Online Multi-Kernel Learning with Graph-Structured Feedback

Pouya M Ghari [1] **Yanning Shen** [1]

Abstract

Multi-kernel learning (MKL) exhibits reliable performance in nonlinear function approximation tasks. Instead of using one kernel, it learns the optimal kernel from a pre-selected dictionary of kernels. The selection of the dictionary has crucial impact on both the performance and complexity of MKL. Specifically, inclusion of a large number of irrelevant kernels may impair the accuracy, and increase the complexity of MKL algorithms. To enhance the accuracy, and alleviate the computational burden, the present paper develops a novel scheme which *actively* chooses relevant kernels. The proposed framework models the pruned kernel combination as feedback collected from a graph, that is refined 'on the fly.' Leveraging the random feature approximation, we propose an online scalable multi-kernel learning approach with graph feedback, and prove that the proposed algorithm enjoys sublinear regret. Numerical tests on real datasets demonstrate the effectiveness of the novel approach.

1. Introduction

Function approximation emerges in various machine learning tasks such as classification, regression and clustering. In present paper, we will focus on supervised function learning tasks, i.e., given samples $\{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_T, y_T)\}_{t=1}^T$ such that $\mathbf{x}_t \in \mathbb{R}^d$ and $y_t \in \mathbb{R}$, the goal is to find a function $f(.)$ such that the discrepancy between $f(\mathbf{x}_t)$ and y_t is minimized. To this end, a cost function $\mathcal{C}(f(\mathbf{x}_t), y_t)$ can be employed to measure the difference between $f(\mathbf{x}_t)$ and y_t and $\sum_{t=1}^T \mathcal{C}(f(\mathbf{x}_t), y_t)$ is minimized. Upon assuming that $f(.)$ belongs to a reproducing kernel Hilbert space (RKHS), the problem becomes tractable (Scholkopf & Smola, 2001). Most of kernel based learning relies on a preselected kernel, while the data-driven multi-kernel learning (MKL) approach is more powerful, as it learns the optimal kernel from a dictionary of kernels, see e.g., (Cortes et al., 2010).

In a number of tasks, it is also required to carry out function learning in an online fashion, especially when data is collected sequentially. Moreover, the sheer volume of data makes it impossible to store it in batch. In these cases, it is imperative to perform learning tasks in an online fashion. However, kernel based learning also suffers from the well-known "curse of dimensionality" (Shawe-Taylor & Cristianini, 2004; Wahba, 1990), which makes it not amenable for sequential processing. In order to address this issue, online single kernel-based learning has attracted increasing attention, see e.g., (Bouboulis et al., 2018; Lu et al., 2016; Zhang & Liao, 2019), where (Zhang & Liao, 2019) developed a random sketching based framework. Kernels can be approximated using finite-dimensional feature maps which can effectively make the learning task scalable, see e.g., (Rahimi & Recht, 2007; Shahrampour & Tarokh, 2018; Williams & Seeger, 2000). Employing finite feature mapping, (Bouboulis et al., 2018; Lu et al., 2016) developed online kernel learning approach based on random feature (RF) approximation (Rahimi & Recht, 2007) . Building upon the online RF-based single kernel learning framework, multi-kernel variants have also been developed in (Lu et al., 2020; Sahoo et al., 2019; Shen et al., 2018; 2019).

A largely overlooked issue for most of the MKL approaches is the proper selection of the kernel dictionary, which has pivotal impact on both accuracy and complexity of MKL. Specifically, inclusion of a large number of irrelevant kernels may impair the accuracy, and increase the complexity of the MKL algorithm. Faced with these challenges, the present paper aims at developing an OMKL framework, which can actively choose relevant kernels from the dictionary, and refine the selection 'on the fly'. The proposed framework can alleviate the computational burden of online MKL due to the reduced number of kernels used at each time instant. Specifically, the online kernel selection is modeled as **g**raph-structured **f**eedback (henceforth termed (OMKL-GF)), where each kernel can be viewed as an expert lying in a graph. At each time slot, instead of collecting feedback from all kernel nodes at each time, the graph is updated and refined so that only a subset of kernels are employed at each time slot. Relative to existing OMKL approaches, our novelty can be summarized as follows:

[1]University of California, Irvine, CA, USA. Correspondence to: Yanning Shen <yannings@uci.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

c1) Different from existing works which employ all kernels in the dictionary, while only learns the combination coefficients, OMKL-GF only uses a subset of kernels each time slot according to a time-varying graph;

c2) We develop an adaptive and disciplined framework to refine the graph structure according to the loss incurred by kernel-based approximants, and prove that the resulting OMKL-GF approach achieves sublinear regret;

c3) Experiments on real datasets demonstrate the effectiveness of the novel algorithm compared with several state-of-art OMKL alternatives.

2. Preliminaries

Given samples $\{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_T, y_T)\}_{t=1}^T$, the goal is to find the nonlinear function $f(.)$, such that $y_t = f(\mathbf{x}_t) + e_t$. In the context of kernel based learning, it is assumed that the sought $f(.)$ belongs to the reproducing Hilbert kernel space (RHKS) $\mathcal{H} := \{f | f(\mathbf{x}) = \sum_{t=1}^{\infty} \alpha_t \kappa(\mathbf{x}, \mathbf{x}_t)\}$, where $\kappa(\mathbf{x}, \mathbf{x}_t) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ measures the similarity between $\mathbf{x}$ and $\mathbf{x}_t$ which is a symmetric positive semidefinite basis function called kernel. A kernel is reproducing if it satisfies $\langle \kappa(\mathbf{x}, \mathbf{x}_t), \kappa(\mathbf{x}, \mathbf{x}_{t'}) \rangle_{\mathcal{H}} = \kappa(\mathbf{x}_t, \mathbf{x}_{t'})$ where $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denotes vector inner product in Hilbert space, with the RKHS norm defined as $\|f\|_{\mathcal{H}}^2 := \sum_t \sum_{t'} \alpha_t \alpha_{t'} \kappa(\mathbf{x}_t, \mathbf{x}_{t'})$. The function approximation problem can henceforth be solved via

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

where $\mathcal{C}(.,.)$ is a cost function, which can be task-specific, e.g., least-squares for regression, and logistic cost for classification. And the regularizer $\Omega(.)$ is a non-decreasing function introduced to control model complexity, and prevent overfitting. Given finite data samples, the representer theorem states that the optimal solution of (1) can be written as (Wahba, 1990)

$$\hat{f}(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t \kappa(\mathbf{x}, \mathbf{x}_t) := \boldsymbol{\alpha}^\top \boldsymbol{\kappa}(\mathbf{x}) \tag{2}$$

where $\boldsymbol{\alpha} := [\alpha_1, ..., \alpha_T]^\top$ collects unknown coefficients to be estimated, and $\boldsymbol{\kappa}(\mathbf{x}) := [\kappa(\mathbf{x}, \mathbf{x}_1), \cdots, \kappa(\mathbf{x}, \mathbf{x}_T)]^\top$. Hence, the RKHS norm can be written as $\|f\|_{\mathcal{H}}^2 := \boldsymbol{\alpha}^\top \mathbf{K} \boldsymbol{\alpha}$, where $\mathbf{K}$ is the kernel matrix whose (t, t')-th entry is $\kappa(\mathbf{x}_t, \mathbf{x}_{t'})$. Note that the dimension of $\boldsymbol{\alpha}$ increases with the number of data samples T, which is also known as 'curse of dimensionality' (Wahba, 1990). This brings challenge for solving (1) in an online fashion.

2.1. Random Feature Approximation

One way to cope with the increasing dimension of the unknown variables is by resorting to random feature approximation for kernels (Rahimi & Recht, 2007). As in (Rahimi & Recht, 2007), we will approximate κ in (2) using shift-invariant kernels that satisfy $\kappa(\mathbf{x}_t, \mathbf{x}_{t'}) = \kappa(\mathbf{x}_t - \mathbf{x}_{t'})$. For $\kappa(\mathbf{x}_t - \mathbf{x}_{t'})$ absolutely integrable, its Fourier transform $\pi_\kappa(\mathbf{v})$ exists and represents the power spectral density, which upon normalizing to ensure $\kappa(\mathbf{0}) = 1$, can also be viewed as a probability density function (pdf); hence,

$$\begin{aligned} \kappa(\mathbf{x}_t - \mathbf{x}_{t'}) &= \int \pi_\kappa(\mathbf{v}) e^{j\mathbf{v}^\top(\mathbf{x}_t - \mathbf{x}_{t'})} d\mathbf{v} \\ &:= \mathbb{E}_{\mathbf{v}}[e^{j\mathbf{v}^\top(\mathbf{x}_t - \mathbf{x}_{t'})}]. \end{aligned} \tag{3}$$

By sampling sufficient number of independent and identically distributed (i.i.d) samples $\{\mathbf{v}_i\}_{i=1}^D$ from $\pi_\kappa(\mathbf{v})$, $\kappa(\mathbf{x}_t - \mathbf{x}_{t'})$ can be approximated by the ensemble mean $\hat{\kappa}_c(\mathbf{x}_t - \mathbf{x}_{t'}) := \frac{1}{T} \sum_{i=1}^D e^{j\mathbf{v}^\top(\mathbf{x}_t - \mathbf{x}_{t'})}$, the real part of which also constitutes an unbiased estimator of $\kappa(\mathbf{x}_t - \mathbf{x}_{t'})$ (Rahimi & Recht, 2007)

$$\hat{\kappa}(\mathbf{x}_t - \mathbf{x}_{t'}) = \mathbf{z}_{\mathbf{v}}^\top(\mathbf{x}_t) \mathbf{z}_{\mathbf{v}}(\mathbf{x}_{t'}) \tag{4}$$

where

$$\begin{aligned} &\mathbf{z}_{\mathbf{v}}(\mathbf{x}) \\ &= \frac{1}{\sqrt{D}} [\sin \mathbf{v}_1^\top \mathbf{x}, ..., \sin \mathbf{v}_D^\top \mathbf{x}, \cos \mathbf{v}_1^\top \mathbf{x}, ..., \cos \mathbf{v}_D^\top \mathbf{x}]^\top. \end{aligned} \tag{5}$$

Hence, $\hat{f}(\mathbf{x})$ in (2) can be approximated as

$$\hat{f}_{\text{RF}}(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t \mathbf{z}_{\mathbf{v}}^\top(\mathbf{x}_t) \mathbf{z}_{\mathbf{v}}(\mathbf{x}) := \boldsymbol{\theta}^\top \mathbf{z}_{\mathbf{v}}(\mathbf{x}) \tag{6}$$

where $\boldsymbol{\theta}$ is a $2D$ vector whose dimension does not increase with the number of data samples. Thus, the nonlinear function that is optimal in sense of (1) is approximated by a linear function in the random feature space. Hence, using RF approximation resolves the growing size of parameters, and makes the problem (10) amenable for scalable online implementation.

Note that kernel based learning approaches rely on a preselected kernel, which often requires task-specific prior information that may not be available. To cope with this challenge, MKL has been proposed which learns the kernel as a combination of a prescribed and sufficiently rich dictionary of kernels $\{\kappa_n\}_{n=1}^N$. The kernel combination belongs to the convex hull $\bar{\mathcal{K}} := \{\bar{\kappa} = \sum_{n=1}^N \bar{\beta}_n \kappa_n, \bar{\beta}_n \geq 0, \sum_{n=1}^N \bar{\beta}_n = 1\}$, and is itself a kernel (Scholkopf & Smola, 2001). Hence, the function approximation can be performed by seeking functions in the following form

$$f(\mathbf{x}_t) = \sum_{n=1}^{N} \bar{w}_n f_n(\mathbf{x}_t) \tag{7}$$

where $\sum_{n=1}^N \bar{w}_n = 1$ and $f_n(\mathbf{x}_t)$ is in $\mathcal{H}_n$ which is a RKHS induced by κ_n. Substituting (7) into (1) and resorting

Jensen's inequality, the function approximation problem can then be solved in an online fashion by minimizing the upper bound of the original objective function. Upon replacing $f_n(.)$ using its RF approximation

$$\hat{f}_{\text{RF},n}(\mathbf{x}_t) = \boldsymbol{\theta}_n^\top \mathbf{z}_n(\mathbf{x}_t) \tag{8}$$

the function approximation problem can be written as (Shen et al., 2019)

$$\min_{\{\bar{w}_{n,t},\boldsymbol{\theta}_n\}} \sum_{t=1}^{T}\sum_{n=1}^{N} \bar{w}_{n,t}\left(\mathcal{C}(\boldsymbol{\theta}_n^\top \mathbf{z}_n(\mathbf{x}_t), y_t) + \lambda\Omega(\|\boldsymbol{\theta}_{n,t}\|^2)\right)$$
$$\text{s.t.} \sum_{n=1}^{N} \bar{w}_{n,t} = 1, \;\; \bar{w}_{n,t} \geq 0, \;\; \forall t : 1 \leq t \leq T. \tag{9}$$

However, employing a large kernel dictionary (i.e., large N) increases the computational complexity, and may deteriorate the accuracy of function approximation if too many irrelevant kernels are included. This motivates us to investigate data-driven approaches to choose and refine the subset of relevant kernels from a large dictionary.

3. Online MKL with Graph Feedback

Based on RF formulation in Section 2, the present section will introduce an online multi-kernel learning approach which can adaptively refine the kernel selection.

3.1. Time-Varying Kernel Selection

Instead of combining the entire dictionary of the kernels, in present paper, we will consider a subset of kernels $\{\kappa_n, n \in \mathcal{S}_t\}$ at time instant t, where $\mathcal{S}_t$ is the index set of the chosen subset of kernels at time instant t, and $|\mathcal{S}_t| = N_t$. Hence, the original function approximation problem boils down to

$$\min_{\{\bar{w}_{n,t},\boldsymbol{\theta}_n\}} \sum_{t=1}^{T}\sum_{n\in\mathcal{S}_t} \bar{w}_{n,t}\left(\mathcal{C}(\boldsymbol{\theta}_n^\top \mathbf{z}_n(\mathbf{x}_t), y_t) + \lambda\Omega(\|\boldsymbol{\theta}_{n,t}\|^2)\right)$$
$$\text{s.t.} \sum_{n\in\mathcal{S}_t} \bar{w}_{n,t} = 1, \;\; \bar{w}_{n,t} \geq 0, \;\; \forall 1 \leq t \leq T. \tag{10}$$

At each time instant t, the loss of n-th kernel is defined as

$$\mathcal{L}(\boldsymbol{\theta}_n^\top \mathbf{z}_n(\mathbf{x}_t), y_t) = \mathcal{C}(\boldsymbol{\theta}_n^\top \mathbf{z}_n(\mathbf{x}_t), y_t) + \lambda\Omega(\|\boldsymbol{\theta}_n\|^2)$$

Upon defining the normalized weights $\bar{w}_{n,t} = \frac{w_n}{\sum_{m\in\mathcal{S}_t} w_m}$, (10) can be re-written as

$$\min_{\{w_n\},\{\boldsymbol{\theta}_n\}} \sum_{t=1}^{T}\sum_{n\in\mathcal{S}_t} \frac{w_n}{\sum_{m\in\mathcal{S}_t} w_m}\mathcal{L}(\boldsymbol{\theta}_n^\top \mathbf{z}_n(\mathbf{x}_t), y_t)$$
$$\text{s.t.} \quad w_n > 0, \quad \forall 1 \leq n \leq N. \tag{11}$$

However, (10) assumes that $\{\mathcal{S}_t\}_{t=1}^T$ are preselected sets. In the following section, we will study data-driven scheme which can adaptively select a subset of kernels 'on the fly' upon receiving new data samples.

3.2. Data-Driven Graph-Structured Kernel Selection

In order to adaptively choose the subset of kernels, the present section models the pruned kernel combination as feedback collected from a graph, that is refined in an online fashion. By doing this, the proposed approach trims irrelevant kernels in the dictionary to both improve the function approximation accuracy and reduce the computational complexity of MKL.

Consider a time varying bipartite graph (Asratian et al., 1998) G_t at time t, which consists of two sets of nodes: N kernel nodes $\{v_1^{(k)}, ..., v_N^{(k)}\}$ and J selective nodes $\{v_1^{(c)}, ..., v_J^{(c)}\}$ where $v_n^{(k)}$ and $v_j^{(c)}$ are the n-th kernel node and j-th selective node, respectively. And the edges of the graph represents the association between the kernel nodes and the selective nodes. Specifically, an edge between $v_n^{(k)}$ and $v_j^{(c)}$ exists at time t if the n-th kernel is assigned to j-th selective node. The construction and refinement of the graph will be discussed in Section 3.3 .

At each time slot, one of selective nodes $v_j^{(c)}$ is chosen, and the subset of kernel nodes connected with $v_j^{(c)}$ will be used for the instantaneous function approximation at time t, [c.f. (11)]. Then, the loss $\mathcal{L}(\hat{f}_{RF,n}(\mathbf{x}_t), y_t)$ is observed for every kernel in the chosen subset and $\boldsymbol{\theta}_{n,t}$ is updated as

$$\boldsymbol{\theta}_{n,t+1} = \boldsymbol{\theta}_{n,t} - \eta\nabla\mathcal{L}(\boldsymbol{\theta}_{n,t}^\top \mathbf{z}_{\mathbf{v},n}(\mathbf{x}_t), y_t). \tag{12}$$

Let $w_{n,t}$ denotes the weighting coefficient w_n at time instant t. We leverage multiplicative update for weights $w_{n,t}$

$$w_{n,t+1} = w_{n,t}\exp(-\eta\frac{\ell_{n,t}}{2^b}) \tag{13}$$

where $b = \lfloor \log_2(J) \rfloor$, and $\ell_{n,t}$ denotes the importance sampling loss estimates (Alon et al., 2017)

$$\ell_{n,t} = \frac{\mathcal{L}(\hat{f}_{\text{RF},n}(\mathbf{x}_t), y_t)}{q_{n,t}}\mathcal{I}\{n \in \mathcal{S}_t\} \tag{14}$$

which is the observed loss $\mathcal{L}(\hat{f}_{RF,n}(\mathbf{x}_t), y_t)$ divided by the probability $q_{n,t}$ which is the probability that the loss of associated kernel is observed. The value of $q_{n,t}$ depends on how the graph is generated, and $\mathcal{I}(.)$ denotes the indicator function. Since we initialize the value of w_n with 1, the multiplicative update in (13) satisfies the constraint in (11). Upon updating parameters $\boldsymbol{\theta}_{n,t+1}$ and $w_{n,t+1}$, the function approximation can be obtained via (7) and (8) using the chosen subset of kernel nodes connected to one of the selective nodes.

Then, the selective nodes are assigned to weights $\{u_{j,t+1}\}$ according to the the kernel nodes' weights $\{w_{n,t+1}\}$. Indeed, each selective node's weight $\{u_{j,t+1}\}$ is the total summation of weights of kernel nodes which are connected to that selective node. Specifically the weight of $v_j^{(c)}$ is obtained via

$$u_{j,t+1} = \sum_{\forall n: v_n^{(k)} \to v_j^{(c)}} w_{n,t+1}. \tag{15}$$

Note that the weights of the selective nodes are determined by its adjacent kernel nodes, which indicates the reliability of the corresponding kernel-based function estimate.Hence, the probability according to which a selective node is chosen in the next time slot can be updated as

$$p_{j,t+1} = (1-\eta_e)\frac{u_{j,t+1}}{U_{t+1}} + \frac{\eta_e}{J} \tag{16}$$

where $U_{t+1} := \sum_{j=1}^{J} u_{j,t+1}$, and $0 < \eta_e \leq 1$ is the exploration rate. The term $\frac{\eta_e}{J}$ is introduced to tradeoff between exploitation and exploration.

To sum up, each kernel is viewed as an expert and at each time instant a subset of function approximations provided by these experts is combined. In this regard, the RF approximation $\hat{f}_{\text{RF},n}(\mathbf{x}_t)$ can be viewed as the feedback provided by n-th kernel node, and the proposed framework models the pruned kernel combination as feedback collected from a graph, where the feedback are combined only if the corresponding kernel node is connected to the chosen selective node.By doing this, the proposed approach trims irrelevant kernels in the dictionary to both improve the function approximation accuracy and reduce the computational complexity of MKL. The graph refining approach will be proposed in the ensuing subsection.

3.3. Online Graph Refinement

Note that generating and refining the time varying graph is of utmost importance, as it affects both function approximation accuracy and computational complexity. In this regard, a graph is successful if it can provide a subset of kernels which results in as less as possible loss. Indeed, considering computational complexity, the graph should provide a limited number of kernels which obtain minimum loss. To this end, we aim to propose a generating method for graph. However, in order to execute more exploration, the graph is generated in a stochastic manner.

Increasing the number of kernel nodes connected to $v_j^{(c)}$, increases the computational complexity of performing function approximation by choosing $v_j^{(c)}$. Therefore, the graph generation algorithm should be designed to limit the number of kernel nodes connected to each selective node. Let M denote the maximum number of kernel nodes connected to each selective node. The procedure to generate the graph G_t is presented in Algorithm 1. Let $\mathbf{A}_t$ represents $N \times J$ sub-adjacency matrix between two disjoint subsets $\{v_1^{(c)}, ..., v_J^{(c)}\}$ and $\{v_1^{(k)}, ..., v_N^{(k)}\}$. Also, $\mathbf{A}_t(n,j)$ represents n-th row and j-th column of the sub-adjacency matrix $\mathbf{A}_t$ and it is equal to 1 if $v_n^{(k)}$ is connected to $v_j^{(c)}$, and 0 otherwise.

Algorithm 1 Generating Graph G_t

Input: Kernels κ_n, $n = 1, ..., N$, exploration coefficient $\eta_e > 0$, the maximum number of connected kernel nodes to each computation node M and weighting coefficient $w_{n,t}$ for kernels.
Initialize: Sub-adjacency matrix $\mathbf{A}_t = \mathbf{0}_{N\times J}$.
for $j = 1, ..., J$ **do**
  for $n = 1, ..., N$ **do**
    Set $p_{t,j}^{(\kappa_n)} = (1-{\eta_e}^j)\frac{w_{n,t}}{\sum_{n=1}^{N} w_{n,t}} + \frac{{\eta_e}^j}{N}$.
  end for
  for $k = 1, ..., M$ **do**
    Choose one of nodes $v_{k,i}$ drawn according to PMF $p_{t,j}^{(\kappa)} = (p_{t,j}^{(\kappa_1)}, ..., p_{t,j}^{(\kappa_N)})$.
    Set $\mathbf{A}_t(n,j) = 1$.
  end for
end for

Each selective node $v_j^{(c)}$ draws kernel nodes $v_n^{(k)}$ in M independent trials and in each trial selective node draws only one kernel node. We put more weight on kernels which obtain less loss. The probability that selective node $v_j^{(c)}$ draws the kernel node $v_n^{(k)}$ in a trial at time t is

$$p_{t,j}^{(\kappa_n)} = (1-{\eta_e}^j)\frac{w_{n,t}}{\sum_{n=1}^{N} w_{n,t}} + \frac{{\eta_e}^j}{N} \tag{17}$$

Note that the first term in (17) discriminates between kernels based on their weights which is determined by their loss in function approximation [c.f. (13)]. Furthermore, the second term allows exploration over all kernel nodes. Specially, the selective node $v_j^{(c)}$ draws kernel nodes according to uniform distribution if $\eta_e = 1$. Furthermore, note that ${\eta_e}^j$ is a non-increasing function of j for $0 < \eta_e \leq 1$. The selective node $v_1^{(c)}$ puts more weight on exploration in comparison with others while $v_J^{(c)}$ considers more exploitation than all the other selective nodes. Therefore, the selective nodes entail different level of exploration and exploitation.

Based on the definition of $p_{t,j}^{(\kappa_n)}$ in (17), the probability that the n-th kernel node is connected to $v_j^{(c)}$ is $1-(1-p_{t,j}^{(\kappa_n)})^M$. In fact, $(1-p_{t,j}^{(\kappa_n)})^M$ is the probability that n-th kernel node is chosen by $v_j^{(c)}$ in none of M trials. Therefore, the probability of observing the loss of n-th kernel at time t is

Algorithm 2 OMKL with Graph Feedback (OMKL-GF)

Input: Kernels κ_n, $n = 1, ..., N$, step size $\eta > 0$, the number of features D and b such that $J \in [2^b, 2^{b+1} - 1]$.
Initialize: $\boldsymbol{\theta}_{n,1} = \mathbf{0}$, $w_{n,1} = 1$, $n = 1, ..., N$
for $t = 1, ..., T$ **do**
  Receive one datum $\mathbf{x}_t$.
  Generate G_t using Algorithm 1.
  Set $u_{j,t} = \sum_{\forall n: v_n^{(k)} \to v_j^{(c)}} w_{n,t}$.
  Obtain $p_{j,t}$ via (16).
  Choose one selective node $v_j^{(c)}$ according to PMF $p_t = (p_{1,t}, ..., p_{J,t})$.
  Predict $\hat{f}_t(\mathbf{x}_t) = \sum_{n \in S_t} \frac{w_{n,t}}{\sum_{m \in S_t} w_{m,t}} \hat{f}_{\text{RF},n}(\mathbf{x}_t)$ with $\hat{f}_{\text{RF},n}(\mathbf{x}_t)$ in (8).
  Obtain loss $\mathcal{L}(\hat{f}_{RF,n}(\mathbf{x}_t), y_t)$ for all $n \in \mathcal{S}_t$.
  Update $\boldsymbol{\theta}_{n,t+1}$ via (12) for all $n \in \mathcal{S}_t$.
  Update $w_{n,t+1}$ via (13).
end for

given by

$$q_{n,t} = \sum_{j=1}^{J} p_{j,t} \left(1 - (1 - p_{t,j}^{(\kappa_n)})^M\right) \tag{18}$$

for $1 \leq n \leq N$. The value of $q_{n,t}$ is computed and used for importance sampling loss estimate in (14). The graph generation framework is summarized in Algorithm 1

At each time slot t, a graph G_t is generated, and used for choosing a selective node, and henceforth subset of the kernels. Then the weights of the selected kernels are updated according to the loss [c.f.(12) and (13)]. Then the graph can be refined as it is presented in Algorithm 1, and henceforth resulting in a better graph, leads to better function approximation. Given the graph, the function approximation will be carried out by choosing one of the selective nodes which leads to selecting a subset of kernels. Our proposed online multi-kernel learning with graph-structured feedback (OMKL-GF) is summarized in Algorithm 2.

Memory Requirement. At time instant t, OMKL-GF needs to store a real $2D$ random feature vector in addition to a weighting vector for each kernel in conjunction with a weighting vector for each selective node. As the number of kernels is in general larger than the number of selective nodes, the required memory is of order $\mathcal{O}(dDN)$.

Computational Complexity. The per-iteration complexity of our OMKL-GF (e.g. calculating inner products) is $\mathcal{O}(dDM + JN)$. In comparison, the per-iteration complexity of OMKR developed (Sahoo et al., 2014) is $\mathcal{O}(tdN)$, while more contemporary online RF-based OMKL approaches proposed in (Sahoo et al., 2019; Shen et al., 2019) both have per-iteration complexity $\mathcal{O}(dDN)$. Hence, OMKL-GF can significantly reduce the per iteration complexity especially when $J \leq M << N$.

Comparison with Online Expert Learning. Note that the updates resembles those in online learning with expert advice (Cesa-Bianchi et al., 1997; Littlestone & Warmuth, 1994; Vovk, 1998), and online expert learning with graph-structured feedback (Alon et al., 2015; 2017; Cortes et al., 2019; Liu et al., 2018; Mannor & Shamir, 2011), where each kernel can be viewed as an expert. However, relative to the generic expert advice problem, there exist three innovative differences in OMKL-GF: i) the kernel-based function estimator itself performs efficient online learning scheme for self-improvement; ii) unlike conventional online learning with expert advice which combines feedback from all experts, OMKL-GF only collects feedback from a subset of experts based on a graph; iii) we proposed an adaptive scheme to actively refine feedback graph based on the incurred online loss.

Comparison with Raker (Shen et al., 2019). Comparing OMKL-GF with Raker (Shen et al., 2019), it can be readily observed that both algorithms rely on RF-approximation to make online kernel based learning more scalable. While Raker employs all kernels in the dictionary for function approximation, our proposed OMKL-GF chooses a *time-varying* subset of kernels at each time instant by adaptively pruning irrelevant kernels. Experiments on real datasets will be presented in Section 5 to show that OMKL-GF can attain lower MSE and execution time in comparison with Raker by *actively* choosing a subset of kernels.

4. Regret Analysis

In order to analyze the performance of our OMKL-GF, we assume that the following conditions hold:

(a1) At each time instance t the loss function $\mathcal{L}(\boldsymbol{\theta}_{n,t}^\top \mathbf{z}_n(\mathbf{x}_t), y_t)$ is convex with respect to $\boldsymbol{\theta}$.

(a2) For $\boldsymbol{\theta}$ in a bounded set $\boldsymbol{\Theta}$ which satisfies $\|\boldsymbol{\theta}\| \leq C_{\boldsymbol{\theta}}$ the loss is bounded as $\mathcal{L}(\boldsymbol{\theta}_{n,t}^\top \mathbf{z}_{\mathbf{v},n}(\mathbf{x}_t), y_t) \in [0, 1]$. Also, in this case the loss has bounded gradient which means $\|\nabla \mathcal{L}(\boldsymbol{\theta}_{n,t}^\top \mathbf{z}_{\mathbf{v},n}(\mathbf{x}_t), y_t)\| \leq L$.

(a3) Kernels $\{\kappa_n\}_{n=1}^N$ are shift-invariant, standardized, and bounded, that is $\kappa_n(\mathbf{x}_i, \mathbf{x}_j), \forall \mathbf{x}_i, \mathbf{x}_j$. Also, each datum is bounded i.e. $\|\mathbf{x}_t\| \leq 1$.

Note that (a1) is satisfied by many convex loss functions such as least-squares loss. Furthermore, (a2) states that the losses are bounded and L-Lipschitz continuous. In addition, a number of kernels satisfy (a3), e.g., Gaussian, Laplacian and Cauchy kernels(Rahimi & Recht, 2007). Generally, the above conditions are standard in online convex optimization

(Hazan, 2016) and kernel learning (Rahimi & Recht, 2007; Shen et al., 2019).

We consider stochastic regret, which is a typical criterion for analyzing online convex optimization schemes (Hazan, 2016). Specifically, it measures the difference between expected aggregate loss of the online algorithm and the best function approximant in the hindsight. Given a sequence of approximations $\hat{f}_t(.)$ obtained by the algorithm $\mathcal{A}$, we have

$$\mathbb{E}[R^s_{\mathcal{A}}(T)]=\sum_{t=1}^{T}\mathbb{E}[\mathcal{L}(\hat{f}_t(\mathbf{x}_t),y_t)]-\sum_{t=1}^{T}\mathcal{L}(f^*(\mathbf{x}_t),y_t). \quad (19)$$

where $f^*(.)$ denotes the best function approximant in the hindsight and it can be obtained as follows

$$f^*(.) \in \arg\min_{f^*_n,n\in\{1,\dots,N\}} \sum_{t=1}^{T}\mathcal{L}(f^*_n(\mathbf{x}_t),y_t) \quad (20a)$$

$$f^*_n \in \arg\min_{f\in\mathcal{H}_n} \sum_{t=1}^{T}\mathcal{L}(f(\mathbf{x}_t),y_t) \quad (20b)$$

where $\mathcal{H}_n$ denotes the RKHS induced by κ_n. Thus, to compute the stochastic static regret analysis we have

$$\mathbb{E}[R^s_{\mathcal{A}}(T)]=\sum_{t=1}^{T}\mathbb{E}[\mathcal{L}(\hat{f}_t(\mathbf{x}_t),y_t)]-\sum_{t=1}^{T}\mathcal{L}(f^*(\mathbf{x}_t),y_t). \quad (21)$$

Note that in this paper, $\mathbb{E}[.]$ at time t denotes conditional expected value given $\{\mathcal{L}(\hat{f}_\tau(\mathbf{x}_\tau),y_\tau)\}_{\tau=1}^{t-1}$.

In order to analyze the regret for OMKL-GF, we first establish an intermediate result in the following lemma.

Lemma 1. *The regret of our proposed algorithm under (a1), (a2) and with* $\mathcal{F}_n = \{\hat{f}_n|\hat{f}_n(\mathbf{x}) = \boldsymbol{\theta}^\top\mathbf{z}_n(\mathbf{x}), \forall\boldsymbol{\theta}\in\mathbb{R}^{2D}\}$ *satisfies the following bound*

$$\sum_{t=1}^{T}\mathbb{E}[\mathcal{L}(\hat{f}_t(\mathbf{x}_t),y_t)] - \sum_{t=1}^{T}\mathcal{L}(\hat{f}^*_{t,n}(\mathbf{x}_t),y_t) < \frac{2^b}{\eta}\ln N + \frac{\|\boldsymbol{\theta}^*_n\|^2 NJ}{2\eta\eta_e^2} + \frac{\eta L^2 T}{2} + \eta_e JT + \frac{\eta N^2 JT}{2^{b+1}\eta_e^2} \quad (22)$$

where $\boldsymbol{\theta}^*_n$ *is associated with the best RF function approximant* $\hat{f}^*_{t,n}(\mathbf{x}_t) = \boldsymbol{\theta}^{*\top}_n\mathbf{z}_n(\mathbf{x}_t)$.

The next theorem further characterizes the difference between the loss of OMKL-GF relative to the best functional estimator in the RKHS.

Theorem 2. *The following bound holds with probability at least* $1-2^8(\frac{\sigma_n}{\epsilon})^2\exp(-\frac{D\epsilon^2}{4d+8})$ *under (a1)-(a3) for* $\epsilon>0$ *and with* f^*_n *belonging to RKHS* $\mathcal{H}_n$ *as in* (20b)

$$\sum_{t=1}^{T}\mathbb{E}[\mathcal{L}(\hat{f}_t(\mathbf{x}_t),y_t)] - \min_{n\in\{1,\dots,N\}}\sum_{t=1}^{T}\mathcal{L}(f^*_n(\mathbf{x}_t),y_t) < \frac{2^b}{\eta}\ln N + \frac{NJ(1+\epsilon)C^2}{2\eta\eta_e^2} + \frac{\eta L^2 T}{2} + \eta_e JT + \epsilon LTC + \frac{\eta N^2 JT}{2^{b+1}\eta_e^2} \quad (23)$$

where C *is a constant, and* σ_n^2 *is the second order moment of the RF vector norm which can be defined as* $\sigma_n^2 := \mathbb{E}^{\pi_{\kappa_n}}_{\mathbf{v}}[\|\mathbf{v}\|^2]$.

According to Theorem 2, by setting $\eta=\epsilon=\mathcal{O}(\frac{1}{\sqrt{T}})$ and $\eta_e=\mathcal{O}(T^{-1/6})$ in (23), the stochastic static regret in (19) satisfies $\mathbb{E}[R^s_{\mathcal{A}}(T)]=\mathcal{O}(T^{\frac{5}{6}})$. Thus, by selecting appropriate parameters, our proposed OMKL-GF achieves sublinear regret in expectation with respect to the best static function approximant in (19).

Note that while proper settings of ϵ and η relies on the knowledge of T, such information may not be necessary, via employing, e.g., doubling trick (Cesa-Bianchi & Lugosi, 2006). Considering (23), the probability $1-2^8(\frac{\sigma_n}{\epsilon})^2\exp(-\frac{D\epsilon^2}{4d+8})$ is an increasing function of D such that for a fixed ϵ, always there are some values for D which result in positive probability. Furthermore, (23) shows that by setting $D=\mathcal{O}(T\log(T))$, a sublinear regret can be obtained with high probability. In addition, more recent results about RF approximation, e.g., (Rudi & Rosasco, 2017) can also be readily applied and show that sublinear regret can be obtained with $\mathcal{O}(\sqrt{T}\log(T))$ random features.

5. Experiments

In this section, our proposed online OMKL-GF framework is tested in terms of accuracy and computational complexity. We evaluate our proposed online OMKL-GF in comparison with the following kernel learning baselines

- **OMKR**: Online MKL Hedge algorithm developed in (Sahoo et al., 2014).
- **OMKR-FA**: Random feature based online MKL via function approximation (Sahoo et al., 2019).
- **Raker**: Random feature based online MKL (Shen et al., 2019).

Mean square error (MSE) is used to evaluate the accuracy of MKL algorithms when it comes to performing online regression. For random feature based MKL algorithms including our proposed OMKL-GF, Raker and OMKR-FA, we generate 50 different sets of random features and the mean value

of MSE for all sets of random features is reported. Thus, in this section we compute MSE at time instant t as follows

$$\text{MSE} = \frac{1}{N_{\text{MSE}}} \sum_{i=1}^{N_{\text{MSE}}} \frac{1}{t} \sum_{\tau=1}^{t} (\hat{y}_\tau - y_\tau)^2 \qquad (24)$$

where $\hat{y}_\tau$ is the approximation of y_t provided by the MKL algorithms. Also, N_{MSE} denotes the number of rounds that MSE is computed. For our proposed OMKL-GF, Raker and OMKR-FA the value of N_{MSE} is set to 50 whereas the value of N_{MSE} for OMKR is set to 1. The number of random features D is fixed to 50 for all random feature-based MKL algorithms. Also, the regularization coefficient λ is set to 10^{-3}. In addition, for MKL algorithms we consider a dictionary of 17 radial basis function (RBF) kernels with different bandwidths. Let $\sigma^2_{i,\text{dic}}$ denotes the variance of i-th RBF kernel in the dictionary. In this case, the value of $\sigma^2_{i,\text{dic}}$ can be expressed as $\sigma^2_{i,\text{dic}} = 10^{\frac{i-9}{2}}$ where $1 \leq i \leq 17$. Moreover, for all MKL algorithms, stepsize η and the exploration rate η_e are set to $\frac{1}{\sqrt{t}}$.

Algorithm 2 requires to observe the graph to choose a subset of kernels. The graph G_t is generated by Algorithm 1 and disclosed. However, generating graph in every time instant can increase the computational complexity of the proposed OMKL-GF while it cannot improve MSE considerably. To further decrease the computational complexity of our proposed OMKL-GF, we consider the following scenario. The graph is generated until an amount of loss is achieved. For this scenario, the graph will not be generated in $\tau_{\text{ter}} + 1, \ldots, T$ if the condition $(\hat{y}_{\tau_{\text{ter}}} - y_{\tau_{\text{ter}}})^2 < 10^{-4}$ is met at time instant τ_{ter}.

The performance of MKL algorithms are tested for online regression, over the following real datasets downloaded from UCI Machine Learning Repository are used.

- **Air Quality.** The dataset contains 9358 instances of hourly averaged responses from an array of 5 sensors located on the field in a significantly polluted area. Data samples contain 13 features which are averaged sensors response. It is aimed at predicting y_t which is polluting chemical concentration in the air. (Vito et al., 2008).

- **Istanbul Stock Exchange.** Data is organized with regard to working days in Istanbul Stock Exchange which contains 536 instances with 7 features including stock market return indices. The goal is to predict Istanbul stock exchange national 100 index (Akbilgic et al., 2014).

- **Twitter.** This dataset contains 14000 samples from a micro-blogging platform Twitter with 77 features including the length of discussion on a given topic and the number of new interactive authors. Also, y_t represents the average number of active discussion (popularity) on a certain topic (Kawala et al., 2013).

- **Tom's Hardware.** dataset contains 10000 samples from a worldwide new technology forum with 96 features including the number of discussions on a topic. Moreover, y_t represents the average number of display about a certain topic on Tom's hardware (Kawala et al., 2013).

- **Naval Propulsion Plants.** Data has been generated from a sophisticated simulator of gas turbines. Dataset contains 11934 samples with 15 features including ship speed. Furthermore, y_t represents lever position (Coraddu et al., 2016).

Table 1. MSE ($\times 10^{-3}$) and execution time of MKL algorithms for Air Quality dataset when $\eta = \frac{1}{\sqrt{t}}$.

Algorithms	M	J	MSE	Execution time (s)
OMKR	-	-	3.3	2533.96s
OMKR-FA	-	-	41.2	1.67s
Raker	-	-	4.7	2.88s
OMKL-GF	1	1	8.1	**0.74**s
OMKL-GF	7	1	**4.2**	1.47s
OMKL-GF	10	1	**3.9**	1.68s
OMKL-GF	17	1	4.1	2.56s
OMKL-GF	1	2	7.7	**0.75**s
OMKL-GF	1	4	11.2	0.78s
OMKL-GF	7	2	4.6	1.46s
OMKL-GF	7	4	6.8	1.67s

Table 2. MSE ($\times 10^{-3}$) and execution time of MKL algorithms for Istanbul Exchange dataset when $\eta = \frac{1}{\sqrt{t}}$.

Algorithms	M	J	MSE	Execution time (s)
OMKR	-	-	10.0	17.16s
OMKR-FA	-	-	187.8	0.11s
Raker	-	-	11.3	0.19s
OMKL-GF	1	1	61.9	**0.05**s
OMKL-GF	7	1	13.3	0.12s
OMKL-GF	10	1	12.2	0.14s
OMKL-GF	17	1	11.3	0.18s
OMKL-GF	1	2	38.5	**0.05**s
OMKL-GF	1	4	38.4	**0.05**s
OMKL-GF	7	2	12.9	0.17s
OMKL-GF	7	4	15.2	0.15s

Table 1, Table 2, Table 3 and Table 4 list performance of MKL algorithms in terms of both MSE and execution time for each dataset. Note that while OMKR provides better

Table 3. MSE ($\times 10^{-3}$) and execution time of MKL algorithms for Twitter dataset when $\eta = \frac{1}{\sqrt{t}}$.

Algorithms	M	J	MSE	Execution time (s)
OMKR	-	-	3.3	9099.95s
OMKR-FA	-	-	29.6	4.06s
Raker	-	-	5.3	6.06s
OMKL-GF	1	1	16.5	**1.46**s
OMKL-GF	7	1	5.3	2.80s
OMKL-GF	10	1	**4.7**	4.03s
OMKL-GF	17	1	4.6	5.17s
OMKL-GF	1	2	12.2	**1.50**s
OMKL-GF	1	4	14.4	1.56s
OMKL-GF	7	2	5.1	3.94s
OMKL-GF	7	4	8.7	3.49s

Table 4. MSE ($\times 10^{-3}$) and execution time of MKL algorithms for Tom's Hardware dataset when $\eta = \frac{1}{\sqrt{t}}$.

Algorithms	M	J	MSE	Execution time (s)
OMKR	-	-	2.2	4474.20s
OMKR-FA	-	-	21.4	3.04s
Raker	-	-	4.6	4.64s
OMKL-GF	1	1	12.3	**1.03**s
OMKL-GF	7	1	3.7	2.26s
OMKL-GF	10	1	**3.5**	2.65s
OMKL-GF	17	1	3.7	3.67s
OMKL-GF	1	2	12.1	**1.07**s
OMKL-GF	1	4	13.0	1.08s
OMKL-GF	7	2	5.2	2.39s
OMKL-GF	7	4	8.5	2.46s

accuracy compared with RF-based alternatives, it is also much more computationally complex, without resorting to RF approximation. It can be observed that the proposed OMKL-GF outperforms OMKR-FA in terms of accuracy. Also, the results show that when the maximum number of kernels that can be chosen is one ($M = 1$), OMKL-GF can achieve higher accuracy with lower execution time in comparison with OMKR-FA. Furthermore, Tables 1, 2 3 and 4 show that MSE obtained by OMKL-GF is comparable to that of Raker while OMKL-GF reaches this with lower execution time. It can also be seen that when $M = 7$, 10 and 17, the proposed OMKL-GF can provide more accurate function approximations compared to Raker. Furthermore, our proposed OMKL-GF is much more scalable in comparison with OMKR and using our OMKL-GF framework can reduce the execution time of function approximation considerably while preserving comparatively accurate estimates.

The results in Tables 1, 2, 3 and 4 also demonstrate that for

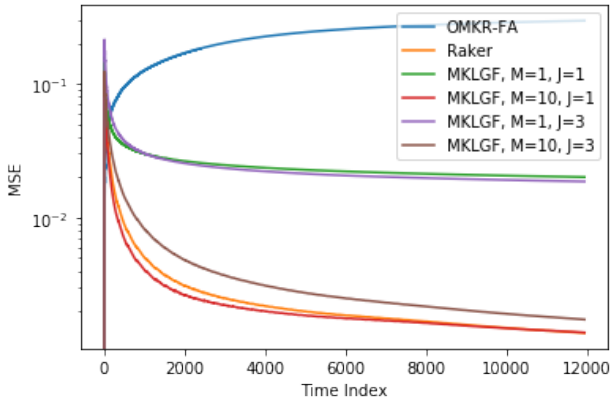

Figure 1. MSE of MKL Algorithms over time for Naval Propulsion Plants dataset.

Table 5. MSE ($\times 10^{-3}$) and execution time of MKL algorithms for Naval Propulsion Plants dataset when $\eta = \frac{1}{\sqrt{t}}$.

Algorithms	M	J	MSE	Execution time (s)
OMKR	-	-	1.0	9690.24s
OMKR-FA	-	-	294.9	2.42s
Raker	-	-	**1.4**	4.14s
OMKL-GF	1	1	20.1	1.24s
OMKL-GF	10	1	**1.4**	2.49s
OMKL-GF	1	3	18.6	**1.24**s
OMKL-GF	10	3	1.7	2.62s

the same value of M, OMKL-GF achieves lower MSE by decreasing J, the number of selective nodes in virtually all cases. It can also be observed that by increasing the value of M, the MSE obtained by OMKL-GF reduces in most of cases. On the other hand, larger M (i.e. the maximum number of kernels chosen at each time) leads to higher the execution time as well. Also, increase in the number of selective nodes leads to increase in the execution time.

Furthermore, Fig. 1 illustrates the MSE of competative algorithms for Naval Propulsion Plants dataset over time. As it can be observed from Fig. 1, our proposed OMKL-GF can provide lower MSE in comparison with OMKR-FA. When $M = 10$ and $J = 1$ in almost all time indices our proposed OMKL-GF provides similar MSE compared to Raker. Also, Fig. 1 shows that increasing the number of selective nodes from 1 to 3 leads to increase in MSE for OMKL-GF. Execution time of all algorithms for Naval Propulsion Plants dataset is reported in Table 5, which shows OMKL-GF runs faster than Raker while it can still provide comparable MSE. This validates the effectiveness of the proposed graph based kernel selection scheme.

6. Conclusion

The present paper developed an online MKL approach for nonlinear function approximation based on random feature approximation. By generating graph feedback, proposed alogrithmic scheme chose a subset of relevant kernels based on losses observed in prior time instants. By choosing a subset of relevant kernels, our proposed approach trimmed irrelevant kernels to enhance the accuracy, and reduce the computational complexity. We proved that our proposed approach achieve sublinear regret in expectation. Numerical tests on several real datasets reveal merits of our proposed algorithmic framework in comparison with other online MKL benchmarks.

References

Akbilgic, O., Bozdogan, H., and Balaban, M. E. A novel hybrid rbf neural networks model as a forecaster. *Statistics and Computing*, 24(3):365 – 375, May 2014.

Alon, N., Cesa-Bianchi, N., Dekel, O., and Koren, T. Online learning with feedback graphs: Beyond bandits. In *Proceedings of Conference on Learning Theory*, volume 40, pp. 23–35, Jul 2015.

Alon, N., Cesa-Bianchi, N., Gentile, C., Mannor, S., Mansour, Y., and Shamir, O. Nonstochastic multi-armed bandits with graph-structured feedback. *SIAM Journal on Computing*, 46(6):1785–1826, 2017.

Asratian, A. S., Denley, T. M. J., and Häggkvist, R. *Bipartite Graphs and Their Applications*. Cambridge University Press, 1998.

Bouboulis, P., Chouvardas, S., and Theodoridis, S. Online distributed learning over networks in rkh spaces using random fourier features. *IEEE Transactions on Signal Processing*, 66(7):1920–1932, Apr 2018.

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

Cesa-Bianchi, N., Freund, Y., Haussler, D., Helmbold, D. P., Schapire, R. E., and Warmuth, M. K. How to use expert advice. *Journal of the ACM*, 44(3):427–485, May 1997.

Coraddu, A., Oneto, L., Ghio, A., Savio, S., Anguita, D., and Figari, M. Machine learning approaches for improving condition-based maintenance of naval propulsion plants. *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment*, 230(1):136–153, 2016.

Cortes, C., Mohri, M., and Talwalkar, A. On the impact of kernel approximation on learning accuracy. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, volume 9, pp. 113–120, May 2010.

Cortes, C., Desalvo, G., Gentile, C., Mohri, M., and Yang, S. Online learning with sleeping experts and feedback graphs. In *Proceedings of International Conference on Machine Learning*, pp. 1370–1378, Jun 2019.

Hazan, E. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2(3-4):157–325, 2016.

Kawala, F., Douzal-Chouakria, A., Gaussier, E., and Dimert, E. Prédictions d'activité dans les réseaux sociaux en ligne. In *4ième conférence sur les modèles et l'analyse des réseaux : Approches mathématiques et informatiques*, pp. 16, France, October 2013.

Littlestone, N. and Warmuth, M. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.

Liu, F., Buccapatnam, S., and Shroff, N. B. Information directed sampling for stochastic bandits with graph feedback. In *Proceedings of AAAI Conference on Artificial Intelligence*, Feb 2018.

Lu, J., Hoi, S. C., Wang, J., Zhao, P., and Liu, Z.-Y. Large scale online kernel learning. *Journal of Machine Learning Research*, 17(47):1–43, 2016.

Lu, Q., Karanikolas, G., Shen, Y., and Giannakis, G. B. Ensemble gaussian processes with spectral features for online interactive learning with scalability. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, volume 108, pp. 1910–1920, Aug 2020.

Mannor, S. and Shamir, O. From bandits to experts: On the value of side-observations. In *Proceedings of International Conference on Neural Information Processing Systems*, pp. 684–692, Dec 2011.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. In *Proceedings of International Conference on Neural Information Processing Systems*, pp. 1177–1184, Dec 2007.

Rudi, A. and Rosasco, L. Generalization properties of learning with random features. In *Proceedings of International Conference on Neural Information Processing Systems*, pp. 3218–3228, 2017.

Sahoo, D., Hoi, S. C., and Li, B. Online multiple kernel regression. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 293–302, 2014.

Sahoo, D., Hoi, S. C. H., and Li, B. Large scale online multiple kernel regression with application to time-series prediction. *ACM Transactions on Knowledge Discovery from Data*, 13(1), Jan 2019.

Scholkopf, B. and Smola, A. J. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001.

Shahrampour, S. and Tarokh, V. Learning bounds for greedy approximation with explicit feature maps from multiple kernels. In *Advances in Neural Information Processing Systems*, pp. 4690–4701, Dec 2018.

Shawe-Taylor, J. and Cristianini, N. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA, 2004.

Shen, Y., Chen, T., and Giannakis, G. Online ensemble multi-kernel learning adaptive to non-stationary and adversarial environments. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, volume 84, pp. 2037–2046, Apr 2018.

Shen, Y., Chen, T., and Giannakis, G. B. Random feature-based online multi-kernel learning in environments with unknown dynamics. *Journal of Machine Learning Research*, 20(1):773–808, Jan 2019.

Vito, S. D., Massera, E., Piga, M., Martinotto, L., and Francia, G. D. On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario. *Sensors and Actuators B: Chemical*, 129(2):750 – 757, 2008.

Vovk, V. A game of prediction with expert advice. *Journal of Computer and System Sciences*, 56(2):153–173, Apr 1998.

Wahba, G. *Spline Models for Observational Data*. Society for Industrial and Applied Mathematics, 1990.

Williams, C. K. I. and Seeger, M. Using the nyström method to speed up kernel machines. In *Proceedings of the International Conference on Neural Information Processing Systems*, pp. 661–667, Jan 2000.

Zhang, X. and Liao, S. Incremental randomized sketching for online kernel learning. In *Proceedings of International Conference on Machine Learning*, pp. 7394–7403, Jun 2019.