Communication-Efficient Randomized Algorithm for Multi-Kernel Online Federated Learning

Songnam Hong, Member, IEEE and Jeongmin Chae, Student Member, IEEE

Abstract—Online federated learning (OFL) is a promising framework to learn a sequence of global functions from distributed sequential data at local devices. In this framework, we first introduce a single kernel-based OFL (termed S-KOFL) by incorporating random-feature (RF) approximation, online gradient descent (OGD), and federated averaging (FedAvg). As manifested in the centralized counterpart, an extension to multi-kernel method is necessary. Harnessing the extension principle in the centralized method, we construct a vanilla multi-kernel algorithm (termed vM-KOFL) and prove its asymptotic optimality. However, it is not practical as the communication overhead grows linearly with the size of a kernel dictionary. Moreover, this problem cannot be addressed via the existing communication-efficient techniques (e.g., quantization and sparsification) in the conventional federated learning. Our major contribution is to propose a novel randomized algorithm (named eM-KOFL), which exhibits similar performance to vM-KOFL while maintaining low communication cost. We theoretically prove that eM-KOFL achieves an optimal sublinear regret bound. Mimicking the key concept of eM-KOFL in an efficient way, we propose a more practical pM-KOFL having the same communication overhead as S-KOFL. Via numerical tests with real datasets, we demonstrate that pM-KOFL yields the almost same performance as vM-KOFL (or eM-KOFL) on various online learning tasks.

Index Terms—Federated learning, online learning, kernel-based learning, online biconvex optimization, RKHS

1 INTRODUCTION

Federated learning has emerged as a promising decentralized machine learning framework, in which distributed nodes (e.g., mobile phones, wearable devices, etc.) learn a global function collaboratively under the coordination of a server without sharing raw data placed at edge nodes [1], [2]. To be specific, federated learning optimizes a global function (or model) by repeating the two operations: i) local model optimizations at edge nodes; ii) global model update (e.g., model averaging) at the server [3]. This distributed learning framework has myriad of applications: activities of mobile phone users, predicting a low blood sugar, heart attack risk from wearable devices, or detecting burglaries within smart homes [4], [5], [6].

In many real-world applications, function learning tasks are expected to be performed in an online fashion. For instance, online learning is required when data is generated as a function of time (e.g., time-series predictions) [7], [8] and when a large number of data makes it hard to carry out data analytic in batch form [9]. Focusing on a centralized network, this challenging problem has been efficiently addressed via online multiple kernel learning (OMKL) [10], [11], [12]. OMKL learns a sequence of functions (or models) from continuous streaming data, which can predict the label of a newly incoming data in real time. Using multiple kernels, OMKL can yield a superior accuracy and enjoy a great flexibility compared with a single kernel-based online learning [11], [12]. Despite its effectiveness, OMKL is restricted to the centralized network only and an extension to a decentralized network (e.g., federated learning) has not been investigated yet.

Motivated by the success of OMKL, this paper considers a novel kernel-based OFL (KOFL) framework, in which the objective is to learn a sequence of global (or common) functions \( f(x; \tilde{m}_t) \) with a model parameter \( \tilde{m}_t \in \mathbb{R}^M \) using streaming data across a large number of distributed nodes. In particular, our learned function \( f(x; \tilde{m}_t) \) belongs to a reproducing Hilbert kernel space (RKHS) \( \mathcal{H} \) [13], i.e., \( f(x; \tilde{m}_t) \in \mathcal{H} \). Specifically, at every time \( t \), each node \( k \in \{1, 2, \ldots, K\} \) estimates a local model \( \tilde{m}_{k,t+1} \) from the current global model \( \tilde{m}_t \) and an incoming data \( (x_{k,t+1}, y_{k,t+1}) \). Then, it sends \( \tilde{m}_{[k,t+1]} \) to the server. This transmission is called uplink. Leveraging the aggregated local models, the server constructs an updated global model as

\[
\tilde{m}_{t+1} = h(\tilde{m}_{[1,t+1]}, \ldots, \tilde{m}_{[K,t+1]}),
\]

and broadcasts it to the \( K \) edge nodes. The mapping \( h \) should be carefully designed according to the structures of model parameters. One representative example is averaging (called FedAvg) [3]. Given the global model, every node \( k \) can estimate the label of a newly incoming data \( x_{k,t+1} \) such as
\[ \hat{y}_{k,t+1} = f(x_{k,t+1}, \tilde{m}_{t+1}) . \] (2)

In KOFL framework, numerous methods can be constructed according to underlying optimization and learning algorithms. We first introduce a single kernel-based algorithm (termed S-KOFL) by properly incorporating the random-feature (RF) approximation [14], online gradient descent (OGD) [15], and FedAvg [3]. This method is closely related to a conventional federated learning especially based on stochastic gradient descent (SGD) [16]. In S-KOFL, each node \( k \) optimizes the parameter of a kernel function (denoted by \( w_{(k,t)} \)) via OGD using a local streaming data. Harnessing the similarities of OGD and SGD, the other operations exactly follow the conventional FedAvg [3, 16]. The primary challenge in federated learning is the high communication cost to exchange model information between edge nodes and the server. Lossy compression of local models is a practical solution to reduce the communication overhead, which can be performed via sparsification or quantization [17], [18], [19], [20]. It is remarkable that S-KOFL can be further enhanced in terms of communication cost, by taking the above techniques immediately.

As manifested in [11, 12, 21, 22], a single kernel-based learning (including S-KOFL) yields a poor performance since it is demanding to choose a proper single kernel. Thus, it is necessary to construct a communication-efficient multi-kernel based algorithm for KOFL framework. We emphasize that it poses a new challenging problem, which has not been caused in existing federated learning frameworks as well as the centralized counterpart. One natural extension of S-KOFL into multiple kernels just follows the extension principle in the centralized network [11, 12]. This vanilla method is referred to as VM-KOFL. In this method, every node should transmit the \( P \) pairs of local parameters to the server and vice versa, where \( P \) is the size of a kernel dictionary. The model size of VM-KOFL is equal to \( P \times M \), whereas that of S-KOFL is equal to \( M \). Accordingly, the downlink (or uplink) communication overhead of VM-KOFL is \( P \) times larger than that of S-KOFL. Throughout the paper, the communication overhead for downlink (or uplink) is defined as the number of channel uses per node at each time. Accordingly, the communication overhead of S-KOFL is equal to \( M \) for downlink (resp. uplink) as the model parameter \( \tilde{m}_{t+1} \in \mathbb{R}^M \) (resp. \( \tilde{m}_{(k,t+1)} \in \mathbb{R}^M \)). Also, the communication overhead of VM-KOFL is equal to \( PM \) for both downlink and uplink. Generally, to attain a higher learning accuracy, VM-KOFL requires a sufficiently large number of kernels and thus, the communication cost becomes indeed trouble. Moreover, the existing communication-efficient techniques (e.g., quantization or sparsification) [17], [18], [19], [20] can only reduce the term \( M \) as \( M' \ll M \), whereas the term \( P \) should be unchanged. Thus, VM-KOFL can achieve an improved learning accuracy over S-KOFL at the expense of an additional communication overhead. A natural question arises: under KOFL framework, can we enjoy the advantage of multiple kernels while maintaining the communication overhead of S-KOFL?

In this paper, we contribute to the aforementioned open problem. A novel randomized algorithm (named eM-KOFL) is proposed. Enjoying the merit of multiple kernels probabilistically, eM-KOFL can yield the same performance as VM-KOFL while maintaining the low communication cost. Notably, the communication overhead of eM-KOFL is irrespective of the size of a kernel dictionary (i.e., \( P \)). To achieve a higher learning accuracy, thus, a sufficiently large number of kernels can be used for free. Our major contributions are summarized as follows.

- We propose a delayed-Exp, from which the server constructs a sequence of probability mass functions (PMFs) \( q_t = (\tilde{q}_{1,t}, \ldots, \tilde{q}_{P,t}) \), \( t = 1, 2, \ldots, T \), in a distributed and online fashion. At every time \( t \), the server chooses one kernel index out of \( P \) kernels randomly according to the PMF \( \tilde{q}_t \). The proposed strategy guarantees that the selected kernel converges to the best kernel in hindsight with high probability. Likewise VM-KOFL, therefore, the proposed eM-KOFL can operate as S-KOFL with the best kernel in hindsight.

- Leveraging a martingale argument, we theoretically prove that the proposed eM-KOFL achieves an optimal sublinear regret bound \( O(\sqrt{T}) \) compared with the best kernel function in hindsight. This analysis also implies that eM-KOFL can asymptotically achieve the same learning accuracy as the centralized OMKL in [11, 12] without sharing raw data (i.e., preserving an edge-node privacy).

- Regarding the communication overhead (per node), S-KOFL and VM-KOFL respectively have \( M \) and \( PM \) for both downlink and uplink. Due to the clever use of multiple kernels, the communication overheads of eM-KOFL are reduced as \( M + 1 \) and \( P + M \) for downlink and uplink, respectively. We further reduce the uplink communication overhead of eM-KOFL by mimicking the delayed-Exp strategy efficiently. The resulting algorithm is named pM-KOFL, which has the communication overheads \( M + 1 \approx M \) for both downlink and uplink. These results are summarized in Table 4.2.

| Methods         | Communication overheads |
|-----------------|-------------------------|
| S-KOFL          | \( M \) / \( M \)       |
| M-KOFL          | \( P(M + 1) \approx PM \) / \( P(M + 1) \approx PM \) |
| eM-KOFL         | \( M + P \leq 2M \) / \( M + 1 \approx M \) |
| pM-KOFL         | \( M + 1 \approx M \) / \( M + 1 \approx M \) |

### Table 1: Comparisons of Communication Overhead for Various KOFL Methods (\( M \): the Size of Random Features in (4) and \( P \): the Number of Kernels)
framework. In Section 3, a vanilla multi-kernel method (termed vM-KOFL) is constructed as a baseline approach and its drawback is identified. In Section 4, we propose a randomized method (named eM-KOFL) and its communication-efficient variant pM-KOFL. We provide theoretical analyses for the proposed methods in Section 5. Some numerical experiments on real datasets are conducted to demonstrate the effectiveness of the proposed methods for online regression and time-series prediction tasks. We conclude this paper in Section 7.

Notations: Bold lowercase letters denote the column vectors. For any vector $x, x^T$ denotes the transpose of $x$ and $||x||$ denote the $\ell_2$-norm of $x$. Also, $E[\cdot]$ represents the expectation over an associated probability distribution. To simplify the notations, we let $[n] \triangleq \{1, \ldots, n\}$ for any positive integer $n$. Also, $k, t, s$ and $i$ will be used to stand for the indices of node, time, and kernel, respectively. The number of nodes, kernels in a kernel dictionary, and total incoming data are denoted as $K, P$, and $T$, respectively.

2 PRELIMINARIES

We introduce an online federated learning (OFL) and then define our kernel-based OFL (KOFL) framework.

2.1 Online Federated Learning (OFL)

The objective of OFL is to learn a sequence of global (or common) functions using sequential data (e.g., time-series data) across a large number of distributed edge nodes. In detail, at every time $t$, the server distributes the latest global model $\hat{m}_t$ (i.e., the parameter of a global function $f(x; \hat{m}_t)$) to the decentralized $K$ nodes. Then, each node $k$ receives the global model $\hat{m}_t$ and an incoming data $(x_{k,t}, y_{k,t})$, where $x_{k,t} \in X \subseteq \mathbb{R}^N$ and $y_{k,t} \in Y \subseteq \mathbb{R}$ represent the feature and the label, respectively. Using them, each node $k$ updates its local model, which is denoted as $\hat{m}_{k,t+1} \in \mathbb{R}^M$ for $k \in [K]$. Every node $k$ sends $\hat{m}_{k,t+1}$ back to the server, from which the server constructs an updated global model as

$$\hat{m}_{t+1} = h(\hat{m}_{1,t+1}, \ldots, \hat{m}_{K,t+1}).$$

The mapping $h$ should be carefully designed according to the structure of a learned function. In the existing federated learning frameworks, one representative mapping is to average the aggregated local models, called FedAvg [3], [16]. Our learned global function at time $t$ is given as $f(x; \hat{m}_{t+1})$, from which each node $k$ can estimate the label of a newly incoming data $x_{k,t+1}$ in real time.

2.2 KOFL Framework

We briefly describe a kernel-based function learning and then by incorporating it into OFL, we define our kernel-based OFL (KOFL) framework reproducing kernel Hilbert space (RKHS) $\mathcal{H}$, defined as $\mathcal{H} = \{ f : f(x) = \sum a_k \kappa(x, x_i) \}$, where $\kappa(x, x_i) : X \times X \rightarrow \mathcal{Y}$ is a symmetric positive semidefinite basis function (called kernel) [23]. The major drawback of the above learning technique is that the dimension of a parameter to be optimized grows linearly with the number of incoming data. Thus, it is not applicable for online learning with continuous streaming data. This problem has been circumvented in [24] by presenting the random-feature (RF) approximation, in which a kernel is well-approximated with a fixed and small number of random features. The RF-based kernel function is represented with a parameter $\tilde{w} \in \mathbb{R}^M$ as

$$f(x; \tilde{w}) = \tilde{w}^T z(x) \in \mathcal{H},$$

where a feature mapping $z(x)$, which relies on a kernel $\kappa$, is defined as

$$z(x) = \frac{1}{\sqrt{d}} \left[ \sin v_1^T x, \ldots, \sin v_d^T x, \cos v_1^T x, \ldots, \cos v_d^T x \right]^T,$$

and where $\{v_i : i \in [d]\}$ denotes an independent and identically distributed (i.i.d.) samples from the Fourier transform of a given kernel function $\kappa(\cdot, \cdot)$. Via the RF approximation, the hyper-parameter $M = 2d$ can be chosen independently from the number of incoming data.

The accuracy of the above RF-based kernel learning fully relies on a predetermined kernel $\kappa$, which can be chosen manually either by a task-specific priori knowledge or by some intensive cross-validation process. As shown in [11], [12], [22], a multiple kernel learning, using a preselected set of $P$ kernels $K = \{k_1, k_2, \ldots, k_P\}$ (called a kernel dictionary), is more powerful as it can enlarge a function space for optimization. In this case, the RF-based multi-kernel function is represented as

$$f(x; \{\tilde{q}_i, \tilde{w}_i\}) = \sum_{i=1}^P \tilde{q}_i \tilde{w}_i^T z_i(x),$$

where $\tilde{q}_i \in [0, 1]$ denotes the combination weight (or reliability) of the associated kernel function $f(x; \tilde{w}_i) = \tilde{w}_i^T z_i(x) \in \mathcal{H}_i$.

Now we are ready to define KOFL framework formally. It is a class of OFL, which seeks a sequence of kernel functions in the form of (6). An algorithm (or method) for KOFL framework is evaluated in terms of a learning accuracy and a communication overhead:

- **Learning accuracy**: As in centralized OMKL [11], [12], the learning accuracy of an algorithm is measured by the cumulative regret over $T$ time slots

$$\text{regret}_T = \sum_{t=1}^T \sum_{k=1}^K \mathcal{L}(f(x_{k,t}; \hat{m}_t), y_{k,t}) - \min_{w \in \mathbb{R}^M} \min_{1 \leq s \leq P} \sum_{k=1}^K \sum_{t=1}^T \mathcal{L}(w_s^T z_i(x_{k,t}), y_{k,t}),$$

where $\mathcal{L}(\cdot, \cdot)$ denotes a loss (or cost) function. This metric compares the cumulative loss of our algorithm to the cumulative loss of the static optimal function from the best kernel.

- **Communication overhead (per node)**: It is measured by the number of transmissions between each node and the server (i.e., the dimension of local (resp. global) model parameter for uplink (resp. downlink)). For example, if a learned function has the form of (6) (i.e., $\hat{m}_t = \{\tilde{q}_i, \tilde{w}_i : i \in [P]\}$), the communication overhead is equal to $PM + P$ for downlink and
uplink, respectively. Throughout the paper, $M$ denotes the dimension of random features in (5) (i.e., the dimension of the parameter of each single-kernel function).

3 Vanilla Methods

In this section, we present two vanilla methods for KOFL framework, which can be constructed by properly combining some existing techniques. A single kernel-based method (termed S-KOFL) is first introduced, which is constructed by incorporating the RF approximation [14], online gradient descent (OGD) [15], and FedAvg [3]. In addition, harnessing the idea of multi-kernel extension in the centralized counterpart [11, 12], a vanilla multi-kernel KOFL (termed vM-KOFL) is constructed. Our key observation is that such extension causes a heavy communication overhead compared with S-KOFL, and the existing overhead-reduction techniques in federated learning cannot address such problem. This is the motivation of our work.

3.1 S-KOFL

In S-KOFL, the goal is to seek a sequence of RF-based kernel functions $f(x; \tilde{w}_1), \ldots, f(x; \tilde{w}_T)$ defined in (4), such as

$$f(x; \tilde{w}_i) = \tilde{w}_i^T z(x) \in \mathcal{H},$$

where $z(x)$ is given in (5) according to a preselected single kernel $k$. To optimize the above global models $\tilde{w}_t \in R^M, t = 1, \ldots, T$ in an online fashion, S-KOFL operates with the following two steps.

Local Model Update. At time $t$, every node $k$ receives the current global model $\tilde{w}_t$ form the server and an incoming data $(x_{k,t}, y_{k,t})$. Using them, it updates the local model via OGD [15]

$$\tilde{w}_{[k,t+1]} = \tilde{w}_t - \eta_t \nabla \mathcal{L}(\tilde{w}_t, z(x_{k,t}, y_{k,t})), \quad (8)$$

with a step size $\eta_t > 0$ and the initial value $\tilde{w}_1 = 0$, where $\nabla \mathcal{L}(\tilde{w}_t, z(x_{k,t}, y_{k,t})$ is the gradient at the point $\tilde{w}_t$. Then, each node $k$ sends the updated local model $\tilde{w}_{[k,t+1]} \in R^M$ back to the server.

Global Model Update. The server updates a global model from the aggregated models $\{\tilde{w}_{[k,t+1]} : k \in [K]\}$ via FedAvg [16]

$$\tilde{w}_{t+1} = \frac{1}{K} \sum_{k=1}^{K} \tilde{w}_{[k,t+1]} \quad (9)$$

Then, it distributes the updated global model $\tilde{w}_{t+1} \in R^M$ to the $K$ nodes. In S-KOFL, the uplink/downlink communication overhead is equal to $M$.

3.2 vM-KOFL

In vM-KOFL, a global learned function at time $t$ is represented as

$$f(x; \{\tilde{w}_{[i,j]} \}) = \sum_{i=1}^{P} \tilde{q}_{[i,j]} \tilde{w}_{[i,j]}^T z_i(x), \quad (10)$$

where the global model is defined as $\hat{w}_0 = \{\tilde{w}_{[i,j]} ; \tilde{q}_{[i,j]} : i \in [P]\}$. Then, vM-KOFL consists of the following two operations (see Algorithm 1).

Local Model Update. Given the global model $\{\tilde{w}_{[i,j]} : i \in [P]\}$ and an incoming data $(x_{k,t}, y_{k,t})$, each node $k$ updates its local parameters $\{\tilde{w}_{[k,t+1,i]} \}$ via OGD

$$\tilde{w}_{[k,t+1,i]} = \tilde{w}_{[k,t,i]} - \eta_i \nabla \mathcal{L}(\tilde{w}_{[k,t,i]}^T z_i(x_{k,t}, y_{k,t}), \quad (11)$$

for $\forall i \in [P]$, where $\eta_i > 0$ is a step size and the initial value is $\tilde{w}_{[k,t,i]} = 0$ for $\forall i \in [P]$. Also, it computes the reliabilities of the $P$ kernels with respect to its own local data

$$\hat{\epsilon}_{[k,t+1,i]} = \exp \left[-\eta \frac{1}{t} \sum_{r=1}^{t} \mathcal{L}(\tilde{w}_{[k,t,i]}^T z_i(x_{k,r}, y_{k,r}) \right], \quad (12)$$

with the initial values $\hat{\epsilon}_{[k,1,i]} = 1$ for $\forall i \in [P]$. Then, it sends the updated local model $\{\tilde{w}_{[k,t+1,i]} ; \hat{\epsilon}_{[k,t+1,i]} : i \in [P]\}$ back to the server.

Algorithm 1. Vanilla Method (vM-KOFL)

1: Input: $K$ edge nodes, a set of preselected $P$ kernels ($\kappa_p : p \in [P]$), and hyper-parameters ($\eta_i, \eta_h$).
2: Output: A sequence of global functions: $f(x; \{\tilde{q}_{[i,j]} , \tilde{w}_{[i,j]} : i \in [P]\} , t = 1, \ldots, T$.
3: Initialization: $\tilde{w}_{[i,j]} = 0$ for $\forall i \in [P]$. Each node has an identical random features $z_i(\cdot)$ for $\forall i \in [P]$.
4: Iteration: $t = 1, \ldots, T$
   - For every node $k \in [K]$:
     - Receive a streaming data $(x_{k,t}, y_{k,t})$.
     - Construct $z_i(x_{k,t}), \forall i \in [P]$ via (5).
     - Update the parameters $\{\tilde{w}_{[k,t+1,i]} : i \in [P]\}$ via (11).
     - Compute the reliabilities $\{\hat{\epsilon}_{[k,t+1,i]} : i \in [P]\}$ via (12).
     - Send the updated local model $\{\tilde{w}_{[k,t+1,i]} ; \hat{\epsilon}_{[k,t+1,i]} : i \in [P]\}$ to the server.
   - At the server:
     - Update the global parameter $w_{[t+1,i]}$ via (13).
     - Obtain the reliabilities $\hat{\epsilon}_{[t+1,i]}$ via (14).
     - Send the updated global model $\{w_{[t+1,i]}, \hat{\epsilon}_{[t+1,i]} : i \in [P]\}$ to the $K$ nodes.
   - S-KOFL follows the above procedures with a predetermined single kernel $k$ (i.e., $P = 1$).

Global Model Update. The server receives the updated local models $\{w_{[k,t+1,i]}, \hat{\epsilon}_{[k,t+1,i]} : i \in [P], k \in [K]\}$ from the $K$ nodes. First, the parameters of the $P$ kernel functions are obtained via FedAvg

$$w_{[t+1,i]} = \frac{1}{K} \sum_{k=1}^{K} w_{[k,t+1,i]}, \forall i \in [P]. \quad (13)$$

Then, the weights for combining the $P$ kernel functions, which is determined on the basis of the entire losses of the $K$ nodes, are computed as

$$\hat{q}_{[t+1,i]} = \frac{\prod_{k=1}^{K} \hat{\epsilon}_{[k,t+1,i]}}{\sum_{k=1}^{K} \prod_{k=1}^{K} \hat{\epsilon}_{[k,t+1,i]}}. \quad (14)$$

The above method to compute the combining weights is known as Exp strategy [25]. Finally, the server broadcasts the
updated global model \(\{\hat{w}_{t+1,i} : i \in [P]\}\) to the \(K\) nodes. In vM-KOFL, the uplink/downlink communication overhead is equal to \(P(M + 1)\).

**Remark 1.** We observe that vM-KOFL can enjoy the merit of multiple kernels at the expense of communication overheads. Especially, the uplink/downlink communication overhead of vM-KOFL grows linearly with the size of a kernel dictionary \(P\). Definitely, as in S-KOFL, the communication-reduction techniques in federated learning (e.g., quantization or sparsification) [17], [18], [20], [26] can be applied to vM-KOFL. We remark that these methods can only reduce the overhead term \(M\) as \(M \ll M\), whereas the term \(P\) is unchanged. This poses a new demanding problem on the construction of a communication-efficient multi-kernel method for KOFL framework. We contribute to this subject in the next section.

| Algorithm 2. Proposed Method (eM-KOFL) |
|----------------------------------------|
| 1: **Input:** \(K\) edge nodes, a set of preselected \(P\) kernels \(\{\kappa_i : i \in [P]\}\), and hyper-parameters \((\eta_q, \eta_w)\). |
| 2: **Output:** A sequence of common functions: \(t \in [T + 1]\) |
| \(f(x; \{\hat{q}_{t,i}, \hat{w}_{t,i} : i \in [P]\}), t = 1, \ldots, T.\) |
| 3: **Initialization:** \(\hat{w}_{1,i} = 0\) for \(\forall i \in [P]\). Each node has an identical random features \(z_i(\cdot)\) for \(\forall i \in [P]\). |
| 4: **Iteration:** \(t = 1, \ldots, T\) |
| \(\diamond \) For every node \(k \in [K]\) |
| \(\bullet \) Receive a streaming data \((x_{t,k}, y_{t,k})\). |
| \(\bullet \) Construct \(z_i(x_{t,k}), \forall i \in [P]\) via (5). |
| \(\bullet \) Update the local parameters \(\{\hat{g}_{t,i,k+1} : i \in [P]\}\) via (17). |
| \(\bullet \) Compute the reliabilities \(\{\hat{\ell}_{t,i,k+1} : i \in [P]\}\) via (12). |
| \(\bullet \) Send the updated local model \(\hat{w}_{t+1,k} = \hat{g}_{t,i,k+1}\) and \(\{\hat{\ell}_{t,i,k+2} : i \in [P]\}\) to the server. |
| \(\diamond \) At the server |
| \(\bullet \) Update the global parameter \(w_{t+1}\) via (20). |
| \(\bullet \) Randomly choose \(\hat{p}_{t+2}\) according to the PMF in (21). |
| \(\bullet \) Send the updated global model \(\{\hat{w}_{t+1,i}, \hat{p}_{t+2}\}\) to the \(K\) nodes. |

### 4 Proposed Methods

In this section, we propose a novel randomized algorithm (named eM-KOFL), which can enjoy the advantage of multiple kernels while having the almost same communication overhead as S-KOFL. It is remarkable that the uplink/downlink communication overhead of eM-KOFL is irrespective of \(P\) (i.e., the number of kernels) and thus, to achieve a higher learning accuracy, a sufficiently large number of kernels can be used without the burden of the communication overhead. The main idea of eM-KOFL is as follows. At every time \(t\), one kernel out of the entire \(P\) kernels is randomly selected according to a carefully designed probability mass function (PMF). Only local functions in the selected kernel are updated globally, while the other functions are updated locally, which ensures that the communication overhead no longer relies on \(P\). More importantly, it is proved that our PMF, obtained by the proposed delayed-Exp strategy, can guarantee that as \(t\) proceeds, the best kernel in hindsight is chosen with high probability.

In this section, we propose a novel randomized algorithm (named eM-KOFL), which can enjoy the advantage of multi-kernel method for KOFL framework. We contribute to this subject in the next section.

#### 4.1 eM-KOFL

In the proposed eM-KOFL, a global learned function at time \(t\) is fully determined by the parameters \(\{\hat{p}_t, \hat{w}_t\}\), i.e.,

\[
f(x; \{\hat{w}_t, \hat{p}_t\}) = \hat{w}_t^T z_{\hat{p}_t}(x) \in \mathcal{H}_{\hat{p}_t},
\]

where the global model is defined as \(m = \{\hat{p}_t, \hat{w}_t\}\). We note that \(\{\hat{p}_t, \hat{w}_t\}\) are random variables and thus, eM-KOFL is a randomized algorithm. Thus, unlike S-KOFL, the associated RKHS in eM-KOFL can change over the time. The proposed eM-KOFL consists of the following two operations (see Algorithm 2):

**Local Model Update.** Every node \(k\) has its own local information \(\{\hat{g}_{t,i,k} : i \in [P]\}\), which is not shared with the server. Also, at time \(t\), it receives the global model \((\hat{p}_{t-1}, \hat{w}_t)\) from the server. Leveraging \(\hat{p}_t\) (received at time \(t = t-1\)) and \(\hat{w}_t\), each node \(k\) updates the parameters of the \(P\) kernel functions as

\[
\hat{w}_{t,k,l} = \begin{cases} 
\hat{g}_{t,i,k+1}, & \text{if } i \neq \hat{p}_t \\
\hat{w}_t, & \text{if } i = \hat{p}_t,
\end{cases}
\]

for \(\forall i \in [P]\). Note that the kernel function belonging to \(\mathcal{H}_{\hat{p}_t}\) is only updated globally. Leveraging them, each node \(k\) updates its local information via OGD

\[
\hat{g}_{t,i,k+1} = \hat{w}_{t,k,l} - \eta_t \nabla \mathcal{L} \bigg( \hat{w}_{t,k,l}^T z_{\hat{p}_t}(x_{t,k}, y_{t,k}) \bigg),
\]

for \(\forall i \in [P]\), where \(\eta_t > 0\) is a step size. Also, from the received parameter \(\hat{p}_{t+1}\), it updates the local model to be conveyed to the server

\[
\hat{w}_{t+1,k} = \hat{g}_{t+1,i,k+1}.
\]

Likewise vM-KOFL, the accumulated losses (or reliabilities) of the \(P\) kernels are computed as

\[
\hat{\ell}_{t,i,k+2} = \exp \left[ -\eta_t \sum_{\tau = 1}^{t} \mathcal{L} \left( \hat{w}_{t,i,k+1}^T z_{\hat{p}_t}(x_{t,k}, y_{t,\tau}) \right) \right],
\]

where \(\hat{\ell}_{t,i,k+1} = 1\) for \(\forall i \in [P]\). Obviously, \(\hat{\ell}_{t,i,k+2}\) can be different from \(\hat{\ell}_{t,i,k+1}\) in (19). Finally, every node \(k\) sends the updated local model \(\hat{w}_{t+1,k}\) and \(\{\hat{\ell}_{t,i,k+2} : i \in [P]\}\) to the server. The corresponding communication overhead in the
uplink is equal to $M + P$. Since $P$ is generally much less than $M$ (e.g., $P = 11$ and $M = 100$ in our experiments), the uplink communication overhead can be less than $2M$.

**Global Model Update.** At time $t$, the server receives the updated local models $\{\hat{w}_{k,t+1} : k \in [K]\}$ and $\{\ell_{[k,t+2,i]} : i \in [P], k \in [K]\}$ from the $K$ nodes. As in S-KOFL, it updates the global parameter via FedAvg

$$
\bar{w}_{t+1} = \frac{1}{K} \sum_{k=1}^{K} \hat{w}_{k,t+1}.
$$

Next, the server chooses a kernel index $\tilde{p}_{t+2}$ by taking the reliabilities of the $P$ kernels into account. Specifically, $\tilde{p}_{t+2} \in [P]$ is chosen according to the following PMF

$$
\bar{q}_{t+2,i} = \frac{\prod_{k=1}^{K} \ell_{[k,t+2,i]} \bar{w}_{t+1,k}^{\tilde{p}_{t+2}}}{\sum_{i=1}^{P} \prod_{k=1}^{K} \ell_{[k,t+2,i]} \bar{w}_{t+1,k}^{\tilde{p}_{t+2}}}, \quad \forall i \in [P].
$$

The proposed method above is called Delayed-Exp strategy. Unlike the conventional Exp strategy [25], our strategy employs the one-time delayed information, i.e., $\bar{q}_{t+2,i}$ is determined on the bases of incoming data up to time $t$, whereas in Exp strategy, it is determined up to time $t + 1$. Finally, the server distributes the updated global model $\{\bar{w}_{t+1, \tilde{p}_{t+2}}\}$ to the $K$ nodes. The downlink communication overhead of eM-KOFL is equal to $M + 1$. We emphasize that unlike vM-KOFL, the uplink/downlink communication overhead of eM-KOFL does not grow with $P$.

### 4.2 pM-KOFL

We propose pM-KOFL as the communication-efficient variant of eM-KOFL, where the corresponding uplink/downlink communication overhead is equal to $M + 1$. Since pM-KOFL almost follows the procedures of eM-KOFL with some modifications, we only highlight such differences. The full descriptions of pM-KOFL are illustrated in Fig. 1.

In pM-KOFL, every node $k$ transmits a candidate kernel index $\tilde{p}_{[k,t+2]}$ rather than sending the accumulated losses $\{\ell_{[k,t+2,i]} : i \in [P]\}$ in eM-KOFL. The corresponding communication overhead can be reduced from $P$ to 1. To be specific, the candidate kernel $\tilde{p}_{[k,t+2]}$ is chosen according to the following local PMF

$$
\bar{q}_{[k,t+2,i]} = \frac{\ell_{[k,t+2,i]}^{\tilde{p}_{[k,t+2]}}}{\sum_{i=1}^{P} \ell_{[k,t+2,i]}^{\tilde{p}_{[k,t+2]}}}, \quad \forall i \in [P],
$$

where $\ell_{[k,t+2,i]}$ is defined in (19). Then, the server chooses a kernel index $\tilde{p}_{t+2}$ from the aggregated candidates $\{\tilde{p}_{[k,t+2]} : k \in [K]\}$ randomly and uniformly. By integrating the randomness at the $K$ nodes and the server, it can be interpreted that $\tilde{p}_{t+2}$ in pM-KOFL is chosen according to the following PMF

$$
\bar{q}_{t+2,i} = \frac{1}{K} \sum_{k=1}^{K} \bar{q}_{[k,t+2,i]},
$$

for $\forall i \in [P]$. Then, $\bar{q}_{t+2,i}$ can be considered as the proxy of the true PMF $\bar{q}_{t+2,i}$ in (21). Definitely pM-KOFL can approach the performance of eM-KOFL, provided that $\bar{q}_{t+2,i}$ is sufficiently close to $\bar{q}_{t+2,i}$ in Section 6, it will be demonstrated that the proxy in (23) is quite accurate so that pM-KOFL yields the same performance as eM-KOFL. This leads us to conclude that pM-KOFL can almost achieve the performance of vM-KOFL while having the same communication overhead as S-KOFL.

### 5 Regret Analysis

In this section, we theoretically prove that eM-KOFL attains the same asymptotic performance as vM-KOFL. Namely, both methods achieve a sublinear regret bound, i.e., $\text{regret}_{T} \leq O(\sqrt{T})$. Our analysis also reveals that eM-KOFL yields the same performance of the centralized counterpart [11], [12] asymptotically without sharing raw data (i.e., preserving an edge-node privacy). Thus, eM-KOFL is asymptotically optimal. Before stating our main theorems, we introduce some useful notations and assumptions. For any fixed kernel $K$, let $f(x; w_{t,i}) = w_{t,i}^{T}z_{i}(x)$ denote the best RF-based function belonging to RKHS $\mathcal{H}_{i}$, i.e.,

$$
\hat{w}_{t,i} = \arg \min_{w} \sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L}(w^{T}z_{i}(x_{t,k}), y_{t,k}).
$$
For any fixed $P$ kernels $\{k_i, i \in [P]\}$, the vanilla
method (vM-KOFL) in Algorithm 1 achieves the following regret bound
\[
\text{regret}_T = \sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L}(\mathbf{w}_{t-1}^T \mathbf{z}(\mathbf{x}_{t,k}), y_{t,k}) - \min_{1 \leq s \leq P} \sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L}(\mathbf{w}_{t-1}^T \mathbf{z}(\mathbf{x}_{t,k}), y_{t,k})
\leq \frac{K C}{2 \eta_t} + \frac{\eta_t K L T}{2} + \frac{\log P}{\eta_y} + \frac{\eta_y K^2 T}{2}.
\]
\[
\text{Proof.} \quad \text{The proof is provided in Section 5.1.}
\]

Using this notation, we can express the regret bound for any fixed kernel $k_i$ as
\[
\text{regret}_T = \sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L}(\mathbf{w}_{t-1}^T \mathbf{z}(\mathbf{x}_{t,k}), y_{t,k}) - \min_{1 \leq s \leq P} \sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L}(\mathbf{w}_{t-1}^T \mathbf{z}(\mathbf{x}_{t,k}), y_{t,k})
\leq \frac{K C}{2 \eta_t} + \frac{\eta_t K L T}{2} + \frac{\log P}{\eta_y} + \frac{\eta_y K^2 T}{2} + K \sqrt{\frac{\log \delta}{2}}.
\]
\[
\text{Proof.} \quad \text{The proof is provided in Section 5.2.}
\]

Unfortunately, it is not possible to theoretically prove if pM-KOFL satisfies the above condition for any decentralized dataset. Via numerical tests, we have confirmed that the above condition easily holds for our real datasets in Section 6. The corresponding result is shown in Fig. 2, where $y$-axis represents $\mathbb{P}(\hat{y}_t = \hat{y}_1)$ (which can clearly capture the left-hand side of (25)).

**Remark 3.** We give some discussions on our theoretical analysis with respect to the two major challenges in the conventional federated learning, such as non-identically and non-independentely distributed (non-IID) data (or data heterogeneity) and partial (node) participation [27]. The partial participation implies that only a random portion (e.g., 10%) of distributed nodes in the network transmit their local models to the server for FedAvg. We emphasize that our analytical results in Theorem 1 and Theorem 2 are valid for any dataset (e.g., non-IID data and any real-world data), provided that all the nodes in the network contribute to the global update. Namely, in the case of full participation, the proposed eM-KOFL is asymptotically optimal irrespective of the degree of heterogeneity of incoming data at the distributed nodes. However, our analysis could not cover the case of partial participation with non-IID data. In a simple single-kernel case, we cannot ensure that S-KOFL, based on OGD and FedAvg, yield an optimal sublinear regret bound. Harnessing the similarity between OGD and SGD, the existing ideas to correct for non-IID data in the conventional federated learning, such as adding a penalty (or proximal) term to local objective functions [28], [29], [30] or designing a new optimizer [31], can be applied to S-KOFL. Yet, the regret analysis of the resulting algorithms is not straightforward and should be further investigated. When considering multiple kernels, the problem becomes more challenging as the combination weights (or PMF) are also affected by the partial participation. An estimated PMF $\hat{q}_{t,i}$ (by some participated nodes) can be different from the true PMF $q_{t,i}$ (by the entire nodes). Obviously, this difference tends to be larger as the degree of data-heterogeneity increases. Under the partial participation and non-IID data, thus, our algorithm can achieve an optimal sublinear
regret bound only when the following condition holds
\[ \sum_{t=1}^{T} \sum_{i=1}^{P} |q_{t,i} - \bar{q}_{t,i}| \leq O(\sqrt{T}). \] (26)

The left-hand side of (26) can reflect the heterogeneity of the dataset. The above condition may not hold for the case of high-degree data heterogeneity and thus, in this case, the proposed delayed-Exp strategy should be further enhanced. In this scenario, it is an interesting research topic to construct an efficient multi-kernel algorithm with an analytical performance guarantee, which is beyond the scope of this paper. Instead, via experimental analysis in Section 6, we demonstrate that pM-KOFL (or eM-KOFL) can provide the robustness to the partial participation and non-IID data to some degree (see Fig. 5).

5.1 Proof of Theorem 1

We provide the proof of Theorem 1. Following the notations in Section 3.2, let \( \{\hat{w}_{t,i} : i \in [P]\} \) and \( \{\bar{q}_{t,i} : i \in [P]\} \) denote the global parameters of a learned function at time \( t \). Using these notations, we derive the two key lemmas.

**Lemma 1.** For any kernel \( k_i \) and any step size \( \eta_t > 0 \), S-KOFL in Section 3.1 achieves the following regret bound
\[
\sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L}\left(\hat{w}_{t,i}^T z_i(x_{t,k}), y_{t,k}\right) - \sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L}\left(\bar{w}_{t,i}^T z_i(x_{t,k}), y_{t,k}\right) \leq KC + \frac{\eta_t LK T}{2}.
\]

**Proof.** Fix a kernel \( k_i \) for any \( i \in [P] \). From OGD update in (8) and FedAvg in (9), the global model is updated as
\[
\hat{w}_{t+1,i} = \hat{w}_{t,i} - \frac{\eta_t}{K} \sum_{k=1}^{K} \nabla k_{t,i},
\] (27)
where for ease of exposition, we let
\[
\nabla k_{t,i} \triangleq \nabla \mathcal{L}\left(\hat{w}_{t,i}^T z_i(x_{t,k}), y_{t,k}\right).
\]
From (27), we have
\[
\left\|\hat{w}_{t+1,i} - \bar{w}_{t,i}\right\|^2 = \left\|\hat{w}_{t,i} - \bar{w}_{t,i}\right\|^2 + \frac{\eta_t^2}{K} \left\| \sum_{k=1}^{K} \nabla k_{t,i} \right\|^2 - \frac{\eta_t}{K} \sum_{k=1}^{K} \nabla k_{t,i} \left(\hat{w}_{t,i} - \bar{w}_{t,i}\right). \]
(29)

Using the convexity of a loss function, we can get
\[
\sum_{k=1}^{K} \mathcal{L}\left(\hat{w}_{t,i}^T z_i(x_{t,k}), y_{t,k}\right) - \mathcal{L}\left(\bar{w}_{t,i}^T z_i(x_{t,k}), y_{t,k}\right) \leq \sum_{k=1}^{K} \nabla k_{t,i} \left(\hat{w}_{t,i} - \bar{w}_{t,i}\right),
\]
(30)

From (29) and (30), we derive the following upper bound
\[
\sum_{k=1}^{K} \mathcal{L}\left(\hat{w}_{t,i}^T z_i(x_{t,k}), y_{t,k}\right) - \mathcal{L}\left(\bar{w}_{t,i}^T z_i(x_{t,k}), y_{t,k}\right) \leq K \left\|\hat{w}_{t,i} - \bar{w}_{t,i}\right\|^2 - \frac{\eta_t^2}{2K} \left\| \sum_{k=1}^{K} \nabla k_{t,i} \right\|^2 + \frac{\eta_t}{2K} \left\| \sum_{k=1}^{K} \nabla k_{t,i} \right\|^2,
\]
(a) \( K \left\|\hat{w}_{t,i} - \bar{w}_{t,i}\right\|^2 - \frac{\eta_t^2}{2K} \left\| \sum_{k=1}^{K} \nabla k_{t,i} \right\|^2 + \frac{\eta_t}{2K} \left\| \sum_{k=1}^{K} \nabla k_{t,i} \right\|^2,
\]
(31)
where (a) is from the Cauchy-Schwartz inequality. Using the telescoping sum over \( t = 1, 2, .., T \), we get
\[
\sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L}\left(\hat{w}_{t,i}^T z_i(x_{t,k}), y_{t,k}\right) - \mathcal{L}\left(\bar{w}_{t,i}^T z_i(x_{t,k}), y_{t,k}\right) \leq K \left\|\hat{w}_{t,i} - \bar{w}_{t,i}\right\|^2 - \frac{\eta_t^2}{2K} \left\| \sum_{k=1}^{K} \nabla k_{t,i} \right\|^2 + \frac{\eta_t}{2K} \left\| \sum_{k=1}^{K} \nabla k_{t,i} \right\|^2,
\]
(32)
where (a) follows \( \hat{w}_{t,i} = 0 \), Assumption 2, and Assumption 3. This completes the proof. \( \Box \)

**Lemma 2.** For any learning rate \( \eta_g > 0 \), Exp strategy guarantees the following regret bound
\[
\frac{\left|\tilde{w}_{t,i}^T z_i(x_{t,k}), y_{t,k}\right|}{\eta_g} \leq \frac{\log P}{\eta_g} + \frac{\eta_t K^2 T}{2}.
\]

**Proof.** The proof is completed from [12, Lemma 2]. \( \Box \)

Note that Lemma 1 holds for any kernel \( k_i \). Thus, combining Lemma 1 and Lemma 2, we complete the proof of Theorem 1.

5.2 Proof of Theorem 2

We provide the proof of Theorem 2. In this proof, we follow the notations in Section 4.1. For example, let \( \{\bar{w}_{t,i} : i \in [P]\} \) denote the local parameters at the node \( k \). Also, let \( w_t \) and \( \bar{q}_t \) denote the global parameters. Using them, we provide the key lemmas for the main proof.

**Lemma 3.** For any kernel \( i \in [P] \) and step size \( \eta_t > 0 \), the local kernel functions of eM-KOFL can achieve the following regret bound
\[
\sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L}(\bar{w}_{t,i}^T z_i(x_{t,k}), y_{t,k}) - \mathcal{L}(\bar{w}_{t,i}^T z_i(x_{t,k}), y_{t,k}) \leq \sum_{k=1}^{K} \nabla k_{t,i} \left(\hat{w}_{t,i} - \bar{w}_{t,i}\right),
\]
(30)
\[
\sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L}\left(\tilde{w}_{[k,t,i]}^T z_t(x_{k,t}), y_{k,t}\right) - \sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L}\left(w_{[k,t,i]}^T z_t(x_{k,t}), y_{k,t}\right)
\leq \frac{KC}{2\eta} + \frac{\eta KLT}{2}.
\]

**Proof.** Given the global parameters \(\bar{p}_t \in [P]\) and \(\bar{w}_{i,t}\), and from (17), the local parameters are updated via OGD as

\[
\check{g}_{[k,t+1,i]} = \bar{w}_{[k,t,i]} - \eta_t \nabla \mathcal{L}(\bar{w}_{[k,t,i]}^T z_t(x_{k,t}), y_{k,t}),
\]

for \(\forall i \in [P]\), where

\[
\bar{w}_{[k,t,i]} = \begin{cases} 
\check{g}_{[k,t,i]} & \text{if } i \neq \bar{p}_t \\
\bar{w}_t - \frac{1}{K} \sum_{k=1}^{K} \check{g}_{[k,t,i]} & \text{if } i = \bar{p}_t.
\end{cases}
\]

Then, from (33), we have that for any \(k \in [K]\)

\[
\|\bar{w}_{[k,t,i]} - w^*\|^2 = \|\bar{w}_{[k,t,i]} - \eta_t \nabla \mathcal{L}(\bar{w}_{[k,t,i]}^T z_t(x_{k,t}), y_{k,t}) - w^*\|^2
\]

\[
= \|\bar{w}_{[k,t,i]} - w^*\|^2 + \eta_t^2 \|\nabla \mathcal{L}(\bar{w}_{[k,t,i]}^T z_t(x_{k,t}), y_{k,t})\|^2 - 2\eta_t \nabla \mathcal{L}(\bar{w}_{[k,t,i]}^T z_t(x_{k,t}), y_{k,t})^T (\bar{w}_{[k,t,i]} - w^*).
\]

For ease of exposition, throughout the proof, we let

\[
\nabla_{[k,t,i]} \defeq \nabla \mathcal{L}(\bar{w}_{[k,t,i]}^T z_t(x_{k,t}), y_{k,t}).
\]

According to the two cases in (34), we can get:

i) When \(\bar{w}_{[k,t,i]} = \check{g}_{[k,t,i]}\) (i.e., \(i \neq \bar{p}_t\)), from (17), we have

\[
\|\check{g}_{[k,t+1,i]} - w^*\|^2 = \|\check{g}_{[k,t,i]} - w^*\|^2 + \eta_t^2 \|\nabla_{[k,t,i]}^T \bar{w}_{[k,t,i]} - w^*\|^2.
\]

ii) when \(\bar{w}_{[k,t,i]} = \frac{1}{K} \sum_{k=1}^{K} \check{g}_{[k,t,i]}\) (i.e., \(i = \bar{p}_t\)), from (17), we have

\[
\sum_{k=1}^{K} \|\check{g}_{[k,t+1,i]} - w^*\|^2 = \sum_{k=1}^{K} \|\bar{w}_{[k,t,i]} - \eta_t \nabla_{[k,t,i]} - w^*\|^2
\]

\[
= \sum_{k=1}^{K} \|\bar{w}_{[k,t,i]} - w^*\|^2 + \eta_t^2 \sum_{k=1}^{K} \|\nabla_{[k,t,i]}\|^2 - 2\eta_t \nabla_{[k,t,i]}^T (\bar{w}_{[k,t,i]} - w^*).
\]

where (a) is due to the fact that for any \(k \in [K]\), we have

\[
\|\bar{w}_{[k,t,i]} - w^*\|^2 = \left\| \frac{1}{K} \sum_{k=1}^{K} \check{g}_{[k,t,i]} - w^* \right\|^2
\]

\[
\leq \frac{1}{K^2} \sum_{k=1}^{K} \|\check{g}_{[k,t,i]} - w^*\|^2
\]

\[
\leq \frac{1}{K} \sum_{k=1}^{K} \|\check{g}_{[k,t,i]} - w^*\|^2,
\]

where (a) is and (b) are from the triangle inequality and Cauchy-Schwartz inequality, respectively. Also, from the convexity of a loss function, we obtain that for any \(k \in \mathcal{V}\)

\[
\mathcal{L}(\bar{w}_{[k,t,i]}^T z_t(x_{k,t}), y_t) - \mathcal{L}(w^T z_t(x_{k,t}), y_t)
\]

\[
\leq \nabla_{[k,t,i]} (\bar{w}_{[k,t,i]}^T z_t(x_{k,t}) - w^T z_t).
\]

Plugging (39) into (37) and (38) separately, and combining the two cases, we can get

\[
\sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L}(\bar{w}_{[k,t,i]}^T z_t(x_{k,t}), y_t) - \mathcal{L}(w^T z_t(x_{k,t}), y_t)
\]

\[
\leq \sum_{t=1}^{T} \sum_{k=1}^{K} \|\check{g}_{[k,t+1,i]} - w^*\|^2 + \eta_t^2 \sum_{k=1}^{K} \|\nabla_{[k,t,i]}\|^2
\]

\[
\leq \frac{2\log P}{\eta_y} + \frac{\eta_y K^2 T}{2}.
\]

**Lemma 4.** For any learning rate \(\eta_y > 0\), the proposed delayed-Exp strategy guarantees the following regret bound

\[
\sum_{t=1}^{T} \sum_{k=1}^{K} \sum_{i=1}^{P} \check{g}_{[k,t,i]} \mathcal{L}(\bar{w}_{[k,t,i]}^T z_t(x_{k,t}), y_t) - \min_{1 \leq j \leq P} \sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L}(w^T z_t(x_{k,t}), y_t)
\]

\[
\leq \frac{2\log P}{\eta_y} + \frac{\eta_y K^2 T}{2}.
\]

**Proof.** For ease of exposition, we define

\[
\tilde{f}_{[k,t,i]}(x) \defeq \bar{w}_{[k,t,i]}^T z_t(x).
\]
The proof will be completed using the upper and lower bounds on $\xi$, which is defined as
\[
\xi = \sum_{t=1}^{T} \log \left[ \sum_{i=1}^{P} \tilde{E}_{k,t+1,i} \right] - \log \left[ \sum_{i=1}^{P} \tilde{E}_{k,t,i} \right] + \log \left[ \sum_{i=1}^{P} \tilde{E}_{k,t-1,i} \right] - \log \left[ \sum_{i=1}^{P} \tilde{E}_{k,t,i} \right] - 2 \log P \leq -2 \eta_g \min_{1 \leq i \leq P} \left[ \mathcal{L} \left( \tilde{w}_{k,t,i}^T \tilde{z}_i(x_{k,t}), y_{k,t} \right) \right] - 2 \log P.
\]

where (a) follows the relation in (46), (b) is from the telescoping sum, (c) is due to the fact that $\sum_{i=1}^{P} \tilde{E}_{k,t+1,i} \geq 0$, (d) is from the fact that the accumulated loss is non-negative. From the lower and upper bounds, we can get
\[
-2 \eta_g \min_{1 \leq i \leq P} \left[ \mathcal{L} \left( \tilde{w}_{k,t,i}^T \tilde{z}_i(x_{k,t}), y_{k,t} \right) \right] - 2 \log P \leq -2 \eta_g \min_{1 \leq i \leq P} \left[ \mathcal{L} \left( \tilde{w}_{k,t,i}^T \tilde{z}_i(x_{k,t}), y_{k,t} \right) \right] + \frac{\eta_g^2 K^2 T}{2}.
\]

Rearranging the above term, we finally obtain
\[
\sum_{t=1}^{T} \sum_{i=1}^{P} \mathcal{L} \left( \tilde{w}_{k,t,i}^T \tilde{z}_i(x_{k,t}), y_{k,t} \right) - 2 \log P \leq -2 \log P + \frac{\eta_g K^2 T}{2},
\]

where we used the fact that
\[
\sum_{t=1}^{T} \sum_{i=1}^{P} \mathcal{L} \left( \tilde{w}_{k,t,i}^T \tilde{z}_i(x_{k,t}), y_{k,t} \right) \geq \min_{1 \leq i \leq P} \sum_{t=1}^{T} \sum_{i=1}^{P} \mathcal{L} \left( \tilde{w}_{k,t,i}^T \tilde{z}_i(x_{k,t}), y_{k,t} \right).
\]

This completes the proof. \(\square\)

We remark that Lemma 3 and Lemma 4 hold for any realizations of our randomized algorithm. We next prove that our randomized algorithm, choosing one kernel at every time instead of the combination of all $P$ kernels, only leads to a bounded loss compared with using the combination of the $P$ kernels.

**Lemma 5.** For some $\delta > 0$, the proposed randomized algorithm achieves the following regret bound with at least probability $1 - \delta$
\[
\sum_{t=1}^{T} \sum_{i=1}^{P} \mathcal{L} \left( \tilde{w}_{k,t,i}^T \tilde{z}_i(x_{k,t}), y_{k,t} \right) - 2 \log P \leq K \sqrt{\frac{\log(\delta^{-1})}{2} T}.
\]

**Proof.** We first define $\tilde{w}_{k,t,i}$ as
\[
\tilde{w}_{k,t,i} = \frac{1}{K} \sum_{k=1}^{K} \tilde{g}_{k,t,i}, \forall i \in [P].
\]

Note that $\tilde{w}_{k,t,i} = \tilde{w}_{k,t,i}$ only when $i = \tilde{p}_t$. Then, we define a random variable $X_i$.
\[ X_{k,t} = \mathcal{L}(\hat{w}_{k}^T z_p(x_k, t), y_{k,t}) - \sum_{i=1}^{P} \tilde{q}_{[k,i]} \mathcal{L}(\hat{w}_{[k,i]}^T z_p(x_{k,i}), y_{k,i}). \]  

(49)

Note that \( \tilde{q}_{[k,i]} \) is obtained as a consequence of random variables \( p_1, \ldots, p_{i-1} \). Also, \( \hat{p}_k \) is chosen according to the PMF \( (\tilde{q}_{[k,1]}, \ldots, \tilde{q}_{[k,P]}). \) Let \( \mathcal{F}_t = \sigma(p_1, p_2, \ldots, \hat{p}_k) \) be the smallest sigma algebra such that \( (p_1, p_2, \ldots, \hat{p}_k) \) is measurable. Then, \( \{ \mathcal{F}_t : t = 1, \ldots, T \} \) is filtration and \( X_{k,t} \) is \( \mathcal{F}_t \)-measurable. Note that the condition on \( \mathcal{F}_{t-1} \), \( \tilde{q}_{[k,i]} \) is fixed and \( p_k \) and \( \hat{w}_t \) are random variables. Using this fact, we have

\[ \mathbb{E}[X_{k,t} | \mathcal{F}_{t-1}] = 0, \]

because

\[ \mathbb{E}[\mathcal{L}(\hat{w}_{k}^T z_p(x_k, t), y_{k,t}) | \mathcal{F}_{t-1}] = \sum_{i=1}^{P} \tilde{q}_{[k,i]} \mathbb{E}[\mathcal{L}(\hat{w}_{[k,i]}^T z_p(x_{k,i}), y_{k,i})]. \]

Hence, \( \{ X_{k,t} : t \in [T] \} \) is a martingale difference sequence and \( X_{k,t} \in [B_t, B_t + \epsilon_t] \) is bounded, where \( B_t \) is a random variable and \( \mathcal{F}_{t-1} \) measurable, and \( \epsilon_t = 1 \), where

\[ B_t \triangleq - \sum_{i=1}^{P} \tilde{q}_{[k,i]} \mathcal{L}(\hat{w}_{[k,i]}^T z_p(x_{k,i}), y_{k,i}). \]

(50)

From Azuma-Hoeffding’s inequality, the following bound holds for some \( \delta > 0 \) with at least probability \( 1 - \delta \)

\[ \sum_{t=1}^{T} X_{k,t} \leq \sqrt{\frac{\log \delta^{-1}}{2} T}. \]

(51)

Since this is true for any \( k \in \mathcal{V} \), we have

\[ \sum_{k=1}^{K} \sum_{t=1}^{T} X_{k,t} \leq K \sqrt{\frac{\log \delta^{-1}}{2} T}. \]

(52)

Also, we have

\[ \sum_{k=1}^{K} \sum_{i=1}^{P} \tilde{q}_{[k,i]} \mathcal{L}(\hat{w}_{[k,i]}^T z_p(x_{k,i}), y_{k,i}) \]

\[ \overset{(a)}{=} \sum_{k=1}^{K} \sum_{i=1}^{P} \tilde{q}_{[k,i]} \mathcal{L} \left( \frac{1}{K} \sum_{k=1}^{K} \hat{w}_{[k,i]}^T z_p(x_{k,i}), y_{k,i} \right) \]

\[ \overset{(b)}{\leq} \sum_{k=1}^{K} \sum_{i=1}^{P} \tilde{q}_{[k,i]} \mathcal{L}(\hat{w}_{[k,i]}^T z_p(x_{k,i}), y_{k,i}), \]

(53)

where \( (a) \) is from the definitions of \( \hat{w}_{[k,i]} \) and \( \tilde{q}_{[k,i]} \) and \( (b) \) is due to the convexity of loss function (i.e., Assumption 1). Combining (49), (52), and (53), the proof is completed.

Combining Lemmas 3, 4 and 5, we can complete the proof of Theorem 2.

### 6 Experimental Results

In this section, we demonstrate the superiority of the proposed eM-KOFL and pM-KOFL via experiments with real datasets on online regression and time-series prediction tasks. As benchmark methods, the two vanilla methods as S-KOFL and vM-KOFL are considered. We believe that they are reasonable baseline methods because they are constructed by leveraging the best-known federated learning and multi-kernel learning approaches. Also, to the best of our knowledge, no other method for KOFL framework can be found. In our experiments, we consider a communication network consisting of \( K = 20 \) and \( K = 100 \) decentralize nodes for small-scale and large-scale datasets, respectively. A regularized least-square loss function, i.e.,

\[ \mathcal{L}(w^T z(x), y) = [w^T z(x) - y]^2 + \lambda \| w \|^2. \]

(54)

The learning accuracy at time \( t \) is measured by the cumulative mean-square-error (MSE) as

\[ \text{MSE}(t) = \frac{1}{tR} \sum_{t=1}^{t} \sum_{k=1}^{K} (\hat{y}_{k,t} - y_{k,t})^2, \]

(55)

where \( \hat{y}_{k,t} \) and \( y_{k,t} \) denote a predicted label and a true label, respectively. Due to the randomness of the above methods caused by the RF approximation, the average MSE performances over 50 trials are evaluated. Also, the following hyper-parameters are used

\[ \eta_1 = 0.5, \eta_2 = 0.5, \lambda = 0.01, \quad \text{and} \quad M = 100. \]

(56)

The above parameters can be further elaborated. However, as noticed in [12], such hyper-parameter optimization is still an open problem even in a simpler centralized network. In our experiments, thus, one pair of the hyper-parameters in (56) are used for all test datasets. We build the kernel dictionary consisting of 11 Gaussian kernels (i.e., \( P = 11 \)), each of which is defined with the following basis kernels

\[ k_i(x, x') = \exp \left( -\frac{\| x - x' \|^2}{\sigma_i^2} \right). \]

(57)

with the parameters (or bandwidths)

\[ \sigma_i = 10^{(i-6)/2}, \quad i = 1, 2, \ldots, 11. \]

(58)

Finally, the real datasets for our experiments on online regressions and time-series predictions are described in Sections 6.1 and 6.2, respectively. They are also summarized in Table 2.

Performance Evaluations. We first verify that pM-KOFL can operate as equivalently as eM-KOFL, i.e., the true PMF in (21) can be well-approximated by the proxy PMF in (23). The corresponding numerical result is illustrated in Fig. 2, where \( \mathbb{P}(p_i = \hat{p}_i) \) is computed empirically with 100 samples. Recall that \( \hat{p}_i \) and \( \hat{p}_i \) indicate the best kernel indices at time \( t \) randomly chosen by the true PMF (in eM-KOFL) and the proxy PMF (in pM-KOFL), respectively. It is clearly shown that after a certain time (called a mixing time), pM-KOFL and eM-KOFL operate at the same way with high probability. Furthermore, the mixing time is extremely fast. For this
reason, pM-KOFL can yield the almost same performance as eM-KFOL with not-so-large number of incoming data (e.g., $T$ is finite). Next, we demonstrate the effectiveness of the proposed methods on various online learning tasks. Fig. 3 shows the MSE performances on online regression tasks with real datasets in Section 6.1. We identify that multi-kernel based methods as vM-KOFL, eM-KFOL, and pM-KOFL yield more stable performances than the single kernel methods. In contrast, S-KOFL can provide an attractive performance only when a proper single kernel is preselected. Otherwise, S-KOFL deteriorates the learning accuracy considerably. This situation can be happened in real-world applications and thus, S-KOFL is not recommended in practice. Notably, both eM-KFOL and pM-KOFL attain the almost same performance as vM-KOFL for all real datasets, where the performance of vM-KOFL can be regarded as the best performance (lower bound) under KOFL framework. Namely, they can fully enjoy the advantage of multiple kernels as in vM-KOFL while having a similar communication overhead with S-KOFL. One can expect that without increasing the communication overheads, pM-KOFL (or eM-KFOL) yields an outstanding performance for any real-world application using a sufficiently large number of kernels. This surprising result could not been attained from the vanilla vM-KOFL. The exactly same results have been observed in Fig. 4 on time-series prediction tasks. This verifies that the proposed pM-KOFL and eM-KFOL can give stable performances on various online learning tasks. These numerical results suggest the practicality of the proposed methods.

We next evaluate the robustness of the proposed pM-KOFL on the partial participation and non-IID (or heterogeneous) data. Toward this, the real datasets (e.g., Twitter, Air quality, Parking occupancy, and Power consumption) are partitioned in a non-IID fashion as follows. Given the dataset $\{(x_t, y_t) : t \in [KT]\}$, they are sorted in ascending order with respect to $\{y_t : t \in [KT]\}$ and the sorted dataset is denoted as $\{(x_t, y_t) : j \in [KT]\}$ with $y_1 \leq y_2 \leq \cdots \leq y_{KT}$. Then, the local data of the node $k$ is obtained as

$$D_k = \{(x_{k,t}, y_{k,t}) : t \in [T]\},$$

for FedAvg. Also, the 10% of the nodes are selected uniformly at random and independently from time-to-time (called uniform selection). Only for the purpose of verifying the heterogeneity of the dataset in (59), we consider the so-called biased selection, in which 10% of nodes are selected uniformly at random and independently from the first

---

| Datasets              | # of features | # of data | feature type |
|-----------------------|---------------|-----------|--------------|
| Online regression tasks |               |           |              |
| Twitter               | 77            | 98704     | real & integer |
| Conductivity          | 81            | 11000     | real         |
| Air quality           | 13            | 38563     | real         |
| Time-series prediction tasks |       |           |              |
| Power consumption     | s=5           | 100000    | real         |
| Parking occupancy     | s=5           | 21500     | real         |
| Traffic               | s=5           | 6500      | real         |
nodes. Due to the aforementioned data partition, it is expected that a learned global function based on biased selection is customized to the data with smaller labels. In Fig. 5, MSE results obtained by biased selection verify that our data partition indeed reflects the non-IID nature. From Fig. 5, we observe that the proposed pM-KOFL can provide the robustness to non-IID data and partial participation when online regression and time-series prediction tasks are considered. Namely, our algorithm is bearable to the degree of heterogeneity of the datasets used in our experiments. As explained in Remark 3, nonetheless, the robustness of our algorithm has not been theoretically proved yet. Such rigorous analysis requires extensive additional efforts, which is left for a future work.

6.1 Online Regression Tasks

In the experiments of online regression tasks, the following popular real datasets from UCI Machine Learning Repository are considered:

- **Twitter [32]**: Data contains buzz events from Twitter, where each attribute is used to predict the popularity of a topic. Higher value indicates more popularity.
- **Conductivity [33]**: Data contains samples of extracted from superconductors, where each feature represents critical information to construct superconductor such as density and mass of atoms. The goal is to predict the critical temperature to create superconductor.
- **Air quality [34]**: Data includes samples, of which features include hourly response from an array of chemical sensors embedded in a city of Italy. The task is to predict the concentration of polluting chemicals in the air.

6.2 Time-Series Prediction Tasks

We consider time-series prediction tasks which estimate the future values in online fashion. As in the centralized counterpart [12], the famous time-series prediction method called Autogressive (AR) model is considered. An AR(s) model predicts the future value \( y_t \) assuming the linear dependency on its \( s \) values, i.e.,

\[
y_t = \sum_{i=1}^{s} \gamma_i y_{t-i} + n_t, \tag{60}
\]

where \( \gamma_i \) denotes the weight for \( y_{t-i} \) and \( n_t \) denotes a Gaussian noise at time \( t \). Based on this, the RF-based kernelized AR(s) model, which can explore a nonlinear dependency, is introduced in [12]

\[
y_t = f_t(x_t) + n_t, \tag{61}
\]

where \( x_t = [y_{t-1}, \ldots, y_{t-s}]^T \). The proposed pM-KOFL aims at learning \( f_t(\cdot) \) with a parameterized model \( f(x_t; \{ p_t, w_t \}) = w_t z_{p_t}(x) \). The other methods can be defined similarly. Then, the proposed and benchmark methods are tested with the following univariate time-series datasets from UCI Machine Learning Repository:

- **Power consumption [35]**: Data contains samples, each of which represents the active energy consumed every minute (in watt per hour) in the household by electrical equipment.
- **Parking occupancy [36]**: Data contains samples obtained from the parking lot in Birmingham, each of which indicates the car park occupancy rate.
Traffic [37]: Data contains the time-series traffic data obtained from Minneapolis Department of Transportation in US. Data is collected from hourly interstate 94 Westbound traffic volume for MN DoT ATR station 301, roughly midway between Minneapolis and St Paul, MN.

7 Conclusion

We proposed a novel randomized algorithm (named eM-KOFL) for kernel-based online federated learning (KOFL) framework. It was theoretically proved that eM-KOFL can achieve the same asymptotic performance as the vanilla multi-kernel method (termed vM-KOFL), while having a lower communication overhead. Also, our analysis revealed that eM-KOFL yields the same asymptotic performance as the centralized counterpart without sharing raw data (i.e., preserving an edge-node privacy). Focusing on a practical aspect, we presented the communication-efficient variant of eM-KOFL by mimicking the delayed-Exp strategy in an efficient way. The proposed method is named pM-KOFL. Via experiments with real datasets, we demonstrated the effectiveness of the proposed eM-KOFL and pM-KOFL on various online learning tasks. In particular, pM-KOFL can yield the almost same performance as vM-KOFL while having the 1/P uplink/downlink communication overhead, where P denote the size of a kernel dictionary. These suggest the practicality of pM-KOFL. One interesting extension is to build the so-called collaborative KOFL by integrating collaborating learning with KOFL so as to enable edge nodes to engage in KOFL framework without directly connecting the server. Another interesting future work is to provide more rigorous theoretical analysis on our algorithm by taking into account the impact of partial participation and non-IID data.

REFERENCES

[1] A. Hard et al., “Federated learning for mobile keyboard prediction,” 2018, arXiv:1811.03604.
[2] P. Kairouz et al., “Advances and open problems in federated learning,” 2019, arXiv:1912.04977.
[3] Q. Yang, Y. Liu, T. Chen, and Y. Tong, “Federated machine learning: Concept and applications,” ACM Trans. Intell. Syst. Technol., vol. 10, no. 2, pp. 1–19, 2019.
[4] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, “A public domain dataset for human activity recognition using smartphones.” Esann, vol. 3, 2013, Art. no. 3.
[5] A. Pantelopoulos and N. G. Bourbakis, “A survey on wearable sensor-based systems for health monitoring and prognosis,” IEEE Trans. Syst., Man, Cybern., vol. 40, no. 1, pp. 1–12, Jan. 2010.
[6] P. Rashidi and D. J. Cook, “Keeping the resident in the loop: Adapting the smart home to the user,” IEEE Trans. Syst., Man, Cybern.—Part A: Syst. Hum., vol. 39, no. 5, pp. 949–959, Sep. 2009.
[7] C. Richard, J. C. M. Bermudez, and P. honeine, “Online prediction of time series data with kernels,” IEEE Trans. Signal Process., vol. 57, no. 3, pp. 1058–1067, Mar. 2009.
[8] S. Shen, H. Jiang, and T. Zhang, “Stock market forecasting using machine learning algorithms,” in Proc. Dept. Elect. Eng., Stanford Univ., Stanford, CA, 2002, pp. 1–5.

[9] J. Kivinen, A. J. Smola, and R. C. Williamson, “Online learning with kernels,” IEEE Trans. Signal Process., vol. 52, no. 8, pp. 2165–2175, Aug. 2004.

[10] J. Shawe-Taylor, N. Cristianini, Kernel Methods for Pattern Analysis. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[11] Y. Shen, T. Chen, and G. B. Giannakis, “Random feature-based online multi-kernel learning in environments with unknown dynamics,” J. Mach. Learn. Res., vol. 20, no. 1, pp. 773–808, 2019.

[12] S. Hong and J. Chae, “Active learning with multiple kernels,” IEEE Trans. Neural Netw. Learn. Syst. early access, Jan. 14, 2021.

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

[14] A. Rahimi and B. Recht, “Random features for large-scale kernel machines,” in Proc. Adv. Neural Inf. Process. Syst., 2008, pp. 1177–1184.

[15] E. Hazan et al., “Introduction to online convex optimization,” Found. Trends Optim., vol. 2, no. 3–4, pp. 157–325, 2016.

[16] H. Yuan and T. Ma, “Federated accelerated stochastic gradient descent,” 2020, arXiv:2006.08950.

[17] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, “QSGD: Communication-efficient SGD via gradient quantization and encoding,” in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 1709–1720.

[18] J. Wangni, J. Wang, J. Liu, and T. Zhang, “Gradient sparsification for communication-efficient distributed optimization,” in Proc. 32nd Int. Conf. Neural Inf. Process. Syst., 2018, pp. 1306–1316.

[19] D. Alistarh, T. Hoeffer, M. J ohansson, S. Khrirat, N. Konstantinov, and C. Renggli, “The convergence of sparsified gradient methods,” in Proc. 32nd Int. Conf. Neural Inf. Process. Syst., pp. 5977–5988, 2018.

[20] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, “signSGD: Compressed optimisation for non-convex problems,” in Proc. Int. Conf. Learn. Intell. Optim., 2018, pp. 5977–5988.

[21] M. Gonen and E. Alpaydin, “Multiple kernel learning algorithms,” J. Mach. Learn. Res., vol. 12, pp. 2211–2268, 2011.

[22] J. Chae and S. Hong, “Distributed online learning with multiple kernels,” IEEE Trans. Neural Netw. Learn. Syst., early Access, Aug. 23, 2023, doi: 10.1109/TNNLS.2021.3105146.

[23] S. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, “Scaffold: Stochastic controlled averaging for on-device federated learning,” Proc. 37th Int. Conf. Mach. Learn., 2020, pp. 119:5132–5143.

[24] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, “‘Federated optimization in heterogeneous networks,” 2018, arXiv:1812.06127.

[25] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, “FedDANE: A federated newton-type method,” in Proc. 53rd Asilomar Conf. Signals, Syst. Comput., 2019, pp. 1227–1231.

[26] V. Smith, C.-K. Chiang, M. Sanjabi, and A. Talwalkar, “Federated multi-task learning,” in Proc. 31st Int. Conf. Neural Inform. Process. Syst., 2017, pp. 4427–4437.

[27] E. G. Frazeeos Kawala, Ahlame Douzal-Chouakria and E. Dimert, “Predictions d’activité dans les reseaux sociaux en ligne,” 4i’eme Conférence sur les Modèles et l’Analyse des Reseaux: Approches Mathématiques et Informatiques, 2015.

[28] K. Hamidieh, “A data-driven statistical model for predicting the critical temperature of a superconductor,” Comput. Mater. Sci., vol. 154, pp. 346–354, 2018.

[29] M. P. L. M. Saverio De Vito, Ettore Massera and G. D. Francia, “On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario,” Sensors Actuators B: Chem., vol. 281, pp. 759–769, 2018.

[30] Georges hebrail. uci machine learning repository. Accessed: Apr. 11, 2021. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/

[31] D. H. S. A. Camero, J. Toutouh and E. Alba, “Evolutionary deep learning for car park occupancy prediction in smart cities,” in Proc. Int. Conf. Learn. Intell. Optim., 2018, pp. 386–401.

[32] D. Du and C. Graff, “UCI machine learning repository,” 2017. [Online]. Available: http://archive.ics.uci.edu/ml

Songnam Hong (Member, IEEE) received the BSc and MSc degrees in electrical and computer engineering from Hanyang University, Seoul, South Korea, and the PhD degree in electrical engineering from the University of Southern California, Los Angeles, CA, in 2003, 2005, and 2011, respectively. From 2005 to 2009, he was a research engineer with Telecommunication R&D Center, Samsung Electronics, Suwon, South Korea. From 2014 to 2016, he was a senior research engineer with Ericsson Research, San Jose, CA. From 2016 to 2020, he was an assistant professor with Ajou University, Suwon, South Korea. He is currently an associate professor with Hanyang University, Seoul, South Korea. His research interests include the areas of large-scale optimization, information theory, machine or deep learning, and statistical signal processing.

Jeongmin Chae (Student Member, IEEE) received the MSc degree in electrical and computer engineering from Ajou University, Suwon, South Korea, in 2020. She is currently working toward the PhD degree in electrical engineering from the University of Southern California, Los Angeles, CA, USA, where she has been engaged in structured matrix completion and active matrix factorization. Her research interests include the areas of high-dimensional data inference, theoretical machine learning, and statistical signal processing.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.