Distributed and Quantized Online Multi-kernel Learning

Yanning Shen*, *Member, IEEE*, Saeed Karimi-Bidhendi*, *Student Member, IEEE*, and Hamid Jafarkhani*, *Fellow, IEEE*

***Abstract*—Kernel-based learning has well-documented merits in various machine learning tasks. Most of the kernel-based learning approaches rely on a pre-selected kernel, the choice of which presumes task-specific prior information. In addition, most existing frameworks assume that data are collected centrally at batch. Such a setting may not be feasible especially for large-scale data sets that are collected sequentially over a network. To cope with these challenges, the present work develops an online multi-kernel learning scheme to infer the intended nonlinear function 'on the fly' from data samples that are collected in distributed locations. To address communication efficiency among distributed nodes, we study the effects of quantization and develop a distributed and quantized online multiple kernel learning algorithm. We provide regret analysis that indicates our algorithm is capable of achieving sublinear regret. Numerical tests on real datasets show the effectiveness of our algorithm.**

***Index Terms*—Kernel-based learning, quantization, online optimization, distributed learning**

I. Introduction

The need to collect a massive amount of data and gain access to high computational power to design machine learning algorithms is inherent in numerous applications. In many such tasks, data is collected in a network. Typical examples include units in an IoT system and wearable devices that collect health statistics [1]. The increasing amount of accumulated data in these applications calls for large memory, storage and computational power resources that are usually not available on a single processing unit. Furthermore, one would like to avoid transmitting raw data to a central location to reduce the communication energy and prolong the network lifetime, especially in sensor networks where wireless communication dominates the energy consumption [2, 3, 4, 5, 6, 7, 8]. Also, other challenges like security and privacy may limit the network's ability to transmit the data without restrictions.

One solution to the challenges outlined above is to process the data locally using a decentralized algorithm. One can use on-device intelligence to process the locally collected data and only transmit the relevant content back to a central location for aggregating what has been learned locally. When nodes work together collectively, each node can borrow statistical strength from the local data available at other nodes. This requires developing new distributed machine learning algorithms that only need a limited processed information from each node. The task of distributed and online learning has been well-studied in the literature. Examples include methods that apply alternating direction method of multipliers (ADMM) [9, 10, 11], algorithms based on diffusion protocols [12, 13], distributed support vector machine (SVM) [14] and diffusion-based SVM [15]. In [16, 17], previous linear online distributed methods have been generalized to a non-linear setting in the sense of kernel least mean squares. Since the inception of SVMs, kernel methods have shown great success in numerous tasks such as classification, regression, pattern recognition, and so on [18, 19]. Major efforts have been devoted to scaling up kernel methods in batch settings. Those include approaches to approximate the kernel matrix using low-rank factorization [20, 21, 22], as well as random feature (RF)-based approaches [23, 24, 25].

Tailored for streaming large-scale datasets, online kernel-based learning methods have been designed. A common issue among traditional online kernel learners is that a set of support vectors needs to be stored to represent the kernel-based model; thus, when the dataset is very large, the increasing number of support vectors leads to a notable bottleneck in communication among nodes [26, 27]. Several budget learning methods have been proposed to bound the number of support vectors. Low complexity budgeted kernel learning algorithms include support vector removal [28, 29] [30], support vector projection [31, 32], and support vector merging [33] [34]. A vertical federated kernel learning framework is developed in [35, 36], where the communication cost at Iteration t is $\mathcal{O}(mt)$ given m nodes. Maintaining an affordable budget, online multi-kernel learning (OMKL) methods have been reported for online classification [37, 38, 39] and regression [40, 41]. Recent methods use functional approximation techniques to alleviate the communication cost. In [27], kernel functions are approximated by mapping data into a feature space, where linear online learning algorithms can be applied. Devoid of the need for budget maintenance, online kernel-based learning algorithms based on RF approximation [23] have been developed in [27, 15, 42], but only with a single pre-selected kernel. In [43], a consensus-based decentralized greedy projected penalty method is presented in which function estimation is performed in a reproducing kernel Hilbert space with a consensus constraint; however, since each node needs to transmit its raw local training data to its neighboring nodes at each iteration, the communication complexity increases with the number of nodes, connectivity of the graph, and data dimension. A heterogeneous adaptive kernel learning method is presented in [44] where the consensus constraint is relaxed in favor of enabling nodes to learn distinctive features of their data distribution; however, the communication efficiency suffers

*Y. Shen, S. Karimi-Bidhendi and H. Jafarkhani are with the Dept. of EECS and CPCC, University of California, Irvine, CA; Emails: {yannings,skarimib,hamidj}@uci.edu

from the need to transmit local raw data samples to neighboring nodes. A consensus-based function estimation based on RF approximation is developed in [45], where depending on the number of random features, each node communicates a fixed length weight vector to other nodes. To further reduce the number of transmission rounds, a communication-censored framework is presented in [45], where model parameters are shared only if they differ significantly compared to their values from previous iteration. A similar communication censored ADMM-based algorithm is proposed in [46]. Recently, an online multi-kernel learning method is proposed in [47] that uses an ADMM approach to optimize each kernel function, separately.

In this paper, we extend the centralized online multi-kernel learning framework in [48] to a distributed and quantized online setting, where multi-kernel learning is used for non-linear parameter estimation in classification and regression tasks. In particular, we consider a distributed network in which nodes observe data generated by a non-linear model in an online fashion. In contrast to many prior works such as [15, 43, 45], where single-kernel learning is used, a multi-kernel learning approach is adapted here for parametric estimation; moreover, each node only shares its own estimated gradient of the loss function with its neighbors. The average of the aggregated gradients is then used to locally update the unknown parameters at each node. Unlike previous approaches to ameliorate the communication constraints, such as [45], where an ad-hoc method is used to determine the relevant information to be communicated between the nodes, we take a quantization thoery approach and extend our algorithm so that only quantized information in batch setting is shared between nodes. We further establish the efficacy of our quantized online multi-kernel learning approach through regret analysis. Evaluation over various real datasets for classification and regression tasks shows the superior performance of our algorithm compared to the state-of-the-art decentralized kernel learning methods.

The organization of the paper is as follows. Section II briefly reviews the basics of kernel-based learning, and outlines the main notions used throughout the paper. In Section III, learning procedures for decentralized and online multi-kernel learning is presented and an algorithm is proposed for quantized and mini-batch distributed setting. Regret analysis and further theoretical results are provided in Section IV. Experimental results and performance evaluations are presented in Section V for both classification and regression tasks, and Section VI concludes the paper.

II. Preliminaries

Given samples $\{(\mathbf{x}_1,y_1),\dots,(\mathbf{x}_T,y_T)\}_{t=1}^T$ with $\mathbf{x}_t\in\mathbb{R}^d$ and $y_t\in\mathbb{R}$, the function approximation task is to find a function $f(\cdot)$ such that $y_t=f(\mathbf{x}_t)+e_t$, where e_t denotes noise. Let us assume that $f(\cdot)$ belongs to a reproducing kernel Hilbert space (RKHS) [50] $\mathcal{H}:=\{f|f(\mathbf{x})=\sum_{t=1}^{\infty}\alpha_t\kappa(\mathbf{x},\mathbf{x}_t)\}$, where $\kappa(\mathbf{x},\mathbf{x}_t):\mathbb{R}^d\times\mathbb{R}^d\to\mathbb{R}$ is a kernel function that measures the similarity between $\mathbf{x}$ and $\mathbf{x}_t$. One example for κ is the Gaussian kernel given by $\kappa(\mathbf{x},\mathbf{x}_t)=\exp[-\|\mathbf{x}-\mathbf{x}_t\|^2/(2\sigma^2)]$. A reproducing kernel induces the RKHS norm $\|f\|_{\mathcal{H}}^2:=\sum_t\sum_{t'}\alpha_t\alpha_{t'}\kappa(\mathbf{x}_t,\mathbf{x}_{t'})$. We consider the optimization problem

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

where depending on the application, the cost function $\mathcal{C}(\cdot,\cdot)$ can be selected to be, e.g., the least-squares, the logistic or the hinge loss; $\Omega(\cdot)$ is an increasing function; and, $\lambda>0$ is a regularization parameter that avoids overfitting. According to the representer theorem [15, 49, 50], the optimal solution of (1) admits the form $\hat{f}(\mathbf{x})=\sum_{t=1}^{T}\alpha_t\kappa(\mathbf{x},\mathbf{x}_t):=\boldsymbol{\alpha}^\top\mathbf{k}(\mathbf{x})$ where $\boldsymbol{\alpha}:=[\alpha_1,\dots,\alpha_T]^\top\in\mathbb{R}^T$, and $\mathbf{k}(\mathbf{x}):=[\kappa(\mathbf{x},\mathbf{x}_1),\dots,\kappa(\mathbf{x},\mathbf{x}_T)]^\top$.

Specifying the kernel is critical for kernel learning, since different kernels yield function estimates with different accuracy. To deal with this, combinations of kernels from a prescribed dictionary $\{\kappa_p\}_{p=1}^P$ can be employed in (1). Each combination is also a kernel [49]. With $\bar{\mathcal{H}}$ denoting the RKHS induced by $\bar{\kappa}\in\bar{\mathcal{K}}$, we then solve (1) with $\mathcal{H}$ replaced by $\bar{\mathcal{H}}:=\mathcal{H}_1\bigoplus\cdots\bigoplus\mathcal{H}_P$, where $\{\mathcal{H}_p\}_{p=1}^P$ are the RKHSs induced by $\{\kappa_p\}_{p=1}^P$ [51]. Therefore, the goal now is to find function $f(\mathbf{x}):=\sum_{p=1}^P\bar{w}_pf_p(\mathbf{x})\in\bar{\mathcal{H}}$, with the weights $\{\bar{w}_p:=w_p/\sum_{p=1}^P w_p\}_{p=1}^P$. Plugging this into (1) results in

$$\min_{\{\bar{w}_p\},\{f_p\}}\ \frac{1}{T}\sum_{t=1}^{T}\mathcal{C}\left(\sum_{p=1}^{P}\bar{w}_pf_p(\mathbf{x}_t),y_t\right)+\lambda\Omega\left(\left\|\sum_{p=1}^{P}\bar{w}_pf_p\right\|_{\bar{\mathcal{H}}}^2\right), \tag{2a}$$

$$\text{s. to}\ \sum_{p=1}^{P}\bar{w}_p=1,\ \bar{w}_p\geq 0,\ p\in\mathcal{P},\quad f_p\in\mathcal{H}_p. \tag{2b}$$

Note that (2a) is biconvex if Ω is convex with respect to f. Leveraging biconvexity, existing batch MKL schemes solve (2) via alternating minimization, but typically such schemes are not scalable with regard to P and T [51, 52, 53].

A. RF-based kernel learning

As in [23, 54, 25], we resort to random feature approximation of kernel functions. We consider normalized shift-invariant kernels that satisfy $\kappa(\mathbf{x}_t,\mathbf{x}_{t'})=\kappa(\mathbf{x}_t-\mathbf{x}_{t'})$. For an absolutely integrable $\kappa(\mathbf{x}_t-\mathbf{x}_{t'})$, its Fourier transform $\pi_\kappa(\mathbf{v})$ exists, and due to normalization, we have $\kappa(\mathbf{0})=1$. It can also be viewed as a probability density function (pdf); hence, $\kappa(\mathbf{x}_t-\mathbf{x}_{t'})=\int\pi_\kappa(\mathbf{v})e^{j\mathbf{v}^\top(\mathbf{x}_t-\mathbf{x}_{t'})}d\mathbf{v}:=\mathbb{E}_{\mathbf{v}}\big[e^{j\mathbf{v}^\top(\mathbf{x}_t-\mathbf{x}_{t'})}\big]$. Drawing a number of independent and identically distributed (i.i.d.) samples $\{\mathbf{v}_i\}_{i=1}^D$ from $\pi_\kappa(\mathbf{v})$, the mean can be approximated by the sample average $\hat{\kappa}(\mathbf{x}_t,\mathbf{x}_{t'})=\mathbf{z}_{\mathbf{V}}^\top(\mathbf{x}_t)\mathbf{z}_{\mathbf{V}}(\mathbf{x}_{t'})$, where

$$\mathbf{z}_{\mathbf{V}}(\mathbf{x})=\frac{1}{\sqrt{D}}\left[\sin(\mathbf{v}_1^\top\mathbf{x}),\dots,\sin(\mathbf{v}_D^\top\mathbf{x}),\cos(\mathbf{v}_1^\top\mathbf{x}),\dots,\cos(\mathbf{v}_D^\top\mathbf{x})\right]^\top. \tag{3}$$

Hence, the nonlinear function that is optimal in the sense of (1) can be approximated by

$$\hat{f}^{\mathrm{RF}}(\mathbf{x})=\sum_{t=1}^{T}\alpha_t\mathbf{z}_{\mathbf{V}}^\top(\mathbf{x}_t)\mathbf{z}_{\mathbf{V}}(\mathbf{x}):=\boldsymbol{\theta}^\top\mathbf{z}_{\mathbf{V}}(\mathbf{x}), \tag{4}$$

where $\boldsymbol{\theta}^\top := \sum_{\tau=1}^{T} \alpha_\tau \mathbf{z}_{\mathbf{V}}^\top(\mathbf{x}_\tau)$ is the new weight vector of size $2D$. As a result, the loss function is

$$\begin{aligned}\mathcal{L}\big(f(\mathbf{x}_t)\big) :=&\,\mathcal{C}(f(\mathbf{x}_t), y_t) + \lambda\Omega\left(\|f\|_{\mathcal{H}}^2\right)\\ =&\,\mathcal{C}\big(\boldsymbol{\theta}^\top \mathbf{z}_{\mathbf{V}}(\mathbf{x}_t), y_t\big) + \lambda\Omega\left(\|\boldsymbol{\theta}\|^2\right). \end{aligned} \tag{5}$$

Resorting to RF approximation, given $\mathbf{x}_t$, for each p, an RF vector $\mathbf{z}_p(\mathbf{x}_t)$ can be generated from pdf $\pi_{\kappa_p}(\mathbf{v})$ (cf. (3)), where $\mathbf{z}_p(\mathbf{x}_t) := \mathbf{z}_{\mathbf{V}_p}(\mathbf{x}_t)$. Hence, for each p and slot t

$$\hat{f}_{p,t}^{\mathrm{RF}}(\mathbf{x}_t) = \boldsymbol{\theta}_{p,t}^\top \mathbf{z}_p(\mathbf{x}_t), \tag{6}$$

and $\boldsymbol{\theta}_{p,t}$ can be updated via

$$\boldsymbol{\theta}_{p,t+1} = \boldsymbol{\theta}_{p,t} - \eta \nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top \mathbf{z}_p(\mathbf{x}_t), y_t), \tag{7}$$

where η is the learning rate, and $\nabla\mathcal{L}(\boldsymbol{\theta}_t^\top \mathbf{z}_p(\mathbf{x}_t), y_t)$ is the gradient at $\boldsymbol{\theta} = \boldsymbol{\theta}_{p,t}$. While the un-normalized weights can be updated using exponentiated gradient descent as

$$w_{p,t+1} = w_{p,t} \exp\left(-\eta \mathcal{L}_t\left(\hat{f}_{p,t}^{\mathrm{RF}}(\mathbf{x}_t)\right)\right), \tag{8}$$

where $\eta \in (0,1)$ is a constant stepsize. The normalized weights are obtained as $\bar{w}_{p,t} := w_{p,t}/\sum_{p=1}^{P} w_{p,t}$ using $\{w_{p,t}\}$ in (8) [54].

III. Distributed and quantized OMKL

So far, it has been assumed that the data and the gradient information are available perfectly and without any error or loss at one central location. However, it is not practical to transmit the massive amount of real-world data to one central location and the transmitted information may not be perfect. Therefore, there is a need for scalable distributed algorithms that can process data in distributed nodes and fuse the learning result in an efficient manner. Specifically, suppose there exist J nodes, where the j th node collects data points $\{\mathbf{x}_t^j, y_t^j\}$ over time. Note that in the present work, we mainly focus on the case where nodes in the network can communicate and exchange information at each time, while only data are collected at different locations. In this scenario, the online function learning problem can be solved via

$$\min_{\{\bar{w}_p^j\},\{f_p\}} \frac{1}{JT}\sum_{j=1}^{J}\sum_{t=1}^{T} \mathcal{C}\left(\sum_{p=1}^{P} \bar{w}_p^j f_p(\mathbf{x}_t^j), y_t^j\right) + \lambda\Omega\left(\left\|\sum_{p=1}^{P} \bar{w}_p^j f_p\right\|_{\bar{\mathcal{H}}}^2\right), \tag{9a}$$

$$\text{s. to} \quad \sum_{p=1}^{P} \bar{w}_p^j = 1,\ \bar{w}_p^j \geq 0,\ p \in \mathcal{P} \tag{9b}$$

$$f_p \in \mathcal{H}_p,\ p \in \mathcal{P}. \tag{9c}$$

In the following sections, we explore how to solve (9) in an efficient and distributed fashion.

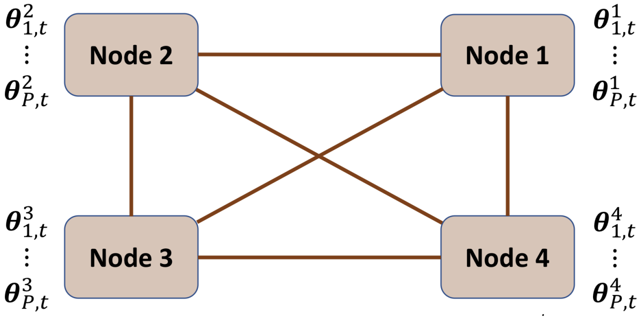

$\boldsymbol{\theta}_{1,t}^2$
⋮
$\boldsymbol{\theta}_{P,t}^2$
Node 2
Node 1
$\boldsymbol{\theta}_{1,t}^1$
⋮
$\boldsymbol{\theta}_{P,t}^1$
$\boldsymbol{\theta}_{1,t}^3$
⋮
$\boldsymbol{\theta}_{P,t}^3$
Node 3
Node 4
$\boldsymbol{\theta}_{1,t}^4$
⋮
$\boldsymbol{\theta}_{P,t}^4$

Figure 1: A network with $J = 4$ nodes where $\boldsymbol{\theta}_{p,t}^j$ denotes the local parameter for the p-th kernel at node j and time slot t.

A. Distributed OMKL

Suppose data samples are collected by J nodes and the nodes communicate with each other point-to-point. In particular, our distributed network can be represented by a complete graph with J vertices where each vertex corresponds to one node and each edge symbolizes the point-to-point communication between two nodes, as depicted in Figure 1. Specifically, each node, say $j \in \{1, \cdots, J\}$, observes local data samples $(\mathbf{x}_{j,t}, y_{j,t})$ and is associated with local parameters $\boldsymbol{\theta}_{p,t}^j$ corresponding to the $p-$th kernel at node j and time slot t. An edge (j_1, j_2) in the network is defined as the connection between nodes j_1 and j_2 in the network to represent the point-to-point communication between nodes j_1 and j_2. In addition, every node stores a copy of the current values of the model variables. In each iteration, every node obtains local updates for $\boldsymbol{\theta}_p$ and communicates these updates to all peers. Then, every node aggregates the received updates, locally, to generate a global model. Specifically, upon acquiring each data sample, the distributed online RF-based OMKL is realized by

$$\begin{aligned}\boldsymbol{\theta}_{p,t+1}^j =&\,\boldsymbol{\theta}_{p,t}^j - \frac{\eta}{J}\sum_{j=1}^{J} \nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top \mathbf{z}_p(\mathbf{x}_t^j), y_t^j)\\ :=&\,\boldsymbol{\theta}_{p,t}^j - \eta\nabla\bar{\mathcal{L}}_t(\boldsymbol{\theta}_{p,t}),\end{aligned} \tag{10}$$

where the second equality holds because of the linearity of the gradient and $\bar{\mathcal{L}}_t(\theta_{p,t}) := -\frac{\eta}{J}\sum_{j=1}^{J}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top \mathbf{z}_p(\mathbf{x}_t^j), y_t^j)$, where $\boldsymbol{\theta}_{p,t} := \{\boldsymbol{\theta}_{p,t}^j\}_{j=1:J}$. Note that the updates in (10) automatically results in $\boldsymbol{\theta}_{p,t+1}^j = \boldsymbol{\theta}_{p,t+1}$, $\forall j$. Meanwhile, the combining weights can be updated via

$$w_{p,t+1}^j = w_{p,t}^j \exp\left(-\eta\bar{\mathcal{L}}_t\left(\boldsymbol{\theta}_{p,t}\right)\right) \tag{11}$$

Similarly, the update in (11) naturally leads to $w_{p,t+1}^j = w_{p,t+1}$, $\forall j$. Moreover, instead of communicating upon acquiring each data sample, every node can communicate the variables and aggregate the global model only after observing B data samples. Specifically, upon acquiring B data samples, the distributed mini-batch OMKL can be realized by

$$\boldsymbol{\theta}_{p,t+1}^j = \boldsymbol{\theta}_{p,t}^j - \frac{\eta}{JB}\sum_{j=1}^{J}\sum_{b=1}^{B} \nabla\mathcal{L}((\boldsymbol{\theta}_{p,t}^j)^\top \mathbf{z}_p(\mathbf{x}_t^{j,b}), y_t^{j,b}), \tag{12}$$

where again the second equality holds due to the linearity of the gradient. Meanwhile, the combining weights can be updated via

$$w^j_{p,t+1} = w^j_{p,t} \exp\left(-\frac{\eta}{JB}\sum_{j=1}^{J}\sum_{b=1}^{B}\mathcal{L}_t\left(\hat{f}^{\mathrm{RF}}_{p,t}(\mathbf{x}_t^{j,b})\right)\right). \quad (13)$$

Obviously, choosing $B = 1$ results in the previous case of communicating and updating after acquiring every sample data.

B. Quantized and distributed OMKL

In practice, it is often infeasible to send real-valued information among distributed nodes. Hence, the information needs to be quantized first before transmission. Let us consider a parameterizable lossy-compression scheme for gradient vectors [55]. We define the scalar quantization function $Q_M(g_i)$, that quantizes the ith element of $\mathbf{g}$ by

$$Q_M(g_i) = \|\mathbf{g}\|_2 \ \mathrm{sign}(g_i) \ \epsilon_i(\mathbf{g}, M), \quad (14)$$

where M denotes the number of quantization levels, and $\epsilon_i(\mathbf{g}, M)$ is a random variable. Letting $0 \leq m < M$ be an integer that characterizes the quantization interval corresponding to $|g_i|/\|\mathbf{g}\|_2$, if $\mathbf{g} \neq \mathbf{0}$, $\epsilon_i(\mathbf{g}, M)$ is defined as

$$\epsilon_i(\mathbf{g}, M) = \begin{cases} m/M & \text{with probability } 1 - \frac{M|g_i|}{\|\mathbf{g}\|_2} + m \\ (m+1)/M & \text{otherwise} \end{cases}. \quad (15)$$

Then, $\tilde{\mathbf{g}} := Q_M(\mathbf{g})$ denotes the quantized version of the vector $\mathbf{g}$ when (14) is applied to every entry of $\mathbf{g}$. Incorporating the above quantization scheme, the $\boldsymbol{\theta}^j_{p,t}$ can be updated via

$$\boldsymbol{\theta}^j_{p,t+1} = \boldsymbol{\theta}^j_{p,t} - \frac{\eta}{J}\sum_{j=1}^{J}\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}^{\top}_{p,t}\mathbf{z}_p(\mathbf{x}^j_t), y^j_t), \quad (16)$$

while the combining weights can be updated via

$$w^j_{p,t+1} = w^j_{p,t} \exp\left(-\frac{\eta}{J}\sum_{j=1}^{J}\tilde{\mathcal{L}}_t\left(\hat{f}^{\mathrm{RF}}_{p,t}(\mathbf{x}^j_t)\right)\right). \quad (17)$$

Hence, the distributed and quantized OMKL can be realized as summarized in Algorithm 1.

C. Communication efficiency

In the peer-to-peer communication protocol considered in this work, each node uses a single broadcast to share its local gradient function; hence, J broadcasts are required, i.e. one for each node, per time slot. In a fusion-based communication protocol, where nodes send their local gradients to a fusion center and the fusion center transmits the aggregated gradient back to all nodes, a total number of $J+1$ broadcasts is needed per time slot. Therefore, the peer-to-peer method is suitable for small local networks where nodes can communicate directly. However, in large networks, where nodes are far from each other, the fusion-based protocol is favorable since the fusion center is equipped with more transmission power and can act as a relay node to make these communications feasible. For a concrete analysis of the communication efficiency, we study the number of bits required per broadcast since the total number of broadcasts per time slot depends on the selected communication protocol.

Algorithm 1 Distribute and Quantized OMKL

```
Input: Kernels $\kappa_p$, $p = 1, \ldots, P$, step size $\eta > 0$, and number of random features $D$.
Initialization: $\boldsymbol{\theta}_1 = \mathbf{0}$.
for $t = 1, 2, \ldots, T$ do
    for $j = 1, 2, \ldots, J$ node do
        Receive a streaming datum $\mathbf{x}^j_t$.
        Construct $\mathbf{z}_p(\mathbf{x}^j_t)$ via (3) using $\kappa_p$ for $p = 1, \ldots, P$.
        Predict $\hat{f}^{\mathrm{RF}}_t(\mathbf{x}^j_t) := \sum_{p=1}^{P}\bar{w}_{p,t}\hat{f}^{\mathrm{RF}}_{p,t}(\mathbf{x}^j_t)$ with $\hat{f}^{\mathrm{RF}}_{p,t}(\mathbf{x}^j_t)$ in (6).
        Observe loss function $\mathcal{L}_t$, incur $\mathcal{L}^j_t(\hat{f}^{\mathrm{RF}}_t(\mathbf{x}^j_t))$.
        for $p = 1, \ldots, P$ do
            Obtain loss $\mathcal{L}(\boldsymbol{\theta}^{\top}_{p,t}\mathbf{z}_p(\mathbf{x}^j_t), y_t)$ or $\mathcal{L}_t(\hat{f}^{\mathrm{RF}}_{p,t}(\mathbf{x}^j_t))$.
            Broadcast and obtain from other nodes $\tilde{\mathcal{L}}_t(\hat{f}^{\mathrm{RF}}_{p,t}(\mathbf{x}^j_t))$ and $\tilde{\nabla}\mathcal{L}_t(\hat{f}^{\mathrm{RF}}_{p,t}(\mathbf{x}^j_t))$
            Update $\boldsymbol{\theta}^j_{p,t+1}$ via (16).
            Update $w^j_{p,t+1}$ via (17).
        end for
    end for
end for
```

Instead of communicating the raw data, our distributed and quantized OMKL algorithm requires nodes to share a gradient vector of fixed length due to RF approximation of kernel functions. Let P be the number of kernels and consider D i.i.d. samples per kernel as in Eq. (3). If M is the number of quantization levels, then the total number of bits that each node communicates per iteration is:

$$\text{bits/node/iteration} = 2D \times P \times \log_2(M). \quad (18)$$

For a $32-$bit CPU architecture and a $d-$dimensional raw data, $32d$ bits are required for each node per iteration. Therefore, communication resources are saved as long as we have:

$$32d > 2D \times P \times \log_2(M). \quad (19)$$

A similar inequality can be written for a $64-$bit hardware. As we will see in Section V, our quantized OMKL algorithm leads to communication efficiency in many applications, especially for high-dimensional data, where the communication bottleneck is more prominent.

IV. Regret analysis

In this section, we study the regret analysis of the proposed quantized and distributed online MKL algorithm. To analyze the performance, we assume that the following conditions are satisfied.

(as1) *Per slot t, the loss function $\mathcal{L}(\boldsymbol{\theta}^{\top}\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t), y_t)$ in* (5) *is convex w.r.t. $\boldsymbol{\theta}$.*

(as2) *For $\boldsymbol{\theta}$ in a bounded set $\boldsymbol{\Theta}$ with $\|\boldsymbol{\theta}\| \leq C_\theta$, the loss is bounded; that is, $\mathcal{L}(\boldsymbol{\theta}^{\top}\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t), y_t) \in [-C, C]$ and has bounded gradient, such that $\|\nabla\mathcal{L}(\boldsymbol{\theta}^{\top}\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t), y_t)\| \leq L$.* Without loss of generality and for the notational convenience, we assume $C = 1$, i.e., the loss function is bounded in $[-1, 1]$.

Convexity of the loss under (as1) is satisfied by the popular loss functions including the square loss and the hinge loss. In addition, (as2) ensures that the losses and their gradients are bounded, meaning they are L-Lipschitz continuous. While boundedness of the losses commonly holds since $\|\boldsymbol{\theta}\|$ is bounded, Lipschitz continuity is also not restrictive. Considering kernel-based regression as an example, the gradient is $(\boldsymbol{\theta}^\top \mathbf{z}_{\mathbf{V}}(\mathbf{x}_t) - y_t)\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t) + \lambda\boldsymbol{\theta}$. Since the loss is bounded, e.g., $\|\boldsymbol{\theta}^\top \mathbf{z}_{\mathbf{V}}(\mathbf{x}_t) - y_t\| \leq 1$, and the RF vector in (3) can be bounded as $\|\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t)\| \leq 1$, the constant is $L := 1 + \lambda C_\theta$ using the Cauchy-Schwartz inequality.

In contrast to the centralized regret analysis, we define the regret in the distributed setting as

$$\mathrm{Reg}^{\mathrm{ds}}_{\mathcal{A}}(T) := \mathbb{E}\left[\sum_{t=1}^{T}\sum_{j=1}^{J}\mathcal{L}_t(\hat{f}_t(\mathbf{x}_t^j))\right] - \sum_{t=1}^{T}\sum_{j=1}^{J}\mathcal{L}_t(\hat{f}^*_p(\mathbf{x}_t^j)), \tag{20}$$

where

$$p = \arg\min_{\{q\in 1,\cdots,P\}} \sum_{t=1}^{T}\sum_{j=1}^{J}\mathcal{L}_t(\hat{f}^*_q(\mathbf{x}_t^j)), \tag{21}$$

$$\text{with} \quad \hat{f}^*_q(\cdot) := (\boldsymbol{\theta}^*_q)^\top \mathbf{z}_q(\cdot) = \arg\min_{f\in\mathcal{F}_q} \sum_{t=1}^{T}\sum_{j=1}^{J}\mathcal{L}_t(f(\mathbf{x}_t^j)).$$

$$\mathcal{F}_q := \{\hat{f}_q | \hat{f}_q(\mathbf{x}) = \boldsymbol{\theta}^\top \mathbf{z}_q(\mathbf{x}),\, \forall \boldsymbol{\theta} \in \mathbb{R}^{2D}\}$$

We first establish the regret of the distributed and online approach in the following lemma.

Lemma 1: *Let us assume (as1), (as2), the sequences $\{\hat{f}_{p,t}\}$ and $\{\bar{w}_{p,t}\}$ generated by the distributed OMKL satisfy*

$$\begin{aligned}&\sum_{t=1}^{T}\sum_{j=1}^{J}\mathcal{L}_t\Big(\sum_{p=1}^{P}\bar{w}_{p,t}\hat{f}_{p,t}(\mathbf{x}_t^j)\Big) - \sum_{t=1}^{T}\sum_{j=1}^{J}\mathcal{L}_t\left(\hat{f}^*_p(\mathbf{x}_t^j)\right)\\ \leq&\frac{J\ln P}{\eta} + \frac{J\|\boldsymbol{\theta}^*_p\|^2}{2\eta} + \frac{\eta J L^2 T}{2} + \eta J T,\end{aligned} \tag{22}$$

where $\boldsymbol{\theta}^*_p$ is associated with the best RF function approximant $\hat{f}^*_p(\mathbf{x}) = \left(\boldsymbol{\theta}^*_p\right)^\top \mathbf{z}_p(\mathbf{x})$.

Proof: See Appendix A. ■

Using Lemma 1 as a step stone, the performance of the distributed and quantized online function learning approach can be bounded as follows.

Theorem 1: Let us assume (as1), (as2), the sequences $\{\hat{f}_{p,t}\}$ and $\{\bar{w}^j_{p,t}\}$ generated by the quantized and distributed OMKL satisfy

$$\begin{aligned}&\mathrm{Reg}^{\mathrm{ds}}(T)\\ =&\mathbb{E}\Big[\sum_{t=1}^{T}\sum_{j=1}^{J}\mathcal{L}_t\Big(\sum_{p=1}^{P}\bar{w}^j_{p,t}\hat{f}_{p,t}(\mathbf{x}_t^j)\Big)\Big] - \sum_{t=1}^{T}\sum_{j=1}^{J}\mathcal{L}_t\left(\hat{f}^*_p(\mathbf{x}_t^j)\right)\\ \leq&\frac{J\ln P}{\eta} + \frac{J\|\boldsymbol{\theta}^*_p\|^2}{2\eta} + \frac{\eta J L^2 T}{2} + \frac{\eta J \sigma_L^2 T}{2} + \eta J T,\end{aligned} \tag{23}$$

where $\sigma_L^2 = \min\left(\frac{2D}{M^2}, \frac{\sqrt{2D}}{M}\right) L^2$.

Proof: See Appendix B. ■

It can be readily established from (23) that the novel quantized and distributed online multi kernel learning approach can achieve sublinear regret if $\eta = 1/O(\sqrt{T})$.

Remark: Note that the regret bound is of order $\mathcal{O}(\ln P)$, which means that the regret bound increases with the increase of P. However, note that there is a difference between tighter regret bounds and better performance. Specifically, if P is smaller, then the regret incurred by the online learner is smaller compared with the best batch learner with the same P. While as P increases, the performance of the best batch learner also improves. Hence, the performance of the online learner is still likely to improve even though the regret bounds is looser. In addition, the randomness in our regret analysis comes from the randomness in the quantization scheme in (14), where the term ϵ_i is a random variable defined in (15), see Appendix for more detailed proof.

V. Experiments

We evaluate our distributed OMKL algorithm against our implementation of three state-of-the-art algorithms, recently proposed in the literature, i.e., Random Fourier Features Distributed Online Kernel-based Learning (RFF-DOKL) [15], Greedy Projected Penalty Method (GPPM) [43], and an extension of the batch random feature alternating direction method of multipliers (RF-ADMM) algorithm [45] to the online setting, where the local updates per time slot t are based on the local data at time t. The RFF-DOKL algorithm is a single-kernel online learning method that uses RF approximation to resort to parametric estimation and circumvent the curse of dimensionality. In each iteration, neighboring nodes transmit their local estimated parameters to each other; then, each node uses a weighted sum of the received local parameters according to the Metropolis rule to update its parameters via gradient descent. The RF-ADMM algorithm is similar to the RFF-DOKL in the sense that it is also a single-kernel method that uses RF approximation to make the method scalable; however, instead of a gradient-based approach, the alternating direction method of multipliers is used to update the parameters at each iteration. The GPPM algorithm is also a single-kernel method; however, instead of using RF approximation to avoid the curse of dimensionality, the kernel orthogonal matching pursuit method is used to keep only relevant training samples for model prediction. In what follows, we compare these methods against our distributed OMKL algorithm which is a multi-kernel method that uses RF approximation for parametric estimation, and a gradient descent approach to update the model parameters. Experiments are carried out for both classification and regression tasks, as outlined below.

A. Classification

Simulations for the classification task are performed on these datasets: (i) Activity ($n = 7352, d = 30$); (ii) Adult ($n = 32562, d = 123$); (iii) Banana ($n = 5300, d = 2$); (iv) Credit-Card ($n = 30000, d = 23$); (v) Device ($n = 3583, d = 60$); (vi) EEG ($n = 14980, d = 14$); (vii) MNIST ($n = 70000, d = 784$); and (viii) Spam-Based ($n = 4601, d = 57$). The Activity dataset contains samples gathered from a group of volunteers wearing a smartphone to monitor their activity, and the binary response variable represents the status of the volunteer as walking or not walking [56]. The Adult dataset uses the census data to predict whether a person makes over $\$50,000$/yr. The Banana dataset consists of two banana shaped clusters, and is obtained from Kaggle. The Credit-Card dataset employs a binary variable, i.e., default or non-default payment methods, as the response variable based on credit history etc. of clients [57]. The Device dataset consists of electricity readings over 15 minutes intervals from different households sampled within a month, and the binary response variable represents whether the electronic device used at a certain interval is a kettle or dishwasher [58]. The EEG dataset contains a continuous EEG measurement with the Emotiv EEG Neuroheadset where the eye state was detected via a camera during the measurement, and the binary response variable represents the state of the eye as either open or closed. The MNIST dataset contains images of handwritten digits for which we classified the digit 8 versus others [59]. The Spam-Based dataset uses attributes such as the number of consecutive capital letters and frequency usage of certain words to classify emails as Spam or Non-Spam. The Adult, Credit-Card and Spam-Based datasets are obtained from the UCI machine learning repository [60].

The network consists of $J = 20$ nodes that communicate with each other point-to-point, and data arrives at each node sequentially. All methods use the Gaussian kernel, i.e., $\kappa(\mathbf{x}_t, \mathbf{x}'_t) = \exp\left[-\|\mathbf{x}_t - \mathbf{x}'_t\|^2/2\sigma^2\right]$, and the parameter σ is fined-tuned via grid-search for each method and dataset, separately. The parameter σ for the RF-ADMM algorithm is set to 1 for all datasets, except for the Banana dataset for which $\sigma = 0.6$ is used. The GPPM algorithm uses $\sigma = 1$ for all datasets. The parameter σ in the RFF-DOKL algorithm is set to 1, 4, 0.6, 1, 1, 1, 5, 2 for the Activity, Adult, Banana, Credit-Card, Device, EEG, MNIST, and Spam-Based datasets, respectively. The distributed OMKL algorithm uses three Gaussian kernels and their variances are optimized for each dataset. The value of σ for these three kernels is provided in Table IV. The regularization parameter is set to $\lambda = 10^{-3}$ for all algorithms. The number of features for random feature (RF) approximation in distributed OMKL, RF-ADMM and RFF-DOKL algorithms are set to $D = 50$. For each method and dataset, z$-$normalization or $[0, 1]-$normalization is applied only if it results in improvement and the original range of the data leads to instability and divergence. The presented distributed OMKL algorithm uses the kernel logistic regression (KLR) loss function for classification. A similar loss function is considered in [43] for the GPPM algorithm. Since the classification task is not addressed in [45], we use the KLR loss function for the RF-ADMM algorithm. Hinge loss is considered in [15] for the RFF-DOKL algorithm. The learning rate is optimized for each method and dataset, separately, either as $\mathcal{O}(1/t)$ or $\mathcal{O}(1/\sqrt{t})$. Since RF-ADMM, RFF-DOKL and our proposed distributed OMKL algorithms are RF-based methods and inherently stochastic, we performed the simulations for ten different sets of i.i.d. random features, and reported the mean and standard deviation of the resulting misclassification rate. However, the GPPM algorithm is deterministic and only performed once.

The misclassification rate and run time of all algorithms are summarized in Tables I and II, respectively. As shown in Table I, our method outperforms other algorithms in five out of eight datasets in terms of the predictive accuracy. Although our distributed online multi-kernel learning algorithm leads to a better performance compare to other methods, it does not sacrifice the computational complexity in favor of better predictive accuracy. As shown in Table II, our method has lower running time compared to other algorithms in all datasets. Both GPPM and RF-ADMM algorithms perform poorly in terms of computational complexity. In order to avoid the memory requirement to grow linearly with time, the GPPM algorithm applies the destructive KOMP procedure at each time step, which makes the algorithm too slow for many practical applications. The RF-ADMM algorithm, equipped with the KLR loss function, solves a minimization task for which a close form solution does not exist. Since a gradient descent method needs to be applied at each time step to numerically solve this optimization problem, the algorithm suffers from high computational complexity. For a fixed exemplary set of random features, Figure 2 shows the evolution of misclassification rate versus time for the four algorithms considered in the paper.

Next, we perform simulations for distributed mini-batch OMKL and distributed quantized OMKL algorithms. The parameter B in mini-batch OMKL is set to $B = 5$ while all other parameters are left unchanged. Similarly, the number of quantization levels in the quantized OMKL algorithm is set to $M = 2$, 4, and 16, corresponding to 1, 2, and $4-$bit representation of the gradient elements, respectively, while the remaining parameters are kept unaltered. For a fair comparison between distributed, mini-batch and quantized OMKL algorithms, a fixed set of random features are used for each dataset. Due to the small number of data in Activity, Device and Spam-Based datasets, which causes the distributed mini-batch OMKL not to reach convergence, the simulations are carried out for three epochs for these three datasets.

The misclassification rate for both mini-batch and quantized distributed OMKL algorithms are summarized in Table III. As expected, mini-batch training increased the error rate since nodes are able to communicate with each other only after B new observations; however, this improved the communication efficiency of the distributed mini-batch OMKL compared to the distributed OMKL by a factor of B. Simulation results for quantized OMKL show that the error rate is slightly increased compared to OMKL without quantization due to the information loss in the gradient vector; however, the misclassification rate approaches that of the OMKL algorithm as the number of quantization levels increases. The effect of mini-batch and quantized online multi-kernel learning algorithms on the progression of the misclassification rate is illustrated in Figure 3.

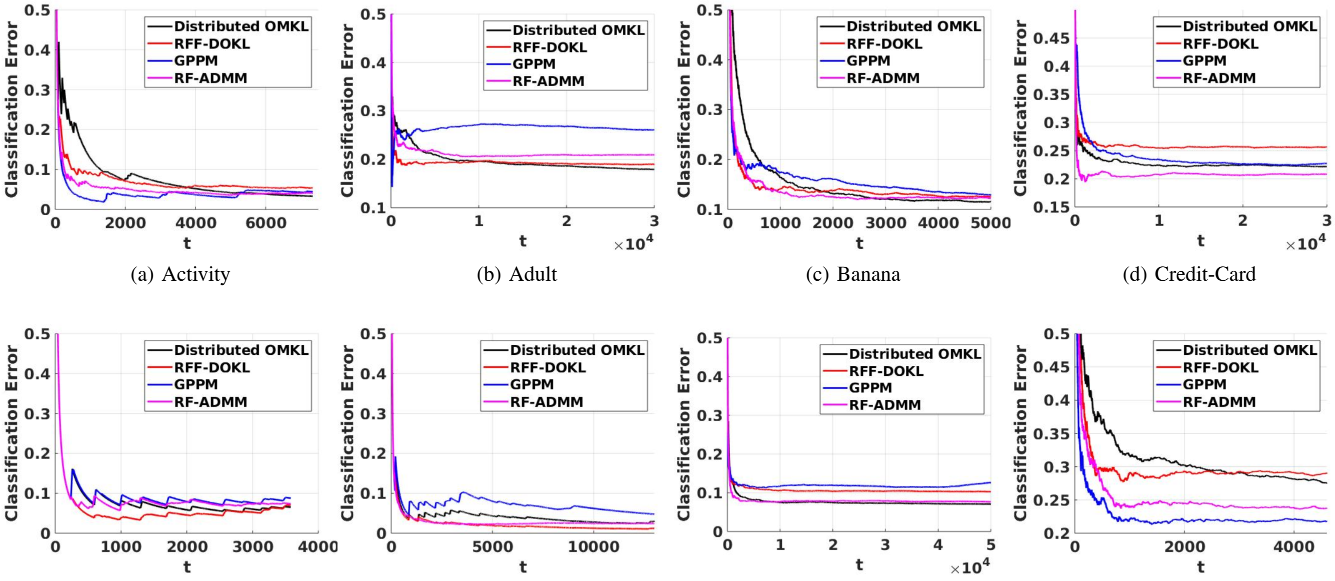
Next, we compare the communication efficiency of these algorithms. Since the raw gradient vector is communicated in the RFF-DOKL algorithm, its communication cost for each node per iteration is $32D$ bits for a $32-$bit hardware.

Table I: Mean (SD) of the misclassification rate $(\%)$ for different algorithms.

	Activity	Adult	Banana	Credit-Card	Device	EEG	MNIST	Spam-Based
RFF-DOKL	5.45(0.30)	18.93(0.44)	12.62(0.94)	25.58(0.45)	7.02(0.47)	**1.22(0.03)**	10.40(0.25)	29.62(0.43)
GPPM	4.47	26.11	12.85	22.73	8.77	4.47	11.92	**21.76**
RF-ADMM	4.10(0.19)	20.86(0.88)	11.82(1.32)	**20.84(0.21)**	7.11(1.32)	2.43(0.11)	7.71(0.07)	23.86(0.57)
OMKL	**3.40(0.17)**	**17.94(0.19)**	**11.53(0.22)**	22.19(0.02)	**6.63(0.16)**	3.14(0.15)	**7.10(0.32)**	27.74(0.43)

Table II: Running time (d: days, h: hours, m: minutes, s:seconds).

	Activity	Adult	Banana	Credit-Card	Device	EEG	MNIST	Spam-Based
RFF-DOKL	1.3s	12.8s	1.1s	10.6s	0.7s	2.7s	1m:9.5s	0.8s
GPPM	11m:20.5s	30m:44.7s	1m:2.9s	43.3s	4m:57.3s	> 1d	> 2d	1m:40.8s
RF-ADMM	8m:1.4s	36m:45.6s	7m:11.1s	33m:41.6s	2m:52.8s	20m:43.9s	5h:36m:24.8s	4m:56.4s
OMKL	**1.2s**	**2.6s**	**0.9s**	**2.0s**	**0.6s**	**2.4s**	**46.6s**	**0.4s**



(a) Activity (b) Adult (c) Banana (d) Credit-Card

(e) Device (f) EEG (g) MNIST (h) Spam-Based

Figure 2: Misclassification rate versus the number of observered training samples per node.

Similar argument reveals that each node in the RF-ADMM algorithm uses $32 \times 2D$ bits per iteration for a $32-$bit CPU operating mode. Since our algorithm uses $P = 3$ kernels, Eq. (18) implies that the quantized OMKL algorithm outperforms RFF-DOKL and RF-ADMM algorithms in terms of communication efficiency if the number of quantization bits, i.e. $\log_2(M)$, is less than or equal to 5 and 10, respectively. As shown in Table III, the performance of the quantized OMKL algorithm converges to that of the OMKL algorithm even for a $4-$bit quantization. Hence, our method outperforms both RFF-DOKL and RF-ADMM algorithms in terms of communication efficiency.

Note that the raw data is communicated in the GPPM algorithm; thus, its communication cost is $32d$ bits for a $32-$bit hardware. According to Eq. (19), our distributed and $4-$bit quantized OMKL algorithm outperforms the GPPM algorithm in terms of communication efficiency if:

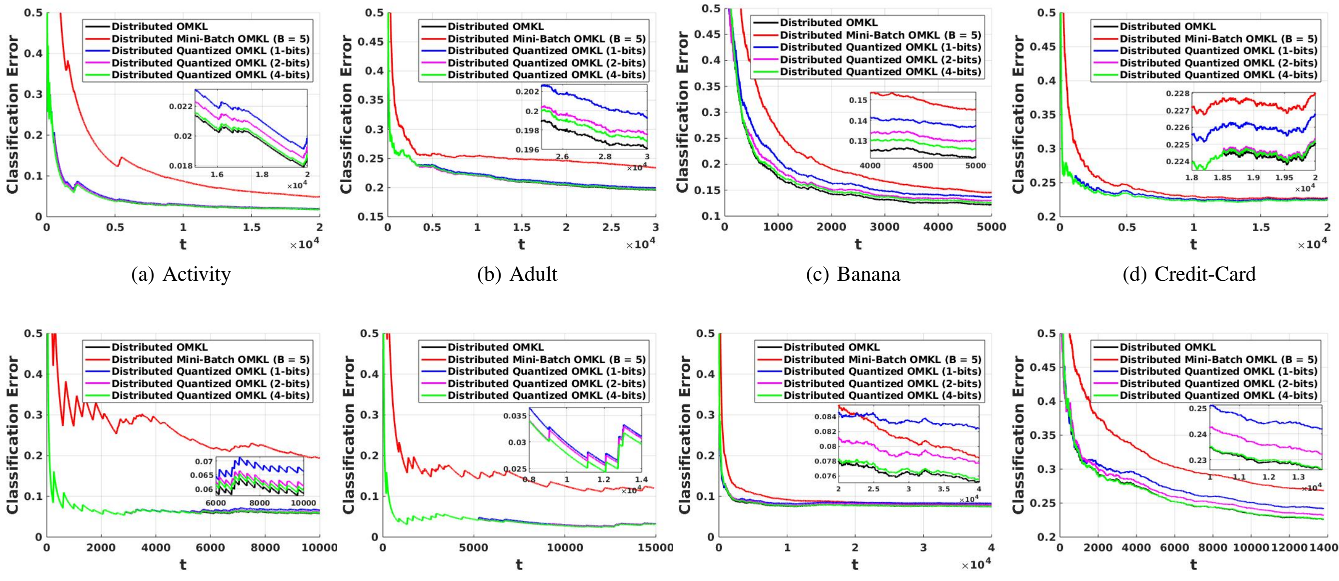
$$d > \frac{2D \times 3 \times 4}{32} = 37.5, \tag{24}$$

which is the case for Adult, Device, MNIST, and Spam-Based datasets. Eq. (24) indicates that for high-dimensional datasets, our algorithm significantly reduces the communication cost compared to sending the raw data. For instance, the distributed and $4-$bit quantized OMKL algorithm reduces the number of communicated bits by a factor of 21 compared to sending the raw data for the MNIST dataset, where $d = 784$. However, for datasets such as Banana, where data is already low-dimensional, the number of bits that each node transmits per iteration is not significant.

Finally, we study the effect of hyperparameters P and D, i.e., the number of kernels and i.i.d. samples, on the classification performance of the distributed OMKL algorithm. We consider three different values $D = 20, 50$, and 100 for the number of i.i.d. samples and three different values $P = 1, 3$, and 5 for the number of Gaussian kernels. Table IV summarizes the mean and standard deviation of the classification error for different datasets. Note that a larger D makes the sample average to be a more accurate approximation of the expected value and kernel

Table III: Misclassification rate $(\%)$ for distributed mini-batch and quantized OMKL algorithms.

	OMKL	**Mini-batch OMKL**	**Quantized OMKL**		
		$B = 5$	$M = 2$	$M = 4$	$M = 16$
Activity	1.73	4.57	1.86	1.81	1.75
Adult	19.50	23.10	19.81	19.67	19.61
Banana	11.96	14.15	13.40	12.74	12.36
Credit-Card	22.17	22.37	22.34	22.23	22.18
Device	6.22	20.36	7.00	6.47	6.32
EEG	3.26	12.03	3.40	3.36	3.26
MNIST	7.33	7.44	8.07	7.57	7.36
Spam-Based	22.61	26.83	24.17	23.20	22.63



(a) Activity

(b) Adult

(c) Banana

(d) Credit-Card

(e) Device

(f) EEG

(g) MNIST

(h) Spam-Based

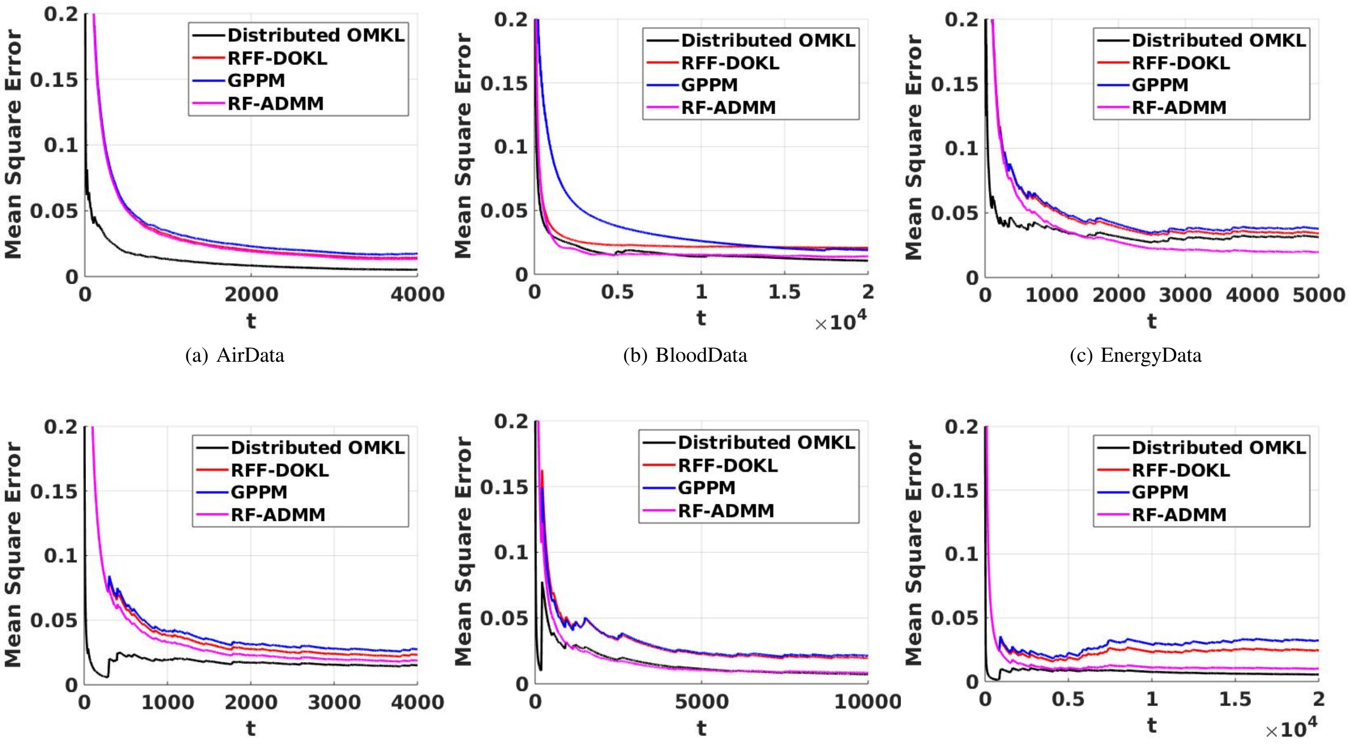
Figure 3: Misclassification rate versus the number of observered training samples per node.

functions; thus, improving the model's accuracy. Moreover, a larger P increases the representability of the function and enhances the performance of the predictive model although a very large value of P can have an adverse effect due to overfitting. This intuition on the effect of P and D over the model's performance is captured in Table IV, where the average error decreases with increasing P and D for each dataset.

B. Regression

Experiments for the regression task are conducted for six datasets: (i) AirData $(n = 7322, d = 13)$; (ii) BloodData $(n = 61000, d = 2)$; (iii) EnergyData $(n = 18604, d = 25)$; (iv) TomData $(n = 9725, d = 96)$; (v) TwitterData $(n = 13813, d = 77)$; and (vi) TwitterDataL $(n = 98704, d = 77)$. The AirData dataset consists of hourly averaged samples from five chemical sensors within a polluted region of Italy, and the goal is to predict the density of the contaminating chemicals in the air [61]. The BloodData contains signals recorded by patient monitors at different hospitals and the goal is to predict an accurate estimation of the blood pressure based on several physiological parameters from Photoplethysmography and Electrocardiogram signals [62]. The EnergyData dataset aims at predicting the energy usage of light fixtures inside the house based on the temperature and humidity levels of indoors and outdoors [63]. The TomData dataset contains information about the number of conversations around a certain topic, and the target variable is the mean amount of display involving a specific topic on Tom's hardware [64]. The TwitterData dataset and its larger version, i.e., the TwitterDataL dataset, use features such as the length of discussions around a certain topic etc. to predict the mean number of active discussions on a particular subject [64]. These datasets are obtained from the UCI machine learning repository [60].

The network contains $J = 20$ nodes that communicate with each other point-to-point. All algorithms employ the Gaussian kernel where the parameter σ is fine-tuned via grid-search for each method individually. The parameter σ for the RF-ADMM algorithm is set to 1, 0.5, 0.1, 0.5, 1, 0.5 for the AirData, BloodData, EnergyData, TomData, TwitterData, and TwitterDataL datasets, respectively. Both GPPM and RFF-DOKL algorithms use $\sigma = 1$ for all datasets. Our distributed OMKL algorithm uses three Gaussian kernels with $\sigma \in \{0.1, 1, 10\}$ values. The regularization parameter is set to $\lambda = 0.001$, and the learning rate is chosen either to be a small constant or decreasing with $\mathcal{O}(1/t)$. The number of features for random feature (RF) approximation in distributed OMKL, RF-



(a) AirData

(b) BloodData

(c) EnergyData

(d) TomData

(e) TwitterData

(f) TwitterDataL

Figure 4: Mean square error versus the number of observered training samples per node.

ADMM, and RFF-DOKL algorithms are set to $D = 20$. Similar to the classification task, the GPPM algorithm is performed once since it is deterministic; however, simulations are carried out for RF-ADMM, RFF-DOKL, and our proposed distributed OMKL algorithms using ten different sets of random features and the mean and standard deviation of the error are reported. No normalization was performed since all algorithms were stable and converged. The remaining experimental setups are similar to that of the classification task.

The mean square error and running time of all methods are summarized in Tables V and VI, respectively. Not only does our method lead to a lower MSE compare to other algorithms in five out of six datasets, but also it has a lower running time in all cases. This shows the advantage of our proposed framework in distributed online learning tasks, especially in time-sensitive applications. For a fixed exemplary set of random features, Figure 4 shows the evolution of mean square error versus time for all four algorithms.

Next, we carried out experiments for distributed mini-batch OMKL and distributed quantized OMKL algorithms. Parameter B in mini-batch OMKL is set to $B = 5$, and three quantization levels $M = 2$, 4, and 16, corresponding to the 1, 2, and $4-$bit element-wise representation of the gradient, are considered. For a fair comparison between distributed, mini-batch, and quantized OMKL algorithms, a fixed set of random features are used for each dataset. All other parameters are kept unchanged.

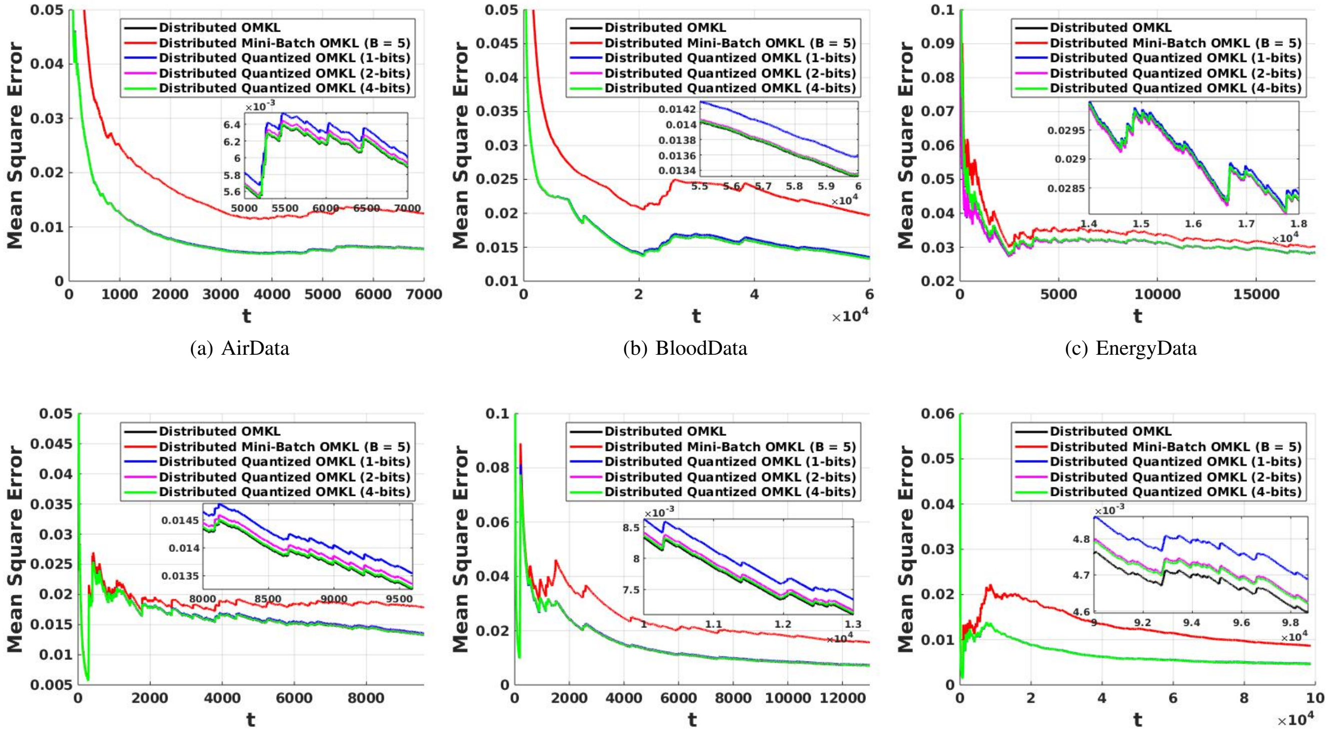
Table VII summarizes the mean square error for OMKL, mini-batch OMKL, and quantized OMKL algorithms. Since parameters are updated after $B = 5$ new observations at each node, the average square error for mini-batch OMKL is higher than that of OMKL; however, the mini-batch OMKL algorithm reduces the number of bits that each node needs to transmit by a factor of B compared to the OMKL algorithm. The mean square error for quantized OMKL is slightly increased due to lossy compression of the gradient vector; however, similar to what we observed in the classification task, the effect of quantization is marginal on the model's performance. The effect of mini-batch and quantized online multi-kernel learning on the progression of the mean square error is illustrated in Figure 5.

An argument similar to that in Section V-A reveals that for a $32-$bit hardware, our quantized OMKL algorithm outperforms RFF-DOKL and RF-ADMM algorithms in terms of communication cost if the number of quantization bits is less than or equal to 5 and 10, respectively. According to Table VII, the performance of the quantized OMKL algorithm converges to that of the OMKL algorithm for $M = 16$, i.e. 4 quantization bits; hence, our algorithm leads to less number of communicated bits compared to RFF-DOKL and RF-ADMM algorithms.

According to Eq. (19), our distributed and $4-$bit quantized OMKL algorithm achieves a lower number of transmitted bits compared to communicating the raw data, which is the case for the GPPM algorithm, if we have:

$$d > \frac{2D \times 3 \times 4}{32} = 15. \qquad (25)$$

This is the case for EnergyData, TomData, TwitterData, and TwitterDataL datasets. Eq. (25) implies that for high-



(a) AirData

(b) BloodData

(c) EnergyData

(d) TomData

(e) TwitterData

(f) TwitterDataL

Figure 5: Mean square error versus the number of observered training samples per node.

dimensional datasets, our algorithm can significantly reduce the number of bits that each node transmits per time slot.

Next, we carry out the regret simulations in a distributed network with $J = 20$ nodes that communicate with each other point-to-point. The regularization parameter and learning rate are set to $\lambda = 0.001$ and $\eta = 0.005$, respectively, and the number of random features is set to $D = 20$. The experiments are performed over the AirData dataset with mean-square-error as the loss function. The loss function for the best single Gaussian kernel RF approximation is obtained via offline learning over the dataset. The online distributed and quantized OMKL uses ten Gaussian kernels where the parameter σ for these kernels are fine-tuned via grid-search. Due to the randomness in selecting the random features, simulations are run ten times and the average regret is depicted in Figure 6. As shown in Figure 6, given the constant learning rate η, the regret closely follows a linear growth with the sample size as we showed analytically in Section IV. Moreover, as shown in Figure 6, the regret curve for the distributed and quantized OMKL closely follows the regret curve for the distributed OMKL. The two curves converge as the number of quantization bits increases.

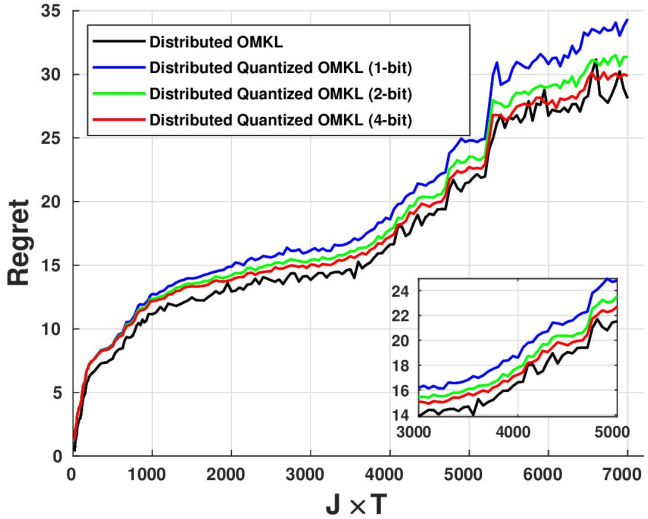


Figure 6: Regret vs sample size for the AirData dataset.

Finally, we study the effect of P, i.e., the number of kernels, on the model's performance and the value of regret in a distributed network with $J = 20$ nodes. The regularization and learning rate parameters are set to $\lambda = 0.001$ and $\eta = 0.005$, respectively, and the number of random features is equal to $D = 20$. The simulations are carried out over the AirData dataset with mean-square-error as the loss function. Offline learning over the same dataset is used to obtain the loss function for the best single Gaussian kernel RF approximation. Five variations of the online distributed OMKL algorithm corresponding to the values $P = 1, 3, 5, 7$ and 10 are simulated where a nested set of σ values are used for these variation, e.g., the set of σ values corresponding to the distributed OMKL with $P = 3$ kernels is a subset of the set of σ values for the one that uses $P = 5$ kernels. Figure 7 depicts the evolution of the residual sum of squares as the dataset size grows. As shown in Figure 7, increasing the number of kernels improves

Table IV: Effect of P and D on the classification error.

Variances of Gaussian kernels	Activity $D = 20$	$D = 50$	$D = 100$
$\{1\}$	$4.96(1.04)$	$4.03(0.41)$	$3.39(0.27)$
$\{0.1, 0.5, 1\}$	$4.65(0.93)$	$3.40(0.17)$	$3.13(0.34)$
$\{0.1, 0.5, 1, 5, 10\}$	$3.22(0.56)$	$2.17(0.37)$	$1.74(0.12)$
Variances of Gaussian kernels	**Adult** $D = 20$	$D = 50$	$D = 100$
$\{1\}$	$23.03(0.86)$	$19.90(0.73)$	$18.76(0.29)$
$\{0.5, 1, 2\}$	$20.72(0.87)$	$17.94(0.19)$	$17.24(0.18)$
$\{0.5, 0.8, 1, 2, 5\}$	$19.67(0.38)$	$17.77(0.16)$	$16.88(0.12)$
Variances of Gaussian kernels	**Banana** $D = 20$	$D = 50$	$D = 100$
$\{3\}$	$14.49(0.57)$	$12.72(0.33)$	$12.04(0.22)$
$\{1, 3, 5\}$	$13.38(0.55)$	$11.53(0.22)$	$11.23(0.15)$
$\{1, 2, 3, 4, 5\}$	$12.65(0.46)$	$11.38(0.11)$	$10.99(0.17)$
Variances of Gaussian kernels	**Credit-Card** $D = 20$	$D = 50$	$D = 100$
$\{3\}$	$24.55(0.19)$	$23.57(0.11)$	$22.82(0.03)$
$\{1, 3, 5\}$	$23.56(0.19)$	$22.19(0.02)$	$22.18(0.01)$
$\{1, 2, 3, 4, 5\}$	$22.73(0.09)$	$22.18(0.01)$	$22.17(0.003)$
Variances of Gaussian kernels	**Device** $D = 20$	$D = 50$	$D = 100$
$\{1\}$	$8.66(0.35)$	$7.59(0.10)$	$7.08(0.21)$
$\{0.1, 0.5, 1\}$	$7.33(0.08)$	$6.63(0.16)$	$6.29(0.17)$
$\{0.1, 0.25, 0.5, 0.75, 1\}$	$7.21(0.04)$	$6.35(0.17)$	$6.11(0.11)$
Variances of Gaussian kernels	**EEG** $D = 20$	$D = 50$	$D = 100$
$\{10\}$	$3.63(0.03)$	$3.26(0.02)$	$2.97(0.05)$
$\{0.1, 1, 10\}$	$3.39(0.12)$	$3.14(0.15)$	$2.84(0.10)$
$\{0.1, 0.5, 1, 5, 10\}$	$3.15(0.08)$	$2.98(0.12)$	$2.65(0.10)$
Variances of Gaussian kernels	**MNIST** $D = 20$	$D = 50$	$D = 100$
$\{1\}$	$9.65(0.15)$	$8.40(0.17)$	$7.49(0.17)$
$\{0.1, 1, 10\}$	$8.72(0.27)$	$7.10(0.32)$	$6.73(0.10)$
$\{0.1, 0.5, 1, 5, 10\}$	$7.85(0.33)$	$6.83(0.19)$	$5.80(0.18)$
Variances of Gaussian kernels	**Spam-Based** $D = 20$	$D = 50$	$D = 100$
$\{0.2\}$	$30.23(0.56)$	$28.93(0.36)$	$27.96(0.23)$
$\{0.1, 0.2, 0.4\}$	$29.41(0.29)$	$27.74(0.43)$	$27.28(0.43)$
$\{0.1, 0.2, 0.3, 0.4, 0.5\}$	$28.92(0.53)$	$27.59(0.30)$	$26.87(0.87)$

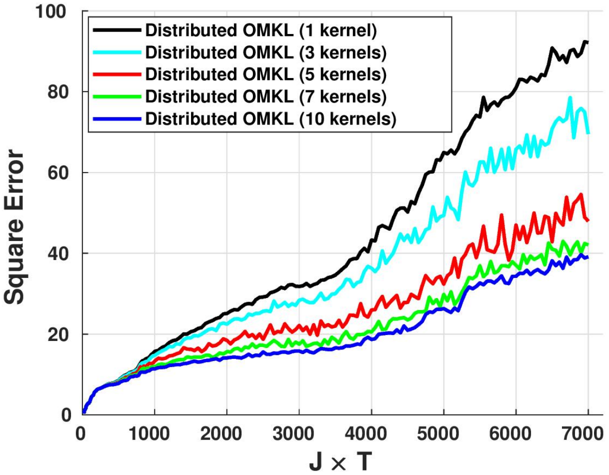


Figure 7: Square error of the distributed OMKL algorithm over the AirData dataset for different number of kernels.

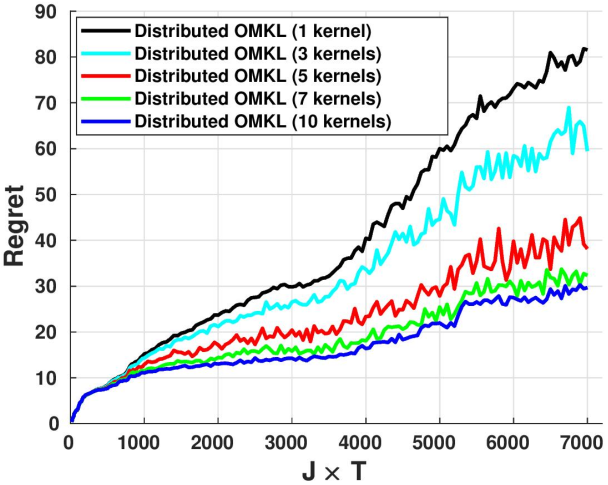


Figure 8: Regret for the distributed OMKL algorithm over the AirData dataset for different number of kernels.

model's performance, as we previously verified in Table IV for the classification task. Figure 8 illustrates the regret for five variations of the distributed OMKL algorithm corresponding to different values of P. Although increasing P leads to a logarithmic increment in the first term of the regret upper bound in Eq. (22), it significantly reduces the square norm of $\boldsymbol{\theta}_p^*$ associated with the best single kernel RF approximation, which leads to an overall decrease in the regret's value and upper bound.

All simulations are performed on a $6-$core workstation with Ubuntu 18.04.3 LTS platform and Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz.

VI. Conclusion

We consider the problem of distributed and quantized online multiple kernel learning. We develop an efficient and scalable algorithm for distributed and quantized OMKL. The algorithm is capable of updating the sought nonlinear model as more and more data samples are collected at distributed locations over a network. In addition, to overcome the communication bottleneck, only quantized information is shared among network nodes. Expected regret bound analysis shows that the algorithm is capable of achieving sublinear regret. Numerical tests on real datasets validate the effectiveness and efficiency of our novel algorithm. The present work opens up a number of possible future directions: (i) extension of the present work to the case where the communication networks entail general connectivity patterns; and (ii) exploring how to cope with possible delays in exchanging the gradients.

Appendix A
Proof of Lemma 1

To prove Lemma 1, we introduce two intermediate lemmas as follows.

Lemma 2: *Let us assume (as1), (as2), and define $\hat{f}_p^*$ in (21) with $\mathcal{F}_p := \{\hat{f}_p | \hat{f}_p(\mathbf{x}) = \boldsymbol{\theta}^\top \mathbf{z}_p(\mathbf{x}), \forall \boldsymbol{\theta} \in \mathbb{R}^{2D}\}$. Let $\{\hat{f}_{p,t}(\mathbf{x}_t)\}$ denote the sequence of estimates generated by the centralized OMKL with a pre-selected kernel κ_p. Then, the following bound holds*

$$\sum_{t=1}^{T}\sum_{j=1}^{J}\mathcal{L}_t(\hat{f}_{p,t}(\mathbf{x}_t^j)) - \sum_{t=1}^{T}\sum_{j=1}^{J}\mathcal{L}_t(\hat{f}_p^*(\mathbf{x}_t^j))$$

Table V: Mean (SD) of the mean square error $\left(10^{-3}\right)$ for different algorithms.

	AirData	**BloodData**	**EnergyData**	**TomData**	**TwitterData**	**TwitterDataL**
RFF-DOKL	16.15(0.68)	22.65(1.15)	29.23(0.07)	20.23(0.10)	18.10(1.34)	19.97(1.03)
GPPM	19.82	19.89	34.08	24.75	19.22	28.37
RF-ADMM	12.76(0.06)	15.56(0.06)	**16.05(0.03)**	15.32(0.11)	7.26(0.08)	8.13(0.44)
OMKL	**5.25(0.20)**	**11.06(0.85)**	28.60(0.08)	**10.85(1.33)**	**6.22(0.30)**	**3.63(0.27)**

Table VI: Running time (in seconds).

	AirData	**BloodData**	**EnergyData**	**TomData**	**TwitterData**	**TwitterDataL**
RFF-DOKL	1.14	19.86	4.02	1.54	2.65	46.07
GPPM	8.47	6843.22	20.14	4.85	19.27	101.45
RF-ADMM	13.92	138.52	37.50	19.36	26.77	232.38
OMKL	**0.62**	**7.69**	**2.05**	**0.90**	**1.34**	**14.62**

Table VII: MSE $\left(10^{-3}\right)$ for distributed mini-batch and quantized OMKL algorithms.

	OMKL	**Mini-batch OMKL**	**Quantized OMKL**		
		$B=5$	$M=2$	$M=4$	$M=16$
AirData	5.75	12.18	5.88	5.80	5.76
BloodData	13.24	19.53	13.49	13.27	13.26
EnergyData	28.60	30.55	28.75	28.64	28.61
TomData	13.11	17.63	13.39	13.20	13.13
TwitterData	6.74	14.82	6.97	6.80	6.77
TwitterDataL	4.60	8.65	4.69	4.63	4.62

$$\leq \frac{J\|\boldsymbol{\theta}_p^*\|^2}{2\eta}+\frac{\eta J L^2 T}{2}, \tag{26}$$

where η is the learning rate, L is the Lipschitz constant in (as2), and $\boldsymbol{\theta}_p^*$ is the corresponding parameter (or weight) vector supporting the best estimator $\hat{f}_p^*(\mathbf{x}) = (\boldsymbol{\theta}_p^*)^\top \mathbf{z}_p(\mathbf{x})$.

Proof: Similar to the regret analysis of online gradient descent [65], using (10) for any fixed $\boldsymbol{\theta}$, we find

$$\begin{aligned}\|\boldsymbol{\theta}_{p,t+1}-\boldsymbol{\theta}\|^2 =& \|\boldsymbol{\theta}_{p,t}-\eta\nabla\bar{\mathcal{L}}(\boldsymbol{\theta}_{p,t})-\boldsymbol{\theta}\|^2 \\ =& \|\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta}\|^2+\eta^2\|\nabla\bar{\mathcal{L}}(\boldsymbol{\theta}_{p,t})\|^2 \\ &-2\eta\nabla^\top\bar{\mathcal{L}}(\boldsymbol{\theta}_{p,t})(\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta}).\end{aligned} \tag{27}$$

Meanwhile, the convexity of the loss under (as1) implies that

$$\bar{\mathcal{L}}(\boldsymbol{\theta}_{p,t})-\bar{\mathcal{L}}(\boldsymbol{\theta}) \leq \nabla^\top\bar{\mathcal{L}}(\boldsymbol{\theta}_{p,t})(\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta}). \tag{28}$$

Plugging (28) into (27) and rearranging terms yield

$$\begin{aligned}&\bar{\mathcal{L}}(\boldsymbol{\theta}_{p,t})-\bar{\mathcal{L}}(\boldsymbol{\theta}) \\ \leq&\frac{\|\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta}\|^2-\|\boldsymbol{\theta}_{p,t+1}-\boldsymbol{\theta}\|^2}{2\eta}+\frac{\eta}{2}\|\nabla\bar{\mathcal{L}}(\boldsymbol{\theta}_{p,t})\|^2.\end{aligned} \tag{29}$$

Summing (29) over $t=1,\ldots,T$, with $\hat{f}_{p,t}(\mathbf{x}_t)=\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t)$ results in

$$\begin{aligned}&\sum_{t=1}^{T}\left(\bar{\mathcal{L}}(\hat{f}_{p,t}(\mathbf{x}_t))-\bar{\mathcal{L}}(\boldsymbol{\theta}^\top\mathbf{z}_p(\mathbf{x}_t))\right) \\ \leq&\ \frac{\|\boldsymbol{\theta}_{p,1}-\boldsymbol{\theta}\|^2-\|\boldsymbol{\theta}_{p,T+1}-\boldsymbol{\theta}\|^2}{2\eta}+\frac{\eta}{2}\sum_{t=1}^{T}\|\nabla\bar{\mathcal{L}}(\boldsymbol{\theta}_{p,t})\|^2 \\ \overset{(a)}{\leq}&\ \frac{\|\boldsymbol{\theta}\|^2}{2\eta}+\frac{\eta L^2 T}{2},\end{aligned} \tag{30}$$

where (a) uses the Lipschitz constant in (as2), the non-negativity of $\|\boldsymbol{\theta}_{p,T+1}-\boldsymbol{\theta}\|^2$, and the initial value $\boldsymbol{\theta}_{p,1}=\mathbf{0}$. Then, the proof of Lemma 2 is complete by choosing $\boldsymbol{\theta}=\boldsymbol{\theta}_p^*=\sum_{t=1}^{T}\alpha_{p,t}^*\mathbf{z}_p(\mathbf{x}_t)$ and $\hat{f}_p^*(\mathbf{x}_t)=\boldsymbol{\theta}^\top\mathbf{z}_p(\mathbf{x}_t)$ in (30). ■

In addition, to bound the difference between the loss of the solution obtained from distributed OMKL and the loss of the best single kernel-based online learning algorithm, the following lemma holds:

Lemma 3: *Let us assume (as1), (as2), and $\{\hat{f}_{p,t}\}$ be generated from the distributed OMKL. Then, the following bound holds*

$$\begin{aligned}&\sum_{t=1}^{T}\sum_{p=1}^{P}\sum_{j=1}^{J}\bar{w}_{p,t}\mathcal{L}_t(\hat{f}_{p,t}(\mathbf{x}_t^j))-\sum_{t=1}^{T}\sum_{j=1}^{J}\mathcal{L}_t(\hat{f}_{p,t}(\mathbf{x}_t^j)) \\ \leq&\eta JT+\frac{J\ln P}{\eta},\end{aligned} \tag{31}$$

where η is the learning rate in (8) *and P is the number of kernels in the dictionary.*

Proof: Letting $W_t := \sum_{p=1}^{P}w_{p,t}$, the weight recursion in (11) implies that

$$\begin{aligned}&W_{t+1} \\ =&\sum_{p=1}^{P}w_{p,t+1}=\sum_{p=1}^{P}w_{p,t}\exp\left(-\eta\bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right)\right) \\ \leq&\sum_{p=1}^{P}w_{p,t}\left(1-\eta\bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right)+\eta^2\bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right)^2\right),\end{aligned} \tag{32}$$

where the last inequality holds because $\exp(-\eta x)\leq 1-\eta x+\eta^2x^2$, for $|\eta|\leq 1$. Furthermore, substituting $\bar{w}_{p,t}:=$

$w_{p,t}/\sum_{p=1}^{P} w_{p,t} = w_{p,t}/W_t$ in (32) results in

$$\begin{aligned} &W_{t+1} \\ &\leq \sum_{p=1}^{P} W_t \bar{w}_{p,t} \left(1 - \eta \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right) + \eta^2 \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right)^2\right) \\ &= W_t \Bigg(1 - \eta \sum_{p=1}^{P} \bar{w}_{p,t} \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right) \\ &\qquad + \eta^2 \sum_{p=1}^{P} \bar{w}_{p,t} \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right)^2\Bigg). \end{aligned} \tag{33}$$

Using $1 + x \leq e^x$, $\forall x$, (33) leads to

$$\begin{aligned} W_{t+1} \leq & W_t \exp\Bigg(-\eta \sum_{p=1}^{P} \bar{w}_{p,t} \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right) \\ & + \eta^2 \sum_{p=1}^{P} \bar{w}_{p,t} \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right)^2\Bigg). \end{aligned} \tag{34}$$

Telescoping (34) from $t = 1$ to T, setting $W_1 = 1$, we have

$$\begin{aligned} W_{T+1} \leq \exp\Bigg(& -\eta \sum_{t=1}^{T} \sum_{p=1}^{P} \bar{w}_{p,t} \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right) \\ & + \eta^2 \sum_{t=1}^{T} \sum_{p=1}^{P} \bar{w}_{p,t} \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right)^2\Bigg). \end{aligned} \tag{35}$$

On the other hand, for any p, we have

$$\begin{aligned} W_{T+1} &\geq w_{p,T+1} \\ &= w_{p,1} \prod_{t=1}^{T} \exp(-\eta \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right)) \\ &= w_{p,1} \exp\left(-\eta \sum_{t=1}^{T} \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right)\right). \end{aligned} \tag{36}$$

Combining (35) with (36), we arrive at

$$\begin{aligned} &\exp\Bigg(-\eta \sum_{t=1}^{T} \sum_{p=1}^{P} \bar{w}_{p,t} \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right) \\ &\qquad + \eta^2 \sum_{t=1}^{T} \sum_{p=1}^{P} \bar{w}_{p,t} \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right)^2\Bigg) \\ &\geq w_{p,1} \exp\left(-\eta \sum_{t=1}^{T} \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right)\right). \end{aligned} \tag{37}$$

Taking the logarithm of the both sides of (37), we find that (cf. $w_{p,1} = 1/P$)

$$\begin{aligned} &-\eta \sum_{t=1}^{T} \sum_{p=1}^{P} \bar{w}_{p,t} \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right) + \eta^2 \sum_{t=1}^{T} \sum_{p=1}^{P} \bar{w}_{p,t} \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right)^2 \\ &\geq -\eta \sum_{t=1}^{T} \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right) - \ln P, \end{aligned} \tag{38}$$

which leads to

$$\begin{aligned} &\sum_{t=1}^{T} \sum_{p=1}^{P} \bar{w}_{p,t} \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right) \\ &\leq \sum_{t=1}^{T} \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right) + \eta \sum_{t=1}^{T} \sum_{p=1}^{P} \bar{w}_{p,t} \bar{\mathcal{L}}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right)^2 + \frac{\ln P}{\eta} \end{aligned} \tag{39}$$

and the proof is complete since $\mathcal{L}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right)^2 \leq 1$ and $\sum_{p=1}^{P} \bar{w}_{p,t} = 1$. ■

To prove Lemma 1, since $\mathcal{L}_t(\cdot)$ is convex under (as1), Jensen's inequality implies that

$$\mathcal{L}_t\left(\sum_{p=1}^{P} \bar{w}_{p,t} \hat{f}_{p,t}(\mathbf{x}_t)\right) \leq \sum_{p=1}^{P} \bar{w}_{p,t} \mathcal{L}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right). \tag{40}$$

Plugging (40) into (31) in Lemma 3, we have

$$\begin{aligned} &\sum_{t=1}^{T} \sum_{j=1}^{J} \mathcal{L}_t\left(\sum_{p=1}^{P} \bar{w}_{p,t} \hat{f}_{p,t}(\mathbf{x}_t^j)\right) \\ &\leq \sum_{t=1}^{T} \sum_{j=1}^{J} \mathcal{L}_t\left(\hat{f}_{p,t}(\mathbf{x}_t^j)\right) + \eta JT + \frac{J \ln P}{\eta} \\ &\overset{(a)}{\leq} \sum_{t=1}^{T} \sum_{j=1}^{J} \mathcal{L}_t\left(\hat{f}_p^*(\mathbf{x}_t)\right) + \frac{J \ln P}{\eta} + \frac{J\|\boldsymbol{\theta}_p^*\|^2}{2\eta} + \frac{\eta J L^2 T}{2} + \eta JT \end{aligned} \tag{41}$$

where (a) follows because of Lemma 2, and $\boldsymbol{\theta}_p^*$ is the optimal solution for any given kernel κ_p. This proves the claim in Lemma 1.

Appendix B
Proof of Theorem 1

To prove Theorem 1, we introduce three intermediate lemmas as follows.

Lemma 4: *Let $\mathcal{L}(.)$ be a convex and L-smooth function. Given repeated independent accesses to an unbiased stochastic gradient with bounded variance, i.e., $\mathbb{E}[\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top \mathbf{z}_p(\mathbf{x}_t), y_t) - \nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top \mathbf{z}_p(\mathbf{x}_t), y_t)] = 0$, $\mathbb{E}[\|\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top \mathbf{z}_p(\mathbf{x}_t), y_t) - \nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top \mathbf{z}_p(\mathbf{x}_t), y_t)\|_2^2] \leq \sigma^2$, each kernel-based learner achieves the following regret:*

$$\begin{aligned} &\sum_{t=1}^{T} \mathbb{E}\left[\mathcal{L}_t(\hat{f}_{p,t}(\mathbf{x}_t))\right] - \sum_{t=1}^{T} \mathcal{L}_t(\hat{f}_p^*(\mathbf{x}_t)) \\ &\leq \frac{\|\boldsymbol{\theta}_p^*\|^2}{2\eta} + \frac{\eta L^2 T}{2} + \frac{\eta \sigma^2 T}{2}. \end{aligned} \tag{42}$$

Proof: For any fixed $\boldsymbol{\theta}$, we find

$$\begin{aligned} &\|\boldsymbol{\theta}_{p,t+1} - \boldsymbol{\theta}\|^2 \\ &= \|\boldsymbol{\theta}_{p,t} - \eta \tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top \mathbf{z}_p(\mathbf{x}_t), y_t) - \boldsymbol{\theta}\|^2. \end{aligned} \tag{43}$$

Adding and abstracting the term $\eta\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)$, we obtain

$$\begin{aligned}&\|\boldsymbol{\theta}_{p,t+1}-\boldsymbol{\theta}\|^2\\
=&\|\boldsymbol{\theta}_{p,t}-\eta\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\eta\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\\
&+\eta\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\boldsymbol{\theta}\|^2\\
=&\|\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta}\|^2+\eta^2\|\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\|^2\\
&+\eta^2\|\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\|^2\\
&-2\eta\nabla^\top\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)(\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta})-2\eta(\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\\
&-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t))^\top(\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta})\\
&+2\eta^2\nabla^\top\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)(\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\\
&-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)).\end{aligned}\tag{44}$$

The convexity of the loss under (as1) implies that

$$\begin{aligned}&\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\mathcal{L}(\boldsymbol{\theta}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\\
\le&\nabla^\top\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)(\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta}).\end{aligned}\tag{45}$$

Substituting (44) in (45) and taking expectation on both sides leads to

$$\begin{aligned}&\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\mathcal{L}(\boldsymbol{\theta}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\\
\le&\frac{\|\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta}\|^2-\|\boldsymbol{\theta}_{p,t+1}-\boldsymbol{\theta}\|^2}{2\eta}+\frac{\eta}{2}\|\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\|^2\\
&+\frac{\eta}{2}\|\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\|^2\\
&-(\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t))^\top(\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta})\\
&+\eta\nabla^\top\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)(\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\\
&-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t))\le\frac{\|\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta}\|^2-\|\boldsymbol{\theta}_{p,t+1}-\boldsymbol{\theta}\|^2}{2\eta}\\
&+\frac{\eta L^2}{2}+\frac{\eta}{2}\|\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\|^2\\
&-(\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t))^\top(\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta})\\
&+\eta\nabla^\top\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)(\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\\
&-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)).\end{aligned}\tag{46}$$

Summing (46) over $t=1,\dots,T$, with $\hat{f}_{p,t}(\mathbf{x}_t)=\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t)$, we arrive at

$$\begin{aligned}&\sum_{t=1}^{T}\Big(\mathcal{L}(\hat{f}_{p,t}(\mathbf{x}_t),y_t)-\mathcal{L}(\boldsymbol{\theta}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\Big)\\
\le&\frac{\|\boldsymbol{\theta}_{p,1}-\boldsymbol{\theta}\|^2-\|\boldsymbol{\theta}_{p,T+1}-\boldsymbol{\theta}\|^2}{2\eta}+\frac{\eta L^2T}{2}\\
&+\frac{\eta}{2}\sum_{t=1}^{T}\|\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\|^2\\
&-\sum_{t=1}^{T}(\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t))^\top(\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta})\\
&+\eta\sum_{t=1}^{T}\nabla^\top\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)(\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\\
&-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t))\\
\le&\frac{\|\boldsymbol{\theta}\|^2}{2\eta}+\frac{\eta L^2T}{2}\\
&+\frac{\eta}{2}\sum_{t=1}^{T}\|\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\|^2\\
&-\sum_{t=1}^{T}(\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t))^\top(\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta})\\
&+\eta\sum_{t=1}^{T}\nabla^\top\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t))(\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\\
&-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)),\end{aligned}\tag{47}$$

where the first inequality uses the Lipschitz constant in (as2), the last inequality uses the non-negativity of $\|\boldsymbol{\theta}_{p,T+1}-\boldsymbol{\theta}\|^2$, and the initial value $\boldsymbol{\theta}_{p,1}=\mathbf{0}$. Taking expectation on both sides of (47) results in

$$\begin{aligned}&\sum_{t=1}^{T}\mathbb{E}\left[\mathcal{L}(\hat{f}_{p,t}(\mathbf{x}_t),y_t)-\mathcal{L}(\boldsymbol{\theta}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\right]\\
\le&\frac{\|\boldsymbol{\theta}\|^2}{2\eta}+\frac{\eta L^2T}{2}\\
&+\frac{\eta}{2}\sum_{t=1}^{T}\mathbb{E}\left[\|\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\|^2\right]\\
&-\sum_{t=1}^{T}\mathbb{E}\Big[(\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t))^\top(\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta})\Big]\\
&+\eta\sum_{t=1}^{T}\mathbb{E}\Big[\nabla^\top\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t))(\tilde{\nabla}\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\\
&-\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t))\Big]\le\frac{\|\boldsymbol{\theta}\|^2}{2\eta}+\frac{\eta L^2T}{2}+\frac{\eta\sigma^2T}{2}.\end{aligned}\tag{48}$$

Then, by choosing $\boldsymbol{\theta}=\boldsymbol{\theta}_p^*=\sum_{t=1}^{T}\alpha_{p,t}^*\mathbf{z}_p(\mathbf{x}_t)$ such that $\hat{f}_p^*(\mathbf{x}_t)=\boldsymbol{\theta}^\top\mathbf{z}_p(\mathbf{x}_t)$ in (48), the proof of Lemma 4 is complete. ■

Building upon the result in Lemma 4, the following lemma can be readily established for distributed and quanitzed OMKL.

Lemma 5: *Let $\mathcal{L}(.)$ be a convex and L-smooth function. Given repeated independent accesses to stochastic gradient with bounded variance, $\mathbb{E}[\|\tilde{\nabla}\mathcal{L}(\mathbf{x}_t^j,y_t^j)-\nabla\mathcal{L}(\mathbf{x}_t^j,y_t^j)\|_2^2]\le\sigma^2$, the distributed and quantized OMKL achieves the following regret:*

$$\begin{aligned}&\sum_{t=1}^{T}\mathbb{E}\left[\mathcal{L}_t(\hat{f}_{p,t}(\mathbf{x}_t))\right]-\sum_{t=1}^{T}\mathcal{L}_t(\hat{f}_p^*(\mathbf{x}_t))\\
\le&\frac{J\|\boldsymbol{\theta}_p^*\|^2}{2\eta}+\frac{\eta JL^2T}{2}+\frac{\eta J\sigma^2T}{2}.\end{aligned}\tag{49}$$

Combining (31) in Lemma 3 with Lemma 5

$$\begin{aligned}&\sum_{j=1}^{J}\sum_{t=1}^{T}\mathbb{E}\left[\mathcal{L}_t\Big(\sum_{p=1}^{P}\bar{w}_{p,t}^j\hat{f}_{p,t}(\mathbf{x}_t^j)\Big)\right]\\
\le&\sum_{j=1}^{J}\sum_{t=1}^{T}\mathcal{L}_t\left(\hat{f}_p^*(\mathbf{x}_t^j)\right)+\frac{J\ln P}{\eta}+\frac{J\|\boldsymbol{\theta}_p^*\|^2}{2\eta}\\
&+\frac{\eta JL^2T}{2}+\frac{\eta J\sigma^2T}{2}+\eta JT.\end{aligned}\tag{50}$$

In addition, the following lemma characterizes the performance of the quantization scheme.

Lemma 6: *For any vector $\mathbf{g} \in \mathbb{R}^{2D}$, using the quantization scheme $Q_M(\mathbf{g})$ in* (14)*, we have $\mathbb{E}[Q_M(\mathbf{g})] = \mathbf{g}$ (unbiasedness) and $\mathbb{E}[\|Q_M(\mathbf{g}) - \mathbf{g}\|_2^2] \leq \min\left(\frac{2D}{M^2}, \frac{\sqrt{2D}}{M}\right) \|\mathbf{g}\|_2^2$ (variance bound).*

Proof: See [55]. ■

Combining Lemma 6 with (50) and letting $\sigma^2 = \sigma_L^2 := \min\left(\frac{2D}{M^2}, \frac{\sqrt{2D}}{M}\right) L^2$ in (50) lead to Theorem 1.

References

[1] K. Slavakis, G. B. Giannakis, and G. Mateos, "Modeling and optimization for big data analytics:(statistical) learning tools for our era of data deluge," *IEEE Signal Processing Magazine*, 31(5):18–31, 2014.

[2] H. Yousefi'zadeh, H. Jafarkhani, and M. Moshfeghi, "Power optimization of wireless media systems with space-time code building blocks," *IEEE Transactions on Image Processing*, 13(7):873-884, 2004.

[3] S. Karimi-Bidhendi, J. Guo, and H. Jafarkhani, "Using Quantization to Deploy Heterogeneous Nodes in Two-Tier Wireless Sensor Networks," *IEEE International Symposium on Information Theory*, pp. 1502-1506, July 2019.

[4] G. Anastasi, M. Conti, M. D. Francesco, and A. Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad hoc networks*, 7(3):537–568, 2009.

[5] J. Guo, S. Karimi-Bidhendi, and H. Jafarkhani, "Energy-Efficient Node Deployment In Wireless Ad-Hoc Sensor Networks," *IEEE International Conference on Communications*, pp. 1-6, June 2020.

[6] M. A. Razzaque and S. Dobson, "Energy-efficient sensing in wireless sensor networks using compressed sensing," *Sensors*, 14(2):2822–2859, 2014.

[7] J. Guo and H. Jafarkhani, "Sensor deployment with limited communication range in homogeneous and heterogeneous wireless sensor networks," *IEEE Transactions on Wireless Communications*, 15(19):6771-6784, 2016.

[8] S. Karimi-Bidhendi, J. Guo, and H. Jafarkhani, "Energy-Efficient Node Deployment in Heterogeneous Two-Tier Wireless Sensor Networks with Limited Communication Range," *IEEE Transactions on Wireless Communications*, 20(1):40-55, Sep 2020.

[9] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

[10] I. D. Schizas, G. Mateos, and G. B. Giannakis, "Distributed lms for consensus-based in-network adaptive processing," *IEEE Transactions on Signal Processing*, 57(6):2365–2382, 2009.

[11] G. Mateos, I. D. Schizas, and G. B. Giannakis, "Distributed recursive least-squares for consensus-based in-network adaptive estimation," *IEEE Transactions on Signal Processing*, 57(11):4583–4588, 2009.

[12] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," *IEEE Transactions on Signal Processing*, 56(7):3122–3136, 2008.

[13] R. L. G. Cavalcante, I. Yamada, and B. Mulgrew, "An adaptive projected subgradient approach to learning in diffusion networks," *IEEE Transactions on Signal Processing*, 57(7):2762–2774, 2009.

[14] P. A. Forero, A. Cano, and G. B. Giannakis, "Consensus-based distributed support vector machines," *Journal of Machine Learning Research*, 11(May):1663–1707, 2010.

[15] P. Bouboulis, S. Chouvardas, and S. Theodoridis, "Online distributed learning over networks in rkh spaces using random fourier features," *IEEE Transactions on Signal Processing*, 66(7):1920–1932, 2017.

[16] R. Mitra and V. Bhatia, "The diffusion-klms algorithm," In *2014 International Conference on Information Technology*, pp. 256–259, IEEE, 2014.

[17] S. Chouvardas and M. Draief, "A diffusion kernel lms algorithm for nonlinear adaptive networks," In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4164–4168, IEEE, 2016.

[18] J. Shawe-Taylor, N. Cristianini, et al., *Kernel methods for pattern analysis*, Cambridge university press, 2004.

[19] T. Hofmann, B. Schölkopf, and A. J. Smola, "Kernel methods in machine learning," *The annals of statistics*, 36(3):1171-1220, 2008.

[20] C. K. I. Williams and M. Seeger, "Using the nyström method to speed up kernel machines," In *Advances in neural information processing systems*, pp. 682–688, 2001.

[21] F. Sheikholeslami, D. Berberidis, and G. B. Giannakis, "Large-scale kernel-based feature extraction via low-rank subspace tracking on a budget," *IEEE Transactions on Signal Processing*, 66(8):1967–1981, 2018.

[22] C. Cortes, M. Mohri, and A. Talwalkar, "On the impact of kernel approximation on learning accuracy," In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 113–120, 2010.

[23] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," In *Advances in neural information processing systems*, pp. 1177–1184, 2007.

[24] B. Dai, B. Xie, N. He, Y. Liang, A. Raj, M. F. Balcan, and L. Song, "Scalable kernel methods via doubly stochastic gradients," In *Advances in Neural Information Processing Systems*, pp. 3041–3049, 2014.

[25] F. X. X. Yu, A. T. Suresh, K. M. Choromanski, D. N. Holtmann-Rice, and S. Kumar, "Orthogonal random features," In *Advances in Neural Information Processing Systems*, pp. 1975–1983, 2016.

[26] Y. Low, D. Bickson, J. Gonzalez, C. Guestrin, A. Kyrola, and J. M. Hellerstein, "Distributed graphlab: a framework for machine learning and data mining in the cloud," *Proceedings of the VLDB Endowment*, 5(8):716–727, 2012.

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

[28] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," *IEEE transactions on signal processing*, 52(8):2165–2176, 2004.

[29] O. Dekel, S. Shalev-Shwartz, and Y. Singer, "The forgetron: A kernel-based perceptron on a budget," *SIAM Journal on Computing*, 37(5):1342–1372, 2008.

[30] G. Cavallanti, N. Cesa-Bianchi, and C. Gentile, "Tracking the best hyperplane with a simple budget perceptron," *Machine Learning*, 69(2-3):143–167, 2007.

[31] F. Orabona, J. Keshet, and B. Caputo, "The projectron: a bounded kernel-based perceptron," *Proceedings of the International Conference on Machine Learning*, pp. 720–727, 2008.

[32] Z. Wang and S. Vucetic, "Online passive-aggressive algorithms on a budget," *International Conference on Artificial Intelligence and Statistics*, pp. 908–915, 2010.

[33] Z. Wang, K. Crammer, and S. Vucetic, "Breaking the curse of kernelization: Budgeted stochastic gradient descent for large-scale svm training," *Journal of Machine Learning Research*, 13(Oct):3103–3131, 2012.

[34] Z. Wang and S. Vucetic, "Twin vector machines for online learning on a budget," *International Conference on Data Mining*, pp. 906–917. SIAM, 2009.

[35] Z. Dang, B. Gu, and H. Huang, "Large-Scale Kernel Method for Vertical Federated Learning," *Federated Learning*, pp. 66-80. Springer, 2020.

[36] B. Gu, Z. Dang, X. Li, and H. Huang, "Federated doubly stochastic kernel learning for vertically partitioned data," *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2483-2493. 2020.

[37] S. C. H. Hoi, R. Jin, P. Zhao, and T. Yang, "Online multiple kernel classification," *Machine Learning*, 90(2):289–316, 2013.

[38] R. Jin, S. C. H. Hoi, and T. Yang, "Online multiple kernel learning: Algorithms and mistake bounds," In *International conference on algorithmic learning theory*, pp. 390–404. Springer, 2010.

[39] D. Sahoo, S. Hoi, and P. Zhao, "Cost sensitive online multiple kernel classification," In *Asian Conf. on Machine Learning*, pp. 65–80, 2016.

[40] D. Sahoo, S. C. H. Hoi, and B. Li, "Online multiple kernel regression,"

In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 293–302, 2014.

[41] J. Lu, D. Sahoo, P. Zhao, and S. C. H. Hoi, "Sparse passive-aggressive learning for bounded online kernel methods," *ACM Transactions on Intelligent Systems and Technology (TIST)*, 9(4):1–27, 2018.

[42] Y. Ding, C. Liu, P. Zhao, and S. C. H. Hoi, "Large scale kernel methods for online auc maximization," In *2017 IEEE International Conference on Data Mining (ICDM)*, pp. 91–100. IEEE, 2017.

[43] A. Koppel, S. Paternain, C. Richard, and A. Ribeiro, "Decentralized online learning with kernels," *IEEE Transactions on Signal Processing*, 66(12):3240–3255, 2018.

[44] H. Pradhan, A. S. Bedi, A. Koppel, and K. Rajawat, "Adaptive kernel learning in heterogeneous networks," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 7, 2021.

[45] P. Xu, Z. Tian, Z. Zhang, and Y. Wang, "Coke: Communication-censored kernel learning via random features," *IEEE Data Science Workshop*, pp. 32-36, 2019.

[46] Y. Liu, W. Xu, G. Wu, Z. Tian, and Q. Ling, "Communication-censored admm for decentralized consensus optimization," *IEEE Transactions on Signal Processing*, 67(10):2565–2579, 2019.

[47] S. Hong, and J. Chae, "Distributed Online Learning with Multiple Kernels," *IEEE Trans. on Neural Networks and Learning Systems*, 2021.

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

[49] B. Schölkopf, A. J. Smola, F. Bach, et al., *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[50] G. Wahba, *Spline models for observational data*, vol. 59, SIAM, 1990.

[51] C. A. Micchelli and M. Pontil, "Learning the kernel function via regularization," *J. of Mach. Learn. Res.*, vol. 6, pp. 1099–1125, 2005.

[52] C. Cortes, M. Mohri, and A. Rostamizadeh, "ℓ_2-regularization for learning kernels," In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pp. 109–116, Montreal, Canada, Jun. 2009.

[53] M. Gönen and E. Alpaydın, "Multiple kernel learning algorithms," *Journal of machine learning research*, vol. 12, pp. 2211–2268, 2011.

[54] Y. Shen, T. Chen, and G. B. Giannakis, "Online ensemble multi-kernel learning adaptive to non-stationary and adversarial environments," In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics*, pp. 2037-2046, 2018.

[55] D. Alistarh, J. Li, R. Tomioka, and M. Vojnovic, "Qsgd: Randomized quantization for communication-optimal stochastic gradient descent," *arXiv preprint arXiv:1610.02132*, 2016.

[56] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones," In *Euro. Symp. on Artificial Neural Netw., Comp. Intell. and Mach. Learn.*, Bruges, Belgium, Apr. 2013.

[57] I. C. Yeh and C. H. Lien, "The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients," *Expert Systems with Applications*, 36(2):2473–2480, 2009.

[58] J. Lines, A. Bagnall, P. Caiger-Smith, and S. Anderson, "Classification of household devices by electricity usage profiles," In *Intl. Conf. on Intelligent Data Engineering and Automated Learning*, pp. 403–412, Sep. 2011.

[59] Y. LeCun, C. Cortes, and C. J. C. Burges, *The mnist database*, *URL http://yann. lecun. com/exdb/mnist*, 1998.

[60] M. Lichman, *UCI machine learning repository*, 2013.

[61] S. D. Vito, E. Massera, M. Piga, L. Martinotto, and G. D. Francia, "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.

[62] M. Kachuee, M. M. Kiani, H. Mohammadzade, and M. Shabany, "Cuff-less high-accuracy calibration-free blood pressure estimation using pulse transit time," In *IEEE international symposium on circuits and systems (ISCAS)*, pp. 1006-1009, May 2015.

[63] L. M. Candanedo, V. Feldheim, and D. Deramaix, "Data driven prediction models of energy use of appliances in a low-energy house," *Energy and buildings*, 140:81–97, 2017.

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

[65] S. Shalev-Shwartz, "Online learning and online convex optimization," *Foundations and Trends in Machine Learning*, 4(2):107–194, 2011.