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

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 using distributed sequential data at local devices. In this framework, we first introduce a *single* kernel-based OFL (termed S-KOFL) by incorporating the random-feature (RF) approximation, online gradient descent (OGD), and federated averaging (FedAvg) properly. However, it is nontrivial to develop a communication-efficient method with multiple kernels. One can construct a multi-kernel method (termed vM-KOFL) by following the extension principle in the centralized counterpart. This vanilla method is not practical as the communication overhead grows linearly with the size of a kernel dictionary. Moreover, this problem is not addressed via the existing communication-efficient techniques in federated learning such as quantization or sparsification. Our major contribution is to propose a novel randomized algorithm (named eM-KOFL), which can enjoy the advantage of multiple kernels while having a similar communication overhead with S-KOFL. It is theoretically proved that eM-KOFL yields the same asymptotic performance as vM-KOFL, i.e., both methods achieve an optimal sublinear regret bound. Mimicking the key principle of eM-KOFL efficiently, pM-KOFL is presented. Via numerical tests with real datasets, we demonstrate that pM-KOFL can yield the same performances as vM-KOFL and eM-KOFL on various online learning tasks while having the same communication overhead as S-KOFL. These suggest the practicality of the proposed pM-KOFL.

Index Terms—Federated learning, online learning, kernel-based learning, reproducing kernel Hilbert space (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.) train a global function collaboratively under the coordination of a server without sharing raw data placed at edge nodes [1], [2]. Specifically, federated learning optimizes a global 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) using 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.

- *S. Hong is with the Department of Electronic Engineering, Hanyang University, Seoul, 04763, Korea. (E-mail: snhong@hanyang.ac.kr) E-mail: snhong@hanyang.ac.kr*
- *J. Chae is with the Department of Electrical Engineering, University of Southern California, CA, 90089, USA. E-mail: chaej@usc.edu*

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

$$\hat{\mathbf{m}}_{t+1}=h\left(\hat{\mathbf{m}}_{[1,t+1]},...,\hat{\mathbf{m}}_{[K,t+1]}\right), \tag{1}$$

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 (FedAvg) [3]. Given the global model, every node k can estimate the label of a newly incoming data $\mathbf{x}_{k,t+1}$ such as

$$\hat{y}_{k,t+1}=f(\mathbf{x}_{k,t+1};\hat{\mathbf{m}}_{t+1}). \tag{2}$$

In KOFL framework, numerous methods can be constructed according to underlying optimization and learning algorithms. We first introduce a *single* kernel-based method (termed S-KOFL) by properly incorporating the random-feature (RF) approximation [14], online gradient descent (OGD) [15], and the famous FedAvg [3]. This method is closely related to a conventional federated learning especially using stochastic gradient descent (SGD) [16]. In S-KOFL, each node k optimizes the parameter of a kernel function (denoted by $\hat{\mathbf{w}}_{[k,t]}$)) via OGD using a local streaming data. Using 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 exchanged 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 overhead, 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 an efficient multi-kernel based method for KOFL framework. We emphasize that it poses a new challenging problem, which has not been caused in existing federated 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 number of local parameters to the server and vice versa, where P denotes the size of a kernel dictionary. Consequently, the downlink/uplink communication overhead is equal to $P \times M$, whereas that of S-KOFL is M. To attain a higher learning accuracy, vM-KOFL requires a sufficiently large number of kernels and hence, the communication overhead is indeed trouble. In addition, the existing overhead-reduction techniques (e.g., sparsification or quantization) [17], [18], [19], [20] can only reduce the term M as $M' \ll M$, whereas the term P is unchanged. Thus, vM-KOFL can achieve an improved learning accuracy over S-KOFL at the cost 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 above open problem. A novel *randomized* method (named eM-KOFL) is proposed. Enjoying the merit of multiple kernels probabilistically, eM-KOFL can yield the same performance as vM-KOFL. Moreover, the communication overhead of eM-KOFL is irrespective of the size of a kernel dictionary P and thus, to achieve a higher learning accuracy, a sufficiently large number of kernels can be exploited for free. Our major contributions are summarized as follows.

- We present a *delayed-Exp* strategy, from which the server constructs a sequence of probability mass functions (PMFs) $\tilde{\mathbf{q}}_t = (\tilde{q}_{[t,1]}, ..., \tilde{q}_{[t,P]})$, $t = 1, 2, ..., T$, 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 $\tilde{\mathbf{q}}_t$. The proposed strategy guarantees that the selected kernel converges to the best kernel in hindsight with high probability. As in 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 eM-KOFL yields the same asymptotic performance as vM-KOFL, namely, it achieves a sublinear regret bound $\mathcal{O}(\sqrt{T})$ 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 overheads (per node), S-KOFL and vM-KOFL respectively have M and $P \times M$ for both downlink and uplink. Since eM-KOFL can be viewed as S-KOFL potentially with different kernels, the communication overheads 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 in an efficient way. The resulting method is named pM-KOFL, which has the communication overheads $M+1 \approx M$ for both downlink and uplink.
- Via numerical tests on real datasets, we demonstrate the effectiveness of the proposed methods on online regression and time-series prediction tasks. Notably, pM-KOFL can yield the almost same performance as vM-KOFL and eM-KOFL. This gives the positive answer to the question, i.e., pM-KOFL can fully enjoy the advantage of multiple kernels while keeping the communication overhead of S-KOFL.

The remainder of this paper is organized as follows. In Section 2, we formally define the problem setting of KOFL 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 $\mathbf{x}$, $\mathbf{x}^\mathsf{T}$ denotes the transpose of $\mathbf{x}$ and $\|\mathbf{x}\|$ denote the ℓ_2-norm of $\mathbf{x}$. Also, $\mathbb{E}[\cdot]$ represents the expectation over an associated probability distribution. To simplify the notations, we let $[n] \triangleq \{1, ..., n\}$ for any positive integer n. Also, k, t, 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 $\hat{\mathbf{m}}_t$ (i.e., the parameter of a global function $f(\mathbf{x}; \hat{\mathbf{m}}_t)$) to the decentralized K nodes. Then, each node k receives the global model $\hat{\mathbf{m}}_t$ and an incoming data $(\mathbf{x}_{k,t}, y_{k,t})$, where $\mathbf{x}_{k,t} \in \mathcal{X} \subseteq \mathbb{R}^N$ and $y_{k,t} \in \mathcal{Y} \subseteq \mathbb{R}$ represent the feature and the label, respectively. Using them, each node k updates its local model, which is denoted as $\hat{\mathbf{m}}_{[k,t+1]} \in \mathbb{R}^M$ for $k \in [K]$. Every node k sends $\hat{\mathbf{m}}_{[k,t+1]}$ back to the server, from which the server constructs an updated global model as

$$\hat{\mathbf{m}}_{t+1} = h(\hat{\mathbf{m}}_{[1,t+1]}, ...\hat{\mathbf{m}}_{[K,t+1]}). \tag{3}$$

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 $f(\mathbf{x}; \hat{\mathbf{m}}_{t+1})$, from which each node

k can estimate the label of a newly incoming data $\mathbf{x}_{k,t+1}$ 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 $\mathcal{H} = \{f : f(\mathbf{x}) = \sum_t \alpha_t \kappa(\mathbf{x}, \mathbf{x}_t)\}$, where $\kappa(\mathbf{x}, \mathbf{x}_t) : \mathcal{X} \times \mathcal{X} \to \mathcal{Y}$ 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 $\hat{\mathbf{w}} \in \mathbb{R}^M$ as

$$f(\mathbf{x}; \hat{\mathbf{w}}) = \hat{\mathbf{w}}^\mathsf{T} \mathbf{z}(\mathbf{x}) \in \mathcal{H}, \tag{4}$$

where a feature mapping $\mathbf{z}(\mathbf{x})$, which relies on a kernel κ, is defined as

$$\mathbf{z}(\mathbf{x}) = \frac{1}{\sqrt{d}}[\sin \mathbf{v}_1^\mathsf{T}\mathbf{x}, \ldots, \sin \mathbf{v}_d^\mathsf{T}\mathbf{x}, \cos \mathbf{v}_1^\mathsf{T}\mathbf{x}, \ldots, \cos \mathbf{v}_d^\mathsf{T}\mathbf{x}]^\mathsf{T}, \tag{5}$$

and where $\{\mathbf{v}_i : i \in [d]\}$ 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 κ, 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 $\mathcal{K} = \{\kappa_1, \kappa_2, ..., \kappa_P\}$ (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

$$f(\mathbf{x}; \{\hat{q}_i, \hat{\mathbf{w}}_i\}) = \sum_{i=1}^{P} \hat{q}_i \hat{\mathbf{w}}_i^\mathsf{T} \mathbf{z}_i(\mathbf{x}), \tag{6}$$

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

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:

$$\text{regret}_T = \sum_{t=1}^{T}\sum_{k=1}^{T} \mathcal{L}(f(\mathbf{x}_{k,t}; \hat{\mathbf{m}}_t), y_{k,t}) - \min_{1 \le i \le P} \min_{\mathbf{w}_i} \sum_{t=1}^{T}\sum_{k=1}^{K} \mathcal{L}(\mathbf{w}_i^\mathsf{T} \mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}),$$

  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 dimensions of shared parameters).

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 $f(\mathbf{x}; \hat{\mathbf{w}}_1), ..., f(\mathbf{x}; \hat{\mathbf{w}}_T)$ defined in (4), such as

$$f(\mathbf{x}; \hat{\mathbf{w}}_t) = \hat{\mathbf{w}}_t^\mathsf{T} \mathbf{z}(\mathbf{x}) \in \mathcal{H}, \tag{7}$$

where $\mathbf{z}(\mathbf{x})$ is given in (5) according to a preselected *single* kernel κ. To optimize the above global models $\hat{\mathbf{w}}_t \in \mathbb{R}^M, t = 1, ..., T$ 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 $\hat{\mathbf{w}}_t$ form the server and an incoming data $(\mathbf{x}_{k,t}, y_{k,t})$. Using them, it updates the local model via OGD [15]:

$$\hat{\mathbf{w}}_{[k,t+1]} = \hat{\mathbf{w}}_t - \eta_\ell \nabla \mathcal{L}\left(\hat{\mathbf{w}}_t^\mathsf{T} \mathbf{z}(\mathbf{x}_{k,t}), y_{k,t}\right), \tag{8}$$

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

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

$$\hat{\mathbf{w}}_{t+1} = \frac{1}{K} \sum_{k=1}^{K} \hat{\mathbf{w}}_{[k,t+1]}. \tag{9}$$

Then, it distributes the updated global model $\hat{\mathbf{w}}_{t+1} \in \mathbb{R}^M$ 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

$$f(\mathbf{x}; \{\hat{\mathbf{w}}_{[t,i]}, \hat{q}_{[t,i]}\}) = \sum_{i=1}^{P} \hat{q}_{[t,i]} \hat{\mathbf{w}}_{[t,i]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}), \tag{10}$$

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

Local model update: Given the global model $\{\hat{\mathbf{w}}_{[t,i]} : i \in [P]\}$ and an incoming data $(\mathbf{x}_{k,t}, y_{k,t})$, each node k updates its local parameters $\{\hat{\mathbf{w}}_{[k,t+1,i]}\}$ via OGD:

$$\hat{\mathbf{w}}_{[k,t+1,i]} = \hat{\mathbf{w}}_{[t,i]} - \eta_\ell \nabla \mathcal{L}\left(\hat{\mathbf{w}}_{[t,i]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right), \quad (11)$$

for $\forall i \in [P]$, where $\eta_l > 0$ is a step size and the initial value is $\hat{\mathbf{w}}_{[1,i]} = \mathbf{0}$ for $\forall i \in [P]$. Also, it computes the reliabilities of the P kernels with respect to its own local data:

$$\hat{\ell}_{[k,t+1,i]} = \exp\left[-\eta_g \sum_{\tau=1}^{t} \mathcal{L}\left(\hat{\mathbf{w}}_{[\tau,i]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}_{k,\tau}), y_{k,\tau}\right)\right], \quad (12)$$

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

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

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

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

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

The above method to compute the combining weights is known as *Exp strategy* [25]. Finally, the server broadcasts the updated global model $\{\hat{\mathbf{w}}_{[t+1,i]}, \hat{q}_{[t+1,i]} : i \in [P]\}$ 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 method 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.

4 Proposed Methods

In this section, we propose a novel *randomized* method (named eM-KOFL), which can enjoy the advantage of multiple kernels while having a similar communication overhead as S-KOFL. It is emphasized that the uplink/downlink communication overhead of eM-KOFL is irrespective of P and thus, to achieve a higher learning accuracy, a sufficiently large number of kernels can be chosen without the burden of the communication overhead. The key idea of eM-KOFL is as follows. At every time t, one kernel out of P kernels is randomly selected according to a carefully designed probability mass function (PMF). Then, local functions in the chosen kernel are updated globally, whereas the

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 (η_l, η_g).
2: **Output:** A sequence of global functions: $t \in [T+1]$

$$f(\mathbf{x}; \{\hat{q}_{[t,i]}, \hat{\mathbf{w}}_{[t,i]} : i \in [P]\}), \ t = 1, ..., T.$$

3: **Initialization:** $\hat{\mathbf{w}}_{[1,i]} = \mathbf{0}$ for $\forall i \in [P]$. Each node has an identical random features $\mathbf{z}_i(\cdot)$ for $\forall i \in [P]$.
4: **Iteration:** $t = 1, ..., T$

◇ For every node $k \in [K]$

- Receive a streaming data $(\mathbf{x}_{k,t}, y_{k,t})$.
- Construct $\mathbf{z}_i(\mathbf{x}_{k,t}), \ \forall i \in [P]$ via (5).
- Update the parameters $\{\hat{\mathbf{w}}_{[k,t+1,i]} : i \in [P]\}$ via (11).
- Compute the reliabilities $\{\hat{\ell}_{[k,t+1,i]} : i \in [P]\}$ via (12).
- Send the updated local model $\{\hat{\mathbf{w}}_{[k,t+1,i]}, \hat{\ell}_{[k,t+1,i]} : i \in [P]\}$ to the server.

◇ At the server

- Update the global parameter $\mathbf{w}_{[t+1,i]}$ via (13).
- Obtain the reliabilities $\{\hat{q}_{[t+1,i]} : i \in [P]\}$ via (14).
- Send the updated global model $\{\mathbf{w}_{[t+1,i]}, \hat{q}_{[t+1,i]} : i \in [P]\}$ to the K nodes.

◊ S-KOFL follows the above procedures with a predetermined single kernel κ (i.e., $P = 1$).

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 (η_l, η_g).
2: **Output:** A sequence of common functions: $t \in [T+1]$

$$f(\mathbf{x}; \{\tilde{q}_{[t,i]}, \tilde{\mathbf{w}}_{[t,i]} : i \in [P]\}), \ t = 1, ..., T.$$

3: **Initialization:** $\tilde{\mathbf{w}}_{[1,i]} = \mathbf{0}$ for $\forall i \in [P]$. Each node has an identical random features $\mathbf{z}_i(\cdot)$ for $\forall i \in [P]$.
4: **Iteration:** $t = 1, ..., T$

◇ For every node $k \in [K]$

- Receive a streaming data $(\mathbf{x}_{k,t}, y_{k,t})$.
- Construct $\mathbf{z}_i(\mathbf{x}_{k,t}), \ \forall i \in [P]$ via (5).
- Update the local parameters $\{\tilde{\mathbf{g}}_{[k,t+1,i]} : i \in [P]\}$ via (17).
- Compute the reliabilities $\{\tilde{\ell}_{[k,t+1,i]} : i \in [P]\}$ via (12).
- Send the updated local model $\tilde{\mathbf{w}}_{[k,t+1]} = \tilde{\mathbf{g}}_{[k,t+1,\tilde{p}_{t+1}]}$ and $\{\tilde{\ell}_{[k,t+2,p]} : i \in [P]\}$ to the server.

◇ At the server

- Update the global parameter $\mathbf{w}_{t+1}$ via (20).
- Randomly choose $\tilde{p}_{t+2}$ according to the PMF in (21).
- Send the updated global model $\{\tilde{\mathbf{w}}_{t+1}, \tilde{p}_{t+2}\}$ to the K nodes.

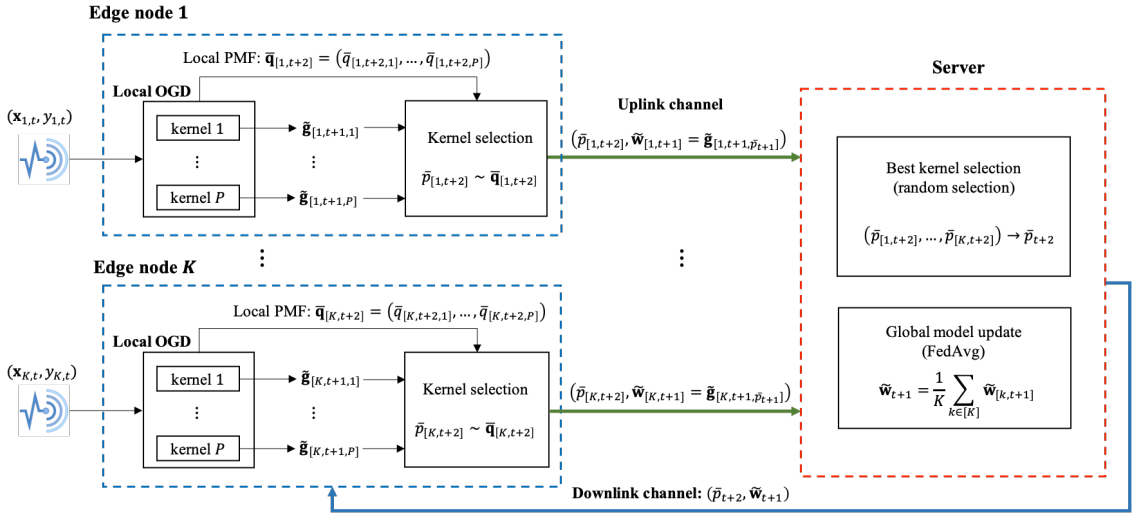


Fig. 1. Description of the proposed pM-KOFL.

other functions are updated locally only. This enables that the communication overhead does not grow with P. Furthermore, it is proved that our PMF, which is constructed by the proposed *delayed-Exp* strategy, can guarantee that as t proceeds, the best kernel in hindsight is selected with high probability. This will be analytically verified in Section 5. The communication overheads of eM-KOFL are equal to $M+1$ and $M+P \leq 2M$ for downlink and uplink respectively. We further reduce the uplink communication overhead of eM-KOFL from $M+P$ to $M+1$, by mimicking the delayed-Exp strategy in an efficient way. This variant of eM-KOFL is named pM-KOFL. The communication overheads of S-KOFL, vM-KOFL, eM-KOFL, and pM-KOFL are summarized in Table I.

4.1 eM-KOFL

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

$$f(\mathbf{x}; \{\tilde{\mathbf{w}}_t, \tilde{p}_t\}) = \tilde{\mathbf{w}}_t^{\mathsf{T}} \mathbf{z}_{\tilde{p}_t}(\mathbf{x}) \in \mathcal{H}_{\tilde{p}_t}, \tag{15}$$

where the global model is defined as $\hat{\mathbf{m}} = \{\tilde{p}_t, \tilde{\mathbf{w}}_t\}$. We note that $\{\tilde{p}_t, \tilde{\mathbf{w}}_t\}$ 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 $\{\tilde{\mathbf{g}}_{[k,t,i]} : i \in [P]\}$, which is not shared with the server. Also, at time t, it receives the global model $(\tilde{p}_{t+1}, \tilde{\mathbf{w}}_t)$ from the server. Leveraging $\tilde{p}_t$ (received at time $t-1$) and $\tilde{\mathbf{w}}_t$, each node k updates the parameters of the P kernel functions as

$$\tilde{\mathbf{w}}_{[k,t,i]} = \begin{cases} \tilde{\mathbf{g}}_{[k,t,i]}, & \text{if } i \neq \tilde{p}_t \\ \tilde{\mathbf{w}}_t, & \text{if } i = \tilde{p}_t, \end{cases} \tag{16}$$

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

$$\tilde{\mathbf{g}}_{[k,t+1,i]} = \tilde{\mathbf{w}}_{[k,t,i]} - \eta_\ell \nabla \mathcal{L}\left(\tilde{\mathbf{w}}_{[k,t,i]}^{\mathsf{T}} \mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right), \tag{17}$$

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

$$\tilde{\mathbf{w}}_{[k,t+1]} = \tilde{\mathbf{g}}_{[k,t+1,\tilde{p}_{t+1}]}. \tag{18}$$

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

$$\tilde{\ell}_{[k,t+2,i]} = \exp\left[-\eta_g \sum_{\tau=1}^{t} \mathcal{L}\left(\tilde{\mathbf{w}}_{[k,\tau,i]}^{\mathsf{T}} \mathbf{z}_i(\mathbf{x}_{k,\tau}), y_{k,\tau}\right)\right], \tag{19}$$

where $\tilde{\ell}_{[k,1,i]} = \tilde{\ell}_{[k,2,i]} = 1$ for $\forall i \in [P]$. Obviously, $\hat{\ell}_{[k,t+2,i]}$ can be different from $\tilde{\ell}_{[k,t+2,i]}$ in (19). Finally, every node k sends the updated local model $\tilde{\mathbf{w}}_{[k,t+1]}$ and $\{\tilde{\ell}_{[k,t+2,i]} : i \in [P]\}$ 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 $\{\tilde{\mathbf{w}}_{[k,t+1]} : k \in [K]\}$ and $\{\tilde{\ell}_{[k,t+2,i]} : i \in [P], k \in [K]\}$ from the K nodes. As in S-KOFL, it updates the global parameter via FedAvg:

$$\tilde{\mathbf{w}}_{t+1} = \frac{1}{K} \sum_{k=1}^{K} \tilde{\mathbf{w}}_{[k,t+1]}. \tag{20}$$

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:

$$\tilde{q}_{[t+2,i]} = \frac{\prod_{k=1}^{K} \tilde{\ell}_{[k,t+2,i]}}{\sum_{i=1}^{P} \prod_{k=1}^{K} \tilde{\ell}_{[k,t+2,i]}}, \quad \forall i \in [P]. \tag{21}$$

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., $\hat{q}_{[t+2,i]}$ 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 $\{\tilde{\mathbf{w}}_{t+1}, \tilde{p}_{t+2}\}$ 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.

TABLE 1
Comparisons of communication overhead for various KOFL methods

Methods	Number of transmissions	
	Uplink	Downlink
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	$M+1 \approx M$	$M+1 \approx M$

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

$$\bar{q}_{[k,t+2,i]} = \frac{(\tilde{\ell}_{[k,t+2,i]})^K}{\sum_{i=1}^{P}(\tilde{\ell}_{[k,t+2,i]})^K}, \quad \forall i \in [P], \tag{22}$$

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

$$\bar{q}_{[t+2,i]} = \frac{1}{K}\sum_{k=1}^{K} \bar{q}_{[k,t+2,i]}, \tag{23}$$

for $\forall i \in [P]$. Then, $\{\bar{q}_{[t+2,i]}\}$ can be considered as the proxy of the true PMF $\{\hat{q}_{[t+2,i]}\}$ in (21). Definitely pM-KOFL can approach the performance of eM-KOFL, provided that $\bar{q}_{[t+2,i]}$ is sufficiently close to $\tilde{q}_{[t+2,i]}$. 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 achieves a sublinear regret bound, i.e., $\text{regret}_T \leq \mathcal{O}(\sqrt{T})$. 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 κ_i, let $f(\mathbf{x}; \mathbf{w}_{[\star,i]}) = \mathbf{w}_{[\star,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x})$ denote the best RF-based function belonging to RKHS $\mathcal{H}_i$, i.e.,

$$\mathbf{w}_{[\star,i]} \triangleq \arg\min_{\mathbf{w}} \sum_{t=1}^{T}\sum_{k=1}^{K} \mathcal{L}\left(\mathbf{w}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right). \tag{24}$$

For our analysis, we make the following assumptions, which are usually assumed for the analysis of online convex optimization and online learning [11], [12], [15], [25]:

Assumption 1. For any fixed $\mathbf{z}_i(\mathbf{x}_t)$ and y_t, the loss function $\mathcal{L}(\mathbf{w}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_t), y_t)$ is convex with respect to $\mathbf{w}$, and is bounded as $\mathcal{L}(\mathbf{w}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_t), y_t) \in [0,1]$.

Assumption 2. For any fixed kernel κ_i, the optimal parameter $\mathbf{w}_{[\star,i]}$ is assumed to be bounded, i.e., $\|\mathbf{w}_{[\star,i]}\|^2 \leq C$.

Assumption 3. The loss function is L-Lipschitz continuous, i.e., $\|\nabla\mathcal{L}(\mathbf{w}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_t), y_t)\|^2 \leq L$ for any $i \in \forall[P]$.

Under Assumption 1 - Assumption 3, the main results of this section are derived as follows.

Theorem 1. Given any set of P kernels $\{\kappa_i,\ i \in [P]\}$, the *vanilla* method (vM-KOFL) in Algorithm 1 achieves the following regret bound:

$$\begin{aligned}\text{regret}_T = &\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}\left(\sum_{i=1}^{P}\hat{q}_{[t,i]}\hat{\mathbf{w}}_{[t,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right)\\ &- \min_{1\leq i\leq P}\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}\left(\mathbf{w}_{[\star,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right)\\ \leq &\frac{KC}{2\eta_\ell} + \frac{\eta_\ell LKT}{2} + \frac{\log P}{\eta_g} + \frac{\eta_g K^2 T}{2}.\end{aligned}$$

Proof: The proof is provided in Section 5.1. □

Theorem 2. Given any set of P kernels $\{\kappa_i,\ i \in [P]\}$, the proposed eM-KOFL in Algorithm 2 achieves the following regret bound with probability $1-\delta$:

$$\begin{aligned}\text{regret}_T = &\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}\left(\tilde{\mathbf{w}}_t^{\mathsf{T}}\mathbf{z}_{\bar{p}_t}(\mathbf{x}_{k,t}), y_{k,t}\right)\\ &- \min_{1\leq i\leq P}\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}\left(\mathbf{w}_{[\star,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right)\\ \leq &\frac{KC}{2\eta_\ell} + \frac{\eta_\ell KLT}{2} + \frac{2\log P}{\eta_g} + \frac{\eta_g K^2 T}{2} + K\sqrt{\frac{\log\delta^{-1}}{2}}.\end{aligned}$$

Note that $\{\tilde{p}_t, \tilde{\mathbf{w}}_t\}, t = 1, ..., T$ are random sequences.

Proof: The proof is provided in Section 5.2. □

Setting $\eta_\ell = \eta_g = \mathcal{O}(1/\sqrt{T})$ in Theorem 1, the vanilla vM-KOFL achieves a sublinear regret bound $\mathcal{O}(\sqrt{T})$. Likewise, with $\eta_\ell = \eta_g = \mathcal{O}(1/\sqrt{T})$ in Theorem 2, the proposed eM-KOFL achieves the same sublinear regret bound with high probability. Thus, in the asymptotic case, eM-KOFL can yield the same performance as vM-KOFL while having the $\frac{1}{P}$ downlink/uplink communication overhead.

Remark 2. Definitely pM-KOFL can approach the performance of eM-KOFL, provided that $\bar{q}_{[t,i]}$ in (23) is sufficiently close to the true PMF $\hat{q}_{[t,i]}$ in (21). In an asymptotic analysis, it is easily proved that pM-KOFL can achieve a sublinear regret bound as in eM-KOFL, if the following condition holds:

$$\sum_{t=1}^{T}\sum_{i=1}^{P}|\bar{q}_{[t,i]} - \tilde{q}_{[t,i]}| \leq \mathcal{O}(\sqrt{T}). \tag{25}$$

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 $\mathbb{P}(\hat{p}_t = \bar{p}_t)$ (which can clearly capture the left-hand side of (25)).

5.1 Proof of Theorem 1

We provide the proof of Theorem 1. Following the notations in Section 3.2, let $\{\hat{\mathbf{w}}_{[t,i]} : i \in [P]\}$ and $\{\hat{q}_{[t,i]} : i \in [P]\}$ 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 κ_i and any step size $\eta_l > 0$, S-KOFL in Section 3.1 achieves the following regret bound:

$$\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}\left(\hat{\mathbf{w}}_{[t,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right) - \sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}\left(\hat{\mathbf{w}}_{[\star,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right) \leq \frac{KC}{2\eta_\ell} + \frac{\eta_\ell LKT}{2}.$$

Proof: Fix a kernel κ_i for any $i \in [P]$. From OGD update in (8) and FedAvg in (9), the global model is updated as

$$\hat{\mathbf{w}}_{[t+1,i]} = \hat{\mathbf{w}}_{[t,i]} - \frac{\eta_\ell}{K}\sum_{k=1}^{K}\nabla_{[k,t,i]}, \tag{26}$$

where for ease of exposition, we let

$$\nabla_{[k,t,i]} \triangleq \nabla\mathcal{L}(\hat{\mathbf{w}}_{[t,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}). \tag{27}$$

From (26), we have:

$$\begin{aligned}\|\hat{\mathbf{w}}_{[t+1,i]} - \mathbf{w}_{[\star,i]}\|^2 &= \|\hat{\mathbf{w}}_{[t,i]} - \mathbf{w}_{[\star,i]}\|^2 + \eta_\ell^2\left\|\frac{1}{K}\sum_{k=1}^{K}\nabla_{[k,t,i]}\right\|^2 \\ &\quad - 2\frac{\eta_\ell}{K}\sum_{k=1}^{K}\nabla_{[k,t,i]}^{\mathsf{T}}(\hat{\mathbf{w}}_{[t,i]} - \mathbf{w}_{[\star,i]}).\end{aligned} \tag{28}$$

Using the convexity of a loss function, we can get:

$$\sum_{k=1}^{K}\mathcal{L}\left(\hat{\mathbf{w}}_{[t,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right) - \mathcal{L}\left(\hat{\mathbf{w}}_{[\star,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right) \leq \sum_{k=1}^{K}\nabla_{[k,t,i]}^{\mathsf{T}}(\hat{\mathbf{w}}_{[t,i]} - \mathbf{w}_{[\star,i]}). \tag{29}$$

From (28) and (29), we derive the following upper bound:

$$\begin{aligned}&\sum_{k=1}^{K}\mathcal{L}\left(\hat{\mathbf{w}}_{[t,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right) - \mathcal{L}\left(\mathbf{w}_{[\star,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right) \\ &\leq K\frac{\|\hat{\mathbf{w}}_{[t,i]} - \mathbf{w}_{[\star,i]}\|^2 - \|\hat{\mathbf{w}}_{[t+1,i]} - \mathbf{w}_{[\star,i]}\|^2}{2\eta_\ell} + \frac{\eta_\ell}{2K}\left\|\sum_{k=1}^{K}\nabla_{[k,t,i]}\right\|^2 \\ &\overset{(a)}{\leq} K\frac{\|\hat{\mathbf{w}}_{[t,i]} - \mathbf{w}_{[\star,i]}\|^2 - \|\hat{\mathbf{w}}_{[t+1,i]} - \mathbf{w}_{[\star,i]}\|^2}{2\eta_\ell} + \frac{\eta_\ell}{2K}\left[\sum_{k=1}^{K}\left\|\nabla_{[k,t,i]}\right\|\right]^2,\end{aligned} \tag{30}$$

where (a) is from the Cauchy-Schwartz inequality. Using the telescoping sum over $t = 1, 2, .., T$, we can get:

$$\begin{aligned}&\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}\left(\hat{\mathbf{w}}_{[t,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right) - \mathcal{L}\left(\hat{\mathbf{w}}_{[\star,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right) \\ &\leq K\frac{\|\hat{\mathbf{w}}_{[1,i]} - \mathbf{w}_{[\star,i]}\|^2 - \|\hat{\mathbf{w}}_{[T+1,i]} - \mathbf{w}_{[\star,i]}\|^2}{2\eta_\ell} + \frac{\eta_\ell}{2K}\sum_{t=1}^{T}\left[\sum_{k=1}^{K}\left\|\nabla_{[k,t,i]}\right\|\right]^2 \\ &\overset{(a)}{\leq} \frac{KC}{2\eta_\ell} + \frac{\eta_\ell LKT}{2},\end{aligned} \tag{31}$$

where (a) follows $\hat{\mathbf{w}}_{[1,i]} = \mathbf{0}$, Assumption 2, and Assumption 3. This completes the proof. □

Lemma 2. For any learning rate $\eta_g > 0$, *Exp strategy* guarantees the following regret bound:

$$\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}\left(\sum_{i=1}^{P}\hat{q}_{[t,i]}\hat{\mathbf{w}}_{[t,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right) - \min_{1\leq i\leq P}\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}\left(\mathbf{w}_{[t,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right) \leq \frac{\log P}{\eta_g} + \frac{\eta_g K^2 T}{2}.$$

Proof: The proof is completed from [12, Lemma 2]. □

Note that Lemma 1 holds for any kernel κ_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 $\{\tilde{\mathbf{w}}_{[k,t,i]} : i \in [P]\}$ denote the local parameters at the node k. Also, let $\tilde{\mathbf{w}}_t$ and $\tilde{p}_t$ denote the global parameters. Using them, we provide the the key lemmas for the main proof.

Lemma 3. For any kernel $i \in [P]$ and step size $\eta_l > 0$, the local kernel functions of eM-KOFL can achieve the following regret bound:

$$\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}\left(\tilde{\mathbf{w}}_{[k,t,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right) - \sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}\left(\mathbf{w}_{[\star,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right) \leq \frac{KC}{2\eta_l} + \frac{\eta_l KLT}{2}.$$

Proof: Given the global parameters $\tilde{p}_t \in [P]$ and $\tilde{\mathbf{w}}_t$, and from (17), the local parameters are updated via OGD as

$$\tilde{\mathbf{g}}_{[k,t+1,i]} = \tilde{\mathbf{w}}_{[k,t,i]} - \eta_\ell\nabla\mathcal{L}(\tilde{\mathbf{w}}_{[k,t,i]}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}), \tag{32}$$

for $\forall i \in [P]$, where

$$\tilde{\mathbf{w}}_{[k,t,i]} = \begin{cases}\tilde{\mathbf{g}}_{[k,t,i]} & \text{if } i \neq \tilde{p}_t \\ \tilde{\mathbf{w}}_t = \frac{1}{K}\sum_{k=1}^{K}\tilde{\mathbf{g}}_{[k,t,i]} & \text{if } i = \tilde{p}_t.\end{cases} \tag{33}$$

Then, from (32), we have that for any $k \in [K]$:

$$\begin{aligned}
&\|\tilde{\mathbf{g}}_{[k,t+1,i]} - \mathbf{w}_{[\star,i]}\|^2 \\
&= \|\tilde{\mathbf{w}}_{[k,t,i]} - \eta_\ell \nabla \mathcal{L}(\tilde{\mathbf{w}}_{[k,t,i]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}) - \mathbf{w}_{[\star,i]}\|^2 \\
&= \|\tilde{\mathbf{w}}_{[k,t,i]} - \mathbf{w}_{[\star,i]}\|^2 + \eta_\ell^2 \|\nabla \mathcal{L}(\tilde{\mathbf{w}}_{[k,t,i]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t})\|^2 \\
&- 2\eta_\ell \nabla \mathcal{L}(\tilde{\mathbf{w}}_{[k,t,i]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t})^\mathsf{T} (\tilde{\mathbf{w}}_{[k,t,i]} - \mathbf{w}_{[\star,i]}). \qquad (34)
\end{aligned}$$

Foe ease of exposition, throughout the proof, we let

$$\nabla_{[k,t,i]} \triangleq \nabla \mathcal{L}(\tilde{\mathbf{w}}_{[k,t,i]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}). \qquad (35)$$

According to the two cases in (33), we can get:

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

$$\begin{aligned}
\|\tilde{\mathbf{g}}_{[k,t+1,i]} - \mathbf{w}_{[\star,i]}\|^2 &= \|\tilde{\mathbf{g}}_{[k,t,i]} - \mathbf{w}_{[\star,i]}\|^2 \\
&+ \eta_\ell^2 \|\nabla_{[k,t,i]}\|^2 - 2\eta_\ell \nabla_{[k,t,i]}^\mathsf{T} (\tilde{\mathbf{w}}_{[k,t,i]} - \mathbf{w}_{[\star,i]}). \qquad (36)
\end{aligned}$$

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

$$\begin{aligned}
&\sum_{k\in[K]} \left\|\tilde{\mathbf{g}}_{[k,t+1,i]} - \mathbf{w}_{[\star,i]}\right\|^2 = \sum_{k=1}^K \left\|\tilde{\mathbf{w}}_{[k,t,i]} - \eta_\ell \nabla_{[k,t,i]} - \mathbf{w}_{[\star,i]}\right\|^2 \\
&= \sum_{k=1}^K \left\|\tilde{\mathbf{w}}_{[k,t,i]} - \mathbf{w}_{[\star,i]}\right\|^2 + \eta_l^2 \sum_{k=1}^K \left\|\nabla_{[k,t,i]}\right\|^2 \\
&\quad - 2\eta_\ell \nabla_{[k,t,i]}^\mathsf{T} (\tilde{\mathbf{w}}_{[k,t,i]} - \mathbf{w}_{[\star,i]}) \\
&\overset{(a)}{\leq} \sum_{k=1}^K \left\|\tilde{\mathbf{g}}_{[k,t,i]} - \mathbf{w}_{[\star,i]}\right\|^2 + \eta_l^2 \sum_{k=1}^K \left\|\nabla_{[k,t,i]}\right\|^2 \\
&\quad - 2\eta_\ell \nabla_{[k,t,i]}^\mathsf{T} (\tilde{\mathbf{w}}_{[k,t,i]} - \mathbf{w}_{[\star,i]}), \qquad (37)
\end{aligned}$$

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

$$\begin{aligned}
\|\tilde{\mathbf{w}}_{[k,t,i]} - \mathbf{w}_{[\star,i]}\|^2 &= \left\|\frac{1}{K}\sum_{k=1}^K \tilde{\mathbf{g}}_{[k,t,i]} - \mathbf{w}_{[\star,i]}\right\|^2 \\
&\overset{(a)}{\leq} \frac{1}{K^2}\left[\sum_{k=1}^K \|\tilde{\mathbf{g}}_{[k,t,i]} - \mathbf{w}_{[\star,i]}\|\right]^2 \\
&\overset{(b)}{\leq} \frac{1}{K}\sum_{k=1}^K \|\tilde{\mathbf{g}}_{[k,t,i]} - \mathbf{w}_{[\star,i]}\|^2,
\end{aligned}$$

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

$$\begin{aligned}
&\mathcal{L}\left(\tilde{\mathbf{w}}_{[k,t,i]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}_{k,t}), y_t\right) - \mathcal{L}\left(\mathbf{w}_{[\star,p]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}_{k,t}), y_t\right) \\
&\leq \nabla_{[k,t,i]}^\mathsf{T} (\tilde{\mathbf{w}}_{[k,t,i]} - \mathbf{w}_{[\star,i]}). \qquad (38)
\end{aligned}$$

Plugging (38) into (36) and (37) separately, and combining the two cases, we can get

$$\begin{aligned}
&\sum_{k=1}^K \mathcal{L}\left(\tilde{\mathbf{w}}_{[k,t,i]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}_{k,t}), y_t\right) - \mathcal{L}\left(\mathbf{w}_{[\star,i]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}_{k,t}), y_t\right) \\
&\leq \sum_{k=1}^K \frac{\|\tilde{\mathbf{g}}_{[k,t,i]} - \mathbf{w}_{[\star,i]}\|^2 - \|\tilde{\mathbf{g}}_{[k,t+1,i]} - \mathbf{w}_{[\star,i]}\|^2}{2\eta_\ell} \\
&+ \frac{\eta_\ell}{2}\sum_{k=1}^K \|\nabla_{[k,t,i]}\|^2. \qquad (39)
\end{aligned}$$

Summing (39) over $t = 1, ..., T$, we obtain that for any fixed $i \in [P]$,

$$\begin{aligned}
&\sum_{t=1}^T \sum_{k=1}^K \mathcal{L}\left(\tilde{\mathbf{w}}_{[k,t,i]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}_{k,t}), y_t\right) - \mathcal{L}\left(\mathbf{w}_{[\star,i]}^\mathsf{T} \mathbf{z}_p(\mathbf{x}_{k,t}), y_t\right) \\
&\overset{(a)}{\leq} \sum_{k=1}^K \frac{\|\tilde{\mathbf{g}}_{[k,1,i]} - \mathbf{w}_{[\star,i]}\|^2}{2\eta_\ell} + \frac{\eta_\ell}{2}\sum_{t=1}^T \sum_{k=1}^K \|\nabla_{[k,t,i]}\|^2 \\
&\overset{(b)}{\leq} \frac{KC}{2\eta_\ell} + \frac{\eta_\ell KLT}{2}, \qquad (40)
\end{aligned}$$

where (a) is due to the telescoping sum and (b) is from $\tilde{\mathbf{g}}_{[k,1,i]} = \mathbf{0}$, Assumption 2 and Assumption 3. This completes the proof. □

Lemma 4. For any learning rate $\eta_g > 0$, the proposed *delayed-Exp strategy* guarantees the following regret bound:

$$\begin{aligned}
&\sum_{t=1}^T \sum_{k=1}^K \sum_{i=1}^P \tilde{q}_{[t,i]} \mathcal{L}\left(\tilde{\mathbf{w}}_{[k,t,i]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right) \\
&\qquad - \min_{1\leq i\leq P} \sum_{t=1}^T \sum_{k=1}^K \mathcal{L}\left(\tilde{\mathbf{w}}_{[k,t,i]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t}\right) \\
&\leq \frac{2\log P}{\eta_g} + \frac{\eta_g K^2 T}{2}.
\end{aligned}$$

Proof: For ease of exposition, we define:

$$\tilde{f}_{[k,t,i]}(\mathbf{x}) \triangleq \tilde{\mathbf{w}}_{[k,t,i]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}). \qquad (41)$$

The proof will be completed using the upper and lower bounds on ζ, which is defined as

$$\zeta \triangleq \sum_{t=1}^T \log\left[\sum_{i=1}^P \tilde{q}_{[t,i]} \exp\left(-\eta_g \sum_{k=1}^K \sum_{\tau=t-1}^t \mathcal{L}(\tilde{f}_{[k,\tau,i]}(\mathbf{x}_{k,\tau}), y_{k,\tau})\right)\right].$$

We first derive the upper bound on ζ:

$$\begin{aligned}
\zeta &\overset{(a)}{=} \sum_{t=1}^T \log \mathbb{E}\left[\exp\left(-\eta_g \sum_{k=1}^K \sum_{\tau=t-1}^t \mathcal{L}(\tilde{f}_{[k,\tau,I_t]}(\mathbf{x}_{k,\tau}), y_{k,\tau})\right)\right] \\
&\overset{(b)}{\leq} -\eta_g \sum_{t=1}^T \sum_{k=1}^K \mathbb{E}\left[\mathcal{L}(\tilde{f}_{[k,t,I_t]}(\mathbf{x}_{k,t}), y_{k,t})\right] \\
&- \eta_g \sum_{t=1}^T \sum_{k=1}^K \mathbb{E}\left[\mathcal{L}(\tilde{f}_{[k,t-1,I_t]}(\mathbf{x}_{k,t-1}), y_{k,t-1})\right] + \frac{\eta_g^2 K^2 T}{2}, \qquad (42)
\end{aligned}$$

where the expectation in (a) over the random variable $I_t \sim (\hat{q}_{t,1}, ..., \hat{q}_{t,P})$ and (b) follows the Hoeffding's inequality with the bounded random variable.

We next derive the lower bound on ζ. First, we define:

$$\bar{\ell}_{[k,t,i]} = \exp\left[-\eta_g \sum_{\tau=1}^{t-1} \mathcal{L}\left(\tilde{\mathbf{w}}_{[k,\tau,i]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}_{k,\tau}), y_{k,\tau}\right)\right], \qquad (43)$$

Also, recall that

$$\tilde{\ell}_{[k,t,i]} = \exp\left[-\eta_g \sum_{\tau=1}^{t-2} \mathcal{L}\left(\tilde{\mathbf{w}}_{[k,\tau,i]}^\mathsf{T} \mathbf{z}_i(\mathbf{x}_{k,\tau}), y_{k,\tau}\right)\right]. \qquad (44)$$

Obviously, we have:

$$\bar{\ell}_{[k,t,i]} = \tilde{\ell}_{[k,t+1,i]} \text{ and } \bar{\ell}_{[k,t,i]} \leq \tilde{\ell}_{[k,t,i]}. \qquad (45)$$

Then, using the above definitions, we have:

$$\begin{aligned}
\zeta &= \sum_{t=1}^{T} \log \left[\frac{\sum_{i=1}^{P} \bar{\ell}_{[k,t+1,i]}}{\sum_{i=1}^{P} \tilde{\ell}_{[k,t,i]}} \right] \\
&= \sum_{t=1}^{T} \log \left[\frac{\sum_{i=1}^{P} \bar{\ell}_{[k,t+1,i]}}{\sum_{i=1}^{P} \bar{\ell}_{[k,t,i]}} \right] + \sum_{t=1}^{T} \log \left[\frac{\sum_{i=1}^{P} \bar{\ell}_{[k,t,i]}}{\sum_{i=1}^{P} \tilde{\ell}_{[k,t,i]}} \right] \\
&\overset{(a)}{=} \sum_{t=1}^{T} \log \left[\frac{\sum_{i=1}^{P} \bar{\ell}_{[k,t+1,i]}}{\sum_{i=1}^{P} \bar{\ell}_{[k,t,i]}} \right] + \sum_{t=1}^{T} \log \left[\frac{\sum_{i=1}^{P} \tilde{\ell}_{[k,t+1,i]}}{\sum_{i=1}^{P} \tilde{\ell}_{[k,t,i]}} \right] \\
&\overset{(b)}{=} \log \left[\sum_{i=1}^{P} \bar{\ell}_{[k,T+1,i]} \right] - \log \left[\sum_{i=1}^{P} \bar{\ell}_{[k,1,i]} \right] \\
&+ \log \left[\sum_{i=1}^{P} \tilde{\ell}_{[k,T+1,i]} \right] - \log \left[\sum_{i=1}^{P} \tilde{\ell}_{[k,1,i]} \right] \\
&\overset{(c)}{\geq} 2 \log \left[\sum_{i=1}^{P} \bar{\ell}_{[k,T+1,i]} \right] - 2 \log P \\
&\overset{(d)}{\geq} 2 \log \left[\max_{1 \leq p \leq P} \bar{\ell}_{[k,T+1,p]} \right] - 2 \log P \\
&= -2\eta_g \min_{1 \leq i \leq P} \sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L} \left(\tilde{\mathbf{w}}_{[k,t,i]}^{\mathsf{T}} \mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t} \right) - 2 \log P
\end{aligned} \tag{46}$$

where (a) follows the relation in (45), (b) is from the telescoping sum, (c) is due to the fact that $\sum_{i=1}^{P} \tilde{\ell}_{[k,T+1,i]} \geq \sum_{i=1}^{P} \bar{\ell}_{[k,T+1,i]}$, and (d) is from the fact that the accumulated loss is non-negative. From the lower and upper bounds, we can get:

$$\begin{aligned}
&- 2\eta_g \min_{1 \leq i \leq P} \sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L} \left(\tilde{\mathbf{w}}_{[k,t,i]}^{\mathsf{T}} \mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t} \right) - 2 \log P \\
&\leq -\eta_g \sum_{t=1}^{T} \sum_{k=1}^{K} \mathbb{E} \left[\mathcal{L}(\tilde{f}_{[k,t,I_t]}(\mathbf{x}_{k,t}), y_{k,t}) \right] \\
&- \eta_g \sum_{t=1}^{T} \sum_{k=1}^{K} \mathbb{E} \left[\mathcal{L}(\tilde{f}_{[k,t-1,I_t]}(\mathbf{x}_{k,t-1}), y_{k,t-1}) \right] + \frac{\eta_g^2 K^2 T}{2}.
\end{aligned}$$

Rearranging the above term, we finally obtain:

$$\begin{aligned}
&\sum_{t=1}^{T} \sum_{k=1}^{K} \mathbb{E} \left[\mathcal{L}(\tilde{f}_{[k,t,I_t]}(\mathbf{x}_{k,t}), y_{k,t}) \right] \\
&- \min_{1 \leq i \leq P} \sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L} \left(\tilde{\mathbf{w}}_{[k,t,i]}^{\mathsf{T}} \mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t} \right) \\
&\leq \frac{2 \log P}{\eta_g} + \frac{\eta_g K^2 T}{2},
\end{aligned}$$

where we used the fact that

$$\begin{aligned}
&\sum_{t=1}^{T} \sum_{k=1}^{K} \mathbb{E} \left[\mathcal{L}(\tilde{f}_{[k,t-1,I_t]}(\mathbf{x}_{k,t-1}), y_{k,t-1}) \right] \\
&\geq \min_{1 \leq i \leq P} \sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L} \left(\tilde{\mathbf{w}}_{[k,t,i]}^{\mathsf{T}} \mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t} \right).
\end{aligned}$$

This completes the proof. □

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 $1 - \delta$:

$$\begin{aligned}
&\sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L} \Big(\tilde{\mathbf{w}}_t^{\mathsf{T}} \mathbf{z}_{\tilde{p}_t}(\mathbf{x}_{k,t}), y_{k,t} \Big) \\
&\qquad - \sum_{t=1}^{T} \sum_{k=1}^{K} \sum_{i=1}^{P} \tilde{q}_{[t,i]} \mathcal{L} \left(\tilde{\mathbf{w}}_{[k,t,i]}^{\mathsf{T}} \mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t} \right) \\
&\leq K \sqrt{\frac{\log(\delta^{-1})}{2} T}.
\end{aligned}$$

Proof: We first define $\bar{\mathbf{w}}_{[k,t,i]}$ as

$$\bar{\mathbf{w}}_{[k,t,i]} = \frac{1}{K} \sum_{k=1}^{K} \tilde{\mathbf{g}}_{[k,t,i]}, \forall i \in [P]. \tag{47}$$

Note that $\bar{\mathbf{w}}_{[k,t,i]} = \hat{\mathbf{w}}_{[k,t,i]}$ only when $i = \hat{p}_t$. Then, we define a random variable X_t:

$$\begin{aligned}
X_{k,t} &= \mathcal{L} \left(\tilde{\mathbf{w}}_t^{\mathsf{T}} \mathbf{z}_{\tilde{p}_t}(\mathbf{x}_{k,t}), y_{k,t} \right) \\
&\qquad - \sum_{i=1}^{P} \tilde{q}_{[t,i]} \mathcal{L} \left(\bar{\mathbf{w}}_{[k,t,i]}^{\mathsf{T}} \mathbf{z}_p(\mathbf{x}_{k,t}), y_{k,t} \right).
\end{aligned} \tag{48}$$

Note that $\tilde{q}_{[t,i]}$ is obtained as a consequence of random variables $\tilde{p}_1, ..., \tilde{p}_{t-1}$. Also, $\tilde{p}_t$ is chosen according to the PMF $(\tilde{q}_{[t,1]}, ..., \tilde{q}_{[t,P]})$. Let $\mathcal{F}_t = \sigma(\tilde{p}_1, \tilde{p}_2, ..., \tilde{p}_t)$ be the smallest sigma algebra such that $(\tilde{p}_1, \tilde{p}_2, ..., \tilde{p}_t)$ is measurable. Then, $\{\mathcal{F}_t : t = 1, ..., T\}$ is filtration and $X_{k,t}$ is $\mathcal{F}_t$-measurable. Note that the condition on $\mathcal{F}_{t-1}$, $\tilde{q}_{[t,i]}$ is fixed and $\tilde{p}_t$ and $\tilde{\mathbf{w}}_t$ are random variables. Using this fact, we have:

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

because

$$\begin{aligned}
&\mathbb{E} \left[\mathcal{L} \left(\tilde{\mathbf{w}}_t^{\mathsf{T}} \mathbf{z}_{\tilde{p}_t}(\mathbf{x}_{k,t}), y_{k,t} \right) \Big| \mathcal{F}_{t-1} \right] \\
&\qquad = \sum_{i=1}^{P} \tilde{q}_{[t,i]} \mathcal{L} \left(\bar{\mathbf{w}}_{[k,t,i]}^{\mathsf{T}} \mathbf{z}_i(\mathbf{x}_{k,t}), y_{k,t} \right).
\end{aligned}$$

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

$$B_t \triangleq - \sum_{i=1}^{P} \tilde{q}_{[t,i]} \mathcal{L}(\bar{\mathbf{w}}_{[k,t,i]}^{\mathsf{T}} \mathbf{z}_p(\mathbf{x}_{k,t}), y_{k,t}). \tag{49}$$

From Azuma-Hoeffding's inequality, the following bound holds for some $\delta > 0$ with at least probability $1 - \delta$:

$$\sum_{t=1}^{T} X_{k,t} \leq \sqrt{\frac{\log \delta^{-1}}{2} T}. \tag{50}$$

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

$$\sum_{t=1}^{T} \sum_{k=1}^{K} X_{k,t} \leq K \sqrt{\frac{\log \delta^{-1}}{2} T}. \tag{51}$$

Also, we have

$$\begin{aligned}
&\sum_{k=1}^{K}\sum_{i=1}^{P}\tilde{q}_{[t,i]}\mathcal{L}(\bar{\mathbf{w}}_{[k,t,i]}^{\mathsf{T}}\mathbf{z}_p(\mathbf{x}_{k,t}),y_{k,t})\\
&\overset{(a)}{=}\sum_{k=1}^{K}\sum_{i=1}^{P}\tilde{q}_{[t,i]}\mathcal{L}\left(\frac{1}{K}\sum_{k=1}^{K}\tilde{\mathbf{w}}_{[k,t,i]}^{\mathsf{T}}\mathbf{z}_p(\mathbf{x}_{k,t}),y_{k,t}\right)\\
&\overset{(b)}{\leq}\sum_{k=1}^{K}\sum_{i=1}^{P}\mathcal{L}(\tilde{\mathbf{w}}_{[k,t,i]}^{\mathsf{T}}\mathbf{z}_p(\mathbf{x}_{k,t}),y_{k,t}),
\end{aligned}\tag{52}$$

where (a) is from the definitions of $\tilde{\mathbf{w}}_{[k,t,i]}$ and $\bar{\mathbf{w}}_{[k,t,i]}$ and (b) is due to the convexity of loss function (i.e., Assumption 1). Combining (48), (51), and (52), the proof is completed. □

Combining Lemma 3, Lemma 4 and Lemma 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 $K = 20$ decentralized nodes and a regularized least-square loss function, i.e.,

$$\mathcal{L}(\mathbf{w}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x}),y)=\left[\mathbf{w}^{\mathsf{T}}\mathbf{z}_i(\mathbf{x})-y\right]^2+\lambda\|\mathbf{w}\|^2.\tag{53}$$

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

$$\mathrm{MSE}(t)=\frac{1}{tK}\sum_{\tau=1}^{t}\sum_{k=1}^{K}(\hat{y}_{k,\tau}-y_{k,\tau})^2,\tag{54}$$

where $\hat{y}_{k,\tau}$ and $y_{k,\tau}$ 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

$$\eta_l=0.5,\ \eta_g=0.5,\lambda=0.01,\ \text{and}\ M=100.\tag{55}$$

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 (55) 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

$$\kappa_i(\mathbf{x},\mathbf{x}')=\exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{\sigma_i^2}\right),\tag{56}$$

with the parameters (or bandwidths)

$$\sigma_i=10^{(i-6)/2},\ i=1,2,...,11.\tag{57}$$

Finally, the real datasets for our experiments on online regressions and time-series predictions are described in Section 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 $\mathbb{P}(\hat{p}_t = \bar{p}_t)$ is computed empirically with 100 samples. Recall that $\hat{p}_t$ and $\bar{p}_t$ 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-KOFL 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-KOFL) 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-KOFL can give stable performances on various online learning tasks. These numerical results suggest the practicality of the proposed methods.

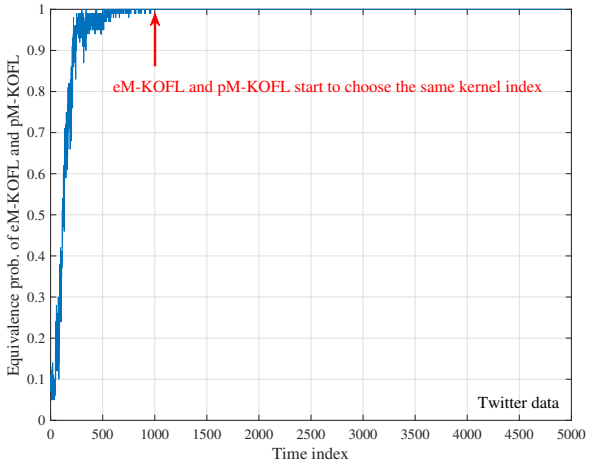
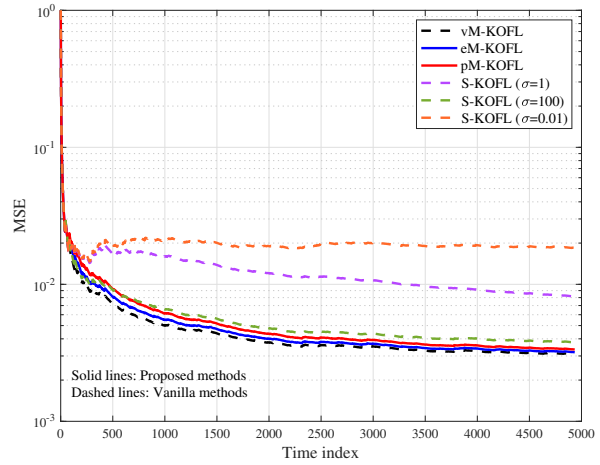


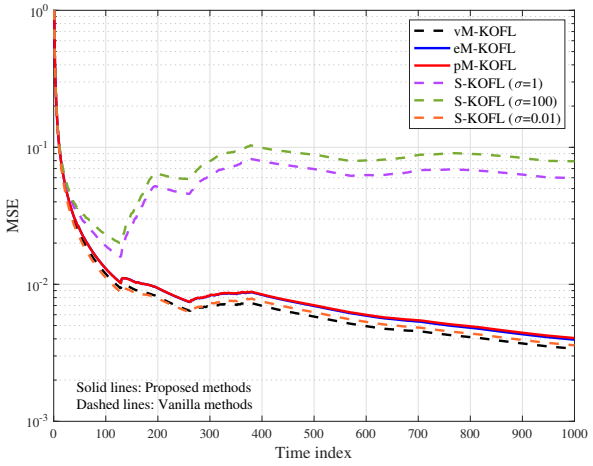
Fig. 2. Comparisons of the proposed eM-KOFL and pM-KOFL in terms of a chosen kernel index. The y-axis measures $\mathbb{P}(\hat{p}_t = \bar{p}_t)$ empirically using Twitter data.

6.1 Online Regression Tasks

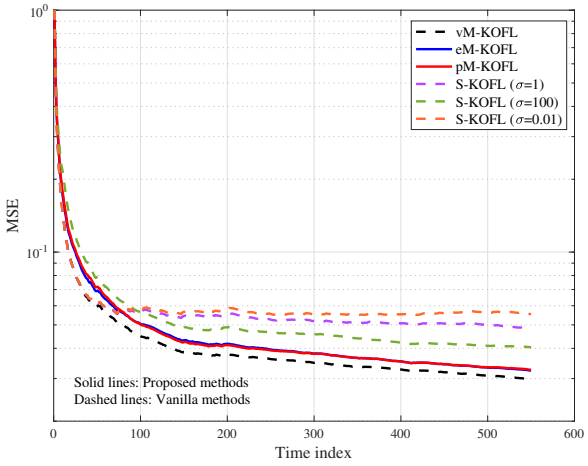
In the experiments of online regression tasks, the following popular real datasets from UCI Machine Learning Repository are



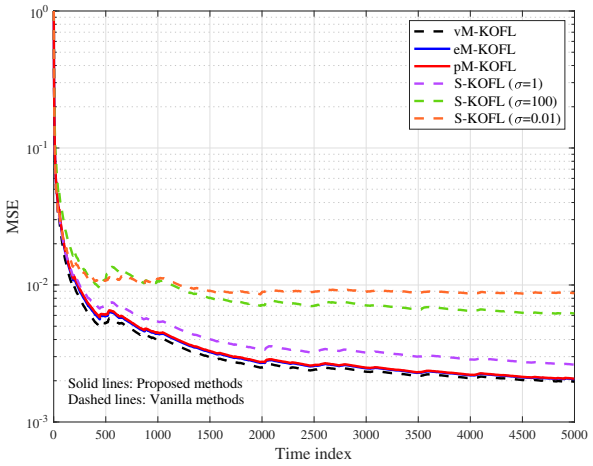
(a) Twitter data



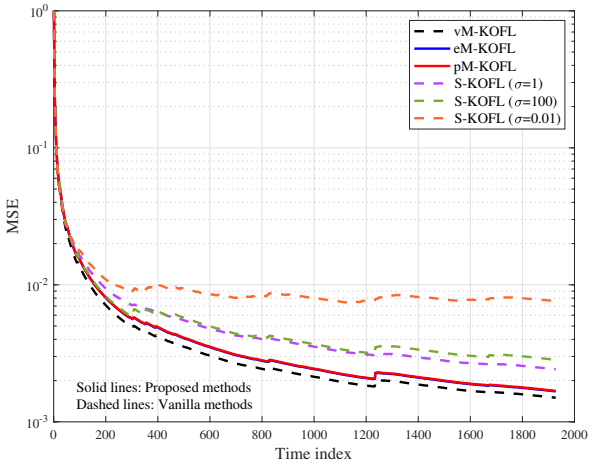
(a) Parking occupancy data



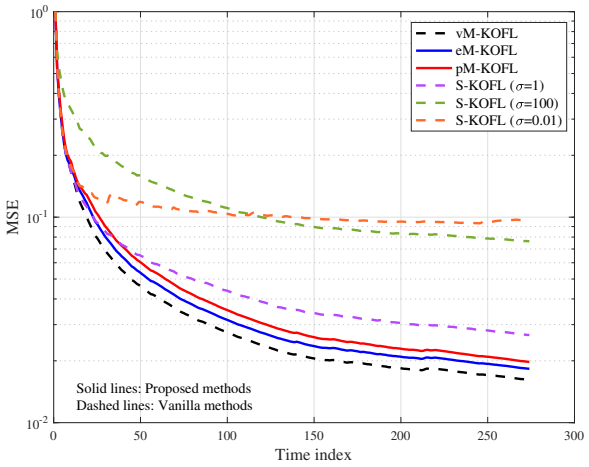
(b) Conductivity data



(b) Power consumption data



(c) Air quality data



(c) Traffic data

Fig. 3. Comparisons of MSE performances of various algorithms on **online regressions tasks**.

Fig. 4. Comparisons of MSE performances of various algorithms on **time-series prediction tasks**.

considered:

- **Twitter [27]:** Data contains buzz events from Twitter, where each attribute is used to predict the popularity of a topic. Higher value indicates more popularity.

TABLE 2
Summary of Real Datasets for Experiments

Datasets	# of features	# of data	feature type
Online regression tasks			
Twitter	77	98704	real & integer
Conductivity	81	11000	real
Tom's hardware	96	9725	real& integer
Air quality	13	38563	real
Time-series prediction tasks			
Power consumption	5	100000	real
Parking occupancy	5	21500	real
Traffic	5	6500	real

- **Conductivity [28]:** 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 [29]:** 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 $\mathrm{AR}(s)$ model predicts the future value y_t assuming the linear dependency on its s values, i.e.,

$$y_t = \sum_{i=1}^{s} \gamma_i y_{t-i} + n_t, \tag{58}$$

where γ_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 $\mathrm{AR}(s)$ model, which can explore a nonlinear dependency, is introduced in [12]:

$$y_t = f_t(\mathbf{x}_t) + n_t, \tag{59}$$

where $\mathbf{x}_t = [y_{t-1}, ..., y_{t-s}]^\mathsf{T}$. The proposed pM-KOFL aims at learning $f_t(\cdot)$ with a parameterized model $f(\mathbf{x}; \{\bar{p}_t, \bar{\mathbf{w}}_t\}) = \bar{\mathbf{w}}_t \mathbf{z}_{\bar{p}_t}(\mathbf{x})$. 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 [30]:** 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 [31]:** Data contains samples obtained from the parking lot in Birmingham, each of which indicates the car park occupancy rate.
- **Traffic [32]:** 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 future work is to extend pM-KOFL into a wireless KOFL framework (or over-the-air KOFL framework). Another 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.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF-2020R1A2C1099836).

References

[1] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, "Federated learning for mobile keyboard prediction," *arXiv preprint arXiv:1811.03604*, 2018.

[2] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *arXiv preprint arXiv:1912.04977*, 2019.

[3] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, 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." in *Esann*, vol. 3, 2013, p. 3.

[5] A. Pantelopoulos and N. G. Bourbakis, "A survey on wearable sensor-based systems for health monitoring and prognosis," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 1, pp. 1–12, 2009.

[6] P. Rashidi and D. J. Cook, "Keeping the resident in the loop: Adapting the smart home to the user," *IEEE Transactions on systems, man, and cybernetics-part A: systems and humans*, vol. 39, no. 5, pp. 949–959, 2009.

[7] C. Richard, J. C. M. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *IEEE Transactions on Signal Processing*, vol. 57, no. 3, pp. 1058–1067, 2008.

[8] S. Shen, H. Jiang, and T. Zhang, "Stock market forecasting using machine learning algorithms," *Department of Electrical Engineering, Stanford University, Stanford, CA*, pp. 1–5, 2012.

[9] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," *IEEE transactions on signal processing*, vol. 52, no. 8, pp. 2165–2176, 2004.

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

[11] 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*, vol. 20, no. 1, pp. 773–808, 2019.

[12] S. Hong and J. Chae, "Active learning with multiple kernels," *accepted to IEEE Transactions on neural networks and learning systems. [Online] arXiv preprint arXiv:2005.03188*, 2020.
[13] B. Scholkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.
[14] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Advances in neural information processing systems*, 2008, pp. 1177–1184.
[15] E. Hazan *et al.*, "Introduction to online convex optimization," *Foundations and Trends® in Optimization*, vol. 2, no. 3-4, pp. 157–325, 2016.
[16] H. Yuan and T. Ma, "Federated accelerated stochastic gradient descent," *arXiv preprint arXiv:2006.08950*, 2020.
[17] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "Qsgd: Communication-efficient sgd via gradient quantization and encoding," *Advances in Neural Information Processing Systems*, vol. 30, pp. 1709–1720, 2017.
[18] J. Wangni, J. Wang, J. Liu, and T. Zhang, "Gradient sparsification for communication-efficient distributed optimization," in *NeurIPS*, 2018.
[19] D. Alistarh, T. Hoefler, M. Johansson, S. Khirirat, N. Konstantinov, and C. Renggli, "The convergence of sparsified gradient methods," *arXiv preprint arXiv:1809.10505*, 2018.
[20] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, "signsgd: Compressed optimisation for non-convex problems," in *International Conference on Machine Learning*. PMLR, 2018, pp. 560–569.
[21] M. Gönen and E. Alpaydın, "Multiple kernel learning algorithms," *The Journal of Machine Learning Research*, vol. 12, pp. 2211–2268, 2011.
[22] J. Chae and S. Hong, "Distributed online learning with multiple kernels," *arXiv preprint arXiv:2102.12733*, 2021.
[23] M. J. Wainwright, *High-dimensional statistics: A non-asymptotic viewpoint*. Cambridge University Press, 2019, vol. 48.
[24] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," *Advances in neural information processing systems*, vol. 20, pp. 1177–1184, 2007.
[25] S. Bubeck, "Introduction to online optimization," *Lecture Notes*, vol. 2, 2011.
[26] S. Shi, Q. Wang, K. Zhao, Z. Tang, Y. Wang, X. Huang, and X. Chu, "A distributed synchronous sgd algorithm with global top-k sparsification for low bandwidth networks," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2019, pp. 2238–2247.
[27] E. G. François Kawala, Ahlame Douzal-Chouakria and E. Dimert, "Prédictions d'activité dans les réseaux sociaux en ligne," *4i'eme Conférence sur les Modéles et l'Analyse des Réseaux: Approches Mathématiques et Informatiques*, 2013.
[28] K.Hamidieh, "A data-driven statistical model for predicting the critical temperature of a superconductor," *Computational Materials Science*, pp. 346–354.
[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 and Actuators B: Chemical*, vol. 129, no. 2, pp. 750–757, 2008.
[30] "Georges hebrail. uci machine learning repository, url 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," *International Conference on Learning and Intelligent Optimization*, 2019.
[32] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml