the case when the function of interest e.g., represents the state in brain networks, or, captures the temporal processes propagating over time-varying networks. Tackling online kernel-based learning tasks in non-stationary and possibly adversarial environments remains a largely uncharted territory [7, 8].

In accordance with these needs and desiderata, the *primary goal* of this paper is an algorithmic pursuit of online multi-kernel learning in possibly adversarial environments, along with its performance guarantees. Major challenges arise due to: i) the well-known curse of dimensionality in kernel-based learning, since the size of the kernel matrix grows quadratically with the number of data [9], and the associated complexity to find even the *single* kernel-based predictor that is cubic of data size; and, ii) the difficulty of tracking unknown time-varying functions without future information. Regarding i), we apply the recent variance-reduced random feature approximation to circumvent the curse of dimensionality; while for ii), we propose a novel online algorithm which achieves sub-linear regret, on average "no-regret" relative to the best static counterpart.

## 2. PRELIMINARIES

Given samples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)\}_{t=1}^T$ with $\mathbf{x}_t \in \mathbb{R}^d$ and $y_t \in \mathbb{R}$, the function learning task is to find a function $f(\cdot)$ such that $y_n = f(\mathbf{x}_n) + e_n$, where $e_n$ denotes an error term representing noise or un-modeled dynamics. Suppose $f$ belongs to the reproducing kernel Hilbert space (RKHS)

$$\mathcal{H} := \{f | f(\mathbf{x}) = \sum_{t=1}^{\infty} \alpha_t \kappa(\mathbf{x}, \mathbf{x}_t)\} \tag{1}$$

where $\kappa(\mathbf{x}, \mathbf{x}_t) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a basis (so-termed kernel) function, which measures the similarity between $\mathbf{x}$ and $\mathbf{x}_t$. Different choices of $\kappa$ specify various bases. One of the popular ones is e.g., the Gaussian kernel $\kappa(\mathbf{x}, \mathbf{x}_t) := \exp[-(\mathbf{x} - \mathbf{x}_t)^2/(2\sigma^2)]$. A kernel is reproducing if it satisfies $\langle \kappa(\mathbf{x}, \mathbf{x}_1), \kappa(\mathbf{x}, \mathbf{x}_2) \rangle = \kappa(\mathbf{x}_1, \mathbf{x}_2)$, which in turn induces the RKHS norm $\|f\|_{\mathcal{H}}^2 = \sum_t \sum_{t'} \alpha_t \alpha_{t'} \kappa(\mathbf{x}_t, \mathbf{x}_{t'})$. Consider the optimization problem

$$\min_{f \in \mathcal{H}} \mathcal{L}(f) := \frac{1}{T} \sum_{t=1}^{T} \ell(f(\mathbf{x}_t), y_t) + \frac{\lambda}{2} \|f\|_{\mathcal{H}}^2 \tag{2}$$

where depending on the application, the loss function $\ell(\cdot, \cdot)$ can be selected to be, e.g., the least-squares cost, the logistic or hinge loss, and $\lambda > 0$ is a regularization parameter. Thanks to the representer theorem, the optimal solution of (2) admits

This work was supported by NSF 1500713, 1514056 and 1711471.

the finite-dimensional form, given by [9]

$$f(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t \kappa(\mathbf{x}, \mathbf{x}_t) \tag{3}$$

where $\{\alpha_t \in \mathbb{R}\}_{t=1}^T$ are combination coefficients. While the scalar $y_t$ is used here for notational brevity, coverage can be readily generalized to the vector form.

Note that (2) relies on two facts: i) a properly pre-selected kernel $\kappa$ is known; and ii) training data $\{\mathbf{x}_t, y_t\}_{t=1}^T$ are available. In the ensuing sections, an online MKL method will be proposed to select the optimal $\kappa$ as a convex combination of multiple kernels, when the data become available online.

## 3. ONLINE MKL WITH RANDOM FEATURES

In this section, we develop an online algorithm to simultaneously deal with kernel basis selections and multiple kernel combinations. Our algorithm leverages the orthogonal random feature techniques [10, 11], which we term **ra**ndom feature-based multi-**ker**nel (**Raker**) learning approach.

### 3.1. Kernel learning via random features

Kernel-based methods are challenged by the curse of dimensionality, due to the fact that the optimal kernel function depends on all the previous data samples [cf. (3)]. Unlike online kernel learning schemes that rely on budget maintenance strategies [12], the present section explores an alternative approach for kernel-based learning to render the subsequent online learning task scalable with the sample size. This approach relies on mapping the original data to random features (RFs), and then applying existing linear learning algorithms in this new feature space [10]. Specifically, given $\mathbf{x}_t$, the RF approach constructs a feature representation $\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t) \in \mathbb{R}^{2D}$, where $D \gg d$, $\mathbf{V} \in \mathbb{R}^{D\times d}$ is a random matrix that will be specified later, and $\mathbf{z}_{\mathbf{V}}(\mathbf{x})$ approximates the kernel by

$$k(\mathbf{x}_i, \mathbf{x}_j) \simeq \mathbf{z}_{\mathbf{V}}^\top(\mathbf{x}_i)\mathbf{z}_{\mathbf{V}}(\mathbf{x}_j). \tag{4}$$

Hence, the function in the corresponding RKHS can be approximated by (cf. (3))

$$f(\mathbf{x}) \simeq \sum_{t=1}^{T} \alpha_t \mathbf{z}_{\mathbf{V}}^\top(\mathbf{x}_t)\mathbf{z}_{\mathbf{V}}(\mathbf{x}) \tag{5}$$

where $\boldsymbol{\theta} := \sum_{t=1}^T \alpha_t \mathbf{z}_{\mathbf{V}}(\mathbf{x}_t)$ denotes the new weight vector that transforms the original kernel-based learning problem into a linear problem in the new $2D$-dimensional feature space, namely

$$f(\mathbf{x}) \simeq \boldsymbol{\theta}^\top \mathbf{z}_{\mathbf{V}}(\mathbf{x}). \tag{6}$$

To efficiently approximate the kernel function, we will confine our class to kernels that are shift invariant; that is, $\kappa(\mathbf{x}_1, \mathbf{x}_2) = \kappa(\boldsymbol{\delta})$ with $\boldsymbol{\delta} := \mathbf{x}_1 - \mathbf{x}_2$, and $\kappa(\mathbf{0}) = 1$. With the shift-invariant property, viewing the positive definite $\kappa(\boldsymbol{\delta})$ as the inverse Fourier transform of $\pi_\kappa(\mathbf{v})$, yields

$$\begin{aligned} \kappa(\mathbf{x}_1, \mathbf{x}_2) &= \int \pi_\kappa(\mathbf{v}) e^{j\mathbf{v}^\top(\mathbf{x}_1 - \mathbf{x}_2)} d\mathbf{v} \\ &= \mathbb{E}_{\mathbf{v}}\left[e^{j\mathbf{v}^\top \mathbf{x}_1} \cdot e^{-j\mathbf{v}^\top \mathbf{x}_2}\right] \end{aligned} \tag{7}$$

where the last equality follows by treating $\pi_\kappa(\mathbf{v})$ as the probability density function (pdf) of $\mathbf{v}$. Taking the Gaussian kernel as an example, where $\kappa(\mathbf{x}_1, \mathbf{x}_2) = \exp\left(\|\mathbf{x}_1 - \mathbf{x}_2\|_2^2/(2\sigma^2)\right)$, the corresponding pdf $\pi_\kappa(\mathbf{v}) = \mathcal{N}(\mathbf{0}, \sigma^{-2}\mathbf{I})$ [10]. Thus, plugging $e^{j\mathbf{v}^\top \mathbf{x}_1} = \cos(\mathbf{v}^\top \mathbf{x}_1) + j\sin(\mathbf{v}^\top \mathbf{x}_1)$ into (7) yields

$$\begin{aligned} \kappa(\mathbf{x}_1, \mathbf{x}_2) &= \mathbb{E}_{\mathbf{v}}\left[\cos(\mathbf{v}^\top \mathbf{x}_1)\cos(\mathbf{v}^\top \mathbf{x}_2) + \sin(\mathbf{v}^\top \mathbf{x}_1)\sin(\mathbf{v}^\top \mathbf{x}_2)\right] \\ &:= \mathbb{E}_{\mathbf{v}}\left[\mathbf{z}^\top(\mathbf{x}_1)\mathbf{z}(\mathbf{x}_2)\right] \end{aligned} \tag{8}$$

where $\mathbf{z}(\mathbf{x}) := [\sin(\mathbf{v}^\top \mathbf{x}), \cos(\mathbf{v}^\top \mathbf{x})]^\top$. Clearly, $D$ realizations of RF $\mathbf{z}(\mathbf{x})$ can be obtained by randomly sampling $\{\mathbf{v}_1, \ldots, \mathbf{v}_D\}$ from $\pi_\kappa(\mathbf{v})$, that is

$$\mathbf{z}_{\mathbf{V}}(\mathbf{x}) := \sqrt{\frac{1}{D}}[\sin(\mathbf{v}_1^\top \mathbf{x}), \cos(\mathbf{v}_1^\top \mathbf{x}), \ldots, \sin(\mathbf{v}_D^\top \mathbf{x}), \cos(\mathbf{v}_D^\top \mathbf{x})] \tag{9}$$

where the entries of $\mathbf{V} := [\mathbf{v}_1, \ldots, \mathbf{v}_D]^\top \in \mathbb{R}^{D\times d}$ are i.i.d. Gaussian. Thanks to (5), the nonparametric learning task is then approximated as a linear learning task in the Fourier feature space. Specifically, with the loss function [cf. (6)]

$$\ell_t(f(\mathbf{x}_t)) := \ell(f(\mathbf{x}_t), y_t) = \ell(\boldsymbol{\theta}^\top \mathbf{z}_{\mathbf{V}}(\mathbf{x}_t), y_t) \tag{10}$$

the online learning task becomes

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^{2D}} \sum_{t=1}^{T} \ell(\boldsymbol{\theta}^\top \mathbf{z}_{\mathbf{V}}(\mathbf{x}_t), y_t). \tag{11}$$

Upon obtaining a new datum $\mathbf{x}_t$, the representations of the data instance $\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t)$ can be generated via (9), and online gradient descent can be applied to refine the estimator

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \nabla \ell(\boldsymbol{\theta}_t^\top \mathbf{z}_{\mathbf{V}}(\mathbf{x}_t), y_t) \tag{12}$$

where $\{\eta_t\}$ is a sequence of stepsizes, and $\nabla \ell(\boldsymbol{\theta}_t^\top \mathbf{z}_{\mathbf{V}}(\mathbf{x}_t), y_t)$ is the gradient with respect to the weight $\boldsymbol{\theta}$ at $\boldsymbol{\theta} = \boldsymbol{\theta}_t$. The update (12) is still *a functional update* in that it is tantamount to updating the linear function $f_t(\cdot) = \boldsymbol{\theta}_t^\top \mathbf{z}_{\mathbf{V}}(\cdot)$.

**Variance-reduced RF.** Even if $\mathbf{z}_{\mathbf{V}}^\top(\mathbf{x}_1)\mathbf{z}_{\mathbf{V}}(\mathbf{x}_2)$ is an unbiased estimator of $\kappa$ [cf. (8)], the variance $\mathbf{z}_{\mathbf{V}}^\top(\mathbf{x}_1)\mathbf{z}_{\mathbf{V}}(\mathbf{x}_2)$ decays as $D$ increases. This explains why the RF vector dimension is chosen to satisfy $D \gg d$. [10]. Next, we will leverage a recent intriguing result from [11] to markedly reduce the variance of RF approximation by enforcing orthogonality on the rows of $\mathbf{V}$. For the original RF approach on a Gaussian kernel with bandwidth $\sigma^2$, recall that $\mathbf{V} = \sigma^{-1}\mathbf{G}$ in (9), where each entry of $\mathbf{G}$ follows a standardized Gaussian pdf. For the variance-

reduced RF method, with $D = d$, $\mathbf{V}_{\rm ORF}$ is formed as

$$\mathbf{V}_{\rm ORF} = \sigma^{-1}\mathbf{S}\mathbf{Q} \tag{13}$$

where $\mathbf{Q} \in \mathbb{R}^{d\times d}$ is a uniformly distributed random orthonormal matrix, and $\mathbf{S}$ denotes a diagonal matrix with diagonal entries drawn i.i.d. from a $\chi$ distribution with $d$ degrees of freedom. Matrix $\mathbf{S}$ is introduced to ensure unbiasedness of the kernel approximation [11]. With $D > d$, several weighted orthonormal matrices can be generated independently from (13), and concatenated to form $\mathbf{V}_{\rm ORF}$. It turns out that $\mathbf{z}_{\mathbf{V}_{\rm ORF}}^\top \mathbf{z}_{\mathbf{V}_{\rm ORF}}$ with $\mathbf{z}_{\mathbf{V}_{\rm ORF}}$ generated as in (9), and $\mathbf{V}_{\rm ORF}$ replacing $\mathbf{V}$, has much smaller variance [11]. Through such orthogonality-promoting RFs, the number of RFs needed to achieve a certain accuracy is markedly reduced.

### 3.2. Online MKL with random features

Here we preselect a dictionary of possible kernel functions, and then adaptively combine kernels in the dictionary [3]. Specifically, given a dictionary of kernels $\{\kappa_p\}_{p=1}^P$ and the RKHS $\mathcal{H}_p$ induced by $\kappa_p$, the solution of (2) is expressible in a separable form as [13]

$$f_t(\mathbf{x}) := \sum_{p=1}^P \bar{w}_{p,t} f_{p,t}(\mathbf{x}) \tag{14}$$

where $f_{p,t}(\mathbf{x})$ belongs to RKHS $\mathcal{H}_p$, for $p = 1, \ldots, P$, and $\bar{w}_{p,t} \in [0, 1]$ denotes the normalized weight for the $p$th kernel-based function estimator at time slot $t$.

To make use of the dictionary, $\{\tilde{w}_{p,t}\}_{p=1}^P$ should be learnt and adjusted in an online fashion. This task fits well the celebrated online learning paradigm, a.k.a., online prediction with expert advice [14]. Specifically, treating $\{\tilde{w}_{p,t}\}$ as an expert, we formulate the multi-kernel based learning problem as an online prediction task with expert advice. Upon obtaining a data sample, the un-normalized weights are updated according to the loss (called regret) incurred by each learner as

$$w_{p,t+1} = w_{p,t} \exp(-\eta \ell_t(f_{p,t}(\mathbf{x}_t))) \tag{15}$$

where $\eta \in (0, 1)$ is a chosen constant that controls the adaptation rate of $\{w_{p,t}\}$. Relative to $\{w_{p,t}\}$, the normalized weights in (14) are $\bar{w}_{p,t} := w_{p,t}/\sum_{p=1}^P w_{p,t}$, $\forall t$. Moreover, it can be observed that when $f_{p,t}$ incurs larger loss relative to other $f_{p',t}$ with $p' \neq p$ at time slot $t$, the corresponding combination weight decreases in the next time slot. In other words, a more accurate learner tends to play more important role in predicting the upcoming data.

However, relative to the generic expert advice problem [14], the difference here is that the kernel-based function estimator itself performs efficient online learning scheme for self-improvement. Indeed, the random feature approximation in Section 3.1 enables the efficient and scalable self-learning for each kernel-specific expert. Specifically, for the expert associated with kernel $p$, a feature representation $\mathbf{z}_p(\mathbf{x}_t)$ will be randomly generated from a kernel-specific distribution given datum $\mathbf{x}_t$ (cf. (9)), where we use $\mathbf{z}_p(\mathbf{x}_t) = \mathbf{z}_{\mathbf{V}_p}(\mathbf{x}_t)$ for notational simplicity; thus, each function estimator that is the advice of each expert at time $t$ can be written as

$$f_{p,t}(\mathbf{x}_t) \simeq \boldsymbol{\theta}_{p,t}^\top \mathbf{z}_p(\mathbf{x}_t) \tag{16}$$

where $\boldsymbol{\theta}_{p,t}$ is the parameter in (6) at time $t$ for kernel $p$. And similar to (12), the $p$th kernel, $\boldsymbol{\theta}_{p,t}$ is updated via

$$\boldsymbol{\theta}_{p,t+1} = \boldsymbol{\theta}_{p,t} - \eta \nabla \ell(\boldsymbol{\theta}_{p,t}^\top \mathbf{z}_p(\mathbf{x}_t), y_t) \tag{17}$$

where we use $\ell(f_{p,t}(\mathbf{x}_t), y_t) = \ell(\boldsymbol{\theta}_{p,t}^\top \mathbf{z}_p(\mathbf{x}_t), y_t)$. The Raker scheme is summarized as Algorithm 1.

**Algorithm 1** Raker: a random feature-based MKL approach

**Input:** Kernels $\kappa_p,\ p = 1, \ldots, P$, step size $\eta > 0$, and number of random features $D$.
**Initialization:** $\{\boldsymbol{\theta}_{p,1} = \mathbf{0}\}$.
**for** $t = 1, 2, \ldots, T$ **do**
  Receive a streaming datum $\mathbf{x}_t$.
  Construct $\mathbf{z}_p(\mathbf{x}_t)$ via (9) for $p = 1, \ldots, P$.
  Predict $f_t(\mathbf{x}_t)$ via (14) with $f_{p,t}(\mathbf{x}_t)$ in (16).
  **for** $p = 1, \ldots, P$ **do**
    Obtain loss $\ell(\boldsymbol{\theta}_{p,t}^\top \mathbf{z}_p(\mathbf{x}_t), y_t)$.
    Update $w_{p,t+1}$ via (15).
    Update $\boldsymbol{\theta}_{p,t+1}$ via (17).
  **end for**
**end for**

**Complexity.** At the $t$-th iteration of Algorithm 1, the memory required is fixed and of order $\mathcal{O}(D)$. Regarding computational overhead, the per-iteration computational complexity is of order $\mathcal{O}(D)$ compared with at least $\mathcal{O}(dt)$ for OMKL [15]; hence, the Raker algorithm is more computationally efficient than existing MKL [4] or OMKL. Along with the later numerical tests, it explains the effectiveness of Raker.

### 3.3. Regret analysis

We assume that the following conditions are satisfied.

**AS1**. Function $\ell(\boldsymbol{\theta}^\top \mathbf{z}_{\mathbf{V}}(\mathbf{x}_t), y_t)$ is convex in $\boldsymbol{\theta}$. If $\boldsymbol{\theta}$ is from a bounded set $\boldsymbol{\Theta}$, the loss and its gradient are bounded; i.e., $|\ell(\boldsymbol{\theta}^\top \mathbf{z}_{\mathbf{V}}(\mathbf{x}_t), y_t)| \leq 1$, and $\|\nabla \ell(\boldsymbol{\theta}^\top \mathbf{z}_{\mathbf{V}}(\mathbf{x}_t), y_t)\| \leq L$.

**AS2**. Each $\kappa_p$ is a shift-invariant kernel with $\kappa_p(\mathbf{x}_i, \mathbf{x}_j) \leq 1$, $\forall \mathbf{x}_i, \mathbf{x}_j$. Also $\|\mathbf{x}\| \leq 1$ and $\|f_{\mathcal{H}_p}^*\|_1 := \sum_{t=1}^T |\alpha_t^*| \leq C$.

AS1 enforces convexity of the loss, and ensures the losses are bounded, but also the gradient of the loss function is bounded, which is also called $L$-Lipschitz continuity that is common in OCO [16]. AS2 bounds the norm of the optimal function [17]. These bounds are assumed without loss of generality, whenever the losses are bounded. AS1-2 are typically satisfied in kernel-based learning tasks [4, 13, 17].

With regard to performance of an online algorithm, static regret is commonly adopted as a metric by most OCO schemes, and measures the difference between the aggregate loss of an OCO algorithm and that of the best fixed solution in hindsight [16, 18]. Specifically, for the sequence of functions $\{f_t\}$ generated by a learning algorithm $\mathcal{A}$, its static regret is

$$\mathrm{Reg}^{\mathrm{s}}_{\mathcal{A}}(T) := \sum_{t=1}^{T} \ell_t(f_t(\mathbf{x}_t)) - \sum_{t=1}^{T} \ell_t(f^*(\mathbf{x}_t)) \tag{18}$$

where the best static function estimator $f^*(\cdot)$ is obtained through the following batch optimization

$$f^*(\cdot) \in \arg\min_{f\in\mathcal{F}} \sum_{t=1}^{T} \ell_t(f(\mathbf{x}_t)) \tag{19}$$

and the function space is $\mathcal{F} := \bigcup_{p\in\mathcal{P}} \mathcal{H}_p$ by default, with $\mathcal{H}_p$ representing the RKHS induced by $\kappa_p$.

The next theorem characterizes the difference between the loss of online MKL algorithm relative to the best functional estimator in the RKHS; see [19] for the proof.

**Theorem 1:** *Suppose AS1-AS2 are satisfied. If $f^*_{\mathcal{H}_p}$ is the best function estimator in* (19) *belonging to $\mathcal{H}_p$, with probability at least $1 - 2^8\left(\frac{\sigma_p}{\epsilon}\right)^2 \exp\left(\frac{-D\epsilon^2}{4d+8}\right)$, the following bound holds*

$$\sum_{t=1}^{T} \ell_t\left(\sum_{p=1}^{P} \bar{w}_{p,t} f_{p,t}(\mathbf{x}_t)\right) - \min_{p\in\{1,\ldots,P\}} \sum_{t=1}^{T} \ell_t\left(f^*_{\mathcal{H}_p}(\mathbf{x}_t)\right)$$
$$\leq \frac{\ln P}{\eta} + \frac{(1+\epsilon)\|f^*_{\mathcal{H}_p}\|_1^2}{2\eta} + \frac{\eta L^2 T}{2} + \eta T + \epsilon L T \|f^*_{\mathcal{H}_p}\|_1 \tag{20}$$

*where $\epsilon > 0$ is a constant, $d$ is the dimension of $\mathbf{x}$, and $D$ is the number of random features, while $\sigma_p^2 := \mathbb{E}^{\pi_{\kappa_p}}_{\mathbf{v}}[\mathbf{v}^\top\mathbf{v}]$ is the second moment of $\mathbf{v}$. Setting $\eta = \epsilon = \mathcal{O}(1/\sqrt{T})$ leads to*

$$\mathrm{Reg}^{\mathrm{s}}_{\mathrm{Raker}}(T) = \mathcal{O}(\sqrt{T}) \tag{21}$$

*where the benchmark in* (18) *belongs to $\bigcup_{p\in\mathcal{P}} \mathcal{H}_p$.*

Observe that the probability in (20) gets larger as $D$ increases. For a given $\epsilon$, one can always find an appropriate $D$ to ensure a positive probability. Theorem 1 establishes that with appropriate choice of parameters, the novel Raker algorithm achieves sub-linear regret, which implies that Raker incurs on average "no-regret" relative to the best static functional estimation belonging to the function space $\bigcup_{p\in\mathcal{P}} \mathcal{H}_p$.

## 4. NUMERICAL TESTS

The present section tests the performance of our novel algorithms for online regression tasks. We compared Raker with online multi-kernel learning (OMKL) [15], and online (single) kernel based learning using Gaussian kernels (RBF) with bandwidth $\sigma^2 = \{0.1, 1, 10\}$. All the considered MKL approaches use Gaussian kernels with $\sigma^2 = \{0.1, 1, 10\}$, step-

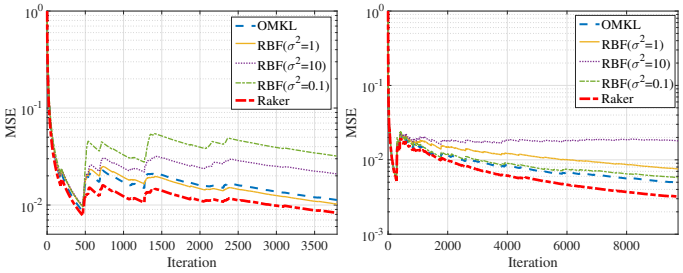


**Fig. 1**: MSE performance: a) Twitter; b) Tom's hardware.

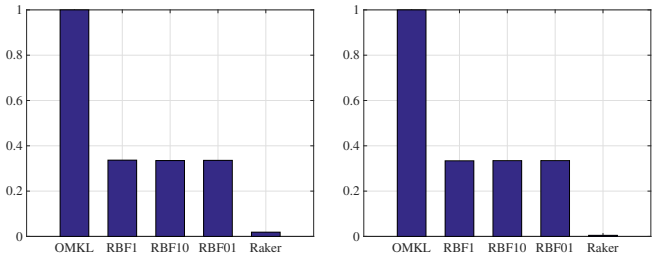


**Fig. 2**: Normalized CPU time: a) Twitter; b) Tom's hardware.

sizes of the single kernel based learning algorithms are set to $\eta = 1/\sqrt{T}$ for all algorithms, and $\eta = 0.5$ and $\lambda = 0.01$ for all MKL approaches. Entries of $\{\mathbf{x}_t\}$ and $\{y_t\}$ are normalized to lie in $[0, 1]$. For RF-based approaches, $D = 50$ *orthogonal random features* were used.

**Datasets.** Performance is tested on several benchmark datasets [20]. The twitter dataset consists of time series of $T = 6,000$ samples with $\mathbf{x}_t \in \mathbb{R}^{77}$ and the Tom's hardware dataset contains $T = 10,000$ feature vectors each of size 96, while $y_t$ represents the average number of active discussions about a certain topic on twitter and Tom's hardware [21].

**Results.** The performance of different algorithms is plotted in Fig. 1 in terms of the mean-square error $\mathrm{MSE}(t) := (1/t)\sum_{\tau=1}^{t}(y_\tau - \hat{y}_\tau)^2$. Clearly, Raker achieves competitive performance, with just $5\%$ of the MKL runtime. Aligned with the motivation of using multiple kernels, all MKL methods outperform the algorithms using only a single kernel. The normalized CPU time of all schemes is depicted in Fig. 2. A sharp observation is that Raker is computationally more efficient than the existing OMKL method.

## 5. CONCLUSIONS

We dealt with online multi-kernel learning problem. Leveraging recent advances in *variance-reduced* random feature approximation, we developed a scalable online *multi-kernel* learning approach that we term Raker. We established that Raker achieves sub-linear regret, meaning that the predictions generated by Raker are no worse than those under the best static function on average. Experiments on real datasets validate the effectiveness of our Raker method.

## 6. REFERENCES


[1] J. Shawe-Taylor and N. Cristianini, *KernelM ethods for Pattern Analysis*. Cambridge, United Kingdom: Cambridge University Press, 2004.

[2] B. Dai, N. He, Y. Pan, B. Boots, and L. Song, "Learning from conditional distributions via dual embeddings," in *Proc. Intl. Conf. on Artificial Intelligence and Statistics*, Fort Lauderdale, FL, Apr. 2017, pp. 1458–1467.

[3] C. Cortes, M. Mohri, and A. Rostamizadeh, "$\ell_2$-regularization for learning kernels," in *Proc. Conf. on Uncertainty in Artificial Intelligence*, Montreal, Canada, Jun. 2009, pp. 109–116.

[4] J. A. Bazerque and G. B. Giannakis, "Nonparametric basis pursuit via sparse kernel-based learning: A unifying view with advances in blind methods," *IEEE Signal Processing Magazine*, vol. 30, no. 4, pp. 112–125, Jul. 2013.

[5] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Identifying suspicious URLs: An application of large-scale online learning," in *Proc. Intl. Conf. Mach. Learn.* ACM, 2009, pp. 681–688.

[6] C. Richard, J. C. M. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *IEEE Trans. Sig. Proc.*, vol. 57, no. 3, pp. 1058–1067, Mar. 2009.

[7] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," *IEEE Trans. Sig. Proc.*, vol. 52, no. 8, pp. 2165–2176, Aug. 2004.

[8] S. C. Hoi, R. Jin, P. Zhao, and T. Yang, "Online multiple kernel classification," *Machine Learning*, vol. 90, no. 2, pp. 289–316, Feb. 2013.

[9] G. Wahba, *Spline Models for Observational Data*. SIAM, 1990.

[10] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Proc. Advances in Neural Info. Process. Syst.*, Vancouver, Canada, Dec. 2008, pp. 1177–1184.

[11] X. Y. Felix, A. T. Suresh, K. M. Choromanski, D. N. Holtmann-Rice, and S. Kumar, "Orthogonal random features," in *Proc. Advances in Neural Info. Process. Syst.*, Barcelona, Spain, Dec. 2016, pp. 1975–1983.

[12] K. Crammer, J. Kandola, and Y. Singer, "Online classification on a budget," in *Proc. Advances in Neural Info. Process. Syst.*, 2004, pp. 225–232.

[13] C. A. Micchelli and M. Pontil, "Learning the kernel function via regularization," *J. Machine Learning Res.*, vol. 6, pp. 1099–1125, Jul. 2005.

[14] N. Cesa-Bianchi and G. Lugosi, *Prediction, Learning, and Games*. Cambridge, United Kingdom: Cambridge University Press, 2006.

[15] D. Sahoo, S. C. Hoi, and B. Li, "Online multiple kernel regression," in *Proc. Intl. Conf. on Knowledge Discovery and Data Mining*, New York, USA, Aug. 2014, pp. 293–302.

[16] E. Hazan, "Introduction to online convex optimization," *Found. and Trends in Mach. Learn.*, vol. 2, no. 3-4, pp. 157–325, 2016.

[17] J. Lu, S. C. Hoi, J. Wang, P. Zhao, and Z.-Y. Liu, "Large scale online kernel learning," *J. Machine Learning Res.*, vol. 17, no. 47, pp. 1–43, Apr. 2016.

[18] S. Shalev-Shwartz, "Online learning and online convex optimization," *Found. and Trends in Mach. Learn.*, vol. 4, no. 2, pp. 107–194, 2011.

[19] Y. Shen, T. Chen, and G. B. Giannakis, "Random feature-based online multi-kernel learning in environments with unknown dynamics," *arXiv preprint:1712.09983*, Dec. 2017. [Online]. Available: https://arxiv.org/abs/1712.09983

[20] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[21] F. Kawala, A. Douzal-Chouakria, E. Gaussier, and E. Dimert, "Prédictions d'activité dans les réseaux sociaux en ligne," in *4ième Conférence sur les Modèles et l'Analyse des Réseaux: Approches Mathématiques et Informatiques*, 2013.