UNIVERSITY OF CALIFORNIA,
IRVINE

Scalable Online Decision Making: Algorithm Design and Fundamental Limits

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Electrical Engineering and Computer Science

by

Pouya Mollaebrahim Ghari

2024

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ACKNOWLEDGMENTS

I would like to thank my advisor, Prof. Yanning Shen, for her efforts and support throughout the research in this thesis. I would also like to thank my peers at the University of California, Irvine, for their discussions and presentations: Oyku Deniz Kose, Yezi Liu, Yashuo Wu, Ruijie Du, Divya Anand Sinha, Erfan Hajihashemi. I am grateful to my thesis committee members, Prof. Ioannis Panageas, Prof. Yasser Shoukry, and Prof. Syed Ali Jafar, for their invaluable time and thoughtful feedback.

Special thanks to the National Science Foundation (Grants ECCS 2207457 and ECCS 2412484) and the Microsoft Academic Grant Award for AI Research for financially supporting the research in this thesis.

Deep gratitude is extended to my friends and family for their constant support throughout this journey.

VITA

Pouya Mollaebrahim Ghari

EDUCATION

PhD in Electrical Engineering and Computer Science — **2024**
University of California, Irvine — *Irvine, California*

RESEARCH EXPERIENCE

Graduate Research Assistant — **2020–2024**
University of California, Irvine — *Irvine, California*

Machine Learning Research Intern — **Summer 2021**
IQVIA — *Plymouth Meeting, Pennsylvania*

Machine Learning Research Intern — **Summer 2023**
Genentech — *South San Francisco, California*

TEACHING EXPERIENCE

Teaching Assistant — **2021–2022**
University of California, Irvine — *Irvine, California*

REFEREED JOURNAL PUBLICATIONS

[J1] **P. M. Ghari** and Y. Shen, "Online Learning With Uncertain Feedback Graphs," *IEEE Transactions on Neural Networks and Learning Systems*, Jul. 2024.

[J2] **P. M. Ghari**, Y. Shen "Budgeted Online Model Selection and Fine-Tuning via Federated Learning," *Transactions on Machine Learning Research (TMLR)*, Feb. 2024.

[J3] **P. M. Ghari**, Y. Shen "Graph-Aided Online Multi-Kernel Learning," in *Journal of Machine Learning Research (JMLR)*, Jan. 2023.

REFEREED CONFERENCE PUBLICATIONS

[C1] **P. M. Ghari**, Y. Shen "Personalized Federated Learning with Mixture of Models for Adaptive Prediction and Model Fine-Tuning," in *Advances in Neural Information Processing Systems (NeurIPS)*, Dec. 2024.

[C2] **P. M. Ghari**, Y. Shen "Personalized Online Federated Learning with Multiple Kernels," in *Advances in Neural Information Processing Systems (NeurIPS)*, Nov. 2022.

[C3] **P. M. Ghari**, Y. Shen "Graph-Assisted Communication-Efficient Ensemble Federated Learning," in *European Signal Processing Conference (EUSIPCO)*, Aug. 2022.

[C4] **P. M. Ghari**, Y. Shen "Online Learning with Probabilistic Feedback," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, May 2022.

[C5] **P. M. Ghari**, Y. Shen "Graph-Aided Online Learning with Expert Advice," in *Asilomar Conference on Signal, Systems, and Computers*, Nov. 2020.

[C6] **P. M. Ghari**, Y. Shen "Online Multi-Kernel Learning with Graph-Structured Feedback," in *International Conference on Machine Learning (ICML)*, Jul. 2020.

ABSTRACT OF THE DISSERTATION

Scalable Online Decision Making: Algorithm Design and Fundamental Limits

By

Pouya Mollaebrahim Ghari

Doctor of Philosophy in Electrical Engineering and Computer Science

University of California, Irvine, 2024

Assistant Professor Yanning Shen, Chair

Decision-making and real-time prediction in non-stationary and dynamic environments present significant challenges for the application of machine learning and artificial intelligence in operational settings. In these environments, deploying pre-trained models for real-time predictions on data streams often yields suboptimal results due to potential distribution shifts between the training data and the incoming data stream. Moreover, the computational complexity of online decision-making algorithms is crucial, as timely decisions must be made before new data points are received. Furthermore, these algorithms should be designed to maintain affordable memory complexity, as the sheer volume of streaming data cannot be stored in batch. This thesis presents several novel online decision-making and real-time prediction algorithms that effectively address the challenges of non-stationary environments while ensuring computational and memory efficiency.

Novel online multi-kernel learning algorithms for real-time prediction on data streams are introduced, with theoretical guarantees and experimental results demonstrating their advantages in both accuracy and computational efficiency over state-of-the-art online kernel learning algorithms. In online decision-making, feedback is typically observed after each decision round through interactions with the environment. This thesis introduces novel algorithms for online decision-making under uncertain observations. Theoretical analysis and experimental results demonstrate the effectiveness of the proposed algorithms in handling uncertainty during decision-making.

Meanwhile, federated learning is well-regarded for its ability to facilitate distributed model training while ensuring data privacy, as clients retain their data without sharing it with the central server that coordinates the collaboration. Most previous research assumes that clients have static batches of training data. However, in many cases, clients need to make real-time predictions on streaming data in non-stationary environments. Another significant challenge in federated learning is the heterogeneity of data distribution among clients. To address these challenges, this thesis introduces novel personalized online federated learning algorithms. The proposed algorithms enjoy theoretical guarantees, and experiments on real datasets demonstrate their superiority over existing online federated learning approaches in real-time prediction, particularly in the presence of heterogeneous data distribution among clients.

Chapter 1

Graph-Aided Online Multi-Kernel Learning

1.1 Introduction

The need for function approximation arises in many machine learning studies including regression, classification, and reinforcement learning, see e.g., [27]. This chapter studies supervised function approximation where given data samples $\{(\boldsymbol{x}_t, y_t)\}_{t=1}^{T}$, the goal is to find the function $f(\cdot)$, such that the difference between $f(\boldsymbol{x}_t)$ and y_t is minimized. In this context, kernel learning methods exhibit reliable performance. Specifically, the function approximation problem becomes tractable under the assumption that $f(\cdot)$ belongs to a reproducing kernel Hilbert space [133]. In some cases, it is imperative to perform function approximation task in an online fashion. For instance, when the volume of data is large and is collected in a sequential fashion, it is impossible to store or process it in batch. Furthermore, suffering from the well-known problem of 'curse of dimensionality' [11, 136], kernel learning methods are not suitable for sequential settings. This has motivated studies on online single kernel learning [104, 41, 12, 162] to address the curse of dimensionality. Specifically, approximating kernels by finite-dimensional feature representations such as random Fourier feature by [123] and Nyström method by [150], function approximation task becomes scalable. Furthermore, kernel approximation with finite-dimensional features has been extensively studied by e.g. [141, 130, 134, 40].

Most of prior studies rely on a pre-selected kernel; however, such selection requires prior information which may not be available. By contrast, utilizing a dictionary of multiple kernel in lieu of a pre-selected kernel provides more flexible approach to obtain more accurate function approximations as it can learn combination of kernels [140, 85]. Multiple kernel learning successfully has been employed in many learning methods as well as practical applications including cross domain learning [45] and computer vision applications [15]. To utilize the merits of multi-kernel learning several algorithms have been emerged (see e.g. [124, 35, 69]) which exhibit well-documented advantages compared to their single kernel learning counterparts. However, the mentioned algorithms are suitable to apply to batch kernel learning cases and are less efficient or even intractable when it comes to performing kernel learning in an online fashion. In order to make multiple kernel learning amenable for online function approximation, several algorithms have been proposed in the literature (see e.g. [77, 131]). However, the aforementioned algorithms suffer from the curse of dimensionality and are not scalable to deal with large volume of data. Enabled by the random feature approximation by [123], scalable online multi-kernel learning algorithms have been developed by [132, 137]. In particular, the aforementioned algorithms perform function approximation by learning the linear combination of random feature kernel approximations.

One of the most important challenges of MKL is the proper selection of kernels in the dictionary, which influences both computational complexity and accuracy of the function approximation significantly. However, selecting an appropriate kernel dictionary requires prior information. When such information is not available, one solution is to include a large number of kernels in the dictionary. In this case, employing all kernels in the dictionary may not be a feasible choice. Data-driven selection of subset of kernels in a given dictionary can alleviate the computational complexity. Furthermore, data-driven subset selection of kernels can enhance the accuracy of function approximation by pruning irrelevant kernels. The goal of the present chapter is to select a subset of kernels in a given dictionary at each time instant in order to alleviate the computational complexity and improve function approximation accuracy. To this end, our proposed algorithms construct a graph whose vertices represent kernels such that a subset of kernels is selected based on the structure of the graph. In this case, function approximations given by the chosen subset of kernels can be viewed as feedback collected from a graph which is called *feedback graph*. Relative to existing online multi-kernel learning approaches, our novelty can be summarized as follows:

- Different from existing works which employ all kernels in the dictionary, while only learning the combination coefficients, our proposed algorithms only use a subset of kernels at each time instant according to a feedback graph.
- An adaptive and disciplined framework is developed to construct a feedback graph at each time instant according to the loss incurred by kernel-based approximants. A novel OMKL algorithm is proposed to select kernels according to the **g**raph-structured **f**eedback (OMKL-GF) which achieves sublinear regret.
- Construction of the feedback graph at each time instant increases the computational burden of multi-kernel learning. To address this issue, a similarity feedback graph is constructed based on the similarity among kernels, which does not require observing the data samples beforehand. The resulting algorithm is called Online Multi-Kernel Learning with Similarity-based Feedback Graph (OMKL-SFG). It is proved that the proposed OMKL-SFG achieves a sub-linear regret.
- A novel algorithm called OMKL-SFG-R is proposed to adaptively refine the structure

of the similarity-based feedback graph 'on the fly.' It is proved that the OMKL-SFG-R enjoys the sublinear regret tighter than OMKL-SFG and OMKL-GF.

- Experiments on real datasets showcase the effectiveness of our proposed algorithms in comparison with other state-of-the-art OMKL baselines.

The remainder of this chapter is organized as follows. Section 1.2 discusses preliminaries of random-feature based multi-kernel learning. Section 1.3 presents the proposed OMKL-GF algorithm and its regret analysis. Furthermore, Section 1.4 presents the OMKL-SFG and OMKL-SFG-R algorithms along with their theoretical performance in terms of regret. Experimental results are provided in Section 1.5 to study performance of MKL algorithms on several real datasets. Finally, Section 1.6 concludes the chapter. It is also worth noting that the materials in this chapter are fully included in [62] and partially included in [58].

1.2 Preliminaries

Given samples $(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_T, y_T)$, with $\boldsymbol{x}_t \in \mathbb{R}^d$ and $y_t \in \mathbb{R}$, the function approximation problem can be written as the following optimization problem

$$\min_{f \in \mathbb{H}} \frac{1}{T} \sum_{t=1}^{T} \left(\mathcal{C}(f(\boldsymbol{x}_t), y_t) + \lambda \Omega(\|f\|_{\mathbb{H}}^2) \right) \tag{1.1}$$

where $\mathcal{C}(\cdot, \cdot)$ denotes the cost function, which is specified according to the learning task. For example, in regression task $\mathcal{C}(\cdot, \cdot)$ can be least square function. In (1.1), λ denotes the regularization coefficient and $\Omega(\cdot)$ represents a non-decreasing function, which is used to prevent over-fitting and control model complexity.

1.2.1 Function Approximation with Reproducing Hilbert Kernel Space

Let $\kappa(\boldsymbol{x}, \boldsymbol{x}_t) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ represent a symmetric positive definite kernel function which measures the similarity between $\boldsymbol{x}$ and $\boldsymbol{x}_t$. In the context of kernel based learning, it is assumed that the sought $f(\cdot)$ belongs to the reproducing Hilbert kernel space (RHKS) $\mathbb{H} := \{f | f(\boldsymbol{x}) = \sum_{t=1}^{\infty} \alpha_t \kappa(\boldsymbol{x}, \boldsymbol{x}_t)\}$. A kernel is reproducing if it satisfies $\langle \kappa(\boldsymbol{x}, \boldsymbol{x}_t), \kappa(\boldsymbol{x}, \boldsymbol{x}_{t'}) \rangle_{\mathbb{H}} = \kappa(\boldsymbol{x}_t, \boldsymbol{x}_{t'})$ where $\langle \cdot, \cdot \rangle_{\mathbb{H}}$ denotes vector inner product in Hilbert space, with the RKHS norm defined as $\|f\|_{\mathbb{H}}^2 := \sum_t \sum_{t'} \alpha_t \alpha_{t'} \kappa(\boldsymbol{x}_t, \boldsymbol{x}_{t'})$. The representer theorem states that the optimal solution of (1.1) can be expressed as follows given finite data samples [148]

$$\hat{f}(\boldsymbol{x}) = \sum_{t=1}^{T} \alpha_t \kappa(\boldsymbol{x}, \boldsymbol{x}_t) := \boldsymbol{\alpha}^\top \boldsymbol{\kappa}(\boldsymbol{x}) \tag{1.2}$$

where $\boldsymbol{\alpha} := [\alpha_1, \ldots, \alpha_T]^\top$ denotes the vector of unknown coefficients to be estimated, and $\boldsymbol{\kappa}(\boldsymbol{x}) := [\kappa(\boldsymbol{x}, \boldsymbol{x}_1), \ldots, \kappa(\boldsymbol{x}, \boldsymbol{x}_T)]^\top$. Furthermore, it can be observed that the dimension of $\boldsymbol{\alpha}$ increases with the number of data samples T. This is known as 'curse of dimensionality' [148] and arises as a major challenge for solving (1.1) in an online fashion.

1.2.2 Random Fourier Feature Approximation

One way to cope with the increasing number of variables to be estimated is to employ random feature (RF) approximation [123]. As in [123], we will approximate $\kappa(\cdot)$ in (1.2) using shift-invariant kernels which satisfy $\kappa(\boldsymbol{x}_t, \boldsymbol{x}_{t'}) = \kappa(\boldsymbol{x}_t - \boldsymbol{x}_{t'})$. However, relying on a pre-selected kernel often requires prior information that may not be available. To cope with this, multi-kernel learning can be exploited which learns the kernel as a combination of a sufficiently rich dictionary of kernels $\{\kappa_i\}_{i=1}^N$. The kernel combination is itself a kernel [133]. Let $\kappa_i(\boldsymbol{x}_t - \boldsymbol{x}_{t'})$ be the i-th kernel in the dictionary of N absolutely integrable kernels. In this case, its Fourier transform $\pi_{\kappa_i}(\boldsymbol{\psi})$ exists and can be viewed as probability density function (PDF) if the kernel is normalized such that $\kappa_i(\mathbf{0}) = 1$. Specifically, it can be written as

$$\kappa_i(\boldsymbol{x}_t - \boldsymbol{x}_{t'}) = \int \pi_{\kappa_i}(\boldsymbol{\psi}) e^{j\boldsymbol{\psi}^\top(\boldsymbol{x}_t - \boldsymbol{x}_{t'})} d\boldsymbol{\psi} := \mathbb{E}_{\pi_{\kappa_i}(\boldsymbol{\psi})}[e^{j\boldsymbol{\psi}^\top(\boldsymbol{x}_t - \boldsymbol{x}_{t'})}]. \tag{1.3}$$

Let $\{\boldsymbol{\psi}_{i,j}\}_{j=1}^D$ be a set of vectors which are independently and identically distributed (i.i.d) samples from $\pi_{\kappa_i}(\boldsymbol{\psi})$. Hence, $\kappa_i(\boldsymbol{x}_t - \boldsymbol{x}_{t'})$ can be approximated by the ensemble mean $\hat{\kappa}_{i,c}(\boldsymbol{x}_t - \boldsymbol{x}_{t'}) := \frac{1}{D}\sum_{j=1}^D e^{j\boldsymbol{\psi}_{i,j}^\top(\boldsymbol{x}_t - \boldsymbol{x}_{t'})}$. Furthermore, the real part of $\hat{\kappa}_{i,c}(\boldsymbol{x}_t - \boldsymbol{x}_{t'})$ also constitutes an unbiased estimator of $\kappa_i(\boldsymbol{x}_t - \boldsymbol{x}_{t'})$ which can be written as $\hat{\kappa}_i(\boldsymbol{x}_t - \boldsymbol{x}_{t'}) = \mathbf{z}_i^\top(\boldsymbol{x}_t)\mathbf{z}_i(\boldsymbol{x}_{t'})$ [123], where

$$\mathbf{z}_i(\boldsymbol{x}_t) := \frac{1}{\sqrt{D}}[\sin(\boldsymbol{\psi}_{i,1}^\top \boldsymbol{x}_t), \cdots, \sin(\boldsymbol{\psi}_{i,D}^\top \boldsymbol{x}_t), \cos(\boldsymbol{\psi}_{i,1}^\top \boldsymbol{x}_t), \cdots, \cos(\boldsymbol{\psi}_{i,D}^\top \boldsymbol{x}_t)].$$

Replacing $\kappa_i(\boldsymbol{x}, \boldsymbol{x}_t)$ with $\hat{\kappa}_i(\boldsymbol{x} - \boldsymbol{x}_t)$, $\hat{f}(\boldsymbol{x})$ in (1.2) can be approximated as

$$\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}) = \sum_{t=1}^T \alpha_t \hat{\kappa}_i(\boldsymbol{x} - \boldsymbol{x}_t) = \sum_{t=1}^T \alpha_t \mathbf{z}_i^\top(\boldsymbol{x})\mathbf{z}_i(\boldsymbol{x}_t) = \boldsymbol{\theta}_i^\top \mathbf{z}_i(\boldsymbol{x}) \tag{1.4}$$

where $\boldsymbol{\theta}_i \in \mathbb{R}^{2D}$ is a vector whose dimension does not increase with the number of data samples. Therefore, utilizing RF approximation can make the function approximation problem amenable for online implementation. Furthermore, the loss of the i-th kernel can be calculated as

$$\mathcal{L}(\boldsymbol{\theta}_i^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t) = \mathcal{C}(\boldsymbol{\theta}_i^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t) + \lambda\Omega(\|\boldsymbol{\theta}_i\|^2). \tag{1.5}$$

1.2.3 Online MKL as Online Learning with Expert Advice

Online learning with expert advice studies the problem where a learner performs the online learning task by interacting with a set of experts, see e.g. [19]). At each time instant, the learner observes the advice given by experts, and then utilize the received advice to make a decision in real time [8, 106, 73]. In multi-kernel learning, each kernel can be viewed as an expert. Specifically, the approximation obtained by the i-th kernel, can be viewed as the advice given by $\kappa_i(\cdot)$. In particular, when multiple kernels are employed, function approximation can be performed by functions in the form $f(\boldsymbol{x}) = \sum_{i=1}^{N} \bar{w}_i f_i(\boldsymbol{x})$ where $\sum_{i=1}^{N} \bar{w}_i = 1$ [133]. Also, $f_i(\boldsymbol{x}) \in \mathbb{H}_i$ where $\mathbb{H}_i$ is an RKHS induced by the kernel κ_i. Replacing $f_i(\boldsymbol{x})$ with $\hat{f}_{\mathrm{RF},i}(\boldsymbol{x})$, the function $f(\boldsymbol{x})$ can be approximated as

$$\hat{f}_{\mathrm{RF}}(\boldsymbol{x}) = \sum_{i=1}^{N} \bar{w}_i \hat{f}_{\mathrm{RF},i}(\boldsymbol{x}), \sum_{i=1}^{N} \bar{w}_i = 1 \tag{1.6}$$

where the approximation $\hat{f}_{\mathrm{RF}}(\boldsymbol{x})$ is a linear combination of approximations (advice) given by kernels in the dictionary. When all kernels are involved in function approximation at every time instants, the multi-kernel learning problem with RF approximation can be formulated as

$$\min_{\{\bar{w}_i, \boldsymbol{\theta}_i\}} \sum_{t=1}^{T} \left(\mathcal{C}\left(\sum_{i=1}^{N} \bar{w}_i \boldsymbol{\theta}_i^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t \right) + \lambda \Omega(\|\boldsymbol{\theta}_i\|^2) \right) \tag{1.7a}$$

$$\text{s.t.} \sum_{i=1}^{N} \bar{w}_i = 1, \quad \bar{w}_i \geq 0, \quad \forall 1 \leq t \leq T. \tag{1.7b}$$

In online MKL where data samples come sequentially, the optimization problem cannot be solved in batch. Online convex optimization methods can be utilized to update $\{\bar{w}_i\}_{i=1}^{N}$, $\{\boldsymbol{\theta}_i\}_{i=1}^{N}$ upon receiving new datum $\boldsymbol{x}_t$ at each time instant t [132, 137]. Let $\bar{w}_{i,t}$ and $\boldsymbol{\theta}_{i,t}$ denote the values of $\bar{w}_i$ and $\boldsymbol{\theta}_i$ at time t. Upon receiving new datum $\boldsymbol{x}_t$ and computing the loss $\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t)$, using the online gradient descent, $\boldsymbol{\theta}_{i,t}$ can be updated as

$$\boldsymbol{\theta}_{i,t+1} = \boldsymbol{\theta}_{i,t} - \eta \nabla \mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t) \tag{1.8}$$

where η is the learning rate. Furthermore, using multiplicative update, the value of $\bar{w}_{i,t}$ can be updated as

$$w_{i,t+1} = w_{i,t} \exp\left(-\eta \mathcal{L}(\boldsymbol{\theta}_{i,t}^{\top} \mathbf{z}_i(\boldsymbol{x}_t), y_t)\right) \tag{1.9a}$$

$$\bar{w}_{i,t+1} = \frac{w_{i,t+1}}{\sum_{j=1}^{N} w_{j,t+1}}. \tag{1.9b}$$

Employing the update rules in (1.8) and (1.9) $\bar{w}_{i,t}$ and $\boldsymbol{\theta}_{i,t}$ can be updated in an online fashion without storing data in batch.

1.2.4 Assumptions

In order to analyze the proposed algorithms, we apply stochastic regret [73] to measure the difference between expected cumulative loss of the proposed algorithms and the best function approximant in the hindsight. Let $f^*(\cdot)$ denote the best function approximant in hindsight which can be obtained as

$$f^*(\cdot) \in \arg \min_{f_i^*, i \in \{1,\ldots,N\}} \sum_{t=1}^{T} \mathcal{L}(f_i^*(\boldsymbol{x}_t), y_t) \tag{1.10a}$$

$$f_i^*(\cdot) \in \arg \min_{f \in \mathbb{H}_i} \sum_{t=1}^{T} \mathcal{L}(f(\boldsymbol{x}_t), y_t). \tag{1.10b}$$

Hence, the stochastic regret is defined as

$$\sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t)] - \sum_{t=1}^{T} \mathcal{L}(f^*(\boldsymbol{x}_t), y_t) \tag{1.11}$$

where $\mathbb{E}_t[\cdot]$ denotes the expected value at time instant t given the loss observations in prior times. Furthermore, the performance of proposed algorithms is analyzed under the following assumptions:

(A1) The loss function $\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t)$ is convex with respect to $\boldsymbol{\theta}_{i,t}$ at each time instant t.

(A2) For $\boldsymbol{\theta}$ in a bounded set $\mathbb{T}$ which satisfies $\|\boldsymbol{\theta}\| \leq C_{\boldsymbol{\theta}}$ the loss and its gradient are bounded as $\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t) \in [0,1]$ and $\|\nabla \mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t)\| \leq L$, respectively.

(A3) Kernels $\{\kappa_i\}_{i=1}^N$ are shift-invariant (i.e. $\kappa_i(\boldsymbol{x}, \boldsymbol{x}') = \kappa_i(\boldsymbol{x} - \boldsymbol{x}')$, $\forall i$: $i = 1, \ldots, N$), standardized and bounded. Each datum $\|\boldsymbol{x}_t\| \leq 1$.

Note that (A1) can be satisfied by many convex loss functions such as least squares loss and logistic loss. Moreover, (A2) states that the losses are bounded and L-Lipschitz continuous. (A3) states that kernels are assumed to be shift-invariant, standardized and bounded, meaning that $|\kappa_i(\boldsymbol{x})| \leq 1$, $\forall i, \forall \boldsymbol{x}$. In what follows, we introduce a general *graph-aided* OMKL framework, where only a subset of kernels in the dictionary are chosen at each time instant.

1.3 Online Multi-Kernel Learning with Bipartite Feedback Graph

The present section introduces an OMKL approach which selects a subset of kernels using a bipartite graph constructed adaptively at each time instant based on the observed losses.

1.3.1 Data-driven Graph-based Kernel Selection

Instead of combining the entire dictionary of the kernels, in the present chapter, we will consider combining a subset of kernels $\{\kappa_i(\cdot), i \in \mathbb{S}_t\}$ at time instant t instead, where $\mathbb{S}_t$ is the index set of the chosen subset of kernels at time instant t. Hence, the original function approximation problem boils down to

$$\min_{\{\bar{w}_{i,t}, \boldsymbol{\theta}_i\}} \sum_{t=1}^{T} \left(\mathcal{C} \left(\sum_{i \in \mathbb{S}_t} \bar{w}_{i,t} \boldsymbol{\theta}_i^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t \right) + \lambda \Omega(\|\boldsymbol{\theta}_i\|^2) \right) \tag{1.12a}$$

$$\text{s.t.} \sum_{i \in \mathbb{S}_t} \bar{w}_{i,t} = 1, \quad \bar{w}_{i,t} \geq 0, \quad \forall 1 \leq t \leq T. \tag{1.12b}$$

Upon defining the normalized weights $\bar{w}_{i,t} = \frac{w_{i,t}}{\sum_{j \in \mathbb{S}_t} w_{j,t}}$, (1.12) can be re-written as

$$\min_{\{w_{i,t}\}, \{\boldsymbol{\theta}_i\}} \sum_{t=1}^{T} \mathcal{L} \left(\sum_{i \in \mathbb{S}_t} \frac{w_{i,t}}{\sum_{j \in \mathbb{S}_t} w_{j,t}} \boldsymbol{\theta}_i^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t \right) \tag{1.13a}$$

$$\text{s.t.} w_{i,t} \geq 0, \quad \forall 1 \leq i \leq N, \quad \forall 1 \leq t \leq T. \tag{1.13b}$$

However, (1.13) assumes that $\{\mathbb{S}_t\}_{t=1}^{T}$ are preselected sets. In this section, we study data-driven scheme which can adaptively select a subset of kernels 'on the fly' upon receiving new data samples. In order to adaptively choose the subset of kernels, the present section models the pruned kernel combination as feedback collected from a graph, that is constructed in an online fashion. By doing this, the proposed approach trims irrelevant kernels in the dictionary to both improve the function approximation accuracy and reduce the computational complexity of MKL.

Consider a time varying bipartite graph [7] $\mathcal{B}_t$ at time t, which consists of two sets of nodes: N kernel nodes $\{v_{k,1}, ..., v_{k,N}\}$ and J selective nodes $\{v_{s,1}, ..., v_{s,J}\}$ where $v_{k,i}$ and $v_{s,j}$ are the i-th kernel node and j-th selective node, respectively. And the edges of the graph represents the association between the kernel nodes and the selective nodes. Specifically, an edge between $v_{k,i}$ and $v_{s,j}$ exists at time t if the i-th kernel is assigned to j-th selective node. The construction of the graph will be discussed in Section 1.3.2 .

At each time instant, one of the selective nodes $v_{s,j}$ is chosen, and the subset of kernel nodes connected to $v_{s,j}$ will be used for the instantaneous function approximation at time t, [c.f.

(1.13)]. Then, the loss $\mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t)$ is observed for every kernel in the chosen subset and $\boldsymbol{\theta}_{i,t}$ is updated as

$$\boldsymbol{\theta}_{i,t+1} = \boldsymbol{\theta}_{i,t} - \eta \frac{\nabla \mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t)}{q_{i,t}} \mathbf{1}_{i \in \mathbb{S}_t}, \tag{1.14}$$

where $q_{i,t}$ is the probability that the loss of associated kernel is observed and $\mathbf{1}_{i \in \mathbb{S}_t}$ denotes the indicator function such that $\mathbf{1}_{i \in \mathbb{S}_t} = 1$ if $i \in \mathbb{S}_t$ and $\mathbf{1}_{i \in \mathbb{S}_t} = 0$ otherwise. The value of $q_{i,t}$ depends on how the bipartite graph is generated. Upon receiving a new datum $\boldsymbol{x}_t$, the value of the weight w_i is updated ‘on the fly’. Let $w_{i,t}$ denote the weighting coefficient w_i at time instant t. We leverage multiplicative update for weights $w_{i,t}$ as

$$w_{i,t+1} = w_{i,t} \exp(-\eta \ell_{i,t}) \tag{1.15}$$

where $\ell_{i,t}$ denotes the importance sampling loss estimates [4] associated with the i-th kernel as follows

$$\ell_{i,t} = \frac{\mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t)}{q_{i,t}} \mathbf{1}_{i \in \mathbb{S}_t} \tag{1.16}$$

which is the observed loss $\mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t)$ divided by the probability $q_{i,t}$. The function approximation can be obtained as

$$\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t) = \sum_{i \in \mathbb{S}_t} \frac{w_{i,t}}{\sum_{m \in \mathbb{S}_t} w_{m,t}} \hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t) \tag{1.17}$$

where $\hat{f}_{\mathrm{RF},i}(\cdot)$ is defined in (1.4).

Then the selective nodes are assigned to weights $\{u_{j,t+1}\}$ according to the the kernel nodes' weights $\{w_{i,t+1}\}$. Indeed, each selective node's weight $\{u_{j,t+1}\}$ is the total summation of weights of kernel nodes which are connected to this selective node. Specifically the weight of

Algorithm 1 Data Driven (Bipartite) Graph-based Kernel Selection

```
1: Input: Shift-invariant kernels $\kappa_i$, $i = 1, \ldots, N$, step size $\eta > 0$, weights $\{w_{i,t}\}_{i=1}^N$, $\{u_{j,t}\}_{j=1}^J$, bipartite graph $\mathcal{B}_t$ and datum $\boldsymbol{x}_t$.
2: Set $u_{j,t} = \sum_{\forall i: v_{k,i} \rightarrow v_{s,j}} w_{i,t}$.
3: Obtain $p_{j,t}$ via (1.19).
4: Choose one selective node $v_{s,j}$ according to PMF $\boldsymbol{p}_t = (p_{1,t}, ..., p_{J,t})$.
5: Predict $\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t) = \sum_{i \in \mathbb{S}_t} \frac{w_{i,t}}{\sum_{m \in \mathbb{S}_t} w_{m,t}} \hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t)$ with $\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t)$ in (1.4).
6: Obtain loss $\mathcal{L}(\hat{f}_{RF,i}(\boldsymbol{x}_t), y_t)$ for all $i \in \mathbb{S}_t$.
7: Update $\boldsymbol{\theta}_{i,t+1}$ via (1.14) for all $i \in \mathbb{S}_t$.
8: Update $w_{i,t+1}$ via (1.15).
9: Output: $\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t)$, $\{w_{i,t+1}\}_{i=1}^N$, $\{\boldsymbol{\theta}_{i,t+1}\}_{i=1}^N$
```

$v_{s,j}$ is obtained via

$$u_{j,t+1} = \sum_{\forall i: v_{k,i} \rightarrow v_{s,j}} w_{i,t+1}. \tag{1.18}$$

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

$$p_{j,t+1} = (1 - \eta_e) \frac{u_{j,t+1}}{U_{t+1}} + \frac{\eta_e}{J} \tag{1.19}$$

where $U_{t+1} := \sum_{j=1}^{J} u_{j,t+1}$, and $0 < \eta_e \leq 1$ is the exploration rate. The term $\frac{\eta_e}{J}$ is introduced to tradeoff between exploitation and exploration. The first term on the right hand side of (1.19) implies choosing a selective node with larger weight $u_{j,t+1}$ with higher probability. And the term $\frac{\eta_e}{J}$ is used to to promote exploration. Algorithm 1 summarizes the data driven kernel selection scheme presented in this section.

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

1.3.2 Online Bipartite Feedback Graph Construction

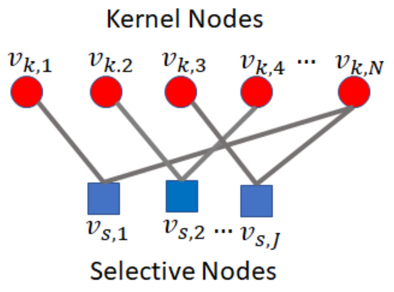
Construction of the time varying graph is of utmost importance, as it affects both function approximation accuracy and computational complexity. In this regard, a graph is successful if it can provide a subset of kernels which results in smallest possible loss. Indeed, considering computational complexity, the graph should provide a limited number of kernels which obtain minimum loss. To this end, we aim to propose a generating method for graph. Specifically, using the weights $\{w_{i,t+1}\}_{i=1}^{N}$ obtained via (1.15), the structure of the graph is reconstructed in a stochastic manner stated below to be leveraged in the next time instant.

Increasing the number of kernel nodes connected to $v_{s,j}$, increases the computational complexity of performing function approximation by choosing $v_{s,j}$. Therefore, the graph generation algorithm should be designed to limit the number of kernel nodes connected to each selective node. Let M denote the maximum number of kernel nodes connected to each selective node. The procedure to generate the graph $\mathcal{B}_{t+1}$ is presented in Algorithm 2. Let $\boldsymbol{A}_{t+1}$ represent the $N \times J$ sub-adjacency matrix between two disjoint subsets $\{v_{s,1}, ..., v_{s,J}\}$ and $\{v_{k,1}, ..., v_{k,N}\}$. Notation $\boldsymbol{A}_{t+1}(i,j)$ represents the element in i-th row and j-th column of the sub-adjacency matrix $\boldsymbol{A}_{t+1}$ and it is equal to 1 if $v_{k,i}$ is connected to $v_{s,j}$, and 0 otherwise.

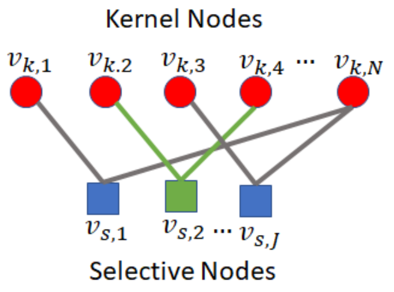
Each selective node $v_{s,j}$ draws kernel nodes $v_{k,i}$ in M independent trials and in each trial selective node draws only one kernel node. We put more weight on kernels which obtain less loss in a sense that the probability that selective node $v_{s,j}$ draws the kernel node $v_{k,i}$ in a

Algorithm 2 Generating Bipartite Feedback Graph

```
1: Input: Shift-invariant kernels $\kappa_i$, weighting coefficient $w_{i,t}$, $i = 1, \ldots, N$, exploration
   coefficient $\eta_e$ and the maximum degree of each selective node $M$.
2: Initialize: Sub-adjacency matrix $\boldsymbol{A}_{t+1} = \mathbf{0}_{N\times J}$.
3: for $j = 1, ..., J$ do
4:     for $i = 1, ..., N$ do
5:         Set $\pi_{ij,t+1} = (1 - {\eta_e}^j)\frac{w_{i,t+1}}{\sum_{i=1}^N w_{i,t+1}} + \frac{{\eta_e}^j}{N}$.
6:     end for
7:     for $k = 1, ..., M$ do
8:         Choose one of nodes $v_{k,i}$ drawn according to the probability mass function (PMF)
           $\boldsymbol{\pi}_{j,t+1} = (\pi_{1j,t+1}, ..., \pi_{Nj,t+1})$.
9:         Set $\boldsymbol{A}_{t+1}(i, j) = 1$.
10:    end for
11: end for
12: Output: Bipartite feedback graph $\mathcal{B}_{t+1}$ with adjacency matrix $\boldsymbol{A}_{t+1}$
```



(a) A bipartite feedback graph consists of J selective nodes and N kernel nodes.



(b) The chosen selective node ($v_{s,2}$ as an example) and edges associated with it are highlighted.

Figure 1.1: A bipartite feedback graph generated by Algorithm 2.

trial at time $t + 1$ is

$$\pi_{ij,t+1} = (1 - {\eta_e}^j)\frac{w_{i,t+1}}{\sum_{k=1}^N w_{k,t+1}} + \frac{{\eta_e}^j}{N} \tag{1.20}$$

Note that the first term in (1.20) discriminates between kernels based on their weights which is determined by their loss in function approximation [c.f. (1.15)]. Furthermore, the second term allows exploration over all kernel nodes. Specially, the selective node $v_{s,j}$ draws kernel nodes according to uniform distribution if $\eta_e = 1$. Furthermore, note that ${\eta_e}^j$ is a non-increasing function of j for $0 < \eta_e \leq 1$. The selective node $v_{s,1}$ puts more weight on exploration in

Algorithm 3 OMKL with (Bipartite) Graph Feedback (OMKL-GF)

```
1: Input: Shift-invariant kernels $\kappa_i$, $i = 1, \ldots, N$, step size $\eta > 0$ and the number of
   random features $D$.
2: Initialize: $\boldsymbol{\theta}_{i,1} = \mathbf{0}$, $w_{i,1} = 1$, $i = 1, ..., N$, generate $\mathcal{B}_1$ using Algorithm 2 given $w_{i,1}$, $\forall i$
3: for $t = 1, ..., T$ do
4:     Receive one datum $\boldsymbol{x}_t$.
5:     Obtain $\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t)$, $\{w_{i,t+1}\}_{i=1}^N$, $\{\boldsymbol{\theta}_{i,t+1}\}_{i=1}^N$ using Algorithm 1 given $\mathcal{B}_t$, $\{w_{i,t}\}_{i=1}^N$, $\{\boldsymbol{\theta}_{i,t}\}_{i=1}^N$.
6:     Generate $\mathcal{B}_{t+1}$ using Algorithm 2 with $\{w_{i,t+1}\}_{i=1}^N$.
7: end for
```

comparison with others while $v_{s,J}$ considers more exploitation than all the other selective nodes. Therefore, the selective nodes entail different level of exploration and exploitation.

Based on the definition of $\pi_{ij,t+1}$ in (1.20), the probability that the i-th kernel node is connected to $v_{s,j}$ is $1 - (1 - \pi_{ij,t+1})^M$, where $(1 - \pi_{ij,t+1})^M$ is the probability that the i-th kernel node is chosen by $v_{s,j}$ in none of M trials. Therefore, the probability of observing the loss of the i-th kernel at time $t + 1$ is given by

$$q_{i,t+1} = \sum_{j=1}^{J} p_{j,t+1} \left(1 - (1 - \pi_{ij,t+1})^M\right) \tag{1.21}$$

for $1 \leq i \leq N$. The value of $q_{i,t+1}$ is computed and used for importance sampling loss estimate in (1.16). The graph generation framework is summarized in Algorithm 2. And Figure 1.1 illustrates an example of the constructed bipartite feedback graph.

At each time instant t, the bipartite graph $\mathcal{B}_t$ is used for choosing a selective node, and henceforth subset of the kernels. Then the weights of the selected kernels are updated according to the loss [c.f.(1.14) and (1.15)]. And the graph can be constructed using Algorithm 2, based on which, the function approximation can be carried out by choosing one of the selective nodes which leads to selecting a subset of kernels. Our proposed online multi-kernel learning with graph-structured feedback (OMKL-GF) is summarized in Algorithm 3.

Computational Complexity. At time instant t, OMKL-GF needs to store a real $2D$ random feature vector in addition to a weighting vector for each kernel in conjunction with a weighting vector for each selective node. As the number of kernels is in general larger than the number of selective nodes, the required memory is of order $\mathcal{O}(dDN)$. The per-iteration complexity of our OMKL-GF (e.g. calculating inner products) is $\mathcal{O}(dDM + JN)$. In comparison, the per-iteration complexity of OMKR developed by [131] is $\mathcal{O}(tdN)$, while more contemporary online RF-based OMKL approaches proposed by [137, 132] both have per-iteration complexity $\mathcal{O}(dDN)$. Hence, OMKL-GF can significantly reduce the per iteration complexity especially when $J \leq M << N$.

1.3.3 Regret Analysis

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

Lemma 1.1. *The regret of the proposed OMKL-GF under (A1) and (A2) with respect to a preselected kernel κ_i where $\mathcal{F}_i = \{\hat{f}_i | \hat{f}_i(\boldsymbol{x}) = \boldsymbol{\theta}^\top \mathbf{z}_i(\boldsymbol{x}), \forall \boldsymbol{\theta} \in \mathbb{R}^{2D}\}$ satisfies the following bound*

$$\begin{aligned}&\sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(\hat{f}_{RF}(\boldsymbol{x}_t), y_t)] - \sum_{t=1}^{T} \mathcal{L}(\hat{f}_i^*(\boldsymbol{x}_t), y_t) \\ &< \frac{\ln N}{\eta} + \frac{\|\boldsymbol{\theta}_i^*\|^2}{2\eta} + \eta_e JT + \frac{\eta NT}{2(1-\eta_e)} + \frac{\eta L^2 NJT}{2\eta_e^2}\end{aligned} \tag{1.22}$$

where $\boldsymbol{\theta}_i^*$ is associated with the best RF function approximant $\hat{f}_i^*(\boldsymbol{x}_t) = \boldsymbol{\theta}_i^{*\top} \mathbf{z}_i(\boldsymbol{x}_t)$ and the expectation at time t is taken with respect to PMFs $\boldsymbol{p}_t$ and $\boldsymbol{\pi}_{j,t}$ in (1.19) *and* (1.20)*, respectively.*

Proof. see Appendix A.2. □

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

Theorem 1.1. *The following bound holds with probability at least* $1 - 2^8(\frac{\sigma_i}{\epsilon})^2\exp(-\frac{D\epsilon^2}{4d+8})$ *under (A1)–(A3) for* $\epsilon > 0$ *and with* f_i^* *belonging to RKHS* $\mathbb{H}_i$ *as in* (1.10b)

$$\sum_{t=1}^{T}\mathbb{E}_t[\mathcal{L}(\hat{f}_{RF}(\boldsymbol{x}_t), y_t)] - \min_{i\in\{1,\ldots,N\}}\sum_{t=1}^{T}\mathcal{L}(f_i^*(\boldsymbol{x}_t), y_t)$$
$$< \frac{\ln N}{\eta} + \frac{\|\boldsymbol{\theta}_i^*\|^2}{2\eta} + \eta_e JT + \epsilon LTC + \frac{\eta NT}{2(1-\eta_e)} + \frac{\eta L^2 NJT}{2\eta_e^2} \tag{1.23}$$

where C *is a constant, and* σ_i^2 *is the second order moment of the RF vector norm which can be defined as* $\sigma_i^2 := \mathbb{E}_{\pi_{\kappa_i}(\boldsymbol{\psi})}[\|\boldsymbol{\psi}\|^2]$*. The expectation at time t is taken with respect to the randomness in* $\boldsymbol{p}_t$ *and* $\boldsymbol{\pi}_{j,t}$ *defined in* (1.19) *and* (1.20)*, respectively.*

Proof. see Appendix A.3. □

According to Theorem 1.1, by setting

$$\eta = \mathcal{O}\left(\sqrt{\frac{\ln N}{N}}T^{-\frac{3}{4}}\right), \eta_e = \mathcal{O}\left(N^{\frac{1}{6}}T^{-\frac{1}{4}}\right), J = \mathcal{O}\left(N^{\frac{1}{3}}\right), \epsilon = \mathcal{O}\left(T^{-\frac{1}{4}}\right) \tag{1.24}$$

in (1.23), the stochastic static regret in (1.11) satisfies $\mathcal{O}\left(\sqrt{N\ln N}T^{\frac{3}{4}}\right)$. Thus, by selecting appropriate parameters, our proposed OMKL-GF achieves sublinear regret in expectation with respect to the best static function approximant in (1.11). Note that while proper settings of ϵ and η relies on the knowledge of T, such information may not be necessary, via employing, e.g., doubling trick [19]. Considering (1.23), the probability $1-2^8(\frac{\sigma_i}{\epsilon})^2\exp(-\frac{D\epsilon^2}{4d+8})$ is an increasing function of D such that for a fixed ϵ, always there are some values for D which result in positive probability. Furthermore, (1.23) shows that by setting $\epsilon = \mathcal{O}(T^{-\frac{1}{4}})$ and $D = \mathcal{O}(\sqrt{T}\ln T)$, the sublinear regret of $\mathcal{O}(\sqrt{N\ln N}T^{\frac{3}{4}})$ can be obtained with high probability of $1 - \mathcal{O}(\frac{1}{\sqrt{T}})$.

The computational complexity of kernel learning algorithms play an important role in their applicability. Bipartite feedback graph construction at each time instant increases the computational complexity of OMKL-GF. To further alleviate the computational burden of multi-kernel learning the ensuing section investigates the problem of choosing a subset of kernels using a feedback graph while the graph is not constructed at every time instant.

1.4 Online Multi-Kernel Learning with Similarity-based Feedback Graph

Note that OMKL-GF is a data-driven kernel selection scheme where a bipartite feedback graph is constructed at every time instant. However, online feedback graph construction increases the computational complexity of OMKL-GF. This section proposes a novel algorithmic framework to construct the feedback graph in an offline fashion such that the proposed algorithms do not need to construct the feedback graph at every time instant. Moreover, the bipartite feedback graph $\mathcal{B}_t$ constructed by Algorithm 2 do not exploit the relationship among kernels. Hence, in this section, the similarity among kernels is measured which will facilitate constructing the feedback graph in an offline fashion in such a way that at each time instant a subset of dissimilar kernels are chosen to avoid unnecessary computation. The present section first introduces a disciplined way to construct feedback graph based on kernel similarities in an offline fashion. Based on the constructed feedback graph, a novel online MKL algorithm (called OMKL-SFG) is developed to select a subset of kernels which is proved to obtain sub-linear regret. Furthermore, to obtain tighter regret bound a modification of OMKL-SFG algorithm (called OMKL-SFG-R) is proposed which refines the structure of the feedback graph to choose a subset of kernels. OMKL-SFG-R entails more computation than OMKL-SFG.

1.4.1 Offline Similarity-based Feedback Graph Construction

The similarity between two shift invariant kernels κ_i and κ_j is measured through divergence between κ_i and κ_j. As κ_i and κ_j has smaller divergence, they are considered to be more similar. The present chapter measures the divergence between a pair of kernels using the Bregman divergence.

Let Ω be a d-dimensional convex set. Bregman divergence defined for a strictly convex and differentiable function $F(\cdot) : \Omega \to \mathbb{R}$ as (see, e.g. [13, 10])

$$B_F(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) = F(\boldsymbol{\omega}_1) - F(\boldsymbol{\omega}_2) - \nabla F(\boldsymbol{\omega}_2)^\top (\boldsymbol{\omega}_1 - \boldsymbol{\omega}_2). \tag{1.25}$$

Based on Bregman divergence, the divergence between two shift invariant kernels κ_i and κ_j can be measured through the function $\Delta(\kappa_i, \kappa_j)$ which is defined as

$$\Delta(\kappa_i, \kappa_j) = \int B_F\left(\kappa_i(\boldsymbol{\rho}), \kappa_j(\boldsymbol{\rho})\right) d\boldsymbol{\rho} \tag{1.26}$$

where $\boldsymbol{\rho} \in \mathbb{R}^d$. As it can be inferred from (1.26), $\Delta(\kappa_i, \kappa_j)$ measures the divergence between two kernels κ_i and κ_j using the aggregation of Bregman divergence on every point $\boldsymbol{\rho}$ in the input space. As $\Delta(\kappa_i, \kappa_j)$ decreases, kernels κ_i and κ_j are considered to be more similar. Note that instead of defining the divergence as in (1.26), one can define the divergence function $\Delta(\kappa_i, \kappa_j)$ as the expected Bregman divergence over points $\boldsymbol{\rho}$. However, taking the expectation requires knowing the distribution of input data samples which may not be available priori. Furthermore, the distribution of input space may change over time and as a result calculating the expected value of Bregman divergence in an offline fashion over the input space may not be feasible. Moreover, squared Euclidean divergence $B_F\left(\kappa_i(\boldsymbol{\rho}), \kappa_j(\boldsymbol{\rho})\right) = \|\kappa_i(\boldsymbol{\rho}) - \kappa_j(\boldsymbol{\rho})\|^2$ is generated by the function $F(\boldsymbol{\omega}) = \|\boldsymbol{\omega}\|^2$. Let $\Delta_S(\cdot, \cdot)$ denotes the function $\Delta(\cdot, \cdot)$ when the

Bregman divergence is obtained by $F(\boldsymbol{\omega}) = \|\boldsymbol{\omega}\|^2$. In this case, we have

$$\Delta_S(\kappa_i, \kappa_j) = \int |\kappa_i(\boldsymbol{\rho}) - \kappa_j(\boldsymbol{\rho})|^2 d\boldsymbol{\rho}. \tag{1.27}$$

The following Lemma states that the function $\Delta_S(\kappa_i, \kappa_j)$ exists for each pair of absolutely integrable kernels.

Lemma 1.2. *Under the assumption that kernels $\{\kappa_i\}_{i=1}^N$ are absolutely integrable, bounded and normalized such that $\kappa_i(\mathbf{0}) = 1$, $\forall i : 1 \leq i \leq N$, the function $\Delta_S(\kappa_i, \kappa_j)$ is bounded and exists for each pair of kernels $\kappa_i(\cdot)$ and $\kappa_j(\cdot)$.*

Proof. Since kernels $\{\kappa_i(\boldsymbol{\rho})\}_{i=1}^N$ are assumed to be bounded as $0 \leq \kappa_i(\boldsymbol{\rho}) \leq 1$, $\forall \boldsymbol{\rho}, 1 \leq i \leq N$, it can be concluded that $|\kappa_i(\boldsymbol{\rho}) - \kappa_j(\boldsymbol{\rho})|^2 \leq |\kappa_i(\boldsymbol{\rho}) - \kappa_j(\boldsymbol{\rho})|$. Thus, it can be inferred that

$$\int |\kappa_i(\boldsymbol{\rho}) - \kappa_j(\boldsymbol{\rho})|^2 d\boldsymbol{\rho} \leq \int |\kappa_i(\boldsymbol{\rho}) - \kappa_j(\boldsymbol{\rho})| d\boldsymbol{\rho}. \tag{1.28}$$

Furthermore, based on the Triangular inequality, it can be written that

$$\int |\kappa_i(\boldsymbol{\rho}) - \kappa_j(\boldsymbol{\rho})| d\boldsymbol{\rho} \leq \int |\kappa_i(\boldsymbol{\rho})| d\boldsymbol{\rho} + \int |\kappa_j(\boldsymbol{\rho})| d\boldsymbol{\rho}. \tag{1.29}$$

Based on (1.28), (1.29) and the fact that kernels are absolutely integrable, we can conclude that the function $\Delta_S(\kappa_i, \kappa_j)$ is bounded and exists for each pair of kernels $\kappa_i(\cdot)$ and $\kappa_j(\cdot)$. $\square$

Furthermore, the following lemma states that the average difference between function approximations given by each pair of kernels is bounded above in accordance with the function $\Delta_S(\cdot, \cdot)$ defined in (1.27).

Lemma 1.3. *Let $C_m := \max_i \sum_{t=1}^T |\alpha_{i,t}|^2$ where $\{\alpha_{i,t}\}_{t=1}^T$ are weights for* (1.2) *associated with the i-th kernel $\kappa_i(\cdot)$. Also, let $\boldsymbol{x}$ is bounded as $\|\boldsymbol{x}\| \leq 1$ and kernels are absolutely integrable. Then, the average difference between function approximations given by $\kappa_i(\cdot)$ and*

$\kappa_j(\cdot)$ is bounded above as

$$\frac{1}{\mathcal{U}_d}\int |\hat{f}_i(\boldsymbol{x}) - \hat{f}_j(\boldsymbol{x})|^2 d\boldsymbol{x} \leq \frac{2C_m}{\mathcal{U}_d}\sum_{t=1}^{T}(\Delta_S(\kappa_i, \kappa_j) + 2\mathcal{U}_d) \tag{1.30}$$

where $\hat{f}_i(\boldsymbol{x})$ denotes the function approximation given by $\kappa_i(\cdot)$ as in (1.2) *and $\mathcal{U}_d$ represents d-dimensional Euclidean unit norm ball volume.*

Proof. See Appendix A.4. □

Let $\mathcal{G} := (\mathcal{V}, \mathcal{E})$ be a directed graph with vertex $v_i \in \mathcal{V}$ which represents the i-th kernel κ_i. In this case, there is a self-loop for each $v_i \in \mathcal{V}$ which means $(i, i) \in \mathcal{E}$. Furthermore, an edge from v_i to v_j is appended to $\mathcal{E}$ if

$$\frac{1}{|\mathbb{N}_i^{\text{out}}|}\sum_{m \in \mathbb{N}_i^{\text{out}}} \Delta(\kappa_m(\boldsymbol{\rho}), \kappa_j(\boldsymbol{\rho})) \geq \gamma_i \tag{1.31}$$

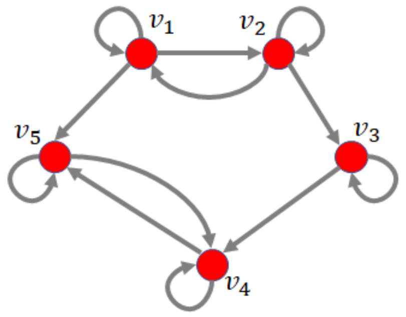
where γ_i is a threshold for v_i and $\mathbb{N}_i^{\text{out}}$ denote the current out-neighborhood set of v_i which means $j \in \mathbb{N}_i^{\text{out}}$ if $(i, j) \in \mathcal{E}$. Using the (1.31) to append edges to the graph $\mathcal{G}$, a vertex v_j associated with the kernel κ_j is added to the out-neighborhood set of v_i if it is dissimilar to the current out-neighbors of v_i. Therefore, the subset of vertices which are out-neighbors of v_i, are jointly dissimilar. Since a subset of function approximations associated with kernels will be chosen using the graph $\mathcal{G}$, the chosen subset of function approximations can be viewed as feedback collected from the graph $\mathcal{G}$ and as a result the graph $\mathcal{G}$ is called *feedback graph*. Specifically in order to restrict the number of out-neighbors for each node to M, the value of γ_i is obtained as

$$\gamma_i = \arg\max_{\gamma}\{\gamma \,|\, |\mathbb{N}_i^{\text{out}}| = M\}. \tag{1.32}$$

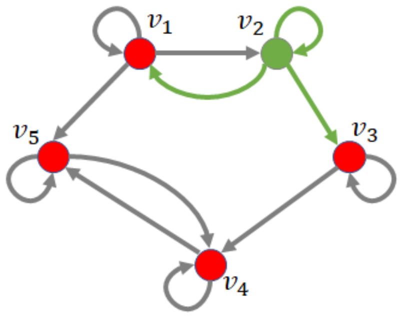
Note that M is a preselected parameter in the algorithm and increasing the value of M

Algorithm 4 Generating Similarity based Feedback Graph

```
1: Input:  Shift-invariant kernels $\kappa_i$, $i = 1, ..., N$.
2: for $i = 1, ..., N$ do
3:     Append $(i, i)$ to $\mathcal{E}$.
4:     Obtain $\gamma_i$ via (1.32).
5:     Append $(i, j)$ to $\mathcal{E}$ if $\frac{1}{|\mathbb{N}_i^{\text{out}}|}\sum_{m\in\mathbb{N}_i^{\text{out}}} \Delta(\kappa_m(\boldsymbol{\rho}), \kappa_j(\boldsymbol{\rho})) \geq \gamma_i$.
6: end for
7: Output: Similarity-based Feedback Graph $\mathcal{G}$.
```



(a) A similarity-based feedback graph with 5 kernel nodes.



(b) As an example, the chosen node (v_2) and its out-neighbors are highlighted.

Figure 1.2: An example of similarity-based feedback graph generated by Algorithm 4.

increases the connectivity of the feedback graph. At each time instant, one of the nodes are drawn and the function approximation is carried out using the combination of a subset of kernels which are out-neighbors of the chosen node. Therefore, increase in M can increase the exploration in the approximation task while it increases the computational complexity. The feedback graph construction procedure is summarized in Algorithm 9. It can be observed from (1.26) that the function $\Delta(\kappa_i, \kappa_j)$ can be considered as a measure of divergence, and henceforth dissimilarity between kernels $\kappa_i(\cdot)$ and $\kappa_j(\cdot)$ without knowing data samples $\{\boldsymbol{x}_t\}_{t=1}^T$. This helps reduction of computational complexity of the function approximation since (dis)similarity among kernels in the dictionary can be measured offline before observing data samples and as a result the computation required to perform Algorithm 4 can be performed offline.

At each time instant, one of the vertices of the feedback graph is drawn according to a PMF as it will be explained in the next section. Then, the function approximation is performed using the kernels associated with out-neighbors of the chosen vertex. Therefore, based on the feedback graph construction in Algorithm 4, at each time instant a subset of dissimilar kernels is chosen to avoid unnecessary computation since it is expected that similar kernels provide comparatively similar approximations. See also Figure 1.2 for an example of similarity based feedback graph where the number of kernels is 5 and v_i represents a Gaussian kernel with bandwidth of 10^{i-3}. For each node v_i, γ_i is selected so that the number of out-neighbors of v_i is 3.

1.4.2 Kernel Selection with Offline Feedback Graph

The present section studies how to select a subset of kernels using the feedback graph $\mathcal{G}$ and prior observations of losses associated with kernels. Assume that each kernel is associated with a set of weights $\{w_{i,t}\}_{i=1}^{N}$ where $w_{i,t}$ is the weight associated with the i-th kernel κ_i. The weight $w_{i,t}$ indicates the accuracy of the function approximation given by the κ_i at time t and its value can be updated when more and more information is being revealed. Furthermore, a set of weights $\{u_{i,t}\}_{i=1}^{N}$ is assigned to $\mathcal{V}$ such that $u_{i,t}$ is the weight associated with $v_i \in \mathcal{V}$ at time instant t, which indicates the accuracy of function approximation when the node v_i is drawn. In order to choose a subset of kernels at time t, one of the vertices in $\mathcal{V}$ is drawn according to the PMF p_t

$$p_{i,t} = (1-\xi)\frac{u_{i,t}}{U_t} + \frac{\xi}{|\mathbb{D}|}\mathbf{1}_{\mathrm{i}\in\mathbb{D}}, i = 1, \ldots, N \tag{1.33}$$

where ξ is the exploration rate and $U_t := \sum_{i=1}^{N} u_{i,t}$. $\mathbb{D}$ represents the dominating set of $\mathcal{G}$, and $|\mathbb{D}|$ denotes the cardinality of $\mathbb{D}$. Let $\mathbb{S}_t$ denote the subset of kernel indices chosen at time t, and I_t denote the index of the kernel drawn according to the PMF p_t in (1.33). Therefore, $i \in \mathbb{S}_t$ if $i \in \mathbb{N}_{I_t}^{\mathrm{out}}$, which means that the loss associated with the i-th kernel is calculated if the i-th kernel is an out-neighbor of the I_t-th node. In turn, the RF-based function approximation can be obtained as

$$\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t) = \sum_{i \in \mathbb{N}_{I_t}^{\mathrm{out}}} \frac{w_{i,t}}{\sum_{j \in \mathbb{N}_{I_t}^{\mathrm{out}}} w_{j,t}} \hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t). \tag{1.34}$$

Furthermore, the importance sampling loss estimate $\ell_{i,t}$ at time instant t is defined as

$$\ell_{i,t} = \frac{\mathcal{L}(\boldsymbol{\theta}_{i,t}^{\top} \mathbf{z}_i(\boldsymbol{x}_t), y_t)}{q_{i,t}} \mathbf{1}_{\mathrm{i} \in \mathbb{S}_{\mathrm{t}}}, i = 1, \ldots, N \tag{1.35}$$

where $q_{i,t}$ is the probability that $i \in \mathbb{S}_t$ and it can be computed as

$$q_{i,t} = \sum_{j \in \mathbb{N}_i^{\mathrm{in}}} p_{j,t} \tag{1.36}$$

where $\mathbb{N}_i^{\mathrm{in}}$ denote the in-neighborhood set of v_i which means $j \in \mathbb{N}_i^{\mathrm{in}}$ if $(j, i) \in \mathcal{E}$. In addition, the importance sampling function approximation estimate $\hat{\ell}_{i,t}$ at time instant t associated with $v_i \in \mathcal{V}$ is defined as

$$\hat{\ell}_{i,t} = \frac{\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t)}{p_{i,t}} \mathbf{1}_{\mathrm{I_t = i}}. \tag{1.37}$$

Using the importance sampling loss estimate in (1.36), $\boldsymbol{\theta}_{i,t}$ can be updated as

$$\boldsymbol{\theta}_{i,t+1} = \boldsymbol{\theta}_{i,t} - \eta \nabla \ell_{i,t} = \boldsymbol{\theta}_{i,t} - \eta \frac{\nabla \mathcal{L}(\boldsymbol{\theta}_{i,t}^{\top} \mathbf{z}_i(\boldsymbol{x}_t), y_t)}{q_{i,t}} \mathbf{1}_{\mathrm{i} \in \mathbb{S}_{\mathrm{t}}}, \tag{1.38}$$

where η is the learning rate. Moreover, the multiplicative update is employed to update $w_{i,t}$ and $u_{i,t}$ based on importance sampling loss estimates in (1.35) and (1.37) as follows

$$w_{i,t+1} = w_{i,t} \exp(-\eta \ell_{i,t}), i = 1, \ldots, N \tag{1.39a}$$

$$u_{i,t+1} = u_{i,t} \exp(-\eta \hat{\ell}_{i,t}), i = 1, \ldots, N. \tag{1.39b}$$

Algorithm 5 OMKL with Similarity-based Feedback Graph (OMKL-SFG)

```
1: Input: Shift-invariant kernels $\kappa_i$, $i = 1, \ldots, N$, learning rate $\eta$, exploration rate $\xi$, the
   number of random features $D$.
2: Initialize: $\boldsymbol{\theta}_{i,1} = \mathbf{0}$, $w_{i,1} = 1$, $i = 1, \ldots, N$, Construct the feedback graph $\mathcal{G}$ via
   Algorithm 4.
3: for $t = 1, \ldots, T$ do
4:     Receive one datum $\boldsymbol{x}_t$.
5:     Draw one of nodes $v_i \in \mathcal{V}$ according to the PMF $p_t = (p_{1,t}, ..., p_{N,t})$ in (1.33).
6:     Predict $\hat{f}_{\text{RF}}(\boldsymbol{x}_t)$ via (1.34).
7:     Calculate loss $\mathcal{L}(\hat{f}_{\text{RF},i}(\boldsymbol{x}_t), y_t)$ for all $i \in \mathbb{S}_t$.
8:     Update $\boldsymbol{\theta}_{i,t+1}$ via (1.38).
9:     Update $w_{i,t+1}$ and $u_{i,t+1}$ via (1.39).
10: end for
```

The procedure to choose a subset of kernels at each time instant for function approximation is summarized in Algorithm 5. This algorithm is called OMKL-SFG which stands for Online Multi Kernel Learning with Similarity based Feedback Graph. Figure 1.2b illustrates the case when v_2 is drawn by the Algorithm 5 as an example. Then the function approximation is performed using kernels associated with out-neighbors of v_2, which are v_1, v_2 and v_3 highlighted in Figure 1.2b.

The following Theorem presents the upper bound for cumulative stochastic regret of OMKL-SFG.

Theorem 1.2. *Under (A1) and (A2), let* $j^* = \arg\min_{\forall j:1\leq j\leq N} \sum_{t=1}^{T} \mathcal{L}(f_j^*(\boldsymbol{x}_t), y_t)$. *Then for any* $i \in \mathbb{N}_{j^*}^{in}$, *the stochastic regret of OMKL-SFG satisfies*

$$\sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(\hat{f}_{RF}(\boldsymbol{x}_t), y_t)] - \sum_{t=1}^{T} \mathcal{L}(f^*(\boldsymbol{x}_t), y_t)$$
$$\leq \frac{\ln N|\mathbb{N}_i^{out}|}{\eta} + \frac{(1+\epsilon)C^2}{2\eta} + \epsilon LTC + (\xi + \frac{\eta N}{2} - \frac{\eta\xi}{2})T + \frac{\eta}{2}\sum_{t=1}^{T}(\frac{1}{\bar{q}_{i,t}} + \frac{L^2}{q_{j^*,t}}) \tag{1.40}$$

with probability at least $1 - 2^8(\frac{\sigma_{j^*}}{\epsilon})^2 \exp(-\frac{D\epsilon^2}{4d+8})$ *under (A1)-(A3) for any* $\epsilon > 0$. *Furthermore,* $\frac{1}{\bar{q}_{i,t}} = \sum_{j\in\mathbb{N}_i^{out}} \frac{w_{j,t}}{q_{j,t}W_{i,t}}$, *C is a constant and* $\sigma_{j^*}^2$ *is the second moment of* $\pi_{\kappa_{j^*}}(\boldsymbol{\psi})$.

Proof. The proof is deferred to Appendix A.5. □

The regret bound in (1.40) depends on $\frac{1}{\bar{q}_{i,t}}$ and $\frac{1}{q_{j^*,t}}$. Since, there is a self-loop for all $v_k \in \mathcal{V}$, it can be written that $q_{k,t} \geq p_{k,t}$. In addition, based on (1.33), we can conclude that $p_{k,t} > \frac{\xi}{|\mathbb{D}|}$, $\forall v_k \in \mathcal{V}$ and as a result $q_{k,t} > \frac{\xi}{|\mathbb{D}|}$, $\forall v_k \in \mathcal{V}$. Therefore, in the worst case where $q_{j^*,t} = \mathcal{O}(\frac{\xi}{|\mathbb{D}|})$, considering

$$\eta = \mathcal{O}\left(\sqrt{\frac{\ln N}{N}} T^{-\frac{2}{3}}\right), \epsilon = \xi = \mathcal{O}\left(T^{-\frac{1}{3}}\right), D = \mathcal{O}\left(T^{\frac{2}{3}} \ln T\right) \tag{1.41}$$

OMKL-SFG can achieve regret bound of $\mathcal{O}(\sqrt{N \ln N} T^{\frac{2}{3}})$ with high probability of $1 - \mathcal{O}(T^{-\frac{1}{3}})$. Moreover, comparing regret bound of OMKL-SFG with that of OMKL-GF, it can be observed that OMKL-SFG obtains tighter regret than OMKL-GF. The reason behind this is that the regret bound of both OMKL-SFG and OMKL-GF depend on $1/q_{j^*,t}$ (c.f. (1.40) and (A.40)) and OMKL-SFG performs more exploration than OMKL-GF in the sense that using OMKL-SFG the lower bound for the probability $q_{i,t}$, $\forall i$ is larger than that of OMKL-GF. Specifically, using OMKL-GF, $q_{i,t} > \eta_e^2/NJ$ (c.f. (A.41)). Setting $\eta_e = \mathcal{O}(N^{\frac{1}{6}} T^{-\frac{1}{4}})$ and $J = \mathcal{O}(N^{\frac{1}{3}})$ as it is specified in (1.24), it can be concluded that

$$q_{i,t} > \mathcal{O}\left(\frac{1}{N\sqrt{T}}\right), \forall i \in \{1, \ldots, N\}, \forall t \in \{1, \ldots, T\} \tag{1.42}$$

when OMKL-GF is employed. Moreover, since using OMKL-SFG, $q_{i,t} > \frac{\xi}{|\mathbb{D}|}$, choosing $\xi = \mathcal{O}(T^{-\frac{1}{3}})$ as in (1.41) and considering the fact that $|\mathbb{D}| \leq N$, the lower bound of $q_{i,t}$ when OMKL-SFG is employed is obtained as

$$q_{i,t} > \mathcal{O}\left(\frac{1}{NT^{\frac{1}{3}}}\right), \forall i \in \{1, \ldots, N\}, \forall t \in \{1, \ldots, T\}. \tag{1.43}$$

Comparing (1.42) with (1.43), it can be concluded that OMKL-SFG can provide larger lower bound for $q_{j^*,t}$ than that of OMKL-GF. This enables OMKL-SFG to obtain tighter regret upper bound than OMKL-GF. Since the regret bound of $\mathcal{O}(\sqrt{T})$ is more satisfactory than the regret bound of $\mathcal{O}(T^{\frac{2}{3}})$, in what follows the structure of the feedback graph is refined at each time instant so that the regret of $\mathcal{O}(\sqrt{T})$ can be achieved.

1.4.3 OMKL with Similarity-based Feedback Graph Refinement

This subsection further improves the OMKL-SFG by refining the offline feedback graph 'on the fly', so that the resulting algorithm achieves a tighter sub-linear regret of $\mathcal{O}(\sqrt{T})$. To this end, at each time instant the offline feedback graph $\mathcal{G}$ constructed by the Algorithm 4 is refined to a feedback graph $\mathcal{G}'_t$ based on the observed losses. To begin with, let's define set $\mathbb{D}'_t$ as

$$\mathbb{D}'_t := \left\{ i \middle| \frac{u_{i,t}}{U_t} \geq \frac{1}{1-\xi}(\beta - \frac{\xi}{N}) \right\} \tag{1.44}$$

where β is a pre-selected constant. According to (1.33), it can be inferred that $p_{i,t} \geq \beta, \forall i \in \mathbb{D}'_t$. Let $\mathcal{G}'_t = (\mathcal{V}, \mathcal{E}'_t)$ be a graph such that $\mathbb{D}'_t$ is a dominating set of $\mathcal{G}'_t$. Suppose at each time instant t, $\mathcal{G}'_t$ is employed as the feedback graph instead of $\mathcal{G}$. In this case, it is ensured that there is at least one edge from $\mathbb{D}'_t$ to each $v_i \in \mathcal{V} \setminus \mathbb{D}'_t$, i.e., $\mathbb{D}'_t$ is a dominating set of $\mathcal{G}'_t$. In this case, we have $q_{i,t} \geq \beta$, $\forall v_i \in \mathcal{V}$. To this end, at each time instant t, $\mathcal{G}'_t$ can be constructed based on $\mathcal{G}$ by expanding $\mathcal{E}$ to $\mathcal{E}'_t$ such that $\mathbb{D}'_t$ would be a dominating set of $\mathcal{G}'_t$. Specifically, at each time instant t, the edge $(d_{i,t}, i)$ is appended to $\mathcal{E}'_t$, if there is not any edge from $\mathbb{D}'_t$ to v_i, where

$$d_{i,t} = \arg\max_{j \in \mathbb{D}'_t} \Delta(\kappa_i, \kappa_j). \tag{1.45}$$

Algorithm 6 OMKL with Similarity Feedback Graph Refinement (OMKL-SFG-R)

```
1:  Input: Shift-invariant kernels $\kappa_i$, $i = 1, \ldots, N$, learning rate $\eta > 0$, exploration rate
    $\xi > 0$, the number of random features $D$ and the constant $\beta > 0$.
2:  Initialize: $\boldsymbol{\theta}_{i,1} = \mathbf{0}$, $w_{i,1} = 1$, $i = 1, \ldots, N$, Construct the feedback graph $\mathcal{G}$ via
    Algorithm 4.
3:  for $t = 1, \ldots, T$ do
4:      Receive one datum $\boldsymbol{x}_t$.
5:      Set $\mathcal{E}'_t = \mathcal{E}$ and obtain $d_{i,t}$, $\forall i \in \mathcal{V} \setminus \mathbb{D}'_t$ by (1.45).
6:      For all $i \in \mathcal{V} \setminus \mathbb{D}'_t$, append $(d_{i,t}, i)$ to $\mathcal{E}'_t$ if $(d_{i,t}, i) \notin \mathcal{E}$.
7:      Draw one of nodes $v_i \in \mathcal{V}$ according to the PMF $p_t = (p_{1,t}, ..., p_{N,t})$ in (1.46).
8:      Predict $\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t)$ via (1.47).
9:      Calculate loss $\mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t)$ for all $i \in \mathbb{S}_t$.
10:     Update $\boldsymbol{\theta}_{i,t+1}$ via (1.38).
11:     Update $w_{i,t+1}$ and $u_{i,t+1}$ via (1.39).
12: end for
```

Hence, there is at least one edge from $\mathbb{D}'_t$ to $v_i \in \mathcal{V} \setminus \mathbb{D}'_t$, meaning $\mathbb{D}'_t$ is a dominating set for $\mathcal{G}'_t$. Then one of the vertices in $\mathcal{V}$ is drawn according to the PMF p_t, with

$$p_{i,t} = (1-\xi)\frac{u_{i,t}}{U_t} + \frac{\xi}{|\mathbb{D}'_t|}\mathbf{1}_{\mathrm{i}\in\mathbb{D}'_{\mathrm{t}}}, i = 1, \ldots, N. \tag{1.46}$$

Let $\mathbb{N}^{\mathrm{out}}_{i,t}$ and $\mathbb{N}^{\mathrm{in}}_{i,t}$ denote sets of out-neighbors and in-neighbors of v_i in $\mathcal{G}'_t$, respectively. According to (1.44) and (1.46), we have $q_{i,t} \geq \beta$, $\forall v_i \in \mathcal{V}$ where $q_{i,t} = \sum_{j\in\mathbb{N}^{\mathrm{in}}_{i,t}} p_{j,t}$. The RF-based function approximation can be written as

$$\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t) = \sum_{i\in\mathbb{N}^{\mathrm{out}}_{I_t}} \frac{w_{i,t}}{\sum_{j\in\mathbb{N}^{\mathrm{out}}_{I_t}} w_{j,t}} \hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t). \tag{1.47}$$

According to (1.47), $\boldsymbol{\theta}_{i,t}$, $w_{i,t}$ and $u_{i,t}$ can be updated using (1.38), (1.39a) and (1.39b), respectively. The procedure is summarized in Algorithm 6, which is called OMKL-SFG-R, and its performance in terms of regret analysis is presented in the following Theorem.

Theorem 1.3. *Under (A1)–(A3), the stochastic regret of OMKL-SFG-R satisfies*

$$\begin{aligned}&\sum_{t=1}^{T}\mathbb{E}_t[\mathcal{L}(\hat{f}_{RF}(\boldsymbol{x}_t),y_t)]-\sum_{t=1}^{T}\mathcal{L}(f^*(\boldsymbol{x}_t),y_t)\\ \leq&\frac{2\ln N}{\eta}+\frac{(1+\epsilon)C^2}{2\eta}+\epsilon LTC+(\xi+\frac{\eta}{2}\frac{L^2+N\beta+1}{\beta}-\frac{\eta\xi}{2})T\end{aligned} \tag{1.48}$$

with probability at least $1-2^8(\frac{\sigma_{j^*}}{\epsilon})^2\exp(-\frac{D\epsilon^2}{4d+8})$ *for any* $\epsilon>0$ *and any* $\beta\leq\frac{1}{N}$.

Proof. The proof can be found in Appendix A.6. □

According to Theorem 1.3, by setting

$$\eta=\mathcal{O}\left(\sqrt{\frac{\ln N}{NT}}\right),\epsilon=\xi=\mathcal{O}\left(\sqrt{\frac{1}{T}}\right),D=\mathcal{O}(T\ln T) \tag{1.49}$$

and $\beta=\mathcal{O}(1)$ such that $\beta\leq\frac{1}{N}$, OMKL-SFG-R can achieve regret bound of $\mathcal{O}(\sqrt{TN\ln N})$ using the feedback graph $\mathcal{G}'_t$ with probability of $1-\mathcal{O}(1)$. According to Theorem 1.3, larger D leads to larger probability that the regret bound in (1.48) holds true. Thus, in order to achieve regret of $\mathcal{O}(\sqrt{TN\ln N})$ with high probability, OMKL-SFG-R should set sufficiently large value of order $\mathcal{O}(T\ln T)$ for D. However, note that since some edges may be added to $\mathcal{G}$ to construct $\mathcal{G}'_t$, using $\mathcal{G}'_t$ instead of $\mathcal{G}$ may cause increase in computational complexity of function approximation.

Computational Complexity. Both OMKL-SFG and OMKL-SFG-R need to store a set of d-dimensional vectors $\{\boldsymbol{\psi}_{i,j}\}_{j=1}^{D}$ per kernel in addition to two weighting coefficients $\{w_{i,t}\}_{i=1}^{N}$ and $\{u_{i,t}\}_{i=1}^{N}$. Furthermore, both OMKL-SFG and OMKL-SFG-R need to store adjacency matrix of $\mathcal{G}$ which is $N\times N$ matrix. In order to perform required computation for (1.45), OMKL-SFG-R needs to store the divergence $\Delta(\kappa_i,\kappa_j)$ between any pair of kernels in the dictionary. Therefore, the memory requirement for both algorithms are $\mathcal{O}(dDN+N^2)$. Consider the case where the number of out-neighbors of each node v_i in $\mathcal{G}$ satisfies $M<N$, the

per-iteration complexity of OMKL-SFG including calculation of inner products is $\mathcal{O}(dDM)$. Therefore, it can be inferred that OMKL-SFG incurs less computational complexity than OMKL-GF since recall that the per-iteration complexity of OMKL-GF is $\mathcal{O}(dDM + JN)$. Therefore, it can be concluded that the offline feedback graph construction indeed can alleviate the computational complexity compared with OMKL-GF in section 1.3. Due to the graph refinement procedure, the complexity of OMKL-SFG-R is higher than OMKL-SFG. According to Algorithm 6, there is a possibility that all nodes in the graph are out-neighbors of one node in $\mathbb{D}'_t$. Therefore, the worst-case per-iteration computational complexity of OMKL-SFG-R is $\mathcal{O}(dDN)$. Furthermore, the computational complexities of OMKL-SFG and OMKL-SFG-R are lower than state-or-art multi-kernel learning algorithms provided that the per-iteration computational complexity of OMKR is $\mathcal{O}(tdN)$ [131], and the per-iteration computational complexity of RF-based online multi-kernel learning algorithms proposed in [132] and [137] are $\mathcal{O}(dDN)$.

Regret Bounds Comparison. The algorithm OMKR [131] achieves regret of $\mathcal{O}(\sqrt{T \ln N})$. However, since per iteration computational complexity of OMKR is $\mathcal{O}(tdN)$, OMKR requires much more computations than the proposed graph-aided OMKL algorithms. Furthermore, Raker [137] obtains regret of $\mathcal{O}(\sqrt{T \ln N})$ with high probability when the number of random features D is set to $\mathcal{O}(T \ln T)$. Therefore, employing all kernels in the dictionary, Raker obtains tighter regret bound than those of the proposed graph-aided OMKL algorithms. However, the proposed graph-aided OMKL algorithms require less computations than Raker.

Comparison with Online Learning. In online learning with expert advice, there is a learner interacts with a set of experts such that at each round of learning the learner choose one of the experts and takes its advice for either decision making or prediction [19, 8]. The learner may observe the loss associated with a subset of experts after decision making and in this regard this can be modeled by a graph which is called feedback graph (see e.g. [106, 29, 4]). In all algorithms proposed in this chapter, each kernel can be viewed as an expert. However, there are two major innovative differences compared with online learning with feedback graph: i) the proposed algorithm constructs and refines the feedback graph to improve the performance of learning task while in online learning, the feedback graph is generated in an adversarial manner. ii) in this chapter, each expert (kernel) is a learner itself and experts implement an online scheme for self-improvement.

1.5 Experiments

This section presents experimental results over real datasets downloaded from UCI Machine Learning Repository [44]. The accuracy of different approaches are evaluated using mean square error (MSE). Due to the randomness in the random features extracted for function approximation, we average the MSE over $R = 20$ different sets of random features. The MSE at time t is computed as

$$\mathrm{MSE}_t = \frac{1}{R}\sum_{r=1}^{R}\frac{1}{t}\sum_{\tau=1}^{t}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_\tau) - y_\tau)^2. \tag{1.50}$$

The number of random features is $D = 50$. The kernel dictionary contains 76 kernels including 51 RBF kernels and 25 Laplacian kernels. The bandwidth of the i-th ($1 \leq i \leq 51$) RBF kernel is 10^{σ_i} with $\sigma_i = \frac{2i-52}{25}$. And the value of the i-th ($1 \leq i \leq 25$) Laplacian kernel's parameter is 10^{λ_i} where $\lambda_i = \frac{i-13}{6}$. For fairness of evaluation, parameters ξ, η and η_e are set to $\frac{0.1}{\sqrt{t}}$ for all algorithms at time step t. More precise results can be obtained with more extensive parameter tuning. The performance of kernel learning algorithms is evaluated through several real datasets:

Airfoil: This dataset comprises $1,503$ different size airfoils at various wind tunnel speeds and each data sample $\boldsymbol{x}_t$ includes 5 features such as frequency and chord length. The output y_t is scaled sound pressure level in decibels [14].

Bias: This dataset contains $7,750$ samples. Each sample has 21 features including maximum and minimum temperatures of present-day, and geographic auxiliary variables for the purpose of bias correction of next-day minimum air temperatures. The goal is to predict next-day minimum air temperature [25].

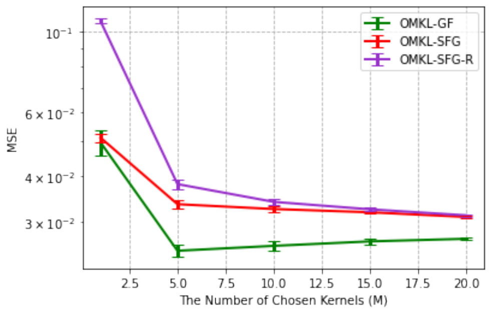
Concrete: This dataset contains $1,030$ samples of 8 features, such as the amount of cement or water in a concrete. The goal is to predict concrete compressive strength [156].

Naval: contains $11,934$ samples of 15 features of a naval vessel characterized by a gas turbine propulsion plant including the ship speed and gas turbine shaft torque. The goal is to predict the lever position [31].

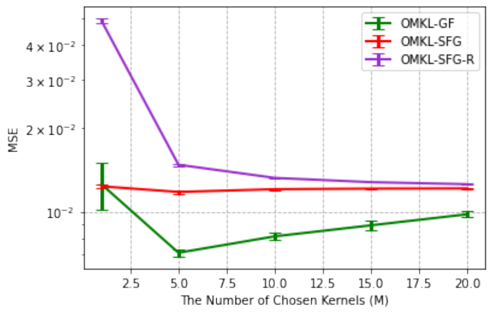
Parameter λ is set to 10^{-3} for all proposed algorithms OMKL-GF, OMKL-SFG and OMKL-SFG-R. Generating the bipartite graph $\mathcal{B}_t$ at every time instant can increase the computational complexity of the proposed OMKL-GF while it cannot improve MSE considerably. To further decrease the computational complexity of our proposed OMKL-GF, we terminate generating $\mathcal{B}_t$ after 300 time instants, meaning that $\mathcal{B}_t = \mathcal{B}_{300}$ if $t > 300$. The number of selective nodes for OMKL-GF is set to be 2. Furthermore, the feedback graph $\mathcal{G}$ in Algorithm 4 is generated with the divergence function $\Delta(\cdot,\cdot)$ defined using Bregman divergence in (1.25) when $F(\boldsymbol{\omega}) = \|\omega\|^2$. The greedy set cover algorithm by [28] is utilized to find the dominating set $\mathbb{D}$ of the feedback graph $\mathcal{G}$. Moreover, in order to speed up OMKL-SFG and OMKL-SFG-R, after 300 time instants, OMKL-SFG-R chooses $I_t = \arg\max_i \frac{u_{i,t}}{U_t}$. All experiments were carried out using Intel(R) Core(TM) i7-10510U CPU @ 1.80 GHz 2.30 GHz processor with a 64-bit Windows operating system. Codes are available at `https://github.com/pouyamghari/Graph-Aided-Online-Multi-Kernel-Learning`.

1.5.1 Number of Selected Kernels

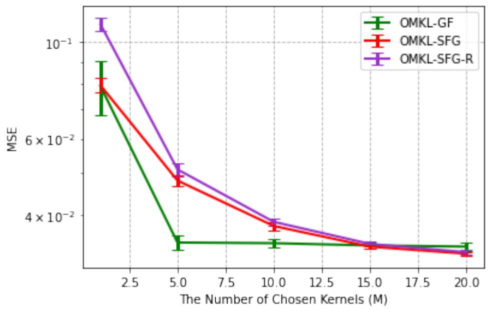
The present subsection studies the effect of the maximum number of selected kernels M on the performance of the proposed OMKL-GF, OMKL-SFG and OMKL-SFG-R. Figure 1.3 illustrates the MSE and its standard deviation of the proposed OMKL-GF, OMKL-SFG and OMKL-SFG-R with the change in the number of selected kernels. The standard deviation and the MSE are obtained over $R = 20$ sets of i.i.d random features (c.f. (1.50)). Figure 1.3 shows the advantage of data-driven kernel selection by OMKL-GF since increasing M from $M = 10$ to $M = 20$ does not result in lower MSE for Airfoil, Bias and Naval datasets. Figure 1.3 indicates that for OMKL-SFG and OMKL-SFG-R choosing moderate value for M such as $M = 10$ obtains MSE comparatively close to the MSE associated with optimal M. Figure 3 illustrates that OMKL-GF can obtain lower MSE than OMKL-SFG although OMKL-SFG achieves tighter regret upper bound than that of OMKL-GF. Regret upper bounds deal with worst cases. According to Theorems 1.1 and 1.2, the worst cases for OMKL-GF and OMKL-SFG happen when the probability of observing the loss of the best kernel (i.e. $q_{j^*,t}$ where j^* defined in Theorem 1.2) is close to its minimum value for almost all t. For both algorithms $q_{j^*,t}$ is close to its minimum value for almost all t if the probability of choosing the best kernel for function approximation is small for all t, which is unlikely to happen although it is not impossible. In addition, both OMKL-GF and OMKL-SFG choose a subset of kernels using a trade-off between exploitation and exploration. OMKL-SFG performs more exploration in choosing a subset of kernels than OMKL-GF in the sense that using OMKL-SFG lower bound for $q_{j^*,t}$ is larger than that of OMKL-GF (see (1.42) and (1.43)). OMKL-GF performs more exploitation than OMKL-SFG since using OMKL-GF the structure of the graph is changing every time instant to enable OMKL-GF to choose optimal subset of kernels while using OMKL-SFG the structure of the graph is fixed and independent of prior loss observations. Results in Figure 1.3 show that the proper selection of M along with data-driven kernel selection can enable OMKL-GF to choose a more powerful subset of



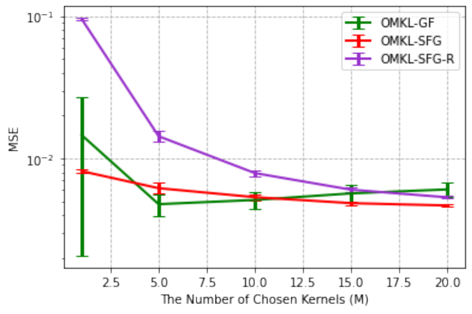
(a) Airfoil dataset.



(b) Bias dataset.



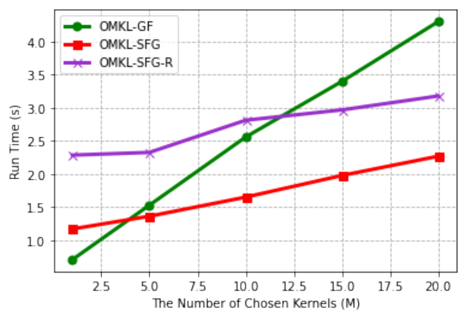
(c) Concrete dataset.



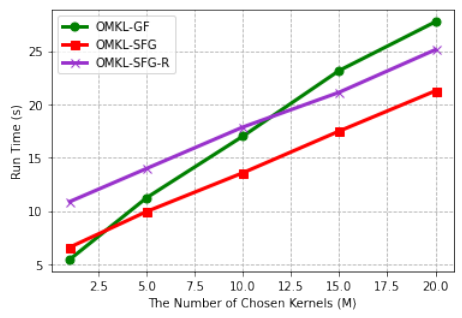
(d) Naval dataset.

Figure 1.3: MSE and standard deviation performance of Graph-aided MKL algorithms on real datasets with the change in the number of selected kernels.

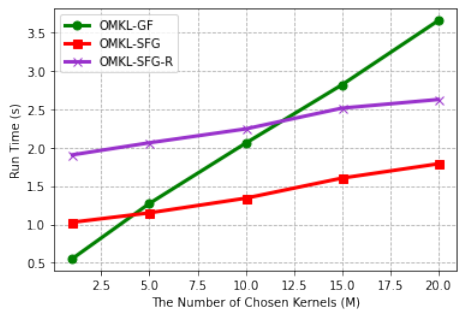
kernels than that of OMKL-SFG which leads to better accuracy of OMKL-GF. Moreover, since using OMKL-SFG-R, $q_{i,t} \geq \beta$, choosing $\beta = \mathcal{O}(1)$, it can be concluded that $q_{j^*,t} \geq \mathcal{O}(1)$ when OMKL-SFG-R is employed. Therefore, OMKL-SFG-R performs more exploration than OMKL-SFG which leads to tighter regret upper bound for OMKL-SFG-R. However, employing OMKL-SFG-R increases the probability that the kernels with comparatively poor prior performance being among the chosen subset of kernels. This is against exploitation and degrades the MSE performance of OMKL-SFG-R compared to OMKL-SFG. Furthermore, Figure 1.4 depicts that the run time of the proposed graph-aided OMKL algorithms with the change in the number of selected kernels. Figure 1.4 shows that a larger M leads to longer run time which is expected as larger M increases the complexity. Moreover, numerical results associated with Figures 1.3 and 1.4 are also presented in Tables A.1, A.2, A.3 in Appendix A.1.



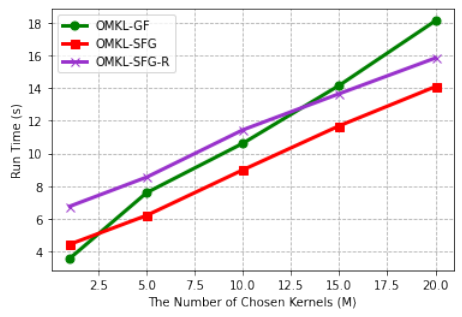
(a) Airfoil dataset.



(b) Bias dataset.



(c) Concrete dataset.



(d) Naval dataset.

Figure 1.4: Run Time performance of Graph-aided MKL algorithms on real datasets with the change in the number of selected kernels.

1.5.2 MSE and Run Time Performance Compared to Baselines

The performance of the proposed algorithms OMKL-GF, OMKL-SFG and OMKL-SFG-R are compared with the following kernel learning benchmarks:

- **OMKR**: online multi-kernel regression approach proposed by [131].
- **RBF-1**: online single kernel regression approach [131] using a radial basis function (RBF) with bandwidth of 1.
- **POLY-2**: online single kernel regression approach [131] using a polynomial kernel with degree of 2.
- **RFOMKR**: online multi-kernel learning approach utilizes RF approximation [132].
- **Raker**: RF-based online multi-kernel learning [137].

Table 1.1: MSE($\times 10^{-3}$) of MKL algorithms on real datasets.

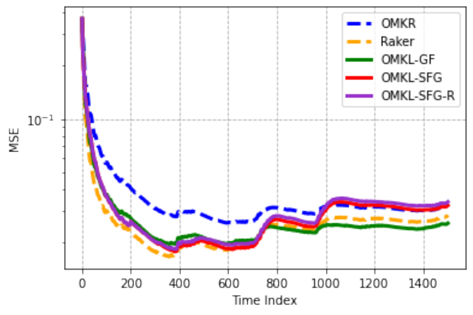
Algorithms	Airfoil	Bias	Concrete	Naval
OMKR	32.68	15.54	41.72	9.22
RBF-1	33.17	15.96	41.73	11.99
POLY-2	355.67	23.36	50.06	90.06
RFOMKR	350.52	405.44	210.42	347.06
Raker	28.64	12.70	35.22	11.35
OMKL-GF	**25.73**	**8.15**	**34.45**	**5.11**
OMKL-SFG	32.49	12.06	37.75	5.35
OMKL-SFG-R	33.99	13.24	38.58	7.89

Table 1.2: Run time(s) of MKL algorithms on real datasets.

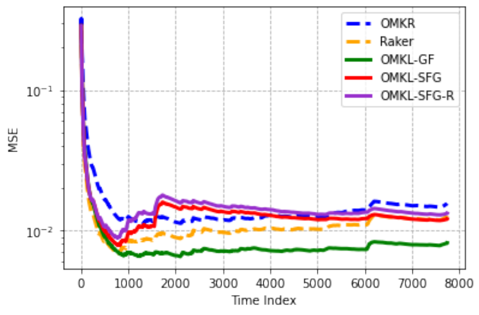
Algorithms	Airfoil	Bias	Concrete	Naval
OMKR	889.36	80010.03	547.93	163688.82
RBF-1	14.39	3914.67	6.60	1499.78
POLY-2	7.22	195.19	3.33	951.93
RFOMKR	2.17	27.72	1.53	18.37
Raker	3.81	46.28	2.71	31.24
OMKL-GF	2.56	17.01	2.06	10.63
OMKL-SFG	**1.65**	**13.58**	**1.34**	**9.00**
OMKL-SFG-R	2.81	17.86	2.24	11.45

The maximum number of kernels chosen by OMKL-GF at each time instant is 10. In addition, for OMKL-SFG, to determine the value of γ_i for each vertex $v_i \in \mathcal{V}$, the number of out-neighbors for each node is set to be 10. For OMKL-SFG-R, at each time, β is set to $\beta = (1-\xi)\bar{u}_{[10,t]} + \frac{\xi}{N}$ where $\bar{u}_{[10,t]}$ denote the tenth greatest value in the sequence $\{\frac{u_{i,t}}{U_t}\}_{i=1}^{N}$.

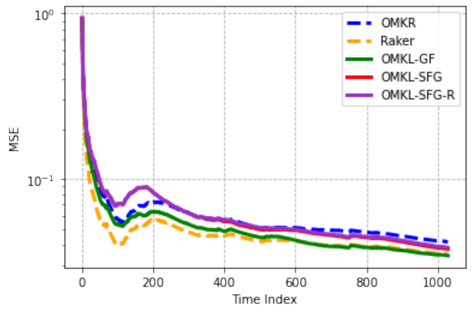
Tables 1.1 and Table 1.2 list MSE and run time performance of alternative algorithms on real datasets, respectively. It can be observed from Table 1.1 that the proposed OMKL-GF significantly outperforms all benchmark algorithms, which corroborate the effectiveness of data-driven feedback graph based kernel pruning. Furthermore, MSEs reported in Table 1.1 indicates that the accuracy of OMKL-SFG is comparable with that of Raker which employs all kernels in the dictionary while OMKL-SFG chooses a subset of kernels at each time



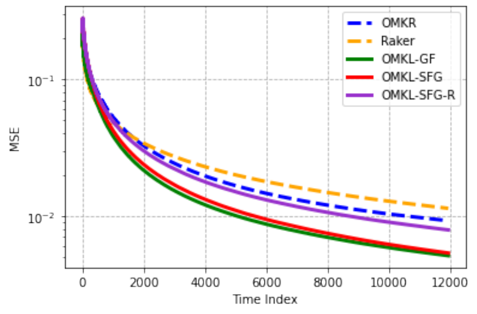
(a) Airfoil dataset.



(b) Bias dataset.



(c) Concrete dataset.

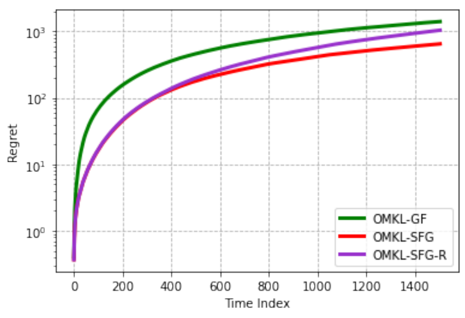


(d) Naval dataset.

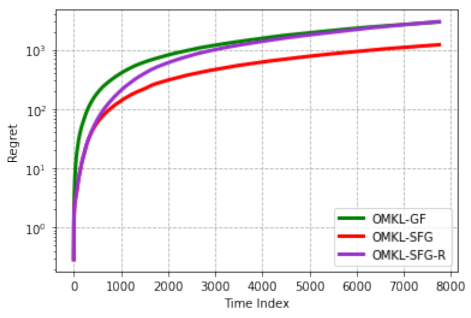
Figure 1.5: MSE performance of MKL algorithms on real datasets.

instant. Table 1.2 shows that OMKL-GF and OMKL-SFG are more efficient than all other alternatives, since thanks to the graph-aided pruning only a subset of kernels instead of all kernels in the dictionary are employed at each time instant. In addition, OMKL-SFG is the fastest due to the offline graph construction. Although OMKL-SFG-R enjoys tighter sub-linear regret than those of OMKL-GF and OMKL-SFG by including a larger number kernels in the selected subset, employing OMKL-SFG-R requires more computation and increases the run time which can be inferred from the results in the Table 1.2.

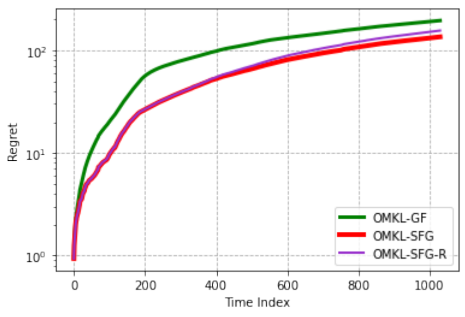
Figure 1.5 illustrates the MSE of OMKR, Raker and proposed algorithms over time. It can be seen that as time goes, performance gain of OMKL-GF becomes more remarkable. This confirms the effectiveness of the data-driven kernel selection in a sense that the proposed OMKL-GF learns the optimal subset of kernels in the dictionary 'on the fly'.



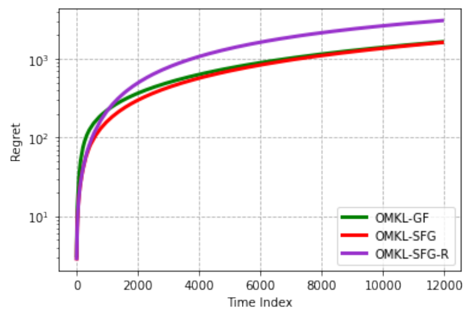
(a) Airfoil dataset.



(b) Bias dataset.



(c) Concrete dataset.



(d) Naval dataset.

Figure 1.6: Regret of proposed OMKL algorithms on real datasets.

1.5.3 Regret Performance

The present section presents the regret performance of the proposed OMKL-GF, OMKL-SFG and OMKL-SFG-R. The maximum number of kernels chosen by OMKL-GF at each time instant is 10. In addition, using OMKL-SFG, at each time instant 10 kernels are chosen to perform the function approximation task. For OMKL-SFG-R, at each time, β is set to $\beta = (1-\xi)\bar{u}_{[10,t]} + \frac{\xi}{N}$. Figure 1.6 illustrates the regret of the proposed algorithms over time.

1.6 Conclusion

This chapter develops online multi-kernel learning algorithms for non-linear function learning. By constructing a bipartite feedback graph at every time instant, OMKL-GF chooses a subset of kernels to both prune irrelevant kernels and decrease the computational complexity. It is proved that OMKL-GF can obtain regret of $\mathcal{O}(T^{\frac{3}{4}})$. To further alleviate the computational burden of multi-kernel learning, a feedback graph is constructed in an offline fashion based on the similarities among kernels. Using the similarity-based feedback graph, a subset of kernels is chosen and the resulting algorithm is called OMKL-SFG. It is proved that OMKL-SFG can achieve sub-linear regret of $\mathcal{O}(T^{\frac{2}{3}})$. Furthermore, refining the similarity-based feedback graph structure at each time instant, OMKL-SFG-R is proposed, which enjoys sub-linear regret of $\mathcal{O}(\sqrt{T})$. Moreover, experiments on real datasets demonstrate that by choosing a subset of kernels, OMKL-GF can obtain lower MSE in comparison with other online kernel learning algorithms including OMKR and Raker. Furthermore, experiments show that OMKL-GF and OMKL-SFG have considerably lower run time compared to online multi-kernel learning algorithms OMKR and Raker.

Chapter 2

Online Learning with Uncertain Feedback Graphs

2.1 Introduction

Online learning with expert advice considers the case where there exists a learner and a set of experts, where the learner interacts with the experts to make a decision [19]. At each time instant, the learner chooses one of the experts and it takes the action advised by the chosen expert, then incurs the loss associated with the taken action. Such framework can be used to model different learning tasks such as online multi-kernel learning see e.g., [137]. Conventional online learning literature mostly focuses on two settings, *full information* setting [101, 18, 74, 127] or *bandit* setting [8, 68, 127, 153]. In the full information setting, at each time instant, the learner can observe the loss associated with all experts. By contrast, in the bandit setting, the learner can only observe the loss associated with the chosen expert. However, in some applications such as the web advertising problem, where a user clicks on an ad and information about other related ads is revealed, the learner can make partial observations of losses associated with a subset of experts. In cases where querying for advice from expert incurs cost, the learner may choose to observe the loss of subset of experts, see e.g. [5, 57]. To cope with this scenario, *online learning with feedback graphs* was developed in [106], where partial observations of losses are modeled using a directed *feedback graph*. Each node represents an expert, and an edge from node i to node j exists if the learner can observe the loss associated with expert j while choosing expert i. The observations of losses associated with other experts are called side observations. The full information and the bandit settings are both special cases of online learning with either a fully connected feedback graph or a feedback graph with only self loops. Given the feedback graph either before or after decision making, [4] has proposed algorithms with sub-linear regret bounds. Online learning with feedback graphs and sleeping experts has been studied in [33] where at each time instant, a subset of experts may not be available. [34] has studied the case where there is a dependency between the feedback graph and expert losses. Moreover, [68] has proposed an algorithm for bandit setting which obtains sub-linear regret with respect to the best switching expert selection strategy.

Most of existing works rely on the assumption that the feedback graph in known *perfectly* before decision making [3, 4, 102, 33, 6], or after decision making [4, 87, 88, 126, 34]. However, such information may not be available in practice. In addition, due to possible uncertainty of the environment, the feedback graph may be uncertain. As an example, consider an online clothing store that offers discount on an item for new customers. Suppose there are two brands A and B producing similar shirts at comparable price. The store has small and medium sizes of brand A and medium and large sizes shirts of brand B in stock. Assuming that the store offers discount on brand B. If the user accepts the offer, and buys a medium size shirt of brand B, it implies the user is also interested in shirts of brand A. Moreover, if the user buys a large size of shirt B, this indicates no interest in shirts of brand A. Otherwise, if the user declines the offer of brand B, it only shows the user is not interested in shirts of brand B but no information is available about the preference of the user on the shirts of brand A. Considering the case where the exact feedback graph may not be available, [29] shows that not knowing the entire feedback graph can make the side observations useless and the learner may simply ignore them. [86] studies the case where the exact feedback graph is unknown but is known to be generated from the Erdös-Rényi model. However, such assumption may not be valid in practice. In addition, both [29] and [86] assume that the loss associated with the chosen expert is guaranteed to be observed. Moreover, the probabilistic feedback graph in stochastic setting has been studied in [95] where the loss of each expert randomly generated using a certain probability distribution.

The present chapter extensively studies the case where the learner only has access to a feedback graph that may contain uncertainties, namely *nominal feedback graph*, and the learner may not be able to observe the loss associated with the chosen expert. Moreover, the present chapter studies non-stochastic adversarial online learning problems where at each time instant, the environment privately selects a loss function. The learner relies on the nominal feedback graph to choose among experts, and then incurs a loss associated with the chosen expert. At the same time, it observes the loss associated with a subset of experts resulting from the unknown actual feedback graph. Furthermore, different from [29] and [86], the present work does not assume that it is guaranteed that the learner observes the loss associated with the chosen expert. This is true in, e.g., apple tasting problem [75]. The present work studies various cases of potential uncertainties, and develops novel online learning algorithms to cope with different uncertainties in the nominal feedback graph. Regret analysis is provided to prove that our novel algorithms can achieve sublinear regret under mild conditions. Experiments on a number of real datasets are presented to showcase the effectiveness of our novel algorithms. It is also worth noting that the materials in this chapter are fully included in [64] and partially included in [60].

2.2 Problem Statement

Consider the case where there exist K experts and the learner chooses to take the advice of one of the experts at each time instant t. Let $\mathcal{G}_t = (\mathcal{V}, \mathcal{E}_t)$ represent the directed nominal feedback graph at time t with a set of vertices $\mathcal{V}$, where the vertex $v_i \in \mathcal{V}$ represents the i-th expert, and there exist an edge from v_i to v_j (i.e. $(i, j) \in \mathcal{E}_t$), if the learner observes the loss associated with the j-th expert (i.e. $\ell_t(v_j)$) with probability p_{ij} while choosing the i-th expert. Let $\mathcal{N}_{i,t}^{\text{in}}$ and $\mathcal{N}_{i,t}^{\text{out}}$ represent in-neighborhood and out-neighborhood of v_i in $\mathcal{G}_t$, respectively. Thus, $v_j \in \mathcal{N}_{i,t}^{\text{out}}$ if there is an edge from v_i to v_j at time t (i.e. $(i, j) \in \mathcal{E}_t$). Similarly, $v_j \in \mathcal{N}_{i,t}^{\text{in}}$ if there is an edge from v_j to v_i at time t (i.e. $(j, i) \in \mathcal{E}_t$). The present chapter considers non-stochastic adversarial online learning problems. At each time instant t, the environment privately selects a loss function $\ell_t(.)$ with $\ell_t(.) : \mathcal{V} \rightarrow [0, 1]$, and the nominal feedback graph $\mathcal{G}_t$ is revealed to the learner before decision making. The learner then chooses one of the experts to take its advice. Then, the learner will incur the loss associated with the chosen expert. Let I_t denote the index of the chosen expert. Note that the learner observes $\ell_t(v_{I_t})$ with probability of $p_{I_t I_t}$, hence the loss remains unknown with the probability of $1 - p_{I_t I_t}$.

The present chapter discusses different potential uncertainties in the feedback graphs, and develops novel algorithms for online learning with uncertain feedback graph. Specifically, two cases are discussed: i) *online learning with informative probabilistic feedback graph*: where the probability p_{ij} associated with each edge is given along with the nominal feedback graph $\mathcal{G}_t$; and ii) *online learning with uninformative probabilistic feedback graph*: where only the nominal feedback graph $\mathcal{G}_t$ is revealed, but not the probabilities.

2.3 Online Learning with Informative Probabilistic Feedback Graphs

First consider the case where $\{p_{ij}\}$ are given along with the $\mathcal{G}_t$. This can be the case in various applications. For instance, consider a network of agents in a wireless sensor network that cooperate with each other on certain tasks such as environmental monitoring. Online learning algorithms distributed over spatial locations have been employed in climate informatics field [17, 111]. Assume that each agent in the network keeps updating its local model, and there is a central unit (learner) wishes to perform a learning task based on models and data samples distributed among agents. In this case, the agents in the network can be viewed as experts. Consider the case where the learner chooses one of the experts and sends a request for the corresponding expert advice through a wireless link. Subset of experts which receive the request, send their advice to the learner. However, due to uncertainty in the environment or power limitation, some of the agents in the network including the chosen one may not detect the request. Therefore, the learner can only observe the advice of subset of agents in the network which detect its request. In this case, the learner can model probable advice that it can receive from experts with a nominal feedback graph. If learner knows the characteristics of the environment which is true in many wireless communication applications, the probabilities associated with edges in the nominal feedback graph is revealed.

At each time instant t, upon selecting an expert and observing the losses of a subset of experts, the weights $\{w_{i,t}\}_{i=1}^{K}$ which indicate the reliability of experts can be updated as follows

$$w_{i,t+1} = w_{i,t} \exp\left(-\eta \hat{\ell}_t(v_i)\right), \quad \forall i \in [K] \tag{2.1}$$

where $[K] := \{1, \ldots, K\}$ and η is the learning rate. Function $\hat{\ell}_t(v_i)$ denotes the importance

Algorithm 7 Exp3-IP: Online learning with informative probabilistic feedback graph

```
1: Input:learning rate $\eta > 0$.
2: Initialize: $w_{i,1} = 1$, $\forall i \in [K]$.
3: for $t = 1, \dots, T$ do
4:     Observe $\mathcal{G}_t = (\mathcal{V}, \mathcal{E}_t)$ and choose one of the experts according to the PMF $\pi_t$ in (2.4).
5:     Observe $\{\ell_t(v_i)\}_{v_i \in \mathcal{S}_t}$ and calculate loss estimate $\hat{\ell}_t(v_i)$, $\forall i \in [K]$ via (2.2).
6:     Update $w_{i,t+1}$, $\forall i \in [K]$ via (2.1).
7: end for
```

sampling loss estimate which can be obtained as

$$\hat{\ell}_t(v_i) = \frac{\ell_t(v_i)}{q_{i,t}} \mathcal{I}(v_i \in \mathcal{S}_t) \tag{2.2}$$

where $\mathcal{S}_t$ represent the set of vertices associated with experts whose losses are observed by the learner at time instant t. The indicator function is denoted by $\mathcal{I}(.)$ and $q_{i,t}$ is the probability that the loss $\ell_t(v_i)$ is observed. Its value depends on the algorithm, and will be specified later.

Let A_t denote the adjacency matrix of the nominal feedback graph $\mathcal{G}_t$ with $A_t(i,j)$ denoting the (i,j)th entry of A_t. Let X_{ij} be a Bernoulli random process with random variables $X_{ij}(t) = 1$ with probability p_{ij}. When the learner chooses the i-th expert at time t, the learner observes $\ell_t(v_j)$ only if $v_j \in \mathcal{N}_{i,t}^{\text{out}}$ and $X_{ij}(t) = 1$. Let F_t denote the number of losses observed by the learner. Due to the stochastic nature of the observations available to the learner, F_t is a random variable. Furthermore, let $F_{i,t}$ denote the expected number of observed losses if the learner chooses the i-th expert at time t. Thus, we can write

$$F_{i,t} = \mathbb{E}_t[F_t | I_t = i, A_t] = \sum_{\forall j : v_j \in \mathcal{N}_{i,t}^{\text{out}}} \mathbb{E}[X_{ij}(t)] = \sum_{\forall j : v_j \in \mathcal{N}_{i,t}^{\text{out}}} p_{ij}. \tag{2.3}$$

The learner then chooses one expert according to the probability mass function (PMF)

$\pi_t := (\pi_{1,t}, \ldots, \pi_{K,t})$ with

$$\pi_{i,t} = (1-\eta)\frac{w_{i,t}}{W_t} + \eta\frac{F_{i,t}}{\sum_{j\in\mathcal{D}_t} F_{j,t}}\mathcal{I}(v_i \in \mathcal{D}_t) \tag{2.4}$$

where $W_t := \sum_{i=1}^{K} w_{i,t}$, and $\mathcal{D}_t$ denotes the dominating set of graph $\mathcal{G}_t$. Note that a dominating set $\mathcal{D}$ of a graph is a subset of vertices such that there is an edge from at least one vertex in $\mathcal{D}$ to any vertex not in $\mathcal{D}$. It can be observed from (2.4) that η controls the trade-off between exploitation and exploration. With a smaller η, more emphasis is placed on the first term which promotes exploitation, and the learner tends to choose the expert with larger $w_{i,t}$. The second term allows the learner to select experts in the dominating set $\mathcal{D}_t$ with certain probability independent of their performance in previous rounds. Based on (2.4), $q_{i,t}$ in (2.2) can be computed as

$$q_{i,t} = \sum_{\forall j: v_j \in \mathcal{N}_{i,t}^{\text{in}}} \pi_{j,t} p_{ji}. \tag{2.5}$$

The overall algorithm for online learning with uncertain feedback graph in the informative probabilistic setting, termed Exp3-IP, is summarized in Algorithm 7. In order to analyze the performance of Algorithm 7, as well as the ensuing algorithms, we first preset two assumptions needed:

(A1) $0 \le \ell_t(v_i) \le 1, \forall t : t \in \{1, \ldots, T\}, \forall i : i \in \{1, \ldots, K\}$.

(A2) If $(i,j) \in \mathcal{E}_t$, the learner can observe the loss associated with the j-th expert with probability at least $\epsilon > 0$ when it chooses the i-th expert, and $(i,i) \in \mathcal{E}_t,\ \forall i$.

Note that (A1) is a general assumption in online learning literature e.g., [3]. And (A2) assumes a nonzero probability of observing (but not guaranteed observation of) the loss associated with the chosen expert $\ell_t(v_{I_t})$. The following theorem presents the regret bound for Exp3-IP.

Theorem 2.1. *Under (A1), the expected regret of Exp3-IP can be bounded by*

$$\sum_{t=1}^{T}\mathbb{E}_t[\ell_t(v_{I_t})]-\min_{v_i\in\mathcal{V}}\sum_{t=1}^{T}\ell_t(v_i)\le\frac{\ln K}{\eta}+\eta(1-\frac{\eta}{2})T+\frac{\eta}{2}\sum_{t=1}^{T}\sum_{i=1}^{K}\frac{\pi_{i,t}}{q_{i,t}}. \tag{2.6}$$

Proof of Theorem 2.1 is included in Appendix B.1. It can be seen from Theorem 2.1 that the value of $\pi_{i,t}/q_{i,t}$ plays an important role in regret bound. Choosing an expert using (2.4), it is ensured that every vertex in $\mathcal{D}_t$ is chosen by the learner with non-zero probability. Moreover, since there is at least one edge from a node in $\mathcal{D}_t$ to any node not in $\mathcal{D}_t$, under (A2), the probability $q_{i,t}$, $\forall i$ is non-zero. Lower bounding $q_{i,t}$, (A2) enables Exp3-IP to achieve sub-linear regret. Building upon Theorem 2.1, the ensuing lemma further explores under which circumstances Exp3-IP can achieve sub-linear regret bound.

Lemma 2.1. *Let the doubling trick (see e.g. [4]) be employed to determine the value of η and greedy set cover algorithm (see e.g. [28]) is exploited to derive a dominating set $\mathcal{D}_t$ for the nominal feedback graph $\mathcal{G}_t$. Under (A1) and (A2), the expected regret of Exp3-IP satisfies*

$$\sum_{t=1}^{T}\mathbb{E}_t[\ell_t(v_{I_t})]-\min_{v_i\in\mathcal{V}}\sum_{t=1}^{T}\ell_t(v_i)\le\mathcal{O}\left(\sqrt{\ln K\ln(\frac{K}{\epsilon}T)\sum_{t=1}^{T}\frac{\alpha(\mathcal{G}_t)}{\epsilon}}+\ln(\frac{K}{\epsilon}T)\right) \tag{2.7}$$

where $\alpha(\mathcal{G}_t)$ denotes the independence number of the nominal feedback graph $\mathcal{G}_t$.

Proof of Lemma 2.1 is included in Appendix B.2. As it is proved in Appendix B.2, the assumption $(i,i)\in\mathcal{E}_t,\ \forall i$ in (A2) guarantees that $\sum_{i=1}^{K}\frac{\pi_{i,t}}{q_{i,t}}\le\mathcal{O}\left(\frac{\alpha(\mathcal{G}_t)}{\epsilon}\ln(\frac{KT}{\epsilon})\right)$ (see Lemma B.1 and (B.25)–(B.28) in Appendix B.2). In order to guarantee the regret bound in (2.7), it is required that $\sum_{i=1}^{K}\frac{\pi_{i,t}}{q_{i,t}}\le\mathcal{O}\left(\frac{\alpha(\mathcal{G}_t)}{\epsilon}\ln(\frac{KT}{\epsilon})\right)$ holds true. Therefore, without (A2), the regret bound in (2.7) cannot be satisfied. Furthermore, if the learner does not know the time horizon T before start decision making, doubling trick can be exploited to determine η. In particular, using the doubling trick, Exp3-IP adjusts the learning rate η 'on the fly' without knowing the time horizon T. At time instant t, as long as

$$\sum_{\tau=1}^{t}(1+\frac{1}{2}\sum_{i=1}^{K}\frac{\pi_{i,\tau}}{q_{i,\tau}}) \leq 2^{r_t} \tag{2.8}$$

holds true, Exp3-IP employs learning rate $\eta = \sqrt{\frac{\ln K}{2^{r_t+1}}}$, where $r_t \geq 0$ is the smallest integer that can satisfy the inequality in (2.8). According to (2.7), Exp3-IP can achieve sub-linear regret. Furthermore, (2.7) shows that the regret bound of Exp3-IP depends on $\frac{1}{\epsilon}$. Larger ϵ indicates that the learner is less uncertain about the nominal feedback graph. In other words higher confidence of the nominal feedback graph leads to a tighter regret bound.

Comparison with [4]. Exp3-DOM of [4] deals with the cases that the feedback graph is certain and revealed to the learner before decision making at each time instant. In this case, Exp3-DOM achieves regret of $\mathcal{O}\left(\ln(K)\sqrt{\ln(KT)\sum_{t=1}^{T}\alpha(\mathcal{G}_t)}+\ln(K)\ln(KT)\right)$ (see Theorem 8 in [4]). When the graph is certain such that $p_{ij}=1$ for all $(i,j)\in\mathcal{E}$, then $\epsilon=1$. Therefore, when the graph is certain and given to the learner, the proposed Exp3-IP achieves regret of $\mathcal{O}\left(\sqrt{\ln K\ln(KT)\sum_{t=1}^{T}\alpha(\mathcal{G}_t)}+\ln(KT)\right)$.

2.4 Online Learning with Uninformative Probabilistic Feedback Graphs

The previous section deals with the case where the nominal feedback graph $\mathcal{G}_t$ can be time-variant and probabilities associated with edges of $\mathcal{G}_t$ are revealed. In this section, we will study the scenario where the nominal feedback graph $\mathcal{G}_t$ is static and is revealed to the learner while the probabilities $\{p_{ij}\}$ associated with edges are not given, which is called *uninformative probabilistic feedback graph.* In this section the nominal feedback graph is denoted by $\mathcal{G}=(\mathcal{V},\mathcal{E})$. In this case, estimates of probabilities $\{p_{ij}\}$ can be updated and employed to assist the learner with future decision making. For example, consider the problem of online advertisement, where a website is trying to decide which product to be advertised via online survey with a multiple choice question. Specifically, users are asked whether they are interested in certain product along with possible reasons (cost, color, etc). Note that the answer to certain product may also indicate the participant's potential interest in other products with similar cost or color. For instance, if the participant indicates that he or she is interested in the product because of its affordable cost, this implies *potential* interest in other products with the same or lower price. In this case, the relationship among products can be modeled by a nominal feedback graph, where an edge exists between two nodes (products) if they share same or similar attributes (cost, color), which implies that users *may be* interested in both products. Such nominal feedback graph can then be used to assist the website to make a decision on which product to advertise . However, the actual relationship between the the user's interests in the products remains uncertain, which leads to uncertainty in the nominal feedback graph. Since attributes (cost, color, etc) of products do not change over time, the nominal feedback graph is static, while the probabilities associated with edges in the nominal feedback graph are unknown. Faced with this practical challenge, two approaches will be developed in this section, to estimate either the unknown probability or the importance sampling loss in (2.2), which will then be employed to assist the learner's decision making.

2.4.1 Estimation-based Approach

In the present subsection, we will further explore the general scenario where the value of p_{ij} may vary across edges, while the nominal feedback graph $\mathcal{G}_t$ is static. Since X_{ij} defined under (2.2) is a mean ergodic random process [120] in this scenario, the sample mean of $\{X_{ij}(t)\}$ converges to p_{ij}, i.e., the expected value of $X_{ij}(t)$. Let $\mathcal{T}_{ij,t}$ represent a set collecting time instants before t when the learner chooses to take the advice of the i-th expert and there is an edge between v_i and v_j in the nominal feedback graph $\mathcal{G}$. In other word, $\mathcal{T}_{ij,t}$ can be defined as

$$\mathcal{T}_{ij,t} = \{\tau | A_\tau(i,j) = 1, I_\tau = i, 0 < \tau < t\}. \tag{2.9}$$

Based on the above discussion, p_{ij} can be estimated as

$$\hat{p}_{ij,t} = \frac{1}{C_{ij,t}} \sum_{\tau \in \mathcal{T}_{ij,t}} X_{ij}(\tau) \tag{2.10}$$

where $C_{ij,t} := |\mathcal{T}_{ij,t}|$ is the cardinality of $\mathcal{T}_{ij,t}$. Since X_{ij} is a mean ergodic Bernoulli random process, $\hat{p}_{ij,t}$ is an unbiased maximum likelihood (ML) estimator of p_{ij}.

Note that a sufficient number of observations of the random process X_{ij} is needed, in order to provide a reliable estimation in (2.10). To this end, the learner performs exploration in the first KM time instants to ensure that $C_{ij,t} \geq M$, $\forall(i,j) \in \mathcal{E}_t$, where the value of M is determined by the learner. Specifically, in the first KM time instants, the learner chooses all experts in $\mathcal{V}$, one by one M times, i.e. the learner selects expert v_k, with $k = t - \left\lfloor \frac{t}{K} \right\rfloor K$ when $t \leq KM$. For $t > KM$, the learner draws one of the experts according to the following PMF

$$\pi_{i,t} = (1-\eta)\frac{w_{i,t}}{W_t} + \frac{\eta}{|\mathcal{D}|}\mathcal{I}(v_i \in \mathcal{D}), \forall i \in [K] \tag{2.11}$$

where $\mathcal{D}$ denotes a dominating set for the nominal feedback graph $\mathcal{G}$. In order to obtain a reliable loss estimate to assist the learner's decision making, we will approximate the importance sampling loss estimate in (2.2) using the estimated probability $\hat{p}_{ij,t}$. In this context, the probability of observing $\ell_t(v_i)$ can be approximated as

$$\hat{q}_{i,t} = \sum_{\forall j : v_j \in \mathcal{N}_{i,t}^{\text{in}}} \pi_{j,t}(\hat{p}_{ji,t} + \frac{\xi}{\sqrt{M}}) \tag{2.12}$$

where $\xi \geq 1$ is a parameter selected by the learner. Then the importance sampling loss

Algorithm 8 Exp3-UP: Online learning with uninformative probabilistic feedback graphs

```
1: Input: learning rate $\eta > 0$, the minimum number of observations $M$, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.
2: Initialize: $w_{i,1} = 1$, $\forall i \in [K]$, $\hat{p}_{ij,1} = 0$, $\forall (i,j) \in \mathcal{E}$.
3: for $t = 1, \ldots, T$ do
4:     if $t \leq KM$ then
5:         Set $k = t - \lfloor \frac{t}{K} \rfloor K$ and draw the expert node $v_k$.
6:     else
7:         Select one of the experts according to the PMF $\pi_t = (\pi_{1,t}, \ldots, \pi_{K,t})$ , with $\pi_{i,t}$ in
           (2.11).
8:     end if
9:     Observe $\{(i, \ell_t(v_i)) : v_i \in \mathcal{S}_t\}$ and compute $\tilde{\ell}_t(v_i)$, $\forall i \in [K]$ as in (2.13).
10:    Update $\hat{p}_{ij,t+1}$, $\forall (i,j) \in \mathcal{E}_t$ via (2.10).
11:    Update $w_{i,t+1}$, $\forall i \in [K]$ via (2.14).
12: end for
```

estimates can be obtained as

$$\tilde{\ell}_t(v_i) = \frac{\ell_t(v_i)}{\hat{q}_{i,t}} \mathcal{I}(v_i \in \mathcal{S}_t). \tag{2.13}$$

With the estimates in hand, the weights $\{w_{i,t}\}_{i=1}^{K}$ can be updated as follows

$$w_{i,t+1} = w_{i,t} \exp\left(-\eta \tilde{\ell}_t(v_i)\right), \quad \forall i \in [K]. \tag{2.14}$$

The procedure that the learner chooses among experts when the probabilities are unknown is presented in Algorithm 8, named Exp3-UP. The following theorem establishes the regret bound of Exp3-UP.

Theorem 2.2. *Under (A1), the expected regret of Exp3-UP satisfies*

$$\begin{aligned}&\sum_{t=1}^{T} \mathbb{E}_t[\ell_t(v_{I_t})] - \min_{v_i \in \mathcal{V}} \sum_{t=1}^{T} \ell_t(v_i)\\ \leq&\frac{\ln K}{\eta} + (K-1)M + \eta(1 - \frac{\eta}{2})(T - KM) + \sum_{t=KM+1}^{T} \sum_{i=1}^{K} \frac{\pi_{i,t}}{q_{i,t}}(\frac{2\xi}{\sqrt{M}} + \frac{\eta}{2})\end{aligned} \tag{2.15}$$

with probability at least

$$\delta_\xi := \prod_{t=KM+1}^{T} \prod_{(i,j)\in\mathcal{E}_t} \left(1 - 2\exp(-\frac{2\xi^2 C_{ij,t}}{M + 4\xi\sqrt{M}})\right).$$

See proof of Theorem 2.2 in Appendix B.3. The following Corollary states conditions under which the regret bound in (2.15) holds with high probability, i.e., $\delta_\xi = 1 - \mathcal{O}(\frac{1}{T})$ and the proof can be found in Appendix B.4.

Corollary 2.1. *If* $M \geq \left(\frac{4\xi\ln(KT)}{\xi^2 - \ln(KT)}\right)^2$ *and* $\xi > \sqrt{\ln(KT)}$*, under (A1) and (A2) the expected regret of Exp3-UP satisfies*

$$\sum_{t=1}^{T} \mathbb{E}_t[\ell_t(v_{I_t})] - \min_{v_i\in\mathcal{V}} \sum_{t=1}^{T} \ell_t(v_i) \leq \mathcal{O}\left(\frac{\alpha(\mathcal{G})}{\epsilon}\ln(KT)\sqrt{K\ln(KT)}T^{\frac{2}{3}}\right) \tag{2.16}$$

with probability at least $1 - \mathcal{O}(\frac{1}{T})$.

Note that according to Algorithm 8 and Corollary 2.1, knowing the value of the time horizon T is required so that the learner can choose the values for M and ξ to achieve the sublinear regret bound in (2.16), which may not be feasible, and can be resolved by resorting to doubling trick. In this case, if $2^b < t \leq 2^{b+1}$ where $b \in \mathbb{N}$, the learner performs the Exp3-UP with parameters

$$\eta = \sqrt{\frac{\ln K}{2^{b+1}}} \tag{2.17a}$$

$$M = \left\lceil 2^{\frac{2(b+1)}{3}} \frac{1}{\sqrt{K}} + \ln 4K \right\rceil \tag{2.17b}$$

$$\xi = \left(2K^{\frac{1}{4}} + \sqrt{4\sqrt{K} + 1}\right)\sqrt{\ln(K2^{b+3})}. \tag{2.17c}$$

When the learner realizes that the value of M needs to be increased, it then performs exploration to guarantee that at least M samples of the mean ergodic random process X_{ij} are observed. The following lemma shows that when doubling trick is employed, Exp3-UP can achieve sub-linear regret without knowing the time horizon beforehand, the proof of which is in Appendix B.5.

Lemma 2.2. *Assuming that the doubling trick is employed to determine the value of η, M and ξ at each time instant and the greedy set cover algorithm is utilized to obtain a dominating set $\mathcal{D}$ of the nominal feedback graph. If $T > K$, the regret of Exp3-UP satisfies*

$$\sum_{t=1}^{T} \mathbb{E}_t[\ell_t(v_{I_t})] - \min_{v_i \in \mathcal{V}} \sum_{t=1}^{T} \ell_t(v_i) \leq \mathcal{O}\left(\frac{\alpha(\mathcal{G})}{\epsilon} \ln(T)\ln(KT)\sqrt{K\ln(KT)}T^{\frac{2}{3}} + \ln T\right) \quad (2.18)$$

with probability at least $1 - \mathcal{O}(\frac{1}{K})$.

2.4.2 Geometric Resampling-based Approach

Another approach to obtain a reliable loss estimate is to employ geometric resampling. Similar to Exp3-UP, if $t \leq KM$ the learner chooses the k-th expert at time instant t where $k = t - \lfloor t/K \rfloor K$. In this way, it is guaranteed that at least M samples of the mean ergodic random process X_{ij} are observed. Based on these observations, a loss estimate is obtained whose expected value is an approximation of the loss $\ell_t(v_i)$, $\forall i \in [K]$. At $t > KM$, the learner draws one of the experts according to the following PMF

$$\pi_{i,t} = (1-\eta)\frac{w_{i,t}}{W_t} + \frac{\eta}{|\mathcal{D}|}\mathcal{I}(v_i \in \mathcal{D}), \quad \forall i \in [K] \quad (2.19)$$

where $\mathcal{D}$ represents a dominating set for $\mathcal{G}$. Furthermore, at each time instant $t > KM$, let $\tau_{i,1}^{(t)}, \ldots, \tau_{i,M}^{(t)}$ denote the last M time instants before t at which the i-th expert was chosen by the learner. If $(i,j) \in \mathcal{E}$, the learner observes $X_{ij}(\tau_{i,1}^{(t)}), \ldots, X_{ij}(\tau_{i,M}^{(t)})$ which are samples of the random process X_{ij} at $\tau_{i,1}^{(t)}, \ldots, \tau_{i,M}^{(t)}$. Let $Y_{ij,1}(t), \ldots, Y_{ij,M}(t)$ denote a random permutation of $X_{ij}(\tau_{i,1}^{(t)}), \ldots, X_{ij}(\tau_{i,M}^{(t)})$. At each time instant t, the learner draws with replacement M experts according to PMF $\{\pi_{i,t}\}$ in (2.19) in M independent trials. Let $P_{i,1}(t), \ldots, P_{i,M}(t)$

be a sequence of random variables associated with v_i at time instant t where $P_{i,u}(t) = 1$ if the learner draws the i-th expert at the u-th trial and $P_{i,u}(t) = 0$ otherwise. Let

$$Z_{i,u}(t) = \sum_{\forall j: v_j \in \mathcal{N}_{i,t}^{\text{in}}} P_{j,u}(t) Y_{ji,u}(t) \tag{2.20}$$

for all $1 \leq u \leq M$. An under-estimate of loss can then be obtained as

$$\tilde{\ell}_t(v_i) = Q_{i,t} \ell_t(v_i) \mathcal{I}(v_i \in \mathcal{S}_t). \tag{2.21}$$

where $Q_{i,t} := \min \{\{u \mid 1 \leq u \leq M, Z_{i,u}(t) = 1\} \cup \{M\}\}$, and the expected value of $\tilde{\ell}_t(v_i)$ can be written as

$$\mathbb{E}_t[\tilde{\ell}_t(v_i)] = \left(1 - (1 - q_{i,t})^M\right) \ell_t(v_i), \tag{2.22}$$

see (B.76) – (B.79) in Appendix B.6 for detailed derivation. Then, the weights $\{w_{i,t}\}_{i=1}^K$ are updated as in (2.14) using the loss estimate $\tilde{\ell}_t(v_i)$ in (2.21). The geometric resampling based online expert learning framework (Exp3-GR) is summarized in Algorithm 9, and Theorem 2.3 presents its regret bound.

Theorem 2.3. *Under (A1) and (A2), the expected regret of Exp3-GR is bounded by*

$$\begin{aligned}
&\sum_{t=1}^{T} \mathbb{E}_t[\ell_t(v_{I_t})] - \min_{v_i \in \mathcal{V}} \sum_{t=1}^{T} \ell_t(v_i) \\
\leq &\frac{\ln K}{\eta} + (K-1)M + \sum_{t=KM+1}^{T} (1 - q_{i,t})^M \\
&+ \eta(1-\eta)(T - KM) + \eta \sum_{t=KM+1}^{T} \sum_{i=1}^{K} \frac{\pi_{i,t}}{q_{i,t}}.
\end{aligned} \tag{2.23}$$

The proof of Theorem 2.3 is presented in Appendix B.6. Building upon Theorem 2.3, the following Corollary presents the conditions under which Exp3-GR can obtain sub-linear

Algorithm 9 Exp3-GR: Exp3 with geometric resampling

```
1: Input:learning rate $\eta > 0$, the minimum number of observations $M$, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.
2: Initialize: $w_{i,1} = 1$, $\forall i \in [K]$.
3: for $t = 1, \ldots, T$ do
4:    if $t \leq KM$ then
5:       Set $k = t - \lfloor \frac{t}{K} \rfloor K$ and draw the expert node $v_k$.
6:    else
7:       Select one expert according to PMF $\pi_t$ in (2.19).
8:       Observe $\{\ell_t(v_i) : v_i \in \mathcal{S}_t\}$ and compute $\tilde{\ell}_t(v_i)$, $\forall i \in [K]$ via (2.21).
9:       Update $w_{i,t+1}$, $\forall i \in [K]$ via (2.14).
10:   end if
11: end for
```

regret.

Corollary 2.2. *Assume that greedy set cover algorithm is employed to find a dominating set of the nominal feedback graph $\mathcal{G}$. If $M \geq \frac{|\mathcal{D}| \ln T}{2\eta\epsilon}$, under (A1) and (A2), Exp3-GR satisfies*

$$\sum_{t=1}^{T} \mathbb{E}_t[\ell_t(v_{I_t})] - \min_{v_i \in \mathcal{V}} \sum_{t=1}^{T} \ell_t(v_i) \leq \mathcal{O}\left(\frac{\alpha(\mathcal{G})}{\epsilon} \ln(KT)\sqrt{KT \ln K}\right). \tag{2.24}$$

Proof. According to (A2), if $(i, j) \in \mathcal{E}$, the learner observes the loss of the j-th expert when it chooses the i-th expert with probability at least ϵ. Recalling (2.19) it can be inferred that $\pi_{i,t} > \eta/|\mathcal{D}|$, $\forall i \in \mathcal{D}$. Combining (2.5) with the fact that for each $v_i \in \mathcal{V}$ there is at least one edge from $\mathcal{D}$ to v_i, $\forall i \in [K]$, $q_{i,t}$ can be bounded below as

$$q_{i,t} > \frac{\eta\epsilon}{|\mathcal{D}|}. \tag{2.25}$$

Combining the condition $M \geq \frac{|\mathcal{D}| \ln T}{2\eta\epsilon}$ with (2.25), we have $Mq_{i,t} \geq \frac{1}{2} \ln T$ which leads to $e^{-Mq_{i,t}} \leq \frac{1}{\sqrt{T}}$. Thus, using the fact $1 + x \leq e^x$, we have

$$(1 - q_{i,t})^M \leq e^{-Mq_{i,t}} \leq \frac{1}{\sqrt{T}}. \tag{2.26}$$

Hence, the third term in (2.23), i.e., $\sum_{t=t'}^{T} (1 - q_{i,t})^M$ can be bounded by $\mathcal{O}(\sqrt{T})$.

Furthermore, consider the case where we have $\eta = \mathcal{O}(\sqrt{\frac{K \ln K}{T}})$. Therefore, taking into account that greedy set cover algorithm is used to determine the dominating set, it can be inferred that $|\mathcal{D}| = \mathcal{O}(\alpha(\mathcal{G}) \ln K)$ (see e.g. [4]) based on which it can be obtained that $M = \mathcal{O}(\frac{\alpha(\mathcal{G})}{\epsilon\sqrt{K}} \ln T \sqrt{T \ln K})$, satisfies the condition $M \geq \frac{|\mathcal{D}| \ln T}{2\eta\epsilon}$. Hence, the expected regret of Exp3-GR satisfies (2.24), and the Corollary 2.2 is proved. □

Achieving the sub-linear regret in (2.24) requires that the learner knows the time horizon T, beforehand which may not be possible in some cases. When the learner does not know T, doubling trick can be utilized to achieve sub-linear regret. The following Lemma is proved in Appendix B.7, shows the regret bound for Exp3-GR when doubling trick is employed to find values of η and M without knowing the time horizon T. In this case, at time instant t, when $2^b < t \leq 2^{b+1}$, parameters η and M can be chosen as $\eta = \sqrt{\frac{K \ln K}{2^{b+1}}}$, $M = \left\lceil \frac{(b+1)\sqrt{2^{b-1}}|\mathcal{D}| \ln 2}{\epsilon\sqrt{K \ln K}} \right\rceil$. When the learner realizes that M needs to be increased, it performs exploration to guarantee that at least M samples of the mean ergodic random process X_{ij} are observed.

Lemma 2.3. *Employing doubling trick to select η and M at each time instant, and supposing that a dominating set for the nominal feedback graph $\mathcal{G}$ is obtained using greedy set cover algorithm, the expected regret of Exp3-GR satisfies*

$$\sum_{t=1}^{T} \mathbb{E}_t[\ell_t(v_{I_t})] - \min_{v_i \in \mathcal{V}} \sum_{t=1}^{T} \ell_t(v_i) \leq \mathcal{O}\left(\frac{\alpha(\mathcal{G}) \ln T}{\epsilon} \ln(KT)\sqrt{KT \ln K}\right). \tag{2.27}$$

Comparing Lemma 2.2 with Lemma 2.3, it can be observed that Exp3-GR achieves a tighter regret bound *with probability 1*. However, note that choosing an appropriate M for Exp3-GR requires knowing ϵ or a lower bound of ϵ, which may not be feasible in general, while such information is not required for Exp3-UP in order to guarantee the regret bound in (2.18).

Comparison with [86]. Note that while Exp3-GR and Exp3-Res proposed in [86] both employ the geometric resampling technique, there exist two major differences: i) Exp3-Res assumes the actual feedback graph is generated from Erdös-Rényi model, and the probabilities of the presence of edges are equal across all edges, while Exp3-GR considers the unequally probable case and does not assume that the probabilities of existence of all edges are equal; and ii) unlike Exp3-Res, Exp3-GR does not assume that the learner is guaranteed to observe the loss associated with the chosen expert. Furthermore, it is useful to compare the regret bound of Exp3-GR with that of Exp3-Res when the actual feedback graph is generated from the Erdös-Rényi model with $p_{ij} = p$, $\forall i, j \in [K]$. In this case, according to Corollary 2.2, Exp3-GR achieves regret of $\mathcal{O}\left(\frac{\ln(KT)}{p}\sqrt{KT \ln K}\right)$. On the other hand, under the assumption that $p \geq \frac{\ln T}{2K-2}$ and knowing that probabilities associated with all edges are equal, Exp3-Res obtains regret of $\mathcal{O}\left(\sqrt{K^2 \ln K + \frac{T \ln K}{p}}\right)$. Hence, having access to knowledge that the probabilities associated with all edges are equal enables Exp3-Res to achieve tighter regret bound than Exp3-GR in this special case.

Dependence of loss and feedback graph. The proposed algorithms Exp3-IP, Exp3-UP and Exp3-GR can also deal with cases where there is dependence between actual feedback graphs and losses. Consider the case that the environment generates $(\mathbf{x}_t, y_t)$ stochastically following certain time-invariant distribution. The i-th expert obtains the input $\mathbf{x}_t$ and outputs the prediction $\hat{y}_{i,t}$. In this case, the loss $\ell_t(v_i)$ can measure the discrepancy between $\hat{y}_{i,t}$ and y_t using some metrics such as squared loss. Furthermore, assume that the actual feedback graph at time t denoted by $\mathcal{H}_t$ depends on $\mathbf{x}_t$. In this case, if the learner knows the possible relations among experts, the learner can construct the nominal feedback graph $\mathcal{G}$ where the existence of each edge depends on $\mathbf{x}_t$. Therefore, the edge between two vertices exist with some time-invariant probability. Before decision making the learner is uncertain about $\mathbf{x}_t$ and as a result the learner can utilize one of the proposed algorithms Exp3-IP, Exp3-UP and Exp3-GR to decide based on the nominal feedback graph.

2.5 Experiments

Performance of the proposed algorithms Exp3-IP, Exp3-UP and Exp3-GR are compared with online learning algorithms Exp3 [8], Exp3.G [3], Exp3-Res [86], Exp3-SET [4] and Exp3-DOM [4]. Exp3 considers bandit setting, and Exp3-Res assumes Erdös-Rényi model for the feedback graph. Furthermore, Exp3.G and Exp3-DOM treats the nominal feedback graph $\mathcal{G}_t$ as the actual one without considering uncertainties. Exp3-SET observes the nominal feedback graph $\mathcal{G}_t$ and the loss of out-neighbors of the chosen expert after decision making. Exp3-SET treats connectivity information given by $\mathcal{G}_t$ associated with nodes other than the chosen one as certain information without considering the uncertainty. Note that Exp3-SET observers the actual feedback graph partially after decision making since Exp3-SET observes the loss of chosen experts' out-neighbors. Performance is tested for regression task on several real datasets downloaded from the UCI Machine Learning Repository [44]:

Air Quality: This dataset contains $9,358$ responses from sensors in a polluted area, each with 13 features. The goal is to predict polluting chemical concentration in the air [147].

CCPP: The dataset has $9,568$ samples, with 4 features such as temperature, collected from a combined cycle power plant. The goal is predicting hourly electrical energy output [146].

Twitter: This dataset contains $14,000$ samples with 77 features including e.g., the length of discussion on a given topic and the number of new interactive authors. The goal is to predict average number of active discussion on a certain topic [83].

Tom's Hardware: The dataset contains $10,000$ samples from a technology forum with 96 features. The goal is to predict the average number of display about a certain topic on Tom's hardware [83].

Let $(\mathbf{x}_i, y_i)$ and $(\bar{\mathbf{x}}_i, \bar{y}_i)$ be the i-th data sample and the normalized one, respectively. The data is normalized as $\bar{\mathbf{x}}_i = \frac{\mathbf{x}_i}{\max_j \|\mathbf{x}_j\|}$, $\bar{y}_i = \frac{y_i - \min_j y_j}{\max_j y_j - \min_j y_j}$. Therefore, $\|\bar{\mathbf{x}}_i\| \leq 1$, $0 \leq \bar{y}_i \leq 1$, $\forall i$. In the experiments, there are 9 experts such that each expert is a trained model. In particular, each expert is trained on 10% of each dataset before the start online learning task

Table 2.1: Cumulative Regret on various datasets and fully connected nominal feedback graph in equally probable setting.

	Air	CCPP	Twitter	Tom's
Exp3	47.56	152.34	71.03	45.39
Exp3.G	39.16	122.93	59.72	40.63
Exp3-Res	33.36	100.22	52.64	38.46
Exp3-SET	38.21	122.62	59.28	39.49
Exp3-DOM	39.04	122.16	61.30	41.10
Exp3-IP	**33.33**	**98.22**	**52.47**	**37.63**
Exp3-UP	35.97	109.87	56.86	38.68
Exp3-GR	33.60	**99.91**	53.33	**38.25**

associated with the corresponding dataset. Among them, 8 experts are trained via kernel ridge regression such that 5 experts exploit RBF kernels with bandwidth of $10^{-2}, 10^{-1}, 1, 10, 100$ while 3 experts employ Laplacian kernels with bandwidth of $10^{-2}, 1, 100$. Moreover, one expert is a trained linear regression model. Performance of algorithms are evaluated based on cumulative regret averaged over 20 independent runs. Recall that cumulative regret of an algorithm is the cumulative difference between the loss of the algorithm and that of the best expert in hindsight over time. In experiments, squared loss function is employed to measure the loss of experts. The learning rate η is set to $\frac{0.5}{\sqrt{T}}$ for all algorithms. Note that online learning algorithms may achieve better regret experimentally with carefully tuned learning rate. However, for fair comparison, the learning rates of all online learning algorithms are set to be the same. Parameter M is set as 25 for both Exp3-UP and Exp3-GR and $\xi = 1$ for Exp3-UP.

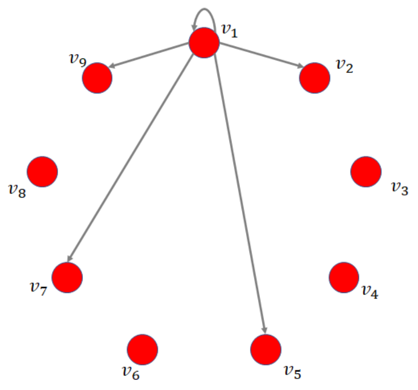
We first tested the equally probable setting where the nominal graph $\mathcal{G}_t$ is fully connected and probabilities $p_{ij} = 0.5$, $\forall i, j$. Table 2.1 lists the regret performance for various datasets. It can be observed that, knowing the exact probability enables Exp3-IP to achieve the lowest regret. Moreover, the proposed Exp3-UP and Exp3-GR obtain lower regret than Exp3.G, Exp3-SET and Exp3-DOM which treat the nominal feedback graph as actual one. Note that in this case, the actual feedback graph is indeed generated from the Erdös-Rényi model.

Table 2.2: Cumulative Regret on various datasets and partially-connected nominal feedback graph $\mathcal{P}$ in unequally probable setting.

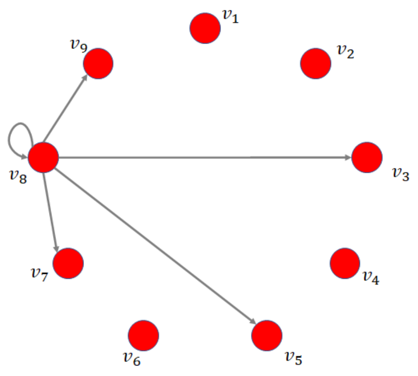
	Air	CCPP	Twitter	Tom's
Exp3	47.56	152.97	71.04	45.39
Exp3.G	41.19	128.74	62.21	41.53
Exp3-Res	35.83	109.62	56.55	39.51
Exp3-SET	40.21	129.32	62.22	40.77
Exp3-DOM	41.19	127.84	63.90	41.98
Exp3-IP	**32.95**	**97.37**	**52.23**	**37.19**
Exp3-UP	38.77	120.97	61.06	40.28
Exp3-GR	**33.60**	**99.59**	**53.04**	**38.01**

The regret of the proposed Exp3-GR is comparable to that of Exp3-Res while Exp3-Res makes decision under the assumption that the actual feedback graph is generated from the Erdös-Rényi model.

We further tested the unequally probable case, when the graph is partially connected. In particular, $v_j \in \mathcal{N}_{i,t}^{\text{out}}$ if j is either the remainder of $i-1$, i, $i+1$, $i+4$ and $i+6$ to 9. Note that if the remainder is zero, it is considered to be 9. The resulting nominal feedback graph in this case is represented by $\mathcal{P}$. Therefore, in the nominal feedback graph $\mathcal{P}$, each node has 5 out-neighbors. As an example, out-neighbors of v_1 and v_8 are illustrated in Figure 2.1. The probability associated with each edge is drawn from uniform distribution $\mathcal{U}[0.25, 0.5]$. Table 2.2 lists the cumulative regret of all algorithms for Air Quality, CCPP, Twitter and Tom's Hardware datasets. It can be observed that Exp3-IP obtains the lowest regret. This shows that knowing the probabilities can indeed help obtain better performance. Furthermore, it can be observed that Exp3-UP and Exp3-GR can achieve lower regret in comparison with Exp3 which shows the effectiveness of using the information given by the uncertain graph. In addition, lower regret of Exp3-UP and Exp3-GR compared to Exp3.G, Expe-SET and Exp3-DOM indicates that considering the uncertain graph $\mathcal{G}_t = \mathcal{P}$ as a certain graph can increase regret. Moreover, it can be observed Exp3-GR outperforms Exp3-Res when the actual feedback graph is not generated by Erdös-Rényi model. It can be observed



(a) Out-neighbors of v_1 in $\mathcal{P}$.



(b) Out-neighbors of v_8 in $\mathcal{P}$.

Figure 2.1: Out-neighbors of v_1 and v_8 are illustrated in partially-connected nominal feedback graph $\mathcal{P}$.

Table 2.3: Cumulative Regret on various datasets and partially-connected certain nominal feedback graph $\mathcal{P}$.

	Air	CCPP	Twitter	Tom's
Exp3	46.46	150.23	70.01	45.05
Exp3.G	32.86	96.80	50.52	36.87
Exp3-Res	32.10	98.47	50.68	37.22
Exp3-SET	31.93	97.31	50.24	36.29
Exp3-DOM	32.84	96.39	52.07	37.22
Exp3-IP	33.19	97.37	52.44	36.88
Exp3-UP	35.92	109.93	56.25	38.44
Exp3-GR	33.34	99.08	52.87	37.60

Exp3-IP achieves lower regret than Exp3-GR and Exp3-UP, since the learner has access to the probabilities, while Exp3-UP and Exp3-GR do not rely on such prior information.

In addition, we tested the performance of algorithms when the nominal feedback graph $\mathcal{P}$ is partially-connected, and the probability associated with each edge is 1. As it can be seen from Table 2.3, Exp3.G, Exp3-SET, Exp3-DOM and the proposed Exp3-IP which utilize the certain feedback graph obtain lower regret than those of Exp3-UP and Exp3-GR which treat the certain feedback graph as uncertain one. In fact, Exp3-UP and Exp3-GR do not know the probability associated with edges. Furthermore, the regret of Exp3-IP is comparable to Exp3.G, Exp3-SET and Exp3-DOM.

2.6 Conclusion

The present chapter studied the problem of online learning with *uncertain* feedback graphs, where potential uncertainties in the feedback graphs were modeled using probabilistic models. Novel algorithms were developed to exploit information revealed by the nominal feedback graph and different scenarios were discussed. Specifically, in the informative case, where the probabilities associated with edges are also revealed, Exp3-IP was developed. It is proved that Exp3-IP can achieve sublinear regret bound. Furthermore, Exp3-UP and Exp3-GR were developed for the uninformative case. It is proved that Exp3-GR can achieve tighter sublinear regret bound than that of Exp3-UP when the number of experts is negligible compared to time horizon, while Exp3-UP requires less prior information than Exp3-GR. Experiments on a number of real datasets were carried out to demonstrate that our novel algorithms can effectively address uncertainties in the feedback graph, and help enhance the learning ability of the learner.

Chapter 3

Personalized Online Federated Learning with Multiple Kernels

3.1 Introduction

Kernel learning exhibits well-documented performance in function approximation tasks, while providing theoretical guarantees associated with different performance metrics, see e.g. [148, 77, 130]. In some cases, a group of learners aims at collaborating to perform function approximation without revealing their data. To this end, federated learning has been emerged as a crucial learning paradigm by enabling a group of learners called clients to collaborate with each other by communicating with a central server to train a centralized model [110, 97, 46, 81]. Through this process, clients send model parameters and updates to the server without revealing their data. Upon receiving updates from clients, the server updates the model. Therefore, federated learning enables clients to perform kernel learning for function approximation. In this context, a server and clients collaborate with each other to learn the optimal kernel. Furthermore, in some practical cases, clients may need to perform the function approximation in an online fashion while they are collaborating with the server to learn the kernel. For example, consider the case where clients may not have enough memory to store data in batch. In addition, data samples may arrive in a sequential manner such that clients are not able to perform the function approximation in batch form. There are major challenges in performing online kernel learning in federated fashion that need to be addressed:

Communication Efficiency: Communication efficiency arises as a bottleneck in federated learning (see e.g. [89, 129, 71, 59]). Specifically, limited clients-to-server communication bandwidth restricts the number of parameters that can be sent from clients to the server.

Heterogeneous Data: The distribution of data observed by a client might be different from others (see e.g. [139, 99, 30]). Thus, the optimal kernel that is aimed to be learned is different across clients.

Computational Complexity: Clients should be able to perform function approximation fast enough in order to make a decision in real-time. Therefore, the computational complexity of kernel learning methods should be affordable for clients.

Conventional online kernel learning approaches (see e.g. [77, 131]) suffer from 'curse of dimensionality' [11] in the sense that the number of parameters that should be learned increases with the number of observed data. This can make employing conventional online kernel learning approaches infeasible to perform online federated kernel learning since clients may be required to send a large number of parameter updates to the server while the available clients-to-server communication bandwidth is not enough for sending such information. Approximating kernels by finite-dimensional feature representations (e.g. Nyström method [150] and random feature method of [123]) makes online kernel learning approaches scalable in the sense that the learner can choose the number of parameters that should be learned, independent of the number of observed data samples (see e.g. [104, 12, 162]). Employing finite-dimensional feature representations of kernels to perform online federated kernel learning, clients can choose the number of parameters that they should send to the server. Therefore, finite-dimensional kernel approximation can better cope with limited clients-to-server communication bandwidth compared to conventional kernel learning approaches. Random feature (RF) approximation [123] has been exploited to perform online federated kernel learning when a single pre-selected kernel function is employed [91, 66]. The choice of the kernel function greatly affects the performance of function approximation when it comes to exploiting only a single kernel function. Employing multiple kernels instead of a single pre-selected one, can lead to obtaining more accurate function approximation since multi-kernel learning (MKL) can learn combination of kernels [85]. Online federated MKL algorithms with theoretical guarantees called vM-KOFL and eM-KOFL have been proposed in [78]. However, eM-KOFL and vM-KOFL do not provide personalized MKL models for clients since they learn the same combination of kernels for all clients.

The present chapter proposes a novel **p**ersonalized **o**nline **f**ederated **MKL** algorithm called POF-MKL that provides a personalized MKL model for each client while it is ensured that the available clients-to-server communication bandwidth can afford communication cost of sending clients' updates to the server. In order to alleviate the communication cost of MKL, the propsoed POF-MKL employs RF approximation of kernels and at each time instant, each client chooses a subset of kernels to send their updates to the server instead of sending the updates of all kernels. The number of kernels in the chosen subset is selected such that the required bandwidth to send all clients' updates does not exceed the available clients-to-server communication bandwidth. Therefore, clients can send their updates to the server independent of the number of kernels in the dictionary and as a result a comparatively large dictionary of kernels can be considered to perform function approximation. Contributions of the present chapter can be summarized as follows:

- Leveraging the proposed POF-MKL, clients can update a subset of kernels' parameters which alleviates computational complexity and communication cost of sending updates to the server.

- Through theoretical analysis, it is proved that using the proposed POF-MKL, each client achieves sub-linear regret with respect to RF approximation of the best kernel in hindsight associated with the corresponding client data samples (c.f. Theorem 3.1). Moreover, it is guaranteed that the server achieves sub-linear regret with respect to the best function approximator (c.f. Theorem 3.2).
- Experiments on real datasets showcase the effectiveness of the proposed POF-MKL compared to existing online federated kernel learning algorithms.

It is worth noting that the materials in this chapter are included in [61].

3.2 Problem Statement and Preliminaries

This section discusses the problem of online federated learning. To make this chapter self-contained, we also review random feature-based online kernel learning, which is covered in greater detail in Section 1.2.

Let there be a set of K clients performing function approximation task in an online fashion. The k-th client's goal is to learn the function f using the stream of data samples $\{(\boldsymbol{x}_{k,t}, y_{k,t})\}_{t=1}^{T}$ such that $\boldsymbol{x}_{k,t} \in \mathbb{R}^d$ is the data sample observed by the k-th client at time t and $y_{k,t}$ is the label associated with $\boldsymbol{x}_{k,t}$. In the kernel learning context, the function f is assumed to belong to a reproducing kernel Hilbert space (RKHS). The present chapter studies the personalized federated supervised function approximation problem

$$\min_{f \in \mathbb{H}} \sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L}(f(\boldsymbol{x}_{k,t}), y_{k,t}) \tag{3.1}$$

where $\mathbb{H}$ represents the RKHS the function f belongs to and $\mathcal{L}(\cdot, \cdot)$ denotes the loss function which can be defined as

$$\mathcal{L}(f(\boldsymbol{x}), y) = \mathcal{C}(f(\boldsymbol{x}), y) + \lambda\Omega(\|f\|^2) \tag{3.2}$$

where $\mathcal{C}(\cdot, \cdot)$ is the cost function (e.g. least squares function for regression task), λ denotes the regularization coefficient and $\Omega(\cdot)$ represents a regularizer function to prevent over-fitting and control the model complexity. Let $\boldsymbol{\Theta}$ be the global parameters of the function f which are learned through collaboration of clients with the server while $\boldsymbol{w}_k$ be the personalized parameter of the function f learned locally by the k-th client. Thus, the goal is that the cumulative difference between $f(\boldsymbol{x}_{k,t}; \boldsymbol{\Theta}, \boldsymbol{w}_k)$ and $y_{k,t}$ over time is minimized. At each time instant t, upon observing the data sample $\boldsymbol{x}_{k,t}$, the k-th client make the prediction $f(\boldsymbol{x}_{k,t}; \boldsymbol{\Theta}, \boldsymbol{w}_k)$ and then observes the true label $y_{k,t}$. Therefore, the function approximation problem that the server aims at solving can be expressed as $\min_{\Theta} \sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L}(f(\boldsymbol{x}_{k,t}; \boldsymbol{\Theta}, \boldsymbol{w}_k), y_{k,t})$. Moreover, finding the local parameters $\boldsymbol{w}_k$ by the k-th client can be expressed as the optimization problem $\min_{\boldsymbol{w}_k} \sum_{t=1}^{T} \mathcal{L}(f(\boldsymbol{x}_{k,t}; \boldsymbol{\Theta}, \boldsymbol{w}_k), y_{k,t})$. In order to perform the function approximation task in an online fashion, the k-th client needs to perform the task with the values of $\boldsymbol{\Theta}$ and $\boldsymbol{w}_k$ at time t denoted by $\boldsymbol{\Theta}_t$ and $\boldsymbol{w}_{k,t}$, respectively. Thus, the values of function parameters $\boldsymbol{\Theta}$ and $\boldsymbol{w}_k$ should be updated 'on the fly'. Since the function $f(\cdot; \cdot, \cdot)$ belongs to a reproducing kernel Hilbert space (RKHS), based on the representer theorem [148], given data samples, the optimal solution for (3.1) can be obtained as

$$\hat{f}(\boldsymbol{x}) = \sum_{t=1}^{T} \sum_{k=1}^{K} \alpha_{k,t} \kappa(\boldsymbol{x}, \boldsymbol{x}_{k,t}) \tag{3.3}$$

where $\kappa(\cdot, \cdot)$ denotes symmetric positive definite kernel function such that $\kappa(\boldsymbol{x}, \boldsymbol{x}')$ measures the similarity between $\boldsymbol{x}$ and $\boldsymbol{x}'$. And $\alpha_{k,t}$ is an unknown coefficient associated with $\kappa(\boldsymbol{x}, \boldsymbol{x}_{k,t})$ which is required to be estimated. In this case, $\hat{f}(\cdot)$ in (3.3) belongs to the RKHS $\mathbb{H} := \{f(\cdot) | f(\boldsymbol{x}) = \sum_{t=1}^{\infty} \sum_{k=1}^{K} \alpha_{k,t} \kappa(\boldsymbol{x}, \boldsymbol{x}_{k,t})\}$ such that RKHS norm is defined as $\|f\|_{\mathbb{H}}^2 := \sum_t \sum_{t'} \alpha_t \alpha_{t'} \kappa(\boldsymbol{x}_t, \boldsymbol{x}_{t'})$. Furthermore, from (3.3), it can be inferred that

$\mathbf{\Theta}_t = [\alpha_{1,1}, \ldots, \alpha_{K,1}, \ldots, \alpha_{1,t}, \ldots, \alpha_{K,t}]$. Therefore, the number of coefficients $\{\alpha_{k,\tau}\}_{\tau=1}^{t}$, $\forall k$ that should be estimated to obtain $\hat{f}(\cdot)$ increases over time. This is known as *curse of dimensionality* [148] since the computational complexity of function approximation increases with time. This brings challenge for federated implementation of function approximation since dimension of updates that should be sent to the server by each client grows over time and when T is large, the available communication bandwidth may not be enough for clients to send their updates.

In order to deal with the increasing number of unknown coefficients, one can employ random Fourier approximation [123]. Assume that $\kappa(\cdot)$ is a shift-invariant kernel meaning that $\kappa(\boldsymbol{x}, \boldsymbol{x}') = \kappa(\boldsymbol{x} - \boldsymbol{x}')$. Let $\pi_\kappa(\boldsymbol{\rho})$ denotes the Fourier transform of $\kappa(\cdot)$. If the kernel function $\kappa(\cdot)$ is normalized such that $\kappa(\mathbf{0}) = 1$, then $\pi_\kappa(\boldsymbol{\rho})$ can be viewed as a probability density function (PDF) (see e.g. [123]). Let $\boldsymbol{\rho}_1, \ldots, \boldsymbol{\rho}_D$ be a set of D independent and identically distributed (i.i.d) vectors drawn from $\pi_\kappa(\cdot)$. Let the vector $\boldsymbol{z}(\boldsymbol{x})$ be defined as

$$\boldsymbol{z}(\boldsymbol{x}) = \frac{1}{\sqrt{D}}[\sin(\boldsymbol{\rho}_1^\top \boldsymbol{x}), \ldots, \sin(\boldsymbol{\rho}_D^\top \boldsymbol{x}), \cos(\boldsymbol{\rho}_1^\top \boldsymbol{x}), \ldots, \cos(\boldsymbol{\rho}_D^\top \boldsymbol{x})]. \tag{3.4}$$

Then, $\hat{\kappa}_r(\boldsymbol{x} - \boldsymbol{x}') = \boldsymbol{z}(\boldsymbol{x})^\top \boldsymbol{z}(\boldsymbol{x}')$ constitutes an unbiased estimator of $\kappa(\boldsymbol{x} - \boldsymbol{x}')$ and the random feature (RF) approximation of $\hat{f}(\boldsymbol{x})$ in (3.3) can be obtained as

$$\hat{f}_{\mathrm{RF}}(\boldsymbol{x}) = \sum_{t=1}^{T} \sum_{k=1}^{K} \alpha_{k,t} \boldsymbol{z}(\boldsymbol{x}_{k,t})^\top \boldsymbol{z}(\boldsymbol{x}) := \boldsymbol{\theta}^\top \boldsymbol{z}(\boldsymbol{x}) \tag{3.5}$$

where in this case $\boldsymbol{\theta} = \sum_{t=1}^{T} \sum_{k=1}^{K} \alpha_{k,t} \boldsymbol{z}(\boldsymbol{x}_{k,t})$. According to (3.4), $\boldsymbol{z}(\boldsymbol{x}_{k,t})$ is a $2D$ vector and as a result it can be concluded that $\boldsymbol{\theta}$ is a $2D$ vector as well. Therefore, using RF approximation, the vector $\boldsymbol{\theta}$ should be estimated whose dimension does not grow over time.

The performance of a kernel learning algorithm depends on the choice of the kernel. Thus, performing the function approximation using a pre-selected kernel requires prior information which may not be available. To cope with this, employing a dictionary of kernels in lieu of a pre-selected single kernel has been proposed in the literature (see e.g. [140, 85, 105]). Specifically, the kernel is learned as a combination of kernels in the dictionary. Let $\kappa_1(\cdot), \ldots, \kappa_N(\cdot)$ be a set of N kernels where $\kappa_i(\cdot)$ denotes the i-th kernel. The function $\bar{\kappa}(\cdot)$ belongs to the convex hull $\mathbb{K} := \{\bar{\kappa}(\boldsymbol{x}) = \sum_{i=1}^{N} \beta_i \kappa_i(\boldsymbol{x}), \beta_i \geq 0, \forall i, \sum_{i=1}^{N} \beta_i = 1\}$ is a kernel [133]. Therefore, in online multi-kernel learning, the goal is to learn the convex combination of kernels in the dictionary to minimize the cumulative regret with respect to the best function approximator in hindsight. The cumulative regret is defined as the cumulative difference between loss of the online multi-kernel learning algorithm and that of the best function approximator in hindsight. Furthermore, for a dataset $\{(\boldsymbol{x}_t, y_t)\}_{t=1}^{T}$, the best function approximator is $f^*(\cdot) \in \arg\min_{f_i^*, i \in [N]} \sum_{t=1}^{T} \mathcal{L}(f_i^*(\boldsymbol{x}_t), y_t)$ where $f_i^*(\cdot) \in \arg\min_{f \in \mathbb{H}_i} \sum_{t=1}^{T} \mathcal{L}(f(\boldsymbol{x}_t), y_t)$ such that $\mathbb{H}_i$ is an RKHS induced by $\kappa_i(\cdot)$ and $[N] := \{1, \ldots, N\}$. Enabled by random feature approximation, centralized and scalable online multi-kernel learning algorithms have been proposed in literature (see e.g. [132, 137]). The present chapter proposes an algorithmic framework for personalized online federated MKL using RF approximation of kernels in the dictionary.

3.3 Personalized Online Federated Kernel Learning

The present section proposes an algorithmic framework for online federated multi-kernel learning which can deal with heterogeneous data among clients. To perform function approximation, RF approximations of kernel functions are employed. For the i-th kernel κ_i, vectors $\boldsymbol{\rho}_{i,1}, \ldots, \boldsymbol{\rho}_{i,D}$ are drawn i.i.d from $\pi_{\kappa_i}(\cdot)$ to construct the random feature vector $\boldsymbol{z}_i(\boldsymbol{x}) = \frac{1}{\sqrt{D}}[\sin(\boldsymbol{\rho}_{i,1}^\top \boldsymbol{x}), \ldots, \sin(\boldsymbol{\rho}_{i,D}^\top \boldsymbol{x}), \cos(\boldsymbol{\rho}_{i,1}^\top \boldsymbol{x}), \ldots, \cos(\boldsymbol{\rho}_{i,D}^\top \boldsymbol{x})]$. Then, at time instant t, the random feature approximation associated with $\kappa_i(\cdot)$ can be obtained as $\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}) = \boldsymbol{\theta}_{i,t}^\top \boldsymbol{z}_i(\boldsymbol{x})$ where $\boldsymbol{\theta}_{i,t}$ is the global function parameter associated the i-th kernel at time t.

3.3.1 Algorithm

At each time instant t, the server transmits global function parameters $\boldsymbol{\theta}_{i,t}$, $\forall i \in [N]$ to all clients. The k-th client, assigns the weight $w_{ik,t}$ to the i-th kernel which indicates the confidence of the k-th client at time t in the function approximation given by the i-th kernel. Upon receiving new data sample $\boldsymbol{x}_{k,t}$, the k-th client performs the function approximation combining kernels' RF approximations as

$$\hat{f}(\boldsymbol{x}_{k,t}; \hat{\boldsymbol{\Theta}}_t, \boldsymbol{w}_{k,t}) = \sum_{i=1}^{N} \frac{w_{ik,t}}{W_{k,t}} \boldsymbol{\theta}_{i,t}^{\top} \boldsymbol{z}_i(\boldsymbol{x}_{k,t}) = \sum_{i=1}^{N} \frac{w_{ik,t}}{W_{k,t}} \hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t}; \boldsymbol{\theta}_{i,t}) \tag{3.6}$$

where $\hat{\boldsymbol{\Theta}}_t = [\boldsymbol{\theta}_{1,t}, \ldots, \boldsymbol{\theta}_{N,t}]$, $\boldsymbol{w}_{k,t} = [w_{1k,t}, \ldots, w_{Nk,t}]$ and $W_{k,t} = \sum_{i=1}^{N} w_{ik,t}$. As it can be inferred from (3.6), each client constructs its own personalized combination of kernels. Upon observing the true label $y_{k,t}$, the k-th client calculates the losses $\mathcal{L}(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t}; \boldsymbol{\theta}_{i,t}), y_{k,t})$, $\forall i \in [N]$. Then, the k-th client leverages calculated losses to locally update both global and local parameters. Let $\boldsymbol{\theta}_{ik,t+1}$ and $w_{ik,t+1}$ denote the k-th client's local updates of $\boldsymbol{\theta}_{i,t}$ and $w_{ik,t}$, respectively. Specifically, the k-th client utilizes multiplicative update rule to update $w_{ik,t}$ as

$$w_{ik,t+1} = w_{ik,t} \exp\left(-\eta_k \mathcal{L}(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t}; \boldsymbol{\theta}_{i,t}), y_{k,t})\right), \forall i \in [N] \tag{3.7}$$

where η_k is the learning rate of the k-th client. Note that the k-th client ($\forall k \in [K]$) does not send its updated local parameter $\boldsymbol{w}_{k,t+1}$ to the server. Clients send their locally updated global parameters to the server (i.e. $\boldsymbol{\theta}_{ik,t+1}$). Aggregating local updates, the server updates global parameters to $\hat{\boldsymbol{\Theta}}_{t+1}$. If all clients send updates associated with all kernels (i.e. $\boldsymbol{\theta}_{ik,t+1}$, $\forall i \in [N]$), this requires sending $2ND$ parameters by each client at each time instant. When the number of both clients and kernels are large, the available client-to-server communication bandwidth may not be enough to afford sending $2NDK$ parameters per time instant even for small values of D. Note that reducing N and D degrade the performance of online federated MKL. Reducing N (the number of kernels), decreases the flexibility of clients to construct their ideal kernel using convex combination of kernels in the dictionary. Reducing D can degrade the accuracy of RF approximation.

The present chapter proposes an algorithmic framework to enable clients to perform online function approximation with sufficiently large dictionary of kernels while the available clients-to-server communication bandwidth can afford sending updates from clients to the server when a desirable value for the number of random features D is chosen. To this end, at each time instant, each client randomly chooses a subset of M kernels among all N kernels in the dictionary. Then, each client updates and sends the global parameters of the chosen M kernels to the server instead of updating and sending global parameters of all kernels. To choose a subset of M kernels, each client splits kernels into some bins and draws randomly one of the bins at each time instant. Each bin contains at most M kernels and each client updates and sends global parameters associated with kernels in the chosen bin. In order to distribute kernels among bins, at first the k-th client sorts kernels in descending order according to kernels' weights $\{w_{ik,t}\}_{i=1}^{N}$. Let $\mathcal{B}_j$ represents the j-th bin of kernels. The k-th client adds kernels from sorted list one by one to $\mathcal{B}_j$ until either all kernels are assigned to a bin or the number of kernels in $\mathcal{B}_j$ reaches M. When there are some kernels that are not assigned to any bins while there are M kernels in $\mathcal{B}_j$, the k-th client opens the bin $\mathcal{B}_{j+1}$ and adds kernels to this bin. This continues until all kernels are assigned to a bin. As it can be inferred from the procedure of distributing kernels into bins, the number of bins at every client is $m = \lceil \frac{N}{M} \rceil$. Furthermore, it can be concluded that $\mathcal{B}_1$ includes M kernels with the largest weights while the bin $\mathcal{B}_m$ includes $N - (m-1)M$ kernels with lowest weights. The k-th client assigns the weight $u_{jk,t}$ at time t to $\mathcal{B}_j$ defined as $u_{jk,t} = \sum_{\kappa_i \in \mathcal{B}_j} w_{ik,t}$. The k-th client draws one of the bins according to the probability mass function (PMF) $\boldsymbol{q}_{k,t}$ defined as

$$q_{jk,t} = (1 - \xi_k)\frac{u_{jk,t}}{U_{k,t}} + \frac{\xi_k}{m}, \forall j \in [m] \tag{3.8}$$

where $U_{k,t} = \sum_{j=1}^{m} u_{jk,t}$ and $0 < \xi_k \leq 1$ is an exploration rate determined by the k-th client.

Algorithm 10 The k-th client kernel subset selection at time t.

```
1:  Input: Weights $w_{ik,t}$, $\forall i \in [N]$, parameter $M$ and exploration rate $0 < \xi_k \leq 1$.
2:  Sort the kernels in descending order with respect to weights $\{w_{ik,t}\}_{i=1}^{N}$.
3:  Obtain the index sequence $s_1, \ldots, s_N$ such that $w_{s_b k,t} \leq w_{s_a k,t}$ if $b > a$, $\forall a, b \in [N]$.
4:  Open bin $\mathcal{B}_1$ and initialize $j = 1$.
5:  for all $i \in [N]$, the $k$-th client do
6:    if the bin $\mathcal{B}_j$ includes less than $M$ kernels then
7:      Adds the $s_i$-th kernel to $\mathcal{B}_j$.
8:    else
9:      Opens new bin $\mathcal{B}_{j+1}$, adds the $s_i$-th kernel to $\mathcal{B}_{j+1}$ and updates $j \leftarrow j + 1$.
10:   end if
11: end for
12: Draw an index $I_{k,t}$ via PMF $\boldsymbol{q}_{k,t}$ in (3.8).
13: Output: $\mathcal{S}_{k,t}$: indices set of kernels in the selected bin $\mathcal{B}_{I_{k,t}}$
```

Let $I_{k,t}$ be the index of the chosen bin by the k-th client at time t. The PMF in (3.8) constitutes trade-off between exploitation and exploration. According to the first term in the right hand side of (3.8), it is more probable that the k-th client draws a bin which includes kernels with larger weights $w_{ik,t}$. Hence, it is more probable that the k-th client collaborates in updating the global parameters of a kernel with larger weight $w_{ik,t}$. Let $\mathcal{S}_{k,t}$ denotes the set which includes the indices of kernels in the chosen bin at time t. The Algorithm 10 summarizes the procedure that the k-th client determines the set $\mathcal{S}_{k,t}$. According to Algorithm 10, kernel subset selection is personalized since each client chooses its own subset of kernels to update their parameters.

Let $p_{ik,t}$ denotes the probability that $i \in \mathcal{S}_{k,t}$. Then $p_{ik,t} = q_{b_i k,t}$ where b_i is the index of the bin which includes the i-th kernel. The k-th client updates global parameters locally as follows

$$\boldsymbol{\theta}_{ik,t+1} = \boldsymbol{\theta}_{i,t} - \eta \frac{\nabla \mathcal{L}(\boldsymbol{\theta}_{i,t}^{\top} \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})}{p_{ik,t}} \mathbf{1}_{i \in \mathcal{S}_{k,t}} \tag{3.9}$$

where $\mathbf{1}_{i \in \mathcal{S}_{k,t}}$ denotes an indicator function and it is 1 when $i \in \mathcal{S}_{k,t}$. The update rule in (3.9) implies that when $i \notin \mathcal{S}_{k,t}$, the k-th client does not update $\boldsymbol{\theta}_{i,t}$ (i.e. $\boldsymbol{\theta}_{ik,t+1} = \boldsymbol{\theta}_{i,t}$). Therefore,

Algorithm 11 Personalized Online Federated Multi-Kernel Learning (POF-MKL)

```
1: Input:Kernels $\kappa_i$, $i = 1, ..., N$, learning rate $\eta > 0$ and the number of random features $D$.
2: Initialize: $\boldsymbol{\theta}_{i,1} = \mathbf{0}$, $w_{ik,1} = 1$, $\forall i \in [N]$, $\forall k \in [K]$.
3: for $t = 1, \dots, T$ do
4:     The server transmits the global parameters $\hat{\boldsymbol{\Theta}}_t = [\boldsymbol{\theta}_{1,t}, \dots, \boldsymbol{\theta}_{N,t}]$ to all clients.
5:     for all $k \in [K]$, the $k$th client do
6:         Receive one datum $\boldsymbol{x}_{k,t}$.
7:         Predicts $\hat{f}(\boldsymbol{x}_{k,t}; \hat{\boldsymbol{\Theta}}_t, \boldsymbol{w}_{k,t})$ via (3.6).
8:         Calculates losses $\mathcal{L}(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t}; \boldsymbol{\theta}_{i,t}), y_{k,t})$, $\forall i \in [N]$.
9:         Updates $w_{ik,t+1}$, $\forall i \in [N]$ via (3.7).
10:        Selects a subset of kernel indices $\mathcal{S}_{k,t}$ using Algorithm 10.
11:        Updates $\boldsymbol{\theta}_{ik,t+1}$, $\forall i \in \mathcal{S}_{k,t}$ via (3.9) and sends $\boldsymbol{\theta}_{ik,t+1}$, $\forall i \in \mathcal{S}_{k,t}$ to the server.
12:    end for
13:    The server updates $\boldsymbol{\theta}_{i,t+1}$, $\forall i \in [N]$ via (3.10).
14: end for
```

the k-th client sends $\boldsymbol{\theta}_{ik,t+1}$ to the server only if $i \in \mathcal{S}_{k,t}$. Therefore, at each time instant, each client needs to send at most $2MD$ parameters to the server. Let $\mathcal{C}_{i,t}$ be a set of client indices such that $k \in \mathcal{C}_{i,t}$ if the k-th client sends $\boldsymbol{\theta}_{ik,t+1}$ to the server. Upon aggregating updates from clients, the server updates $\boldsymbol{\theta}_{i,t}$ as

$$\boldsymbol{\theta}_{i,t+1} = \boldsymbol{\theta}_{i,t} - \frac{1}{K} \sum_{k \in \mathcal{C}_{i,t}} (\boldsymbol{\theta}_{i,t} - \boldsymbol{\theta}_{ik,t+1}) = \boldsymbol{\theta}_{i,t} - \frac{\eta}{K} \sum_{k=1}^{K} \frac{\nabla \mathcal{L}(\boldsymbol{\theta}_{i,t}^{\top} \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})}{p_{ik,t}} \mathbf{1}_{i \in \mathcal{S}_{k,t}}. \quad (3.10)$$

Algorithm 11 summarizes the proposed personalized online federated multi-kernel learning algorithm called POF-MKL. It is useful to note that using our proposed POF-MKL, the server cannot find the gradients $\nabla \mathcal{L}(\boldsymbol{\theta}_{i,t}^{\top} \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})$ from updates received from clients. Instead, the server can find $\nabla \mathcal{L}(\boldsymbol{\theta}_{i,t}^{\top} \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})/p_{ik,t}$ where $p_{ik,t}$ is a time-varying value determined locally by the k-th client. This can promote the privacy of the proposed POF-MKL since exchanging the gradients can be hazardous to the privacy of federated learning (see e.g. [164, 56]).

Complexity. Each client needs to store d-dimensional D random feature vectors for each kernel. Therefore, the memory requirement of each client to implement function approximation using POF-MKL is $\mathcal{O}(dND)$. Using POF-MKL, at each time instant, each client needs to perform $\mathcal{O}(dND)$ operations including inner products and summations. Furthermore, when $\xi_k < 1$, in order to choose a subset of kernels, the k-th client needs to sort kernels which imposes worst case computational complexity of $\mathcal{O}(N \log N)$. However, when $\xi_k = 1$, according to PMF in (3.8), the k-th client chooses one bin uniformly at random and as a result in this case the k-th client does not need to sort kernels. Therefore, setting $\xi_k < 1$, the computational complexity for the k-th client is $\mathcal{O}(dND + N \log N)$ while setting $\xi_k = 1$, the computational complexity of the k-th client at each time instant is $\mathcal{O}(dND)$.

3.3.2 Regret Analysis

The present section analyzes the regret of the proposed POF-MKL. Specifically, two types of regret $\mathcal{R}_{k,T}$ and $\mathcal{R}_{s,T}$ are considered for the k-th client and the server, respectively. The performance of the k-th client utilizing POF-MKL is analyzed in terms of regret defined as

$$\mathcal{R}_{k,T} = \sum_{t=1}^{T} \mathcal{L}(\hat{f}(\boldsymbol{x}_{k,t}; \hat{\boldsymbol{\Theta}}_t, \boldsymbol{w}_{k,t}), y_{k,t}) - \min_{i \in [N]} \sum_{t=1}^{T} \mathcal{L}(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t}; \boldsymbol{\theta}_{i,t}), y_{k,t}) \tag{3.11}$$

where $\mathcal{R}_{k,T}$ measures the cumulative difference between the loss of the k-th client and the loss of the RF approximation of the kernel with minimum loss among all kernels' RF approximations. Let $\alpha^*_{ik,t}$, $\forall t \in [T]$, $\forall k \in [K]$ represents the optimal coefficients associated with the i-th kernel such that $f_i^*(\boldsymbol{x}) = \sum_{t=1}^{T} \sum_{k=1}^{K} \alpha^*_{ik,t} \kappa_i(\boldsymbol{x}, \boldsymbol{x}_{k,t})$. Then the best function approximator is defined as $f^*(\cdot) \in \arg\min_{f_i^*, i \in [N]} \sum_{t=1}^{T} \sum_{k=1}^{K} \mathcal{L}(f_i^*(\boldsymbol{x}_{k,t}), y_{k,t})$. Furthermore, the regret of the server is defined as the cumulative difference between the loss of POF-MKL and that of the best function approximator over all data samples distributed among clients which can be expressed as

$$\mathcal{R}_{s,T} = \sum_{t=1}^{T}\sum_{k=1}^{K} \mathcal{L}(\hat{f}(\boldsymbol{x}_{k,t};\hat{\boldsymbol{\Theta}}_t, \boldsymbol{w}_{k,t}), y_{k,t}) - \sum_{t=1}^{T}\sum_{k=1}^{K} \mathcal{L}(f^*(\boldsymbol{x}_{k,t}), y_{k,t}). \tag{3.12}$$

In order to analyze the regret of POF-MKL, suppose that the following assumptions hold true:

(A1) $\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})$, $\forall k \in [K]$ is convex with respect to $\boldsymbol{\theta}_{i,t}$ at each time instant t.

(A2) For $\boldsymbol{\theta}$ in a bounded set satisfying $\|\boldsymbol{\theta}\| \leq C$, the loss function and its gradient are bounded as $0 \leq \mathcal{L}(\boldsymbol{\theta}^\top \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t}) \leq 1$ and $\|\nabla \mathcal{L}(\boldsymbol{\theta}^\top \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})\| \leq L$. Moreover, each data sample is bounded as $\|\boldsymbol{x}_{k,t}\| \leq 1$, $\forall k \in [K]$, $\forall t \in [T]$.

(A3) Kernels $\kappa_i(\cdot)$, $\forall i \in [N]$ are shift-invariant with $\kappa_i(\mathbf{0}) = 1$, $\forall i \in [N]$.

The following theorem investigates the regret of the k-th client according to the k-th client data. The proof of the following Theorem can be found in Appendix C.1.

Theorem 3.1. *Under (A1)–(A3), the regret of the k-th client with respect to the best kernel satisfies*

$$\sum_{t=1}^{T} \mathcal{L}(\hat{f}(\boldsymbol{x}_{k,t};\hat{\boldsymbol{\Theta}}_t, \boldsymbol{w}_{k,t}), y_{k,t}) - \min_{i\in[N]} \sum_{t=1}^{T} \mathcal{L}(\hat{f}_{RF,it}(\boldsymbol{x}_{k,t};\boldsymbol{\theta}_{i,t}), y_{k,t}) \leq \frac{\ln N}{\eta_k} + \frac{\eta_k}{2}T. \tag{3.13}$$

Theorem 3.1 shows that by setting $\eta_k = \mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$, the k-th client achieves sub-linear regret of $\mathcal{O}(\sqrt{T})$. Furthermore, Theorem 3.1 shows that POF-MKL can deal with heterogeneous data among clients since the regret of each client defined in (3.11) is calculated with respect to the corresponding client data. The following theorem studies the regret of the server with respect to the best function approximator. The proof can be found in Appendix C.2.

Theorem 3.2. *Let $i^* := \arg\min_{i\in[N]} \sum_{t=1}^{T}\sum_{k=1}^{K} \mathcal{L}(f_i^*(\boldsymbol{x}_{k,t}), y_{k,t})$ and σ_i be the second Fourier moment of the i-th kernel. Under (A1)–(A3), the regret of the server with respect to the best* *function approximator satisfies*

$$\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}(\hat{f}(\boldsymbol{x}_{k,t};\hat{\boldsymbol{\Theta}}_t,\boldsymbol{w}_{k,t}),y_{k,t})-\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}(f^*(\boldsymbol{x}_{k,t}),y_{k,t})$$
$$\leq\frac{KC^2}{2\eta}+\frac{\eta}{2}\sum_{t=1}^{T}\sum_{k=1}^{K}\frac{L^2}{p_{i^*k,t}}+\sum_{k=1}^{K}\left(\frac{\ln N}{\eta_k}+\frac{\eta_k}{2}T\right)+\epsilon LKTC \tag{3.14}$$

with probability at least $1-2^8\left(\frac{\sigma_{i^*}}{\epsilon}\right)^2\exp\left(-\frac{D\epsilon^2}{4(d+2)}\right)$ *where* $C:=\max_{i\in[N]}\sum_{t=1}^{T}\sum_{k=1}^{K}\alpha^*_{ik,t}$.

As it can be inferred from (3.14), the regret of the server with respect to the best function approximator depends on $\frac{1}{p_{i^*k,t}}$. From (3.8) and the fact that $p_{ik,t}=q_{b_ik,t}$, it can be concluded that $p_{i^*k,t}>\frac{\xi_k}{m}$. Thus, setting $\xi_k=\mathcal{O}(1)$, then $p_{ik,t}>\mathcal{O}(\frac{M}{N})$. The regret bound in (3.14) shows that setting $\eta=\mathcal{O}\left(\sqrt{\frac{M}{NT}}\right)$ and $\epsilon=\eta_k=\frac{1}{\sqrt{T}}$, $\forall k\in[K]$, the server obtains regret of $\mathcal{O}\left(\sqrt{\frac{N}{M}T}\right)$ with probability at least $1-2^8\left(\frac{\sigma_{i^*}}{\epsilon}\right)^2\exp\left(-\frac{D\epsilon^2}{4(d+2)}\right)$. This shows that increasing M tighten the regret bound and increasing D increases the probability that the regret bound in (3.14) holds true. However, using POF-MKL, each client needs to transmit MD parameters at each time instant. Since both M and D are determined by the algorithm POF-MKL, this shows that POF-MKL can provide flexibility to tighten regret bound while the available clients-to-server communication bandwidth can afford transmission of clients' updates to the server. It is useful to mention that choosing larger value for ξ_k increases the lower bound of $p_{i^*k,t}$ and as a result the optimal choice for ξ_k in terms of regret is $\xi_k=1$. However, choosing smaller values for ξ_k makes the value of $p_{ik,t}$ more dependent on weights $\{w_{ik,t}\}_{k=1}^{K}$ (c.f. (3.8)). Therefore, choosing smaller values for ξ_k makes $p_{ik,t}$ less predictable. This makes estimating $\nabla\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top z_i(x_{k,t}),y_{k,t})$ given $\nabla\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top z_i(x_{k,t}),y_{k,t})/p_{ik,t}$ more difficult which leads to better protection of privacy.

Comparison with personalized federated learning. In order to deal with heterogeneous data among clients, personalized federated learning has been studied extensively in the literature (see [139, 32, 39, 46, 72, 42, 99, 96, 135, 30, 1, 107, 158, 2, 142, 22]). Utilizing model-agnostic meta-learning [49], personalized federated learning algorithms have been proposed in [46, 1]. In [135, 22], personalized federated learning algorithms have been designed by learning hyper-networks [70]. In [107], a personalized model is a linear combination of a set of shared component models such that each client constructs its personalized mixture of models. However, in aforementioned personalized federated learning works, clients are assumed to store a dataset to perform local updates with. Therefore, when clients are not able to store data in batch and they have to make a decision upon receiving a new data sample, aforementioned works in personalized federated learning cannot guarantee sub-linear regret for clients. However, according to Theorems 3.1 and 3.2, POF-MKL provides sub-linear regret for clients when clients cannot store data in batch and make decision in an online fashion.

Comparison with online federated learning [112]. Fed-OMD algorithm has been proposed in [112] which enables clients to perform their learning task in an online and federated fashion while it is proved that Fed-OMD enjoys sub-linear regret when the loss function is convex with respect to parameters required to be learnt at each time instant. The proposed POF-MKL differs from Fed-OMD in the sense that Fed-OMD cannot guarantee sub-linear regret when it comes to performing the online learning task with RF approximations of multiple kernels since the loss function $\mathcal{L}(\sum_{i=1}^{N} w_{i,t}\boldsymbol{\theta}_{i,t}^{\top}\boldsymbol{z}_i(\boldsymbol{x}), y)$ is not convex with respect to both $\boldsymbol{\theta}_{i,t}$ and $w_{i,t}$. However, according to Theorems 3.1 and 3.2, the proposed POF-MKL guarantees sub-linear regret.

Comparison with [78]. Online federated MKL algorithms called vM-KOFL and eM-KOFL have been presented in [78]. Both POF-MKL and algorithms in [78] exploit random feature approximation to alleviate computational complexity of online kernel learning. Furthermore, both POF-MKL and algorithms in [78] learn a linear combination of kernels. The proposed POF-MKL has the following advantages and innovations compared to vM-KOFL and eM-KOFL: i) The proposed POF-MKL allows clients to learn their own personalized combination of kernels (c.f. (3.6)). As it is proved in Theorem 3.1, the proposed POF-MKL can deal with heterogeneous data among clients in the sense that using POF-MKL each client guarantees sub-linear regret with respect to the best kernel RF approximation according to the corresponding client data. However, both vM-KOFL and eM-KOFL are not able to provide such guarantee. ii) Using vM-KOFL, each client needs to send $(N+1)D$ parameters to the server. However, using the proposed POF-MKL, each client needs to send MD parameters to the server such that $M \leq N$ is determined by POF-MKL and can be chosen to be much smaller than N. iii) In eM-KOFL, the server chooses a kernel at each time instant and clients send their local updates associated with the chosen kernel by the server. The proposed POF-MKL provides more flexibility compared to eM-KOFL in the sense that using POF-MKL each client can send local updates of $M \geq 1$ kernels to the server. And each client chooses its own subset of kernels to send their updates to the server. Therefore, even though POF-MKL sets M to 1, it is possible that at a time instant the server receives updates associated with all kernels in the dictionary. It is useful to mention that using eM-KOFL the cumulative regret of all clients is sub-linear with respect to the best kernel RF approximation with probability $1-\delta$ where $0 < \delta \leq 1$. However, utilizing the update rule in (3.9), using the proposed POF-MKL, each client obtains sub-linear regret with respect to RF approximation of its best kernel with probability 1.

3.4 Experiments

We tested the performance of the proposed POF-MKL for online regression task through a set of experiments. The performance of POF-MKL is compared with the baselines PerFedAvg [46], OFSKL [112], OFMKL-Avg [112], vM-KOFL [78] and eM-KOFL [78]. PerFedAvg refers to the personalized federated averaging algorithm in [46]. In the experiments, PerFedAvg employs a fully connected feedforward neural network model. More information about the implementation of PerFedAvg can be found in Appendix C.3. OFSKL and OFMKL-Avg are two variations of Fed-OMD [112]. OFSKL leverages Fed-OMD [112] when a single radial basis function (RBF) with bandwidth of 10 is employed to perform the learning task. In OFMKL-Avg, kernels are learned independently from each other using Fed-OMD [112] and the prediction is the average of approximations given by kernels. Moreover, vM-KOFL and eM-KOFL are online federated MKL algorithms of [78] such that vM-KOFL requires transmission of all kernel updates at every time instant while eM-KOFL requires transmission of a kernel update at each time instant. In the experiments, each client observes 500 samples until the end of the learning task meaning that $T = 500$. The performance of the proposed POF-MKL and other baselines are tested on the following real datasets downloaded from UCI machine learning repository [44]: Naval [31], UJI [145], Air [161] and WEC [117]. More detailed information about datasets can be found in Appendix C.3. Data samples of Naval and UJI datasets are distributed i.i.d among clients. Data samples in Air and WEC datasets are distributed non-i.i.d among clients. More inforamtion about distributing data samples among clients can be found in Appendix C.3. The number of clients for Naval, UJI, Air and WEC datasets are 23, 42, 240 and 560, respectively. The dictionary of kernels consists of 51 RBFs with different bandwidth such that the bandwidth of the i-th kernel is $\sigma_i = 10^{\frac{2i-52}{25}}$. We consider the case where the clients-to-server communication bandwidth is limited such that at each time instant, the maximum number of parameters that a client is allowed to transmit to the server is 1000. Furthermore, the memory and computational capability of clients are limited such that the maximum value can be picked for the number of random features D is 100. The experiments were carried for 20 different sets of random feature vectors. The performance of algorithms is measured using average of mean squared error (MSE) defined as

$$\text{MSE} = \frac{1}{20}\sum_{j=1}^{20}\frac{1}{KT}\sum_{t=1}^{T}\sum_{k=1}^{K}(\hat{y}_{jk,t} - y_{k,t})^2$$

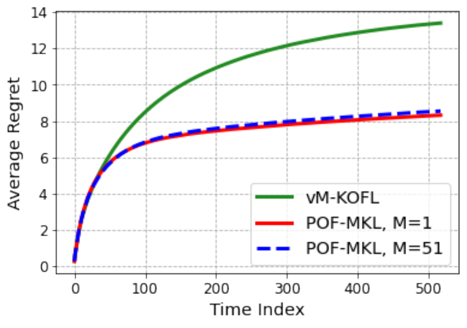
where $\hat{y}_{jk,t}$ denotes the prediction of the k-th client at time instant t corresponding to the j-th set of random feature vectors. Learning rates are set to $\eta = \eta_k = \frac{1}{\sqrt{T}}$, $\forall k$. Also, exploration rates are set to $\xi_k = 1$, $\forall k$. The performance of POF-MKL with different ξ_k is studied in Appendix C.3. Codes are available at `https://github.com/pouyamghari/POF-MKL`.

Table 3.1 presents the MSE and run time performance of online federated kernel learning algorithms on real datasets. Run time refers to average total run time of clients to perform online learning task on the entire data samples that they observe. In Table 3.1, M refers to the number of kernels whose updates are sent to the server after prediction at each time instant. And, D is the number of random features. Comparing MSE of POF-MKL with that of OFMKL-Avg, it can be concluded that learning the weights to combine kernels provides higher accuracy than averaging kernels' predictions. Table 3.1 shows that POF-MKL with $M = 1$ provides lower MSE than eM-KOFL. Using eM-KOFL, at each time instant, the server receives updates belong to only one kernel. However, using POF-MKL with $M = 1$, each client sends an update belongs to a kernel which is selected by the client. Therefore, the server receives updates associated with different kernels even though $M = 1$. Therefore, experimental results show the effectiveness of the personalized kernel selection provided by POF-MKL. It can be observed that POF-MKL with $M = 25$ obtains lower MSE than those of POF-MKL with $M = 51$ and vM-KOFL. Since each client is allowed to send at most 1000 parameters per time instant, if clients send updates of all kernels at every time instant as this is the case in vM-KOFL, D cannot be chosen to be greater than 9. However, setting $M = 25$, POF-MKL can set $D = 20$ which can improve the accuracy of online regression task compared to the case where $D = 9$. Note that according to Theorem 3.2, increase in D increases the probability that the server achieves sub-linear regret with respect to the best function approximator. Furthermore, POF-MKL with $M = 51$ achieves lower MSE than vM-KOFL even if data samples are distributed i.i.d among clients. This shows that the proposed POF-MKL can better cope with heterogeneous data among clients which is in agreement with theoretical results in Theorem 3.1. In fact, the optimal combination of kernels

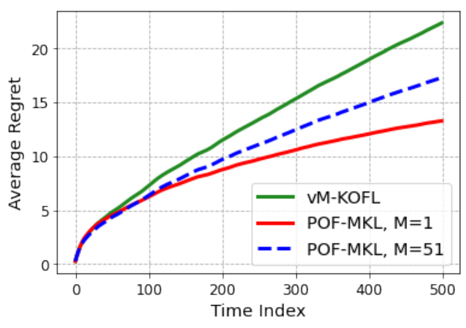
Table 3.1: MSE($\times 10^{-3}$) and run time of online federated learning algorithms on real datasets.

			MSE($\times 10^{-3}$)				Run time(s)			
Algorithms	M	D	Naval	UJI	Air	WEC	Naval	UJI	Air	WEC
PerFedAvg	-	-	118.60	63.03	13.68	77.33	44.59	41.67	37.40	33.56
OFSKL	1	100	77.77	61.82	13.65	87.87	0.07	0.06	0.08	0.06
OFMKL-Avg	51	9	33.25	55.44	10.63	34.01	1.51	1.73	0.91	0.47
vM-KOFL	51	9	26.42	51.50	10.58	25.17	2.01	2.22	1.37	0.67
eM-KOFL	1	100	28.64	61.08	21.94	20.14	2.27	10.13	1.45	1.70
POF-MKL	1	100	**16.16**	**33.02**	**9.27**	**11.44**	1.22	9.02	1.25	1.10
POF-MKL	25	20	16.82	37.34	9.34	11.58	0.69	2.29	0.63	0.52
POF-MKL	51	9	16.65	41.00	9.38	11.97	0.82	1.07	0.81	0.65

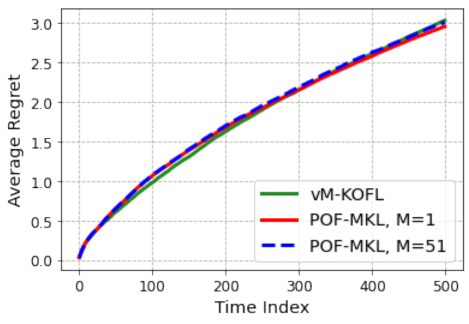
can be different across clients. Using POF-MKL, each client constructs its own personalized combination of kernels which results in lower MSE compared to vM-KOFL. The proposed POF-MKL with $M = 1$ and $M = 25$ runs faster than eM-KOFL. In fact, using POF-MKL, clients only need to update parameters associated with M kernels while employing vM-KOFL and eM-KOFL, clients have to update parameters of all kernels. Moreover, POF-MKL obtains lower MSE than PerFedAvg. Note that since clients are not able to store data in batch, at each time instant clients update PerFedAvg's model using only the newly observed data sample. Therefore, convergence of PerFedAvg is not guaranteed. Experimental results show that POF-MKL achieves higher accuracy than PerFedAvg in online regression task when it is not possible for clients to store data in batch. Since OFSKL employs only a pre-selected single kernel, OFSKL runs faster than POF-MKL. However, utilizing multiple kernels enables POF-MKL to obtain lower MSE than that of OFSKL. In fact, using POF-MKL clients learn a linear combination of kernels which is proved to enjoy sub-linear regret with respect to the best kernel in hindsight while employing OFSKL clients have to make predictions using a pre-selected kernel. Furthermore, Figure 3.1 illustrates the average regret of clients when clients employ vM-KOFL and the proposed POF-MKL with different M parameters. From Figure 3.1, it can be observed that the proposed POF-MKL achieves sub-linear regret.



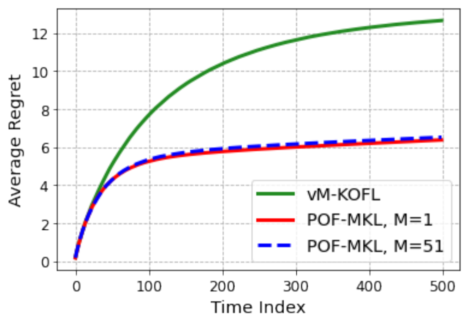
(a) Naval dataset.



(b) UJI dataset.



(c) Air dataset.



(d) WEC dataset.

Figure 3.1: Average regret of clients.

3.5 Conclusion

The present chapter proposed a personalized online federated MKL algorithm called POF-MKL based on RF approximation. Employing the proposed POF-MKL, each client updates the parameters of a subset of kernels which alleviates the computational complexity of the client as well as communication cost of sending updated parameters of kernels. Theoretical analysis proved that using POF-MKL, each client achieves sub-linear regret with respect to the RF approximation of its best kernel in hindsight which indicates that POF-MKL can deal heterogeneous data among clients. While each client updates a subset of kernels, it was proved that the server achieves sub-linear regret with respect to the best function approximator. Experiments on real datasets showcased the advantages of POF-MKL compared with other online federated kernel learning algorithms.

Chapter 4

Online Federated Model Selection

4.1 Introduction

The performance of prediction tasks can be heavily influenced by the choice of model. As a result, the problem of model selection arises in various applications and studies, such as reinforcement learning (see, e.g., [36, 47, 93]) where a learner selects a model from a set of candidate models to be deployed for prediction. Furthermore, in practical scenarios, data often arrives in a sequential manner, rendering the storage and processing of data in batches impractical. Consequently, there is a need to conduct model selection in an online fashion. In this regard, the learner chooses one model among a *dictionary of models* at each learning round, and after performing a prediction with the chosen model, the learner incurs a loss. The best model in hindsight refers to the model with minimum cumulative loss over all learning rounds while regret is defined as the difference between the loss of the chosen model and the best model in hindsight. Performing model selection online, the goal is to minimize cumulative regret. Depending on the available information about the loss, different online model selection algorithms have been proposed in the literature (see, e.g.,, [50, 114, 51, 16, 119, 128]) which are proven to achieve sub-linear regret. The performance of existing online model selection algorithms depends on the performance of models in the dictionary. The choice of dictionary requires information about the performance of models on unseen data, which may not be available a priori. In this case, a richer dictionary of models may improve the performance. However, in practice, the learner may face storage limitations, making storing and operating with a large dictionary infeasible. For example, consider the learner as an edge device performing an online prediction task such as image or document classification using a set of pre-trained models. In this case, the learner may be unable to store all models in the memory. Faced with this challenge, the present work aims at answering the following critical question: *How can learners perform online model selection with a large dictionary of models that requires memory beyond their storage capacity?*

To tackle this problem, the present chapter proposes an **o**nline **f**ederated **m**odel **s**election and **f**ine-**t**uning algorithm called OFMS-FT to enable a group of learners to perform online prediction employing a large dictionary of models. To this end, the online model selection is performed in a federated manner in the sense that the group of learners, also known as *clients*, interacts with a server to choose a model among all models stored in the server. Specifically, a server with a significantly larger storage capacity than clients stores a large number of models. At each round, each client chooses to receive and store a subset of models that can be fit into its memory and performs its prediction using one of the stored models. Each client computes the losses of received models and leverages these observed losses to select a model for prediction in future rounds. Moreover, the distribution of data streams observed by clients may differ from the distribution of data that models are pre-trained on. At each round, clients employ the proposed OFMS-FT to adapt models to their data and send the updates to the server. Upon receiving updates from clients, the server updates models.

In addition to the storage capacity of clients, there are some other challenges that should be taken into account. Communication efficiency is a critical issue in federated learning (see e.g.,, [89, 82, 129, 71]). The available bandwidth for client-to-server communication is usually limited which restricts the amount of information that can be sent to the server. To deal with limited communication bandwidth, using the proposed OFMS-FT, at each round the server chooses a subset of clients to fine-tune their received models. Moreover, in some cases, the distribution of data samples may be different across clients [139, 160, 99]. Thus, different clients can have different models as the best model in hindsight. Through regret analysis, it is proven that by employing OFMS-FT, each client enjoys sub-linear regret with respect to its best model in hindsight. This shows that OFMS-FT effectively deals with heterogeneous data distributions among clients. In addition, the regret analysis of the present chapter proves that the server achieves sub-linear regret with respect to the best model in hindsight. Furthermore, regret analysis shows that increase in communication bandwidth and memory of clients improves the regret bounds. This indicates efficient usage of resources by OFMS-FT. Experiments on regression and image classification datasets showcase the effectiveness of OFMS-FT compared with state-of-the-art alternatives. It is also worth noting that the materials in this chapter are included in [63].

4.2 Problem Statement

Consider a set of N clients interacting with a server to perform sequential prediction. There are a set of K models $f_1(\cdot;\cdot), \ldots, f_K(\cdot;\cdot)$ stored at the server. The set of models $f_1(\cdot;\cdot), \ldots, f_K(\cdot;\cdot)$ is called *dictionary* of models. Due to clients' limited memory, clients are not able to store all models $f_1(\cdot;\cdot), \ldots, f_K(\cdot;\cdot)$ and a server with larger storage capacity stores models. Let $[K]$ denote the set $\{1, \ldots, K\}$. At each learning round t, client i receives a data sample $\boldsymbol{x}_{i,t}$ and makes a prediction for $\boldsymbol{x}_{i,t}$. Specifically, at learning round t, client i picks a model among the dictionary of models $f_1(\cdot;\cdot), \ldots, f_K(\cdot;\cdot)$ and makes prediction $f_{I_{i,t}}(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{I_{i,t},t})$ where $I_{i,t}$ denote the index of the chosen model by client i at learning round t and $\boldsymbol{\theta}_{k,t}$ represents the parameter of model k at learning round t. After making prediction, client i, $\forall i \in [N]$ incurs loss $\mathcal{L}(f_{I_{i,t}}(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{I_{i,t},t}), y_{i,t})$ and observes label $y_{i,t}$ associated with $\boldsymbol{x}_{i,t}$. This continues until the time horizon T. The loss function $\mathcal{L}(\cdot,\cdot)$ is the same across all clients and depends on the task performed by clients. For example, if clients perform regression tasks, the loss function $\mathcal{L}(\cdot,\cdot)$ can be the mean square error function while if clients perform classification tasks the loss function $\mathcal{L}(\cdot,\cdot)$ can be chosen to be cross-entropy loss function. Furthermore, the present chapter studies the adversarial setting where the label $y_{i,t}$ for any i and t is determined by the environment through a process unknown to clients. This means that the distribution of the data stream $\{(\boldsymbol{x}_{i,t}, y_{i,t})\}_{t=1}^{T}$ observed by client i can be non-stationary (i.e., it can change over learning rounds) while client i, $\forall i \in [N]$ does not know the distribution from which $(\boldsymbol{x}_{i,t}, y_{i,t})$, $\forall t$ is sampled. Moreover, data distribution can be different across clients.

The performance of clients can be measured through the notion of regret which is the difference between the client's incurred loss and that of the best model in hindsight. Therefore, the regret of the i-th client can be formalized as

$$\mathcal{R}_{i,T} = \sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(f_{I_{i,t}}(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{I_{i,t},t}), y_{i,t})] - \min_{k\in[K]} \sum_{t=1}^{T} \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t}), \tag{4.1}$$

where $\mathbb{E}_t[\cdot]$ denote the conditional expectation given observed losses in prior learning rounds. The objective for each client is to minimize their regret by choosing a model from a dictionary in each learning round. This can be accomplished if clients can identify a subset of models that perform well with the data they have observed. Assessing the losses of multiple models, including the selected one, can help clients better evaluate model performance. This can expedite the identification of models with superior performance, ultimately reducing prediction loss. However, due to memory constraints, clients cannot compute the loss for all models. To address this limitation, clients must select a subset of models that can fit within their memory and calculate the loss for this chosen subset. This information aids in selecting a model for future learning rounds. Furthermore, to enhance model performance, clients and the server collaborate on fine-tuning models. Let $\boldsymbol{\theta}_k^*$ be the optimal parameter for the k-th model which is defined as

$$\boldsymbol{\theta}_k^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \sum_{t=1}^{T} \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}), y_{i,t}). \tag{4.2}$$

In this context, the regret of the server in fine-tuning model k is defined as

$$\mathcal{S}_{k,T} = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t}) - \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_k^*), y_{i,t}). \tag{4.3}$$

The objective of the server is to orchestrate model fine-tuning to minimize its regrets. This chapter introduces an algorithmic framework designed to enable clients with limited memory to conduct online model selection and fine-tuning. Regret analysis in Section 4.4 demonstrates that employing this algorithm leads to sub-linear regret for clients and the server. Additionally, the analysis in Section 4.4 illustrates that increasing the memory budget for clients results in a tighter upper bound on regret for the proposed algorithm.

4.3 Online Federated Model Selection

The present section introduces a disciplined way to enable clients to pick a model among a dictionary of models beyond the storage capacity of clients while clients enjoy sub-linear regret with respect to the best model in the dictionary. The proposed algorithm enables clients to collaborate with each other to fine-tune models in order to adapt models to clients' data.

4.3.1 Online Federated Model Selection

Let c_k be the cost to store model k. For example, c_k can be the amount of memory required to store the k-th model. Let B_i denote the budget associated with the i-th client, which can be the memory available by the i-th client to store models. At each learning round, each client i stores a subset of models that can be fit into the memory of client i. The i-th client fine-tunes and computes the loss of the stored subset of models. Computing the loss of multiple models at each learning round enables clients to obtain better evaluation on the performance of models which can lead to identifying the best model faster. Moreover, fine-tuning multiple models can result in adapting models to clients' data faster. In Section 4.4, it is demonstrated that an increase in the number of models that clients can evaluate and fine-tune leads to tighter regret bounds for the proposed algorithm

At learning round t, the i-th client assigns weight $z_{ik,t}$ to the k-th model, which indicates the credibility of the k-th model with respect to the i-th client. Upon receiving a new data sample, the i-th client updates the weights $\{z_{ik,t}\}_{k=1}^{K}$ based on the observed losses. The update rule for weights $\{z_{ik,t}\}_{k=1}^{K}$ will be specified later in (4.8). Using $\{z_{ik,t}\}_{k=1}^{K}$, at learning round t, the i-th client selects one of the models according to the probability mass function (PMF) $\boldsymbol{p}_{i,t}$ as

$$p_{ik,t} = \frac{z_{ik,t}}{Z_{i,t}}, \forall k \in [K], \tag{4.4}$$

where $Z_{i,t} = \sum_{k=1}^{K} z_{ik,t}$. Let $I_{i,t}$ denote the index of the selected model by the i-th client at learning round t. Then client i splits all models except for $I_{i,t}$-th model into clusters $\mathbb{D}_{i1,t}, \ldots, \mathbb{D}_{im_{iI_{i,t}},t}$ such that the cumulative cost of models in each cluster $\mathbb{D}_{ij,t}$ does not exceed $B_i - c_{I_{i,t}}$. Note that $m_{ij,t}$ denote the number of clusters constructed by client i at learning round t if $I_{i,t} = j$. As it will be clarified in Section 4.4, packing models into minimum number of clusters helps clients to achieve tighter regret bound. Packing models into the minimum number of clusters can be viewed as a bin packing problem (see e.g.,, [54]) which is NP-hard [26]. Several approximation methods have been proposed and analyzed in the literature [80, 38, 43, 76]. In the present work, models are packed into clusters using the first-fit-decreasing (FFD) bin packing algorithm (see Algorithm 15 in Appendix D.1). It has been proved that FFD can pack models into at most $\frac{11}{9}m^* + \frac{2}{3}$ clusters where m^* is the minimum number of clusters (i.e., optimal solution for the clustering problem) [43].

After packing models into clusters, the i-th client chooses one of the clusters $\mathbb{D}_{i1,t}, \ldots, \mathbb{D}_{im_{iI_{i,t},t},t}$ uniformly at random. Let $J_{i,t}$ be the index of the cluster chosen by client i at learning round t. The i-th client downloads and stores all models in the cluster $\mathbb{D}_{iJ_{i,t},t}$ along with the $I_{i,t}$-th model. Since the cumulative cost of models in $\mathbb{D}_{iJ_{i,t},t}$ does not exceed $B_i - c_{I_{i,t}}$, the cumulative cost of models in $\mathbb{D}_{iJ_{i,t},t}$ in addition to the $I_{i,t}$-th model does not exceed the budget B_i. Let $\mathbb{S}_{i,t}$ represents the subset of models stored by client i at learning round t. Upon receiving a new datum at learning round t, the i-th client performs its prediction task using the $I_{i,t}$-th model. Then, the i-th client incurs loss $\mathcal{L}(f_{I_{i,t}}(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{I_{i,t},t}), y_{i,t})$. After observing $y_{i,t}$, the i-th client computes the loss $\mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t})$, $\forall k \in \mathbb{S}_{i,t}$. After computing the losses, the i-th client obtains the importance sampling loss estimate for the k-th model as

$$\ell_{ik,t} = \frac{\mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t})}{q_{ik,t}} \mathcal{I}(k \in \mathbb{S}_{i,t}), \tag{4.5}$$

where $\mathcal{I}(\cdot)$ denote the indicator function and $q_{ik,t}$ is the probability that the i-th client stores the k-th model at round t. In order to derive $q_{ik,t}$, note that the probability of storing model k at learning round t can be conditioned on $I_{i,t}$ and based on the total probability theorem, it can be obtained that

$$q_{ik,t} = \sum_{j=1}^{K} \Pr[I_{i,t} = j] \Pr[k \in \mathbb{S}_{i,t} | I_{i,t} = j] = \sum_{j=1}^{K} p_{ij,t} \Pr[k \in \mathbb{S}_{i,t} | I_{i,t} = j], \tag{4.6}$$

where $\Pr[\mathcal{A}|\mathcal{B}]$ denote the probability of event $\mathcal{A}$ given that event $\mathcal{B}$ occurred. If client i

chooses the k-th model at learning round t (i.e., $I_{i,t} = k$), then client i stores model k meaning that $\Pr[k \in \mathbb{S}_{i,t} | I_{i,t} = k] = 1$. Let $m_{ij,t}$ denote the number of clusters when client i chooses $I_{i,t} = j$. Note that if $I_{i,t} = j$ client i splits all models except for model j into $m_{ij,t}$ clusters. If client i chooses $I_{i,t} = j$ such that $j \neq k$, then client i stores model k at learning round t if the chosen cluster contains model k. Since only one cluster among $m_{ij,t}$ clusters contains model k and client i chooses one cluster uniformly at random, it can be concluded that $\Pr[k \in \mathbb{S}_{i,t} | I_{i,t} = j, j \neq k] = \frac{1}{m_{ij,t}}$. Therefore, according to (4.6), the probability $q_{ik,t}$ can be obtained as

$$q_{ik,t} = p_{ik,t} + \sum_{\forall j: j \neq k} \frac{p_{ij,t}}{m_{ij,t}}. \tag{4.7}$$

At learning round t, client i updates the weights of models using the multiplicative update rule as

$$z_{ik,t+1} = z_{ik,t} \exp\left(-\eta_i \ell_{ik,t}\right), \tag{4.8}$$

where η_i is the learning rate associated with the i-th client. According to (4.5) and (4.8), if client i does not observe the loss of model k at learning round t, client i keeps the weight $z_{ik,t+1}$ the same as $z_{ik,t}$. Conversely, when the loss is observed, a higher loss corresponds to a greater reduction in the weight $z_{ik,t}$.

4.3.2 Online Model Fine-Tuning

This subsection presents a principled way to enable clients to collaborate with the server to fine-tune models. If client i participates in fine-tuning at learning round t, client i locally updates all its stored models in $\mathbb{S}_{i,t}$ and sends updated models' parameters to the server. Sending updated parameters of a model to the server occupies a portion of communication bandwidth. Therefore, the available communication bandwidth may not be enough such that all clients can send their updates every learning round. In this case, at learning round t, the server has to choose a subset of clients $\mathbb{G}_t$ such that the available bandwidth is sufficient for all clients in $\mathbb{G}_t$ to send their updates to the server.

Let b_k be the bandwidth required to send the updated parameters of the k-th model. If $i \in \mathbb{G}_t$, then the i-th client needs the bandwidth $e_i = \sum_{k \in \mathbb{S}_{i,t}} b_k$ to send updated parameters of its stored models. Let the available communication bandwidth be denoted by E. Given e_i, $\forall i \in [N]$, the server splits clients into α groups $\mathbb{N}_1, \ldots, \mathbb{N}_\alpha$ such that $\sum_{i \in \mathbb{N}_j} e_i \leq E, \forall j \in [\alpha]$. This means that all clients in each group $\mathbb{N}_j$, $\forall j \in [\alpha]$ can send their updated model parameters to the server given the available bandwidth. At each learning round t, the server draws one of the client groups $\mathbb{N}_1, \ldots, \mathbb{N}_\alpha$ uniformly at random. Clients in the chosen group send their updated models' parameters to the server. In other words, $\mathbb{G}_t := \mathbb{N}_{\iota_t}$ where ι_t represents the index of the chosen client group at learning round t. Since the probability of choosing a client group is $\frac{1}{\alpha}$, the probability that a client is chosen by the server to send its updated models parameters is $\frac{1}{\alpha}$ as well. Therefore, to maximize the probability that a client is chosen by the server to be in $\mathbb{G}_t$, the server can split clients into a minimum number of groups. To do this, bin-packing algorithms such as FFD [43] can be employed.

Define the importance sampling loss gradient estimate $\nabla \hat{\ell}_{ik,t}$ associated with client i and model k as

$$\nabla \hat{\ell}_{ik,t} = \frac{\alpha}{q_{ik,t}} \nabla \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t}) \mathcal{I}(i \in \mathbb{G}_t, k \in \mathbb{S}_{i,t}). \tag{4.9}$$

Recall that $\mathcal{I}(\cdot)$ denote the indicator function. The i-th client updates the models' parameters as

$$\boldsymbol{\theta}_{ik,t+1} = \boldsymbol{\theta}_{k,t} - \eta_f \nabla \hat{\ell}_{ik,t}, \forall k \in \mathbb{S}_{i,t}, \tag{4.10}$$

where η_f is the fine-tuning learning rate. According to (4.9) and (4.10), if the i-th client is not chosen for fine-tuning (i.e., $i \notin \mathbb{G}_t$), then $\boldsymbol{\theta}_{ik,t+1} = \boldsymbol{\theta}_{k,t}$. Thus, if $i \in \mathbb{G}_t$, the i-th client sends locally updated models parameters $\{\boldsymbol{\theta}_{ik,t+1}\}_{\forall k \in \mathbb{S}_{i,t}}$ to the server. Let $\mathbb{V}_{k,t}$ denote the index set of clients such that $i \in \mathbb{V}_{k,t}$ if the server receives the update of the k-th model $\boldsymbol{\theta}_{ik,t+1}$ from the i-th client. Upon aggregating information from clients, the server updates model parameters $\{\boldsymbol{\theta}_{k,t}\}_{k=1}^{K}$ as:

$$\boldsymbol{\theta}_{k,t+1} = \boldsymbol{\theta}_{k,t} - \frac{1}{N} \sum_{i \in \mathbb{V}_{k,t}} (\boldsymbol{\theta}_{k,t} - \boldsymbol{\theta}_{ik,t+1}) = \boldsymbol{\theta}_{k,t} - \frac{\eta_f}{N} \sum_{i=1}^{N} \nabla \hat{\ell}_{ik,t}. \tag{4.11}$$

The proposed Online Federated Model Selection and Fine-Tuning (OFMS-FT) algorithm is summarized in Algorithm 12. Steps 5 to 10 in Algorithm 12 outline how client i selects a subset of models during each learning round t. Subsequently, each client i, $\forall i \in [N]$, transmits its required bandwidth e_i (acquired in step 10) to the server. Upon receiving e_i, $\forall i \in [N]$, the server, following steps 12 to 14, determines a subset of clients $\mathbb{G}_t$ eligible to send their updates. Each client i then utilizes its selected model for prediction (step 16) and computes the loss of its stored model subset, updating the weights $z_{ik,t}{}_{k=1}^{K}$ (step 17). If client i is included in the server's selected subset $\mathbb{G}_t$, it transmits its local updates to the server, as depicted in step 19. Finally, in step 22, the server updates the model parameters for use in the subsequent learning round.

4.4 Regret Analysis

The present section analyzes the performance of OFMS-FT in terms of cumulative regret. To analyze the performance of OFMS-FT, it is supposed that the following assumptions hold:

(A1) For any $(\boldsymbol{x}, y)$ and $\boldsymbol{\theta}$, the loss is bounded as $0 \leq \mathcal{L}(f_k(\boldsymbol{x}; \boldsymbol{\theta}), y) \leq 1$.

Algorithm 12 OFMS-FT: Online Federated Model Selection and Fine-Tuning

```
1: **Input:** Models $f_k(\cdot;\boldsymbol{\theta}_{k,0})$, costs $c_k$, $b_k$ and budgets $B_i$, $E$, $\forall i$, $\forall k$.
2: **Initialize:** $z_{ik,1} = 1$, $\forall k$, $\forall i$.
3: **for** $t = 1, \ldots, T$ **do**
4:     **for all** $i \in [N]$, the $i$th client **do**
5:         Chooses a model using PMF $\boldsymbol{p}_{i,t}$ in (4.4). $I_{i,t}$ denote the index of the chosen model.
6:         Splits all models except for $I_{i,t}$-th model into clusters $\mathbb{D}_{i1,t}, \ldots, \mathbb{D}_{im_{i,t},t}$ such that
               $$\sum_{k\in\mathbb{D}_{ij,t}} c_k \leq B_i - c_{I_{i,t}}, \forall j : 1 \leq j \leq m_{i,t}.$$
7:         Chooses one cluster among $\{\mathbb{D}_{ij,t}\}_{j=1}^{m_{i,t}}$ uniformly at random where $J_{i,t}$ is the chosen
           cluster index.
8:         Constructs the set of model indices $\mathbb{S}_{i,t} = \{I_{i,t}\} \cup \{k|\forall k \in \mathbb{D}_{iJ_{i,t},t}\}$.
9:         Downloads all models whose indices are in $\mathbb{S}_{i,t}$ from the server.
10:        Sends the bandwidth cost $e_i = \sum_{k\in\mathbb{S}_{i,t}} b_k$ to the server.
11:    **end for**
12:    The server splits clients into $\alpha$ groups $\mathbb{N}_1, \ldots, \mathbb{N}_\alpha$ such that $\sum_{i\in\mathbb{N}_j} e_i \leq E, \forall j \in [\alpha]$.
13:    The server draws one of the groups $\{\mathbb{N}_j\}_{j=1}^{\alpha}$ uniformly at random.
14:    The server finds $\mathbb{G}_t := \mathbb{N}_{\iota_t}$ with $\iota_t$ as the chosen client group index.
15:    **for all** $i \in [N]$, the $i$th client **do**
16:        Makes prediction $f_{I_{i,t}}(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{I_{i,t},t})$ and computes $\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t})$, $\forall k \in \mathbb{S}_{i,t}$.
17:        Updates $z_{ik,t}$, $\forall k \in \mathbb{S}_{i,t}$ according to (4.8).
18:        **if** $i \in \mathbb{G}_t$ **then**
19:            Sends $\boldsymbol{\theta}_{ik,t+1}$, $\forall k \in \mathbb{S}_{i,t}$ obtained by (4.10) to the server.
20:        **end if**
21:    **end for**
22:    The server updates models parameters as in (4.11).
23: **end for**
```

(A2) The budget satisfies $B_i \geq c_k + c_j$, $\forall k, j \in [K]$, $\forall i \in [N]$.

(A3) The loss function $\mathcal{L}(f_k(\boldsymbol{x};\boldsymbol{\theta}), y)$ is convex with respect to $\boldsymbol{\theta}$, $\forall k \in [K]$.

(A4) The gradient of the loss function is bounded as $\|\nabla\mathcal{L}(f_k(\boldsymbol{x};\boldsymbol{\theta}), y)\| \leq G$, $\forall \boldsymbol{\theta}$, $\forall k \in [K]$. Also, $\boldsymbol{\theta}$ belongs to a bounded set such that $\|\boldsymbol{\theta}\|^2 \leq R$.

Let m^*_{ij} be the minimum number of clusters if client i splits all models except for model j such that the cumulative cost of each cluster does not exceed $B_i - c_j$. Define $\mu_i = \max_j m^*_{ij}$, which can be interpreted as the upper bound for the minimum number of clusters that can be constructed by client i at each learning round. It can be concluded that $\mu_i < K$ and increase in budget B_i leads to decrease in μ_i. The following theorem obtains the upper bound for the regret of clients and the server in terms of μ_i.

Theorem 4.1. *Assume that clients utilize the first fit decreasing algorithm in order to split models into clusters $\mathbb{D}_{i1,t}, \ldots, \mathbb{D}_{im_{i,t},t}$. Under (A1) and (A2), the expected cumulative regret of the i-th client using OFMS-FT is bounded by*

$$\mathcal{R}_{i,T} \leq \frac{\ln K}{\eta_i} + \eta_i \mu_i T, \tag{4.12}$$

which holds for all $i \in [N]$. Under (A1)–(A4), the cumulative regret of the server in fine-tuning model k using OFMS-FT is bounded by

$$\mathcal{S}_{k,T} \leq \frac{R}{2\eta_f} + \frac{1}{N}\sum_{i=1}^{N} \mu_i \alpha \eta_f G^2 T. \tag{4.13}$$

Proof. see Appendix D.2. □

If the i-th client sets $\eta_i = \mathcal{O}\left(\sqrt{\frac{\ln K}{\mu_i T}}\right)$, then the i-th client achieves sub-linear regret of

$$\mathcal{R}_{i,T} \leq \mathcal{O}\left(\sqrt{(\ln K)\mu_i T}\right). \tag{4.14}$$

If the server sets the fine-tuning learning rate as $\eta_f = \frac{1}{\sqrt{\frac{\alpha T}{N}\sum_{i=1}^{N}\mu_i}}$, then the server achieves regret of

$$\mathcal{S}_{k,T} \leq \mathcal{O}\left(\sqrt{\frac{\alpha T}{N}\sum_{i=1}^{N}\mu_i}\right). \tag{4.15}$$

The regret bounds in (4.14) and (4.15) show that a decrease in μ_i leads to tighter regret bound. If the i-th client has a larger budget B_i, the upper bound for the minimum number of model clusters μ_i decreases since client i can split models into clusters with larger budgets. As an example, consider the special case that all models have the same cost $c_k = c$, $\forall k \in [K]$ while $B_i = \delta_i c$ where $\delta_i \geq 2$ is an integer. In this case, according to step 6 in Algorithm 12, the upper bound for the minimum number of clusters is $\mu_i = \frac{(K-1)c}{(\delta_i - 1)c} = \frac{K-1}{\delta_i - 1}$. Therefore, as the budget of a client increases, the client can achieve tighter regret bound. In addition, larger communication bandwidth enables the server to partition clients into smaller number of groups α. Thus, according to (4.15), larger communication bandwidth leads to tighter regret bound for the server.

Challenges of Obtaining Regret in (4.15)**.** Limited memory of clients brings challenges for obtaining the sub-linear regret in (4.15) since clients cannot calculate the gradient loss of all models every learning round. To overcome this challenge, the present chapter proposes update rules (4.9) and (4.10) along with the novel model subset selection presented in steps 5, 6 and 7 of Algorithm 12. In what follows the effectiveness of the proposed update rules and model subset selection is explained. Employing vanilla online gradient descent update rule $\boldsymbol{\theta}_{ik,t+1} = \boldsymbol{\theta}_{k,t} - \nabla\mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t})$ locally by clients, can result in regret of $\mathcal{O}(\sqrt{T})$ if client i knows $\nabla\mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t})$, $\forall k \in [K]$ at every learning round (see e.g., [73]). This is not possible since client i has limited memory and is not able to calculate the gradient loss of all models every learning round. To overcome this challenge, the present chapter proposes the local update rules in (4.9) and (4.10) which use the gradient loss estimate $\nabla\hat{\ell}_{ik,t}$ instead of true loss gradient. Using (D.22) of Appendix D.2, it can be concluded that $\nabla\hat{\ell}_{ik,t}$ is an unbiased estimator of $\nabla\mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t})$ meaning that $\mathbb{E}_t[\nabla\hat{\ell}_{ik,t}] = \nabla\mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t})$. According to (4.9), obtaining $\nabla\hat{\ell}_{ik,t}$ does not require storing model k and calculating $\nabla\mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t})$ every learning round. Hence, $\nabla\hat{\ell}_{ik,t}$ can be obtained every learning round given the limited memory of client i. However, according to (D.21), (D.23) and (D.24) of Appendix D.2, employing update rule of (4.10) the regret of the server grows with $\mathbb{E}_t[\|\nabla\hat{\ell}_{ik,t}\|^2]$ which is upper bounded as $\mathbb{E}_t[\|\nabla\hat{\ell}_{ik,t}\|^2] \leq \frac{\alpha G^2}{q_{ik,t}}$ (see (D.22b) in Appendix D.2). Therefore, the regret of the server grows with $\frac{1}{q_{ik,t}}$ where $q_{ik,t}$ is the probability that client i stores model k and fine-tunes it at learning round t. Using model subset selection method presented in steps 5, 6 and 7 of Algorithm 12, the probability $q_{ik,t}$ is obtained as in (4.7). In (D.14) of Appendix D.2, it is proven that if clients employ FFD algorithm to cluster models in step 6, then it is guaranteed that $q_{ik,t} \geq \frac{1}{2\mu_i}$. This lead to guaranteeing the regret upper bound in (4.15).

4.5 Related Works and Discussions

This Section discusses differences, innovations, and improvements provided by the proposed OFMS-FT compared to related works in the literature.

Online Model Selection. Online model selection algorithms by [50, 114, 51, 16, 119, 128] have studied either *full-information* or *bandit* settings. Full-information refers to cases where the loss of all models can be observed at every round while in bandit setting only the loss of the chosen model can be observed. Regret bounds obtained by full-information based online model selection algorithms cannot be guaranteed if the learner (i.e., a client) cannot store all models. Moreover, it is useful to mention that the present chapter studies the adversarial setting where the losses observed by clients at each round are specified by the environment and may not follow any time-invariant distribution. It is well-known that in the adversarial bandit setting the learner achieves regret upper bound of $\mathcal{O}(\sqrt{KT})$ (see e.g., [119]). The proposed OFMS-FT utilizes the available memory of clients to evaluate a subset of models every round (see step 16 in Algorithm 12) which helps client i to achieve regret of $\mathcal{O}(\sqrt{\mu_i T})$ as presented in (4.14). If client i is able to store more than one model, then $\mu_i < K$ (see below (4.3) and the discussion below Theorem 4.1) which shows that OFMS-FT utilizes the available memory of clients to improve their regret bound compared to bandit setting. Moreover, aforementioned online model selection works have not studied online fine-tuning of models. A model selection algorithm proposed by [118] assumes that each model (called base learner) comes with a candidate regret bound and utilizes this information for model selection. By contrast, the present chapter assumes that there is no available prior information about the performance of models. Moreover, [115] has studied the problem of model selection in linear contextual bandits where the reward (can be interpreted as negative loss in our work) is the linear function of the context (can be interpreted as $\boldsymbol{x}_{i,t}$ in our work). However, the present chapter does not make this assumption in both theoretical analysis and experiments. Furthermore, online model selection when models are kernels has been studied in the literature (see e.g., [154, 163, 94]) where the specific characteristics of kernel functions are exploited to perform model selection to alleviate computational complexity of kernel learning.

Online Learning with Partial Observations. Another line of research related to the focus of the present chapter is online learning with expert advice where a learner interacts with a set of experts such that at each learning round the learner makes decision based on advice received from the experts [19]. The learner may observe the loss associated with a subset of experts after decision making, which can be modeled using a graph called feedback graph [106, 5, 29, 3, 34]. In online federated model selection, each model can be viewed as an expert. Employing the proposed OFMS-FT, in addition to performing online model selection, clients and the server collaborate to fine-tune the models (experts). However, the aforementioned online learning algorithms do not study the case where the learner can influence experts. Performing online model selection and fine-tuning jointly in a federated fashion brings challenges for guarantying sub-linear regret that cannot be overcame using the existing online learning algorithms. Specifically, due to limited client-to-server communication bandwidth and limited memory of clients, all clients are not able to fine-tune all models every learning round. The proposed OFMS-FT introduces a novel model subset selection in steps 5, 6 and 7 of Algorithm 12 and a novel update rule in (4.9) and (4.10) to fine-tune models locally by clients in such a way that given limited memory of clients and limited communication bandwidth, the server achieves sub-linear regret of (4.15).

Online Federated Learning. The problem of online federated model selection and fine-tuning is related to online federated learning [24, 112, 37]. [24] has studied learning a global model when clients receive new data samples while they participate in federated learning. Online decision-making by clients has not been studied by [24] and hence the regret bound for clients cannot be guaranteed. An online federated learning algorithm has been proposed by [37] to cope with the staleness in federated learning. However, it lacks theoretical analysis when clients need to perform online decision-making. An online mirror descent-based federated learning algorithm called Fed-OMD has been proposed in [112]. Fed-OMD obtains sub-linear regret when clients perform their online learning task while collaborating with the server to learn a single model. However, Fed-OMD cannot guarantee sub-linear regret when it comes to performing online model selection if clients are unable to store all models in the dictionary. Furthermore, [78, 66] have studied the problem of online federated learning where each client learns a kernel-based model employing specific characteristics of kernel functions.

Personalized Model Selection and Fine-Tuning. In addition to online model selection, online learning and online federated learning discussed in Section 4.5, personalized federated learning can be related to the focus of this chapter. Employing the proposed OFMS-FT, model selection and fine-tuning is personalized for clients. According to step 5 in Algorithm 12, each client chooses a model locally using the personalized PMF $\boldsymbol{p}_{i,t}$ in (4.4) to make a prediction at round t. This helps each client i, $\forall i \in [N]$ to achieve the sub-linear regret in (4.14). Furthermore, the choice of models to be fine-tuned locally by each client is personalized according to step 19 in Algorithm 12. Particularly, $q_{ik,t}$ in (4.7) is the probability that the client i fine-tunes the model k at round t. The probability $q_{ik,t}$ is determined by client i and it can be inferred that the probability to participate in fine-tuning a model is determined by client i based on its preferences given the limited memory budget. It is useful to add that personalized federated learning is well-studied topic related to the focus of the present chapter. In personalized federated learning framework, aggregating information from clients the server assists clients to learn their own personalized model. Several personalized federated learning approaches have been proposed in the literature for example inspired by model-agnostic meta-learning [49, 46, 1], adding a regularization term to the objective function [72, 42, 96, 103], among others (see e.g., [39, 30, 107, 135]). However, none of the aforementioned works have studied online decision making, online federated model selection and fine-tuning when clients have limited memory and employing them, sub-linear regrets cannot be guaranteed for clients and the server.

Client Selection. Client selection in federated learning has been extensively explored in the literature [23, 79, 9, 116, 52]. However, none of the aforementioned works have specifically studied client selection for online federated learning, where clients utilize the trained model from federated learning for online predictions. In the proposed OFMS-FT, the server selects clients for their participation in model fine-tuning uniformly at random. This choice aims to avoid differentiating among clients and fine-tune models in the favor of any clients. Nevertheless, an intriguing direction for future research is to investigate how alternative client selection strategies, beyond uniform selection, could enhance client regret in the context of online federated learning.

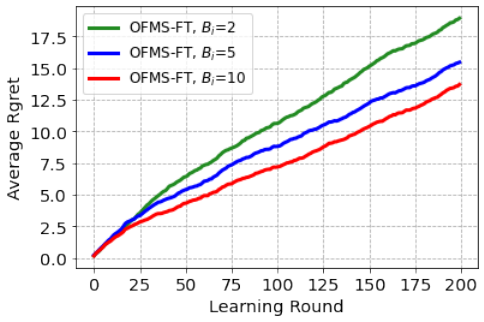
4.6 Experiments

We tested the performance of the proposed OFMS-FT for online model selection through a set of experiments. The performance of OFMS-FT is compared with the following baselines: MAB [8], Non-Fed-OMS, RMS-FT, B-Fed-OMFT, FedOMD [112] and PerFedAvg [46]. MAB refers to the case where the server chooses a model using Exp3 algorithm [8] and transmits the chosen model to all clients. Then, each client sends the loss of the received model to the server. Non-Fed-OMS refers to **non-fed**erated **o**nline **m**odel **s**election where each client stores a fixed subset of models that can be fit into its memory. At each learning round, each client chooses one model from the stored subset of models using Exp3 algorithm. RMS-FT denote a baseline where at each learning round each client chooses a subset of models uniformly at random to fine-tune them. The prediction task is then carried out by selecting one of the chosen models uniformly at random. Furthermore, B-Fed-OMFT stands for Budgeted Federated Online Model Fine-Tuning. In this approach, the server maintains a set of models that can be fit into the memory of all clients. Clients collaborate with the server to fine-tune all models in each learning round. In the B-Fed-OMFT framework, each client employs the Exp3 algorithm to choose one model to perform the prediction task. Using Fed-OMD [112], given a pre-trained model, clients and the server fine-tune the model. PerFedAvg refers to the case where given a pre-trained model, clients and the server fine-tune the model using Personalized FedAvg [46]. The performance of the proposed OFMS-FT and baselines are tested on online image classification and online regression tasks. Image classification tasks are performed over CIFAR-10 [90] and MNIST [92] datasets. Online regression tasks are performed on Air [161] and WEC [117] datasets. For both image classification datasets, the server stores 20 pre-trained convolutional neural networks (CNNs). Based on the number of parameters required to store models, for CIFAR-10 the normalized costs of storing CNNs are either 0.89 or 1, and for MNIST the normalized costs are either 0.66 or 1. The experiments were conducted with a single meta-replication, utilizing a consistent random seed for both the proposed OFMS-FT and all baseline methods. Moreover, for both regression datasets, the server stores 20 pre-trained fully-connected feedforward neural networks. Since all neural networks have the same size, the normalized costs of all of them are 1. More details about models and datasets can be found in Appendix D.3.

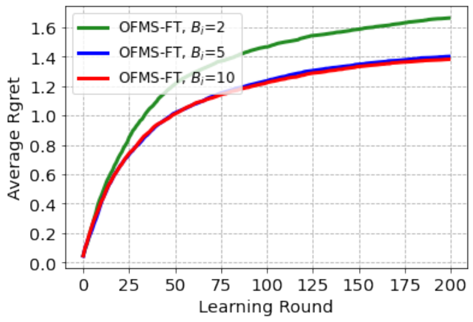
There are 50 clients performing image classification task, and 100 clients performing online regression task. Note that clients are performing the learning task in an online fashion such that at each learning round each client observes one data sample and predicts its label in real time. The learning rates η_i for all methods are set to be $10/\sqrt{T}$ where $T = 200$. Furthermore, the fine-tuning learning rate is set to $\eta_f = 10^{-3}/\sqrt{T}$. Since the required bandwidth to send a model is proportional to the model size, the normalized bandwidth b_k associated with the

Table 4.1: Average and standard deviation of clients' accuracy over CIFAR-10 and MNIST datasets. MSE ($\times 10^{-3}$) and its standard deviation ($\times 10^{-3}$) across clients over Air and WEC datasets.

Algorithms	CIFAR-10	MNIST	Air	WEC
MAB	$61.14\% \pm 10.12\%$	$84.37\% \pm 5.18\%$	8.97 ± 6.82	36.96 ± 8.52
Non-Fed-OMS	$65.76\% \pm 12.24\%$	$85.06\% \pm 7.30\%$	8.82 ± 6.70	64.84 ± 40.30
RMS-FT	$57.86\% \pm 9.75\%$	$88.33\% \pm 4.25\%$	7.87 ± 5.22	18.89 ± 4.02
B-Fed-OMFT	$67.83\% \pm 12.28\%$	$89.46\% \pm 4.71\%$	8.09 ± 5.55	31.94 ± 7.97
Fed-OMD	$64.34\% \pm 9.89\%$	$88.69\% \pm 5.16\%$	11.37 ± 7.16	30.89 ± 10.06
PerFedAvg	$55.65\% \pm 11.94\%$	$89.71\% \pm 4.93\%$	11.29 ± 7.08	30.09 ± 10.17
OFMS-FT	$\mathbf{76.77\% \pm 4.46\%}$	$\mathbf{92.05\% \pm 2.69\%}$	$\mathbf{7.46 \pm 5.10}$	$\mathbf{7.09 \pm 1.67}$



(a) MNIST



(b) WEC

Figure 4.1: Average regret of clients using OFMS-FT with the change in budget B_i.

k-th model are considered to be the same as the normalized cost c_k.

Table 4.1 demonstrates the average and standard deviation of clients' accuracy over CIFAR-10 and MNIST when the test set is distributed in non-i.i.d manner among clients. For CIFAR-10, each client receives 155 testing data samples from one class and 5 samples from each of the other nine classes. In the case of MNIST, each client receives at least 133 samples from one class and at least 5 samples from the other classes. The 200 testing data samples are randomly shuffled and are sequentially presented to each client over $T = 200$ learning rounds. The accuracy of the client i is defined as $\text{Accuracy}_i = \frac{1}{T}\sum_{t=1}^{T} \mathcal{I}(\hat{y}_{i,t} = y_{i,t})$ where $\hat{y}_{i,t}$ denote the class label predicted by the algorithm. The memory budget is $B_i = 5$, $\forall i \in [N]$. Each client is able to store up to 50 images. In order to fine-tune models, clients employ the last 50 observed images. Moreover, Fed-OMD and PerFedAvg fine-tune one of the 20 pre-trained models such that both fine-tune the same pre-trained model. As can be observed from Table 4.1, the proposed OFMS-FT achieves higher accuracy than Non-Fed-OMS and B-Fed-OMFT. This corroborates that having access to larger number of models helps learners to achieve better learning performance, especially in cases where the learners are faced with heterogeneous data, and performance of models are unknown in priori. Moreover, from Table 4.1 it can be seen that OFMS-FT achieves higher accuracy than MAB which admits the effectiveness of observing losses of multiple models at each learning round. Higher accuracy of OFMS-FT compared with Fed-OMD and PerFedAvg indicates the benefit of fine-tuning multiple models rather than one. The superior performance of OFMS-FT in comparison to RMS-FT highlights the efficacy of employing model selection through the PMF defined in equation (4.4), as opposed to choosing models uniformly at random. In addition, as can be seen from Table 4.1, the standard deviation of clients' accuracy associated with OFMS-FT is considerably lower than other baselines. This shows that using OFMS-FT, accuracy across clients shows less variations compared with other baselines. Therefore, the results confirm that OFMS-FT can cope with heterogeneous data of clients in more flexible and henceforth more effective fashion.

Furthermore, Table 4.1 presents the mean square error (MSE) of online regression and its standard deviation across clients for Air and WEC datasets. Specifically, MSE of client i is defined as $\text{MSE}_i = \frac{1}{T}\sum_{t=1}^{T}(\hat{y}_{i,t} - y_{i,t})^2$. In both the Air and WEC datasets, individual data samples are associated with one of four geographical areas. The distribution of data samples across clients is non-i.i.d, with 50 clients observing data samples from one specific site, while the remaining 50 clients observe data samples from another geographical site. At each round, only half of clients are able to send their updated models to the server. All other settings are the same as online image classification setting. Results for online regression tasks are consistent with the conclusions obtained from the results of online image classification task. Moreover, Figure 4.1 illustrates the average regret of clients using the proposed OFMS-FT

Table 4.2: MSE ($\times 10^{-3}$) and its standard deviation ($\times 10^{-3}$) of OFMS-FT across clients over WEC dataset under varying budgets among clients.

$B_i = 3$	$B_i = 5$	MSE
60%	40%	7.96 ± 1.49
50%	50%	7.85 ± 1.66
40%	60%	7.32 ± 1.54

through learning rounds for different values of memory budget B_i. Figure 4.1 depicts the regret of OFMS-FT through learning on MNIST and WEC datasets. As can be seen, the increase in B_i leads to obtaining lower regret by clients. Therefore, the results in Figure 4.1 are in agreement with the regret analysis in Section 4.4. Table 4.2 illustrates the sensitivity of the MSE and its standard deviation, as achieved by the proposed OFMS-FT, to the budget B_i ($\forall i \in [N]$) over the WEC dataset. In this configuration, the budget varies across clients, with a subset having $B_i = 3$ and the remainder $B_i = 5$. The improvement in MSE becomes evident as the number of clients with a budget of $B = 5$ increases. This observation indicates that an increase in budget enhances the performance of OFMS-FT, aligning with the theoretical findings presented in Section 4.4.

4.7 Conclusion

Performing online model selection with a large number of models can improve the performance of online model selection especially when there is not prior information about models. The present chapter developed a federated learning approach (OFMS-FT) for online model selection when clients cannot store all models due to limitations in their memory. To adapt models to clients' data, employing OFMS-FT clients can collaborate to fine-tune models. It was proved that both clients and the server achieve sub-linear regret with respect to the best model in hindsight. Experiments on regression and image classification datasets were carried out to showcase that the proposed OFMS-FT achieved better performance in comparison with non-federated online model selection approach and other state-of-the-art federated learning algorithms which employ a single model rather than a dictionary of models.

Chapter 5

Personalized Federated Learning with Mixture of Models

5.1 Introduction

Federated learning enables a group of learners, known as *clients*, to collaborate and collectively train a model under the coordination of a central *server*, without revealing their data. In this framework, clients perform local model updates and share these updates with the server. By aggregating these local updates, the server globally updates the model. Many prior works in the literature have assumed that each client stores a batch of training data and updates models locally based on this stored data (see e.g., [113, 97, 129, 20, 107, 149]). However, in some cases, clients may need to make real-time predictions, and streams of data arrive sequentially, making it challenging to store and process data in batch. Furthermore, if clients operate in a non-stationary and dynamic environment, employing pre-trained models may fall short in prediction accuracy, requiring clients to fine-tune their models to adapt to their data.

A federated learning framework, commonly referred to as online federated learning [112, 78], specifically addresses situations where clients engage in real-time predictions using a shared global model. After making predictions, clients collaborate with the server to update the global model for subsequent predictions in the future. Specifically, after making predictions, clients incur losses and based on the incurred losses, clients update the model locally and send their local updates to the server. The server updates the global model upon aggregating local updates and then distributes the updated global model to clients for use in the ongoing online prediction task. In this context, the performance of clients can be assessed using the notion of *regret* [19, 8]. The regret of a client at a given time step is defined as the difference between its prediction loss and that of the best model in hindsight. The best model in hindsight is determined as the model that achieves the minimum total prediction loss over time across all clients' data. The primary objective is to minimize the cumulative regret of all clients over time.

In the literature, federated learning algorithms based on online gradient descent have been proposed that achieve sub-linear regret upper bounds [112, 78]. This suggests that over the long run, these algorithms perform as well as the best model in hindsight. However, in Section 5.3, this chapter demonstrates that these federated learning algorithms do not obtain tighter regret bounds compared to the scenario where each client learns its own model locally without participating in federated learning. This indicates that participation in federated learning may not provide any benefit for clients performing online prediction. Specifically, if data is distributed non-i.i.d. among clients and data distributions are time-variant and not known a priori, it is likely that local model training by clients will achieve better prediction accuracy than participation in federated learning. Although the benefit of federated learning in online prediction is not evident in existing theoretical analyses, models learned through federated learning enjoy higher generalizability as they are trained on all data samples distributed among clients. This motivates the idea that combining the model learned through federated learning with the locally learned one may prove effective in scenarios where clients need to perform online prediction.

This chapter proposes the Fed-POE (Federated Learning with Personalized Online Ensemble) algorithm, through which each client constructs a personalized model for online prediction by adaptively ensembling the locally learned model and the model learned through federated learning. With Fed-POE, clients can adapt their local models to their data while benefiting from the higher generalizability of the federated model. Theoretical analysis for convex cases demonstrates that Fed-POE achieves sublinear regret bounds with respect to both the best federated and local models in hindsight. This indicates that Fed-POE effectively leverages the advantages of both federated and local models. Providing such theoretical guarantees may not be feasible for non-convex models such as neural networks. These models may suffer from the forgetting process [144, 125], where fine-tuning on streaming data 'on the fly' can lead to forgetting previously observed data samples. To overcome this challenge, the present chapter proposes a novel framework in which the server periodically stores federated model parameters over time. Each client adaptively selects a personalized subset of stored models on the server based on the performance of these models in the client's online prediction task. Clients then use the selected models, along with the federated and local models, to construct an ensemble model for prediction. Clients select a subset of models to both control the memory and computational complexity of prediction and to prune models with relatively lower accuracy, thereby improving prediction performance. Theoretical analysis proves that Fed-POE achieves sublinear regret with respect to the best model in hindsight among the local model, federated model, and all models stored by the server. The contributions of the present chapter are summarized as follows:

- Fed-POE enables clients to utilize the advantages of both local and federated models for online prediction tasks.
- To address the issue of forgetting in online prediction, Fed-POE introduces a novel federated framework for collaboration between clients and the server.

- Theoretical analyses for both convex and non-convex cases prove that Fed-POE achieves sublinear regret with respect to the best model in hindsight.
- Extensive experiments on regression and image classification datasets show that Fed-POE effectively leverages the benefits of local and federated models, achieving higher online prediction accuracy compared to state-of-the-art federated learning algorithms.

It is worth noting that the materials in this chapter are included in [65].

5.2 Related Works

Personalized Federated Learning. Personalized federated learning involves developing individualized models for each client, derived from a global model learned through federated learning. Several personalized federated learning algorithms have been proposed in the literature [143, 21, 151, 155]. In [72, 42, 96, 103], clients construct their personalized models by adding a regularization term to the local objective and using the global federated model. Algorithms in [46, 1] allow clients to fine-tune the global federated model using model-agnostic meta-learning [49] to learn their personalized models. With Fed-Rep [30], each client generates a representation using the global model and learns its own local head for prediction. Algorithms in [39, 100, 159] enable clients to achieve personalized models by combining local and global models. However, none of these works have addressed the problem of online prediction while clients collaborate on training their personalized models.

Federated Learning with Streaming Data. Several studies have explored the problem of federated learning when clients receive a stream of data in real time. While [24, 109, 108] investigate federated learning scenarios where clients receive new training data in each learning round, they do not address the aspect of online prediction by clients. Consequently, these works cannot provide regret guarantees for online prediction. Additionally, [37] studies the issue of staleness in online federated learning to both train and make prediction with the model; however, it lacks rigorous theoretical analysis. In [112], an online federated learning algorithm is introduced, utilizing online mirror descent, with a proven sublinear regret bound. Similarly, [121, 53] propose online federated learning algorithms with guaranteed sublinear regret. Furthermore, [122] analyzes the benefits of collaboration in online federated learning, particularly in scenarios where only loss values at queried points are available to clients, without access to loss gradients. Regarding online federated kernel learning, algorithms are introduced in [55, 67], albeit without accompanying regret analysis, leading to an absence of guaranteed regret bounds. In [78], multiple kernel-based models and random feature-based online federated kernel learning algorithms are proposed, with demonstrated sublinear regret.

5.3 Preliminaries

This section explains the problem of federated learning for real-time prediction and model training. The present section studies the cases where clients either collaborate in federated learning or employ online gradient descent methods to locally train the model in real-time. This study highlights the motivation behind the proposed ensemble approach to federated learning.

5.3.1 Online Prediction and Federated Learning

Let there are N clients interact with a server to train a model $f(\cdot;\cdot)$. Also, let $[N] := \{1, \ldots, N\}$. At each time step t, client i, $\forall i \in [N]$ receives a data sample $\boldsymbol{x}_{i,t} \in \mathbb{R}^d$ and makes the prediction $f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t)$ where $\boldsymbol{\theta}_t$ denotes the parameter of the model at time step t. Note that generalization to the scenario where at each time step, each client receives a dataset instead of a single data sample is straightforward. After making prediction, client

i, $\forall i \in [N]$ observes label $y_{i,t}$. Then client i, $\forall i \in [N]$ computes the loss of its prediction $\mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t), y_{i,t})$ where $\mathcal{L}(\cdot, \cdot)$ denotes the loss function. After computing the loss, clients send their updates to the server (e.g., by sending the gradient loss $\nabla \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t), y_{i,t})$) and the server aggregates information from clients to update $\boldsymbol{\theta}_t$ to $\boldsymbol{\theta}_{t+1}$ to be used by clients to make predictions at time step $t+1$. For ease of presentation, it is assumed that all clients can send their updates to the server every time step. Generalizing the results for the cases where only a fraction of clients can send their updates is straightforward. Furthermore, it is assumed that the label $y_{i,t}$ for any i and t is determined by the environment through a process unknown to the clients. This implies that the data distribution observed by client i can be non-stationary, and client i, $\forall i \in [N]$ does not know the distribution. Additionally, the data distribution can differ across clients. The goal is to enable clients to collaborate with the server to minimize the cumulative regret of clients over time. The regret of client i at time step t is defined as the difference between the prediction loss $\mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t), y_{i,t})$ and the loss $\mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}^*), y_{i,t})$ corresponding to the model with the optimal parameter $\boldsymbol{\theta}^*$. Therefore, the average cumulative regret of clients up to time horizon T is defined as

$$R_T = \frac{1}{N} \sum_{t=1}^{T} \sum_{i=1}^{N} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t), y_{i,t}) - \frac{1}{N} \sum_{t=1}^{T} \sum_{i=1}^{N} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}^*), y_{i,t}) \tag{5.1}$$

where $\boldsymbol{\theta}^*$ denotes the optimal model parameter in hindsight and can be expressed as

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{t=1}^{T} \sum_{i=1}^{N} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}), y_{i,t}). \tag{5.2}$$

One common way to solve the problem is that clients employ online gradient descent to update models locally and the server exploits federated averaging [98, 112] to update the global model.

5.3.2 Federated Learning with Online Gradient Descent

At each time step t, client i obtains the locally updated model parameter $\boldsymbol{\psi}_{i,t+1}$ as follows:

$$\boldsymbol{\psi}_{i,t+1} = \boldsymbol{\theta}_t - \eta \nabla \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t), y_{i,t}) \tag{5.3}$$

where η is the learning rate. Aggregating locally updated model parameters, the server obtains the updated global model parameter for time step $t+1$ as

$$\boldsymbol{\theta}_{t+1} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\psi}_{i,t+1}. \tag{5.4}$$

This continues up until the time horizon T. This chapter examines regret under some or all of the following assumptions:

(A1) The loss function $\mathcal{L}(f(\boldsymbol{x};\boldsymbol{\theta}), y)$ is convex with respect to $\boldsymbol{\theta}$.

(A2) The gradient of the loss is bounded as $\|\nabla \mathcal{L}(f(\boldsymbol{x};\boldsymbol{\theta}), y)\| \leq G$.

(A3) For any $\boldsymbol{x}$, $\boldsymbol{\theta}$, the loss is bounded as $0 \leq \mathcal{L}(f(\boldsymbol{x};\boldsymbol{\theta}), y) \leq 1$.

The following theorem specifies the regret bound for federated learning employing online gradient descent under (A1) and (A2).

Theorem 5.1. *Employing online gradient descent, the following cumulative regret upper bound is guaranteed for federated learning under assumptions (A1) and (A2):*

$$\frac{1}{N}\sum_{t=1}^{T}\sum_{i=1}^{N} \mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t), y_{i,t}) - \frac{1}{N}\sum_{t=1}^{T}\sum_{i=1}^{N} \mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}^*), y_{i,t}) \leq \frac{\|\boldsymbol{\theta}^*\|^2}{2\eta} + \frac{\eta}{2}G^2 T. \tag{5.5}$$

Proof of Theorem 5.1 can be found in Appendix E.1. According to Theorem 5.1, choosing $\eta = \mathcal{O}(\sqrt{1/T})$, the cumulative regret in (5.5) is bounded from above as $\mathcal{O}(\sqrt{T})$. If the time horizon T is unknown, the doubling trick technique (see e.g., [4]) can be effectively used to set the learning rate to maintain theoretical guarantees. Consider the case where each client learns the model locally using online gradient descent without collaborating with other clients and the server. Let $\boldsymbol{\phi}_{i,t}$ denote the local model parameter learned by client i at time step t, employing online gradient descent locally. The regret of client i in this case is equivalent to federated learning regret where there is only one client. Therefore, substituting $N = 1$ and $\boldsymbol{\theta}_0 = \mathbf{0}$ in (E.7) of Appendix E.1, for the cumulative regret of client i with respect to any $\boldsymbol{\theta}$, we get

$$\sum_{t=1}^{T} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}_{i,t}), y_{i,t}) - \sum_{t=1}^{T} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}), y_{i,t}) \leq \frac{\|\boldsymbol{\theta}\|^2}{2\eta} + \frac{\eta}{2} G^2 T. \tag{5.6}$$

Averaging (5.6) over all clients and substituting $\boldsymbol{\theta}$ with $\boldsymbol{\theta}^*$ in (5.6), we obtain

$$\frac{1}{N} \sum_{t=1}^{T} \sum_{i=1}^{N} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}_{i,t}), y_{i,t}) - \frac{1}{N} \sum_{t=1}^{T} \sum_{i=1}^{N} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}^*), y_{i,t}) \leq \frac{\|\boldsymbol{\theta}^*\|^2}{2\eta} + \frac{\eta}{2} G^2 T. \tag{5.7}$$

Comparing (5.5) with (5.7), it can be inferred that under assumptions (A1) and (A2), federated learning does not achieve tighter regret bound than the case where each client independently learns its local model. Hence, from a theoretical standpoint, it remains uncertain whether collaboration in federated learning yields any improvement over local online model training. Intuitively, collaboration in federated learning may prove advantageous when there exists similarity among data samples observed by clients over time. However, in cases where data distribution is heterogeneous and such similarities are lacking, employing online local training may yield superior results for a client. Given the lack of prior information on data distribution in online scenarios, each client can independently assess over time whether utilizing the model learned through federated learning for predictions is beneficial.

Furthermore, according to assumption (A1), the theoretical guarantees obtained in (5.5) and (5.7) hold if the loss is convex with respect to the model parameter. However, in the case of non-convex models, such as neural networks, these theoretical guarantees are not applicable. Non-convex models, like neural networks, may encounter the forgetting process [144, 125], where the model tends to overfit to recently observed data samples. Consequently, it may not be feasible to derive a single model parameter using online gradient descent that achieves sublinear regret with respect to the best model in hindsight. The subsequent section introduces a novel algorithm aimed at assisting clients in addressing such scenarios.

5.4 Personalized Federated Learning Methods

The current section introduces personalized federated learning algorithms where each client dynamically learns the prediction performance of both the models trained through federated learning over time and the locally learned model.

5.4.1 Ensemble Learning

At each time step, each client constructs an ensemble model comprising the federated model and its locally learned model to make a prediction. Let $\boldsymbol{\phi}_{i,t}$ be the model parameter locally learned by client i such that at each time step t, client i updates $\boldsymbol{\phi}_{i,t}$ via gradient descent as

$$\boldsymbol{\phi}_{i,t+1} = \boldsymbol{\phi}_{i,t} - \eta \nabla \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}_{i,t}), y_{i,t}). \tag{5.8}$$

Furthermore, let clients and the server collaborate to update the *federated* model parameter $\boldsymbol{\theta}_t$ as outlined in (5.3) and (5.4). At time step t, client i makes the prediction for $\boldsymbol{x}_{i,t}$ using its personalized ensemble model $f_{i,t}(\cdot)$ expressed as

$$f_{i,t}(\boldsymbol{x}_{i,t}) = \frac{\alpha_{i,t}}{\alpha_{i,t} + \beta_{i,t}} f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t) + \frac{\beta_{i,t}}{\alpha_{i,t} + \beta_{i,t}} f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}_{i,t}) \tag{5.9}$$

where $\alpha_{i,t}$ and $\beta_{i,t}$ represent the weights assigned by client i to the federated and local models, respectively, indicating the credibility of predictions from each model. After making predictions and observing the label $y_{i,t}$, client i computes the loss of predictions from the federated and local models, updating the weights $\alpha_{i,t}$ and $\beta_{i,t}$ using the multiplicative update rule as follows:

$$\alpha_{i,t+1} = \alpha_{i,t} \exp\left(-\eta_c \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t), y_{i,t})\right), \tag{5.10a}$$

$$\beta_{i,t+1} = \beta_{i,t} \exp\left(-\eta_c \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}_{i,t}), y_{i,t})\right) \tag{5.10b}$$

where η_c is a learning rate. Client i initializes $\alpha_{i,1} = 1$ and $\beta_{i,1} = 1$. The proposed algorithm is personalized since according to (5.9), each client constructs its own ensemble model to perform prediction. The personalized regret of client i is defined as $C_{i,T} = \sum_{t=1}^{T} \mathcal{L}(f_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t}) - \sum_{t=1}^{T} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}_i^*), y_{i,t})$ where $\boldsymbol{\phi}_i^*$ denotes the best hindsight model parameter for client i which can be expressed as $\boldsymbol{\phi}_i^* = \arg\min_{\boldsymbol{\phi}} \sum_{t=1}^{T} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}_i), y_{i,t})$. The following theorem establishes the personalized regret upper bound for client i as well as global regret of all clients with respect to the best model parameter in hindsight.

Theorem 5.2. *Under assumptions (A1)–(A3), employing the ensemble model in* (5.9) *for online prediction, the global regret of all clients is bounded from above as*

$$\begin{aligned}&\frac{1}{N}\sum_{t=1}^{T}\sum_{i=1}^{N} \mathcal{L}(f_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t}) - \frac{1}{N}\sum_{t=1}^{T}\sum_{i=1}^{N} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}^*), y_{i,t}) \\ \leq& \frac{\|\boldsymbol{\theta}^*\|^2}{2\eta} + \frac{\ln(2)}{\eta_c} + \frac{\eta}{2}G^2 T + \frac{\eta_c T}{2}\end{aligned} \tag{5.11}$$

while client i achieves the following personalized regret upper bound:

$$\sum_{t=1}^{T} \mathcal{L}(f_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t}) - \sum_{t=1}^{T} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}_i^*), y_{i,t}) \leq \frac{\|\boldsymbol{\phi}_i^*\|^2}{2\eta} + \frac{\ln(2)}{\eta_c} + \frac{\eta}{2}G^2 T + \frac{\eta_c T}{2}. \tag{5.12}$$

Proof of Theorem 5.2 is given in Appendix E.2. According to (5.11) and (5.12) in Theorem

5.2, setting $\eta = \mathcal{O}(1/\sqrt{T})$, $\eta_c = \mathcal{O}(1/\sqrt{T})$ both the global regret for all clients and the personalized regret of each client i achieve a regret bound of $\mathcal{O}(\sqrt{T})$. This demonstrates that while the ensemble model in (5.9) ensures personalized regret guarantees for each client with respect to its best model in hindsight, it also enables clients to leverage federated learning, thus enjoying sublinear regret in comparison to the best global model in hindsight.

Comparison with Federated and Local Models. The main advantage of using the ensemble model instead of the federated model lies in Theorem 5.2, where it is shown that the ensemble model can attain the global regret guarantee provided by the federated model while employing the federated model, achieving the personalized regret guarantee in (5.12) is not feasible. However, according to (5.6) and (5.7), using the online local training, each client achieves sublinear regret with respect to its best model while all clients achieve sublinear regret with respect to the best global model in hindsight. This efficacy of local training stems from clients adapting the model to their individual data. In contrast, in federated learning, the model is trained on all data samples across clients, potentially leading to higher generalizability compared to its local counterpart. If there are similarities in the distribution of data samples among clients over time, the federated model is anticipated to achieve greater accuracy in online prediction. However, due to the lack of available information regarding the relationships between data samples observed by clients before online prediction, these advantages may not be reflected in theoretical bounds. The proposed method constructs an ensemble to harness the advantages of both federated and local models for online prediction, as evidenced by Theorem 5.2. Experimental results in Subsection in 5.5.1 confirm that the ensemble model achieves superior performance compared to both local and federated models.

5.4.2 Model Selection

Regret guarantees for the ensemble method, as outlined in Theorem 5.2, are contingent upon the model's convexity. However, if the model is non-convex, achieving such guarantees may not be feasible. Particularly, non-convex models such as neural networks are susceptible to the forgetting process [144, 125], wherein applying online gradient descent may lead to overfitting to recently observed data samples. This section introduces a novel algorithm that allows clients to make online predictions using non-convex models while simultaneously collaborating to fine-tune the model. The scenario assumes the existence of a pre-trained model, and the objective for clients is two-fold: to make real-time predictions and to refine the model for alignment with their preferences. This situation may arise, for instance, in fine-tuning large foundation models to tailor them to client preferences.

Let the server and clients collaborate to fine-tune the non-convex model $f(\cdot;\cdot)$. At each time step t, client i updates the model on the batch of recently observed samples with size b as

$$\boldsymbol{\psi}_{i,t+1} = \boldsymbol{\theta}_t - \frac{\eta}{b}\sum_{\tau=t-b}^{t} \nabla\mathcal{L}(f(\boldsymbol{x}_{i,\tau};\boldsymbol{\theta}_t), y_{i,\tau}). \tag{5.13}$$

Then the server aggregates locally updated parameters and updates the federated model parameter as in (5.4). Furthermore, each clients learns its own local model by fine-tuning the pre-trained model locally via online gradient descent as in (5.8) on the batch of b recently observed samples. While online gradient descent methods are well-known for their efficiency in handling dynamic environments, employing the update rule of (5.13) for non-convex models may lead to overfitting to recently observed batches. To mitigate potential forgetting, the server saves the federated model parameters every n time step, where n is an integer hyperparameter.

Let $\mathbb{D}_t$ represent the set of model parameters stored by the server at time step t. At time step $\tau = (j-1)n+1$, the server adds $\boldsymbol{\theta}_\tau$ to $\mathbb{D}_\tau$ meaning that $\mathbb{D}_{\tau+1} = \mathbb{D}_\tau \cup \{\boldsymbol{\theta}_\tau\}$. Let $\boldsymbol{\rho}_j$

Algorithm 13 Model selection by client i at time step t

```
1: Input: $\mathbb{D}_t$, $M$, wights $\{w_{ij,t}\}_{j=1}^{|\mathbb{D}_t|}$ and $\mathbb{M}_{i,t} = \varnothing$.
2: for $m = 1, \dots, M$ do
3:     Sample model index $k$ according to the PMF $p_{ij,t} = \frac{w_{ij,t}}{\sum_{j=1}^{|\mathbb{D}_t|} w_{ij,t}}$, $\forall j \in [|\mathbb{D}_t|]$.
4:     if $k \notin \mathbb{M}_{i,t}$ then
5:         Append model index $k$ to $\mathbb{M}_{i,t}$.
6:     end if
7: end for
8: Output: $\mathbb{M}_{i,t}$.
```

denotes the j-th model parameter in $\mathbb{D}_t$. It can be concluded that $\boldsymbol{\rho}_j = \boldsymbol{\theta}_{(j-1)n+1}$. The server continues saving model parameters every n time steps until time step $U \leq T$. Client i assigns a weight $w_{ij,t}$ to the j-th model in $\mathbb{D}_t$, which assesses the credibility of the predictions given by the model parameter $\boldsymbol{\rho}_j$. Client i initializes $w_{ij,1} = 1$. At each time step t, client i selects M models with replacement from $\mathbb{D}_t$ to construct its model set $\mathbb{M}_{i,t}$. Algorithm 13 illustrates the model selection process conducted by client i at time step t. During each round of model selection, client i chooses a model according to a probability mass function (PMF) proportional to weights $\{w_{ij,t} | 1 \leq j \leq |\mathbb{D}_t|\}$ where $|\cdot|$ denotes the cardinality of a set (see step 3 in Algorithm 13). Client i adds the chosen model to $\mathbb{M}_{i,t}$ if it is not already present. Therefore, it can be concluded that $|\mathbb{M}_{i,t}| \leq M$, $\forall i, t$. Then at each time step t, client i downloads all models in $\mathbb{M}_{i,t}$ from the server. Upon receiving the models, client i constructs an ensemble model $\tilde{f}_{i,t}(\cdot)$ as

$$\tilde{f}_{i,t}(\boldsymbol{x}) = \sum_{j \in \mathbb{M}_{i,t}} \frac{w_{ij,t}}{\sum_{m \in \mathbb{M}_{i,t}} w_{im,t}} f(\boldsymbol{x}; \boldsymbol{\rho}_j). \tag{5.14}$$

Client i makes the prediction for $\boldsymbol{x}_{i,t}$ as

$$\bar{f}_{i,t}(\boldsymbol{x}_{i,t}) = \frac{\gamma_{i,t}}{\gamma_{i,t} + \delta_{i,t}} f_{i,t}(\boldsymbol{x}_{i,t}) + \frac{\delta_{i,t}}{\gamma_{i,t} + \delta_{i,t}} \tilde{f}_{i,t}(\boldsymbol{x}_{i,t}) \tag{5.15}$$

where $f_{i,t}(\boldsymbol{x}_{i,t})$ is the ensemble of local and federated models as defined in (5.9). Furthermore, $\gamma_{i,t}$ and $\delta_{i,t}$ are weights assigned by client i to ensemble models $f_{i,t}(\cdot)$ and $\tilde{f}_{i,t}(\cdot)$, respectively.

Algorithm 14 Fed-POE: Federated Learning with Personalized Online Ensemble

```
1: Input: Model $f(\cdot;\cdot)$, batch size $b$, $n$, $U$, $\mathbb{D}_{i,1} = \varnothing$.
2: for $t = 1, \ldots, T$ do
3:     The server transmits $\boldsymbol{\theta}_t$ to all clients.
4:     for all $i \in [N]$, client $i$ do
5:         Performs model selection given $\mathbb{D}_t$ according to Algorithm 13 to obtain $\mathbb{M}_{i,t}$.
6:         Makes prediction $\bar{f}_{i,t}(\boldsymbol{x}_{i,t})$ as in (5.15) using chosen model set $\mathbb{M}_{i,t}$.
7:         Upon receiving $y_{i,t}$, updates $\alpha_{i,t}$, $\beta_{i,t}$, $\gamma_{i,t}$, $\delta_{i,t}$ and $\{w_{ij,t}\}_{j=1}^{|\mathbb{D}_t|}$ as in (5.10), (5.16) and (5.17).
8:         Updates the local model as $\boldsymbol{\phi}_{i,t+1} = \boldsymbol{\phi}_{i,t} - \frac{\eta}{b}\sum_{\tau=t-b}^{t} \nabla\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\phi}_{i,\tau}), y_{i,t})$.
9:         Updates the federated model as $\boldsymbol{\psi}_{i,t+1} = \boldsymbol{\theta}_t - \frac{\eta}{b}\sum_{\tau=t-b}^{t} \nabla\mathcal{L}(f(\boldsymbol{x}_{i,\tau};\boldsymbol{\theta}_t), y_{i,\tau})$.
10:        Sends $\boldsymbol{\psi}_{i,t+1}$ to the server.
11:    end for
12:    if $t \le U$ and $t \bmod n = 0$ then
13:        The server updates $\mathbb{D}_t$ as $\mathbb{D}_{t+1} = \mathbb{D}_t \cup \{\boldsymbol{\theta}_t\}$.
14:    end if
15:    The server updates $\boldsymbol{\theta}_t$ as $\boldsymbol{\theta}_{t+1} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\psi}_{i,t+1}$.
16: end for
```

Upon observing the label $y_{i,t}$ after prediction, client i updates weights $\gamma_{i,t}$ and $\delta_{i,t}$ as

$$\gamma_{i,t+1} = \gamma_{i,t} \exp\left(-\eta_c \mathcal{L}(f_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t})\right), \tag{5.16a}$$

$$\delta_{i,t+1} = \delta_{i,t} \exp(-\eta_c \mathcal{L}(\tilde{f}_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t})). \tag{5.16b}$$

Furthermore, $\alpha_{i,t}$ and $\beta_{i,t}$ which are used to construct $f_{i,t}(\boldsymbol{x}_{i,t})$ in (5.9) are updated as in (5.10). The weight $w_{ij,t}$ is updated using the importance sampling loss as

$$w_{ij,t+1} = w_{ij,t} \exp\left(-\eta_c \frac{\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\rho}_j), y_{i,t})}{q_{ij,t}} \mathbf{1}_{j\in\mathbb{M}_{i,t}}\right) \tag{5.17}$$

where $\mathbf{1}_{j\in\mathbb{M}_{i,t}}$ denotes the indicator function and is 1 if $j \in \mathbb{M}_{i,t}$. Moreover, $q_{ij,t}$ is the probability that client i selects the j-th model in $\mathbb{D}_t$ to be in $\mathbb{M}_{i,t}$ and can be expressed as $q_{ij,t} = 1 - (1 - p_{ij,t})^M$ where $p_{ij,t}$ is defined in step 3 of Algorithm 13. The proposed algorithm, named Federated Learning with Personalized Online Ensemble (Fed-POE) is summarized in Algorithm 14. It is useful to note that the proposed method in Subsection 5.4.1 is a special case of Fed-POE by setting $M = 0$ and $b = 1$.

Efficiency of Fed-POE. To perform model selection using Fed-POE, clients do not need to store all model parameters in $\mathbb{D}_t$ for all $t \in [T]$. Instead, the server, which has higher storage capacity than the clients, stores all model parameters, and clients download a subset of at most M model parameters. The hyperparameter M can be chosen such that clients can handle the memory and computational requirements of making predictions with the selected subset of models. Let C_F denote the number of computations required to fine-tune the model f, and let C_I represent the number of computations required to make an inference with model f. Assume that the complexity of model selection in Algorithm 1 is negligible compared to fine-tuning and making inferences with model f. According to Algorithm 2, each client performs $2C_F + (M+2)C_I$ computations per time step. Therefore, the computational complexity of Fed-POE for each client is $\mathcal{O}(C_F + MC_I)$. Beyond memory and computational considerations, selecting a subset of models from $\mathbb{D}_t$ helps clients improve their prediction accuracy. Specifically, using the model weights $\{w_{ij,t}\}$ at time step t, client i selects models that perform better on its data while pruning those with lower performance.

Let $h_j(\boldsymbol{x}_{i,t}) = f(\boldsymbol{x}_{i,t}; \boldsymbol{\rho}_j)$ denote the model associated with the j-th model parameter in $\mathbb{D}_t$. Furthermore, $h_{\text{loc}}(\boldsymbol{x}_{i,t}) = f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}_{i,t})$ and $h_{\text{fed}}(\boldsymbol{x}_{i,t}) = f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t)$ represent the local and federated models, respectively, . Let the set of models $\mathbb{H}$ be defined as $\mathbb{H} := \{h_j \mid \forall j : 1 \leq j \leq |\mathbb{D}_T|\} \cup \{h_{\text{loc}}, h_{\text{fed}}\}$. This set $\mathbb{H}$ includes all models that can be employed by each client using Fed-POE. The best model in hindsight h^* and the best model in hindsight for client i, h_i^* are defined as

$$h^* = \min_{h \in \mathbb{H}} \sum_{t=1}^{T} \sum_{i=1}^{N} \mathcal{L}(h(\boldsymbol{x}_{i,t}), y_{i,t}), \tag{5.18a}$$

$$h_i^* = \min_{h \in \mathbb{H}} \sum_{t=1}^{T} \mathcal{L}(h(\boldsymbol{x}_{i,t}), y_{i,t}). \tag{5.18b}$$

The following theorem provides the regret upper bound of Fed-POE.

Theorem 5.3. *Under assumption (A3), employing Fed-POE in Algorithm 14, the expected* *global regret of all clients is bounded from above as*

$$\begin{aligned}&\frac{1}{N}\sum_{t=1}^{T}\sum_{i=1}^{N}\mathbb{E}_t[\mathcal{L}(\bar{f}_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t})] - \frac{1}{N}\sum_{t=1}^{T}\sum_{i=1}^{N}\mathcal{L}(h^*(\boldsymbol{x}_{i,t}), y_{i,t})\\ \leq&\frac{\ln 2U - \ln 2n}{\eta_c} + \frac{\eta_c}{2}(\frac{U}{n}+1)T + (1-\frac{\eta_c}{2n}U)U\end{aligned} \tag{5.19}$$

while client i achieves the following expected personalized regret upper bound:

$$\begin{aligned}&\sum_{t=1}^{T}\mathbb{E}_t[\mathcal{L}(\bar{f}_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t})] - \sum_{t=1}^{T}\mathcal{L}(h_i^*(\boldsymbol{x}_{i,t}), y_{i,t})\\ \leq&\frac{\ln 2U - \ln 2n}{\eta_c} + \frac{\eta_c}{2}(\frac{U}{n}+1)T + (1-\frac{\eta_c}{2n}U)U.\end{aligned} \tag{5.20}$$

The expectation is taken with respect to randomization in model selection.

The proof of Theorem 5.3 can be found in Appendix E.3. According to (5.19) and (5.20), setting $\eta_c = \mathcal{O}(1/\sqrt{T})$, $n = \mathcal{O}(\sqrt{T})$ and $U = \mathcal{O}(\sqrt{T})$, both the personalized and global regrets of clients achieve sublinear regret of $\mathcal{O}(\sqrt{T})$. Since clients construct their ensemble models using a time-varying subset of models, employing existing model selection and ensemble learning approaches [48, 50, 114, 128, 59, 118] may not ensure the sublinear regrets stated in Theorem 5.3. However, by using the proposed Algorithm 13, Fed-POE guarantees sublinear regret bounds while allowing clients the flexibility to select time-varying and personalized subsets of models for their ensembles.

5.5 Experiments

The present section studies the performance of Fed-POE in Algorithm 14 compared to other baselines. Experiments are conducted on both image classification and regression tasks. The performance of federated learning is examined in both convex and non-convex cases. Codes are available at `https://github.com/pouyamghari/Fed-POE`.

5.5.1 Regression

The performance of the proposed Fed-POE is evaluated on online regression tasks. For these tasks, clients and the server collaborate to train a random feature kernel-based model, which is known to be convex [77, 131]. Details about the random feature kernel-based model used in the experiments can be found in Appendix E.4. The performance of Fed-POE is compared with a baseline called Local, where clients train their models locally without participating in federated learning. Additionally, Fed-POE is compared to personalized federated learning baselines Ditto [96] and Fed-Rep [30], the online federated learning baseline Fed-OMD [112], and online federated kernel learning baselines eM-KOFL [78] and POF-MKL. Mean square error (MSE) is used as the metric to evaluate the performance of algorithms on regression task. MSE for client i can be expressed as $\text{MSE}_i = \frac{1}{T}\sum_{t=1}^{T}(\hat{y}_{ij,t} - y_{i,t})^2$ where $\hat{y}_{ij,t}$ denotes the prediction of client i at time step t. The performance of algorithms are tested on two regression datasets Air [161] and WEC [117]. Air dataset is a time-series dataset that each data sample contains air quality features and the goal is to predict the concentration of contamination in the air. Each sample in WEC dataset contains features of different wave energy converters and the goal is to predict power output. Data samples are distributed non-i.i.d among 400 clients. Time horizon T for both datasets is 250. More information about datasets and distributed data among clients is presented in Appendix E.4.

Table 5.1: Average MSE ($\times 10^{-3}$) and its standard deviation ($\times 10^{-3}$) across clients for online regression on Air and WEC datasets.

Methods	Air	WEC
Local	9.12 ± 3.59	17.64 ± 0.44
Ditto	10.65 ± 5.69	33.88 ± 16.08
Fed-Rep	10.48 ± 5.23	35.13 ± 10.38
Fed-OMD	11.48 ± 6.84	32.61 ± 27.38
eM-KOFL	11.51 ± 6.71	72.29 ± 62.48
POF-MKL	10.66 ± 6.07	16.94 ± 15.92
Fed-POE	$\mathbf{9.06 \pm 3.73}$	$\mathbf{11.83 \pm 4.60}$

Table 5.1 presents the MSE of algorithms and their standard deviation across clients. For all algorithms, the learning rates are set to $\eta = \eta_c = 1/\sqrt{T}$. Table 5.1 shows that when data is distributed non-i.i.d. among clients, local model training can achieve higher accuracy compared to federated learning. For the Air dataset, Local achieves lower MSE than other federated learning baselines except for Fed-POE. For the WEC dataset, only POF-MKL achieves lower MSE than Local. This indicates that the performance of federated learning approaches compared to Local depends on the dataset. By utilizing both federated and local models, Fed-POE achieves the lowest MSE. Table 5.1 shows that the performance of Fed-POE relative to other baselines is more consistent across different datasets.

5.5.2 Image Classification

The performance of the proposed Fed-POE on an image classification task is compared with Local, Ditto [96], Fed-Rep [30], Fed-OMD [112], Fed-ALA [159], and Fed-DS [108]. Fed-ALA [159] is a personalized federated learning model suitable for deep neural networks, while Fed-DS [108] is a federated learning algorithm designed to handle data streams. Experiments are conducted on CIFAR-10 [90] and Fashion MNIST (FMNIST) [152] datasets. CIFAR-10 and FMNIST contain $60,000$ and $70,000$ images. For both CIFAR-10 and FMNIST, a CNN with a VGG architecture [138], consisting of two blocks, is pre-trained on a subset of training samples from each dataset. The training datasets are biased towards class 0. More details about training the CNNs can be found in Appendix E.4. For both the CIFAR-10 and FMNIST datasets, $10,000$ test samples are sequentially received by clients. There are 20 clients in total, and the data samples are distributed non-i.i.d. among them. For CIFAR-10, each client is biased towards one specific class, with 55% of the samples belonging to that class and 5% of the samples belonging to each of the other 9 classes. For FMNIST, each client is biased towards two classes, and the distribution of samples is time-variant. More details about the data distribution among clients and experimental setup can be found in

Table 5.3: Average accuracy and standard deviation across clients employing Fed-POE for image classification on CIFAR-10 with varying values of M and batch size b.

	$M = 0$	$M = 4$	$M = 8$	$M = 16$
$b = 1$	$53.80\% \pm 6.71\%$	$62.73\% \pm 8.29\%$	$62.73\% \pm 8.29\%$	$62.73\% \pm 8.26\%$
$b = 10$	$65.55\% \pm 8.77\%$	$66.50\% \pm 8.00\%$	$66.54\% \pm 8.08\%$	$66.46\% \pm 7.98\%$
$b = 20$	$65.72\% \pm 8.62\%$	$66.13\% \pm 8.20\%$	$66.64\% \pm 7.94\%$	$66.53\% \pm 8.00\%$
$b = 30$	$66.83\% \pm 8.54\%$	$66.32\% \pm 7.92\%$	$66.24\% \pm 8.05\%$	$66.39\% \pm 8.02\%$

Appendix E.4. At each time step, all algorithms employ batch of size 10 for model update. The learning rates for all algorithms are set to $\eta = 0.01/\sqrt{T}$ and $\eta_c = 1/\sqrt{T}$. The server stores models every $n = 20$ time steps. The metric to evaluate the performance of algorithms is the accuracy. The accuracy for client i is defined as $\text{Accuracy}_i = \frac{1}{T}\sum_{t=1}^{T} \mathbf{1}_{\hat{y}_{i,t}=y_{i,t}}$ where $\hat{y}_{i,t}$ denotes the label predicted by client i at time step t.

Table 5.2: Average accuracy and its standard deviation across clients for image classification.

Methods	CIFAR-10	FMNIST
Local	$50.35\% \pm 10.11\%$	$78.81\% \pm 2.12\%$
Ditto	$56.87\% \pm 9.06\%$	$78.73\% \pm 1.89\%$
Fed-Rep	$63.86\% \pm 7.97\%$	$79.04\% \pm 1.77\%$
Fed-OMD	$65.09\% \pm 7.39\%$	$74.60\% \pm 6.52\%$
Fed-ALA	$61.48\% \pm 8.88\%$	$75.13\% \pm 6.39\%$
Fed-DS	$64.26\% \pm 7.03\%$	$75.62\% \pm 6.58\%$
Fed-POE	$\mathbf{66.54\% \pm 8.07\%}$	$\mathbf{79.23\% \pm 1.88\%}$

Average accuracy and its standard deviation across clients for CIFAR-10 and FMNIST are reported in Table 5.2. At each time step t, clients can store 10 model parameters. Therefore, M is set to $M = 8$ for Fed-POE. The results indicate that the performance of Local relative to federated learning baselines depends on the dataset. While Local outperforms all federated baselines except for Fed-Rep and Fed-POE on FMNIST, it obtains the worst accuracy on CIFAR-10. Conversely, Fed-POE achieves the highest accuracy for both datasets, indicating that Fed-POE efficiently leverages the advantages of both federated and local models. To analyze the effect of the number of models M and batch size b on the Fed-POE performance, experiments are conducted on the

CIFAR-10 dataset, varying the batch size b and the number of models M selected by each client to construct the ensemble model. As observed in Table 5.3, the batch size $b = 1$ results in the worst accuracy, mainly due to the forgetting process where models overfit to the most recently observed data. However, increasing the batch size from $b = 10$ or $b = 20$ to $b = 30$ does not significantly improve the accuracy. Larger batch sizes may lead the model to perform better on older data, as the model is trained on older data over more iterations. Therefore, this study concludes that a moderate batch size is optimal, considering that increasing the batch size also increases computational complexity. Table E.1 in Appendix E.4 presents the accuracy of Fed-POE on both the CIFAR-10 and FMNIST datasets with varying values of M. The table shows that employing the saved models by the server in $\mathbb{D}_t$ improves performance, as setting $M = 0$ results in the worst accuracy. Moreover, it can be observed that increasing M does not necessarily lead to further accuracy improvement. This aligns with the intuition behind selecting a subset of models rather than using all models.

5.6 Conclusion

This chapter proposed Fed-POE, a personalized federated learning algorithm designed for online prediction and model fine-tuning. Fed-POE constructs an ensemble using local models and federated models stored by the server periodically over time. Theoretical analysis demonstrated that Fed-POE achieves sublinear regret. Experimental results revealed that the relative performance of local models compared to federated models depends on the dataset, making the decision between local model training and federated learning challenging. However, experimental results also show that Fed-POE consistently outperforms both local and federated models across all datasets. This indicates that Fed-POE effectively leverages the advantages of both local and federated models.

Chapter 6

Conclusions

This thesis explored scalability and algorithm design for various challenging scenarios in online decision-making. Online kernel learning techniques, known for their computational efficiency and adaptability to dynamic environments, are particularly suitable for online prediction in non-stationary settings. Chapter 1 introduced online multi-kernel learning algorithms that enable the use of a large set of kernel-based models while maintaining computational efficiency. Experiments on regression datasets demonstrated that the proposed algorithms achieve higher accuracy and greater computational efficiency compared to state-of-the-art online kernel learning methods. Chapter 2 studied the problem of uncertainty in observations within online learning. While most existing online learning algorithms cannot achieve sublinear regret under such uncertainties, it was proven that the proposed algorithms in Chapter 2 attain sublinear regret.

This thesis investigated personalized online federated learning, focusing on scenarios where a set of clients collaborates to learn personalized models for online prediction with heterogeneously distributed data. Chapter 3 explored the use of online multi-kernel learning for personalized online federated learning. In the experiments in Chapter 3, the proposed algorithm, POF-MKL, outperformed other online federated kernel learning algorithms when data was distributed heterogeneously among clients. Chapter 4 introduced a novel online model selection algorithm that leverages federated learning, enabling clients to perform online model selection using a large set of models beyond their memory capacity. This algorithm allows for personalized model selection while enabling collaborative model fine-tuning among clients. Experiments in Chapter 4 demonstrated that the proposed algorithm achieved higher accuracy than other online model selection and federated learning methods. Finally, Chapter 5 highlighted that in online prediction tasks, the benefit of using online federated learning over local online model fine-tuning is not always clear. To address this, Chapter 5 proposed a novel personalized federated learning algorithm in which each client constructs its own model by combining an ensemble of federated models with its local model for online prediction. Experiments in Chapter 5 confirmed that the advantage of online federated learning over local online learning depends on the dataset. However, the proposed algorithm demonstrated robustness, consistently achieving the highest accuracy across all datasets.

Bibliography

[1] D. A. E. Acar, Y. Zhao, R. Zhu, R. Matas, M. Mattina, P. Whatmough, and V. Saligrama. Debiasing model updates for improving personalized federated training. In *Proceedings of International Conference on Machine Learning*, volume 139, pages 21–31, Jul 2021.

[2] I. Achituve, A. Shamsian, A. Navon, G. Chechik, and E. Fetaya. Personalized federated learning with gaussian processes. In *Proceedings of International Conference on Neural Information Processing Systems*, volume 34, pages 8392–8406, Dec 2021.

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

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

[5] K. Amin, S. Kale, G. Tesauro, and D. Turaga. Budgeted prediction with expert advice. In *AAAI Conference on Artificial Intelligence*, Austin, Texas, USA, Feb 2015.

[6] R. Arora, T. V. Marinov, and M. Mohri. Bandits with feedback graphs and switching costs. In *Advances in Neural Information Processing Systems*, pages 10397–10407, Dec 2019.

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

[8] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, Jan 2003.

[9] R. Balakrishnan, T. Li, T. Zhou, N. Himayat, V. Smith, and J. Bilmes. Diverse client selection for federated learning via submodular maximization. In *International Conference on Learning Representations*, 2022.

[10] A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh. Clustering with bregman divergences. *Journal of Machine Learning Research*, 6(58):1705–1749, 2005.

[11] Y. Bengio, O. Delalleau, and N. L. Roux. The curse of highly variable functions for local kernel machines. In *Advances in Neural Information Processing Systems*, pages 107–114, May 2006.

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

[13] L. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7(3):200–217, 1967.

[14] T. F. Brooks, D. S. Pope, and M. A. Marcolini. Airfoil self-noise and prediction. Technical report, Jul 1989.

[15] S. S. Bucak, R. Jin, and A. K. Jain. Multiple kernel learning for visual object recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1354–1369, Jul 2014.

[16] L. Cella, M. Pontil, and C. Gentile. Best model identification: A rested bandit formulation. In *Proceedings of International Conference on Machine Learning*, volume 139, pages 1362–1372, Jul 2021.

[17] N. Cesa-Bianchi, T. Cesari, and C. Monteleoni. Cooperative online learning: Keeping your neighbors updated. In *Proceedings of the International Conference on Algorithmic Learning Theory*, volume 117, pages 234–250, Feb 2020.

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

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

[20] Z. Charles, Z. Garrett, Z. Huo, S. Shmulyian, and V. Smith. On large-cohort training for federated learning. In *Advances in Neural Information Processing Systems*, 2021.

[21] H. Chen, J. Ding, E. W. Tramel, S. Wu, A. K. Sahu, S. Avestimehr, and T. Zhang. Self-aware personalized federated learning. *Advances in Neural Information Processing Systems*, 35:20675–20688, 2022.

[22] H.-Y. Chen and W.-L. Chao. On bridging generic and personalized federated learning for image classification. In *International Conference on Learning Representations*, 2022.

[23] T. Chen, G. B. Giannakis, T. Sun, and W. Yin. Lag: Lazily aggregated gradient for communication-efficient distributed learning. In *Proceedings of International Conference on Neural Information Processing Systems*, page 5055–5065, 2018.

[24] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala. Asynchronous online federated learning for edge devices with non-iid data. In *IEEE International Conference on Big Data (Big Data)*, pages 15–24, Dec 2020.

[25] D. Cho, C. Yoo, J. Im, and D.-H. Cha. Comparative assessment of various machine learning-based bias correction methods for numerical weather prediction model forecasts of extreme air temperatures in urban areas. *Earth and Space Science*, 7(4), Mar 2020.

[26] H. I. Christensen, A. Khan, S. Pokutta, and P. Tetali. Approximation and online algorithms for multidimensional bin packing: A survey. *Computer Science Review*, 24:63–79, May 2017.

[27] W. Chung, S. Nath, A. Joseph, and M. White. Two-timescale networks for nonlinear value function approximation. In *International Conference on Learning Representations*, May 2019.

[28] V. Chvatal. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):233–235, Aug 1979.

[29] A. Cohen, T. Hazan, and T. Koren. Online learning with feedback graphs without the graphs. In *Proceedings of International Conference on Machine Learning*, page 811–819, Jun 2016.

[30] L. Collins, H. Hassani, A. Mokhtari, and S. Shakkottai. Exploiting shared representations for personalized federated learning. In *Proceedings of the International Conference on Machine Learning*, volume 139, pages 2089–2099, Jul 2021.

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

[32] L. Corinzia, A. Beuret, and J. M. Buhmann. Variational federated multi-task learning, 2019.

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

[34] C. Cortes, G. DeSalvo, C. Gentile, M. Mohri, and N. Zhang. Online learning with dependent stochastic feedback graphs. In *Proceedings of International Conference on Machine Learning*, Jul 2020.

[35] C. Cortes, M. Mohri, and A. Rostamizadeh. L2 regularization for learning kernels. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, page 109–116, Jun 2009.

[36] Z. Dai, P. Chandar, G. Fazelnia, B. Carterette, and M. Lalmas. Model selection for production system via automated online experiments. In *Proceedings of International Conference on Neural Information Processing Systems*, volume 33, pages 1106–1116, 2020.

[37] G. Damaskinos, R. Guerraoui, A.-M. Kermarrec, V. Nitu, R. Patra, and F. Taiani. Fleet: Online federated learning via staleness awareness and performance prediction. *ACM Transactions on Intelligent Systems and Technology*, 13(5), Sep 2022.

[38] W. F. de la Vega and G. S. Lueker. Bin packing can be solved within $1 + \epsilon$ in linear time. *Combinatorica*, 1:349–355, 1981.

[39] Y. Deng, M. M. Kamani, and M. Mahdavi. Adaptive personalized federated learning, 2020.

[40] L. Ding, R. Tuo, and S. Shahrampour. Generalization guarantees for sparse kernel approximation with entropic optimal features. In *Proceedings of the International Conference on Machine Learning*, volume 119, pages 2545–2555, Jul 2020.

[41] Y. Ding, C. Liu, P. Zhao, and S. C. H. Hoi. Large scale kernel methods for online AUC maximization. In *IEEE International Conference on Data Mining*, pages 91–100, Nov 2017.

[42] C. T. Dinh, N. H. Tran, and T. D. Nguyen. Personalized federated learning with moreau envelopes. In *Proceedings of International Conference on Neural Information Processing Systems*, page 21394–21405, Dec 2020.

[43] G. Dósa. The tight bound of first fit decreasing bin-packing algorithm is ffd(i) $\leq$ 11/9opt(i) + 6/9. In *International Conference on Combinatorics, Algorithms, Probabilistic and Experimental Methodologies*, page 1–11, Apr. 2007.

[44] D. Dua and C. Graff. UCI machine learning repository, 2017.

[45] L. Duan, I. W. Tsang, and D. Xu. Domain transfer multiple kernel learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3):465–479, Mar 2012.

[46] A. Fallah, A. Mokhtari, and A. Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. In *Advances in Neural Information Processing Systems*, volume 33, pages 3557–3568, Dec 2020.

[47] A. Farahmand and C. Szepesvari. Model selection in reinforcement learning. *Machine Learning*, 85:299–332, Jun 2011.

[48] A. Fern and R. Givan. Online ensemble learning: An empirical study. *Machine Learning*, 53:71–109, 2003.

[49] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of International Conference on Machine Learning*, volume 70, pages 1126–1135, Aug 2017.

[50] D. Foster, S. Kale, M. Mohri, and K. Sridharan. Parameter-free online learning via model selection. In *Proceedings of International Conference on Neural Information Processing Systems*, 2017.

[51] D. J. Foster, A. Krishnamurthy, and H. Luo. Model selection for contextual bandits. In *Proceedings of International Conference on Neural Information Processing Systems*, volume 32, 2019.

[52] L. Fu, H. Zhang, G. Gao, M. Zhang, and X. Liu. Client selection in federated learning: Principles, challenges, and opportunities. *IEEE Internet of Things Journal*, 2023.

[53] B. Ganguly and V. Aggarwal. Online federated learning via non-stationary detection and adaptation amidst concept drift. *arXiv preprint arXiv:2211.12578*, 2022.

[54] M. R. Garey, R. L. Graham, and J. D. Ullman. Worst-case analysis of memory allocation algorithms. In *ACM Symposium on Theory of Computing*, page 143–150, 1972.

[55] F. Gauthier, V. C. Gogineni, S. Werner, Y.-F. Huang, and A. Kuh. Resource-aware asynchronous online federated learning for nonlinear regression. In *IEEE International Conference on Communications*, pages 2828–2833, 2022.

[56] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller. Inverting gradients - how easy is it to break privacy in federated learning? In *Advances in Neural Information Processing Systems*, volume 33, pages 16937–16947, 2020.

[57] P. M. Ghari and Y. Shen. Graph-aided online learning with expert advice. In *Asilomar Conference on Signals, Systems, and Computers*, pages 470–474, 2020.

[58] P. M. Ghari and Y. Shen. Online multi-kernel learning with graph-structured feedback. In *Proceedings of the International Conference on Machine Learning*, volume 119, pages 3474–3483, Jul 2020.

[59] P. M. Ghari and Y. Shen. Graph-assisted communication-efficient ensemble federated learning. In *European Signal Processing Conference (EUSIPCO)*, pages 737–741, 2022.

[60] P. M. Ghari and Y. Shen. Online learning with probabilistic feedback. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4183–4187, May 2022.

[61] P. M. Ghari and Y. Shen. Personalized online federated learning with multiple kernels. In *Advances in Neural Information Processing Systems*, 2022.

[62] P. M. Ghari and Y. Shen. Graph-aided online multi-kernel learning. *Journal of Machine Learning Research*, 24(21):1–44, 2023.

[63] P. M. Ghari and Y. Shen. Budgeted online model selection and fine-tuning via federated learning. *Transactions on Machine Learning Research*, 2024.

[64] P. M. Ghari and Y. Shen. Online learning with uncertain feedback graphs. *IEEE Transactions on Neural Networks and Learning Systems*, 35(7):9636–9650, 2024.

[65] P. M. Ghari and Y. Shen. Personalized federated learning with mixture of models for adaptive prediction and model fine-tuning. In *Advances in Neural Information Processing Systems*, 2024.

[66] V. C. Gogineni, S. Werner, Y.-F. Huang, and A. Kuh. Communication-efficient online federated learning framework for nonlinear regression. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5228–5232, May 2022.

[67] V. C. Gogineni, S. Werner, Y.-F. Huang, and A. Kuh. Communication-efficient online federated learning strategies for kernel regression. *IEEE Internet of Things Journal*, 10(5):4531–4544, 2023.

[68] K. Gokcesu and S. S. Kozat. An online minimax optimal algorithm for adversarial multiarmed bandit problem. *IEEE Transactions on Neural Networks and Learning Systems*, 29(11):5565–5580, Mar. 2018.

[69] M. Gönen and E. Alpaydin. Multiple kernel learning algorithms. *Journal of Machine Learning Research*, 12(64):2211–2268, 2011.

[70] D. Ha, A. M. Dai, and Q. V. Le. Hypernetworks. In *International Conference on Learning Representations*, 2017.

[71] J. Hamer, M. Mohri, and A. T. Suresh. FedBoost: A communication-efficient algorithm for federated learning. In *Proceedings of International Conference on Machine Learning*, volume 119, pages 3973–3983, Jul 2020.

[72] F. Hanzely, S. Hanzely, S. Horváth, and P. Richtárik. Lower bounds and optimal algorithms for personalized federated learning. In *Proceedings of International Conference on Neural Information Processing Systems*, page 2304–2315, Dec 2020.

[73] E. Hazan. *Introduction to Online Convex Optimization*. MIT Press, 2022.

[74] E. Hazan and N. Megiddo. Online learning with prior knowledge. In *Proceedings of Annual Conference on Learning Theory*, page 499–513, Jun 2007.

[75] D. P. Helmbold, N. Littlestone, and P. M. Long. Apple tasting. *Information and Computation*, 161(2):85–139, Sep 2000.

[76] R. Hoberg and T. Rothvoss. A logarithmic additive integrality gap for bin packing. In *Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2616–2625, 2017.

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

[78] S. Hong and J. Chae. Communication-efficient randomized algorithm for multi-kernel online federated learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):9872–9886, 2022.

[79] T. Huang, W. Lin, W. Wu, L. He, K. Li, and A. Y. Zomaya. An efficiency-boosting client selection scheme for federated learning with fairness guarantee. *IEEE Transactions on Parallel and Distributed Systems*, 32(7):1552–1564, 2021.

[80] D. S. Johnson. *Near-optimal bin packing algorithms.* PhD thesis, MIT, 1973.

[81] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, H. Eichner, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konecný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, H. Qi, D. Ramage, R. Raskar, M. Raykova, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

[82] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh. SCAFFOLD: Stochastic controlled averaging for federated learning. In *Proceedings of International Conference on Machine Learning*, volume 119, pages 5132–5143, Jul 2020.

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

[84] M. Kelly, R. Longjohn, and K. Nottingham. UCI machine learning repository, 2023.

[85] M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien. Lp-norm multiple kernel learning. *Journal of Machine Learning Research*, 12:953–997, Jul 2011.

[86] T. Kocák, G. Neu, and M. Valko. Online learning with Erdös-Rényi side-observation graphs. In *Proceedings of Conference on Uncertainty in Artificial Intelligence*, page 339–346, Jun 2016.

[87] T. Kocák, G. Neu, M. Valko, and R. Munos. Efficient learning by implicit exploration in bandit problems with side observations. In *Proceedings of International Conference on Neural Information Processing Systems*, page 613–621, Dec 2014.

[88] T. Kocák, G. Neu, and M. Valko. Online learning with noisy side observations. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, pages 1186–1194, Cadiz, Spain, May 2016.

[89] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon. Federated learning: Strategies for improving communication efficiency, 2017.

[90] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[91] A. Kuh. Real time kernel learning for sensor networks using principles of federated learning. In *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 2089–2093, Dec 2021.

[92] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[93] J. Lee, A. Pacchiano, V. Muthukumar, W. Kong, and E. Brunskill. Online model selection for reinforcement learning with function approximation. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, volume 130, pages 3340–3348, Apr 2021.

[94] J. Li and S. Liao. Worst-case regret analysis of computationally budgeted online kernel selection. *Machine Learning*, 111(3):937–976, Mar 2022.

[95] S. Li, W. Chen, Z. Wen, and K.-S. Leung. Stochastic online learning with probabilistic graph feedback. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):4675–4682, Apr. 2020.

[96] T. Li, S. Hu, A. Beirami, and V. Smith. Ditto: Fair and robust federated learning through personalization. In *Proceedings of the International Conference on Machine Learning*, volume 139, pages 6357–6368, Jul 2021.

[97] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[98] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang. On the convergence of fedavg on non-iid data. In *International Conference on Learning Representations*, 2020.

[99] X. Li, M. Jiang, X. Zhang, M. Kamp, and Q. Dou. FedBN: Federated learning on non-IID features via local batch normalization. In *International Conference on Learning Representations*, 2021.

[100] X.-C. Li, D.-C. Zhan, Y. Shao, B. Li, and S. Song. Fedphp: Federated personalization with inherited private models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 587–602, 2021.

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

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

[103] K. Liu, S. Hu, S. Wu, and V. Smith. On privacy and personalization in cross-silo federated learning. In *Advances in Neural Information Processing Systems*, 2022.

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

[105] S. Lv, J. Wang, J. Liu, and Y. Liu. Improved learning rates of a functional lasso-type svm with sparse multi-kernel representation. In *Advances in Neural Information Processing Systems*, volume 34, pages 21467–21479, Dec 2021.

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

[107] O. Marfoq, G. Neglia, A. Bellet, L. Kameni, and R. Vidal. Federated multi-task learning under a mixture of distributions. In *Proceedings of International Conference on Neural Information Processing Systems*, volume 34, pages 15434–15447, Dec 2021.

[108] O. Marfoq, G. Neglia, L. Kameni, and R. Vidal. Federated learning for data streams. In *Proceedings of The International Conference on Artificial Intelligence and Statistics*, volume 206, pages 8889–8924, Apr 2023.

[109] C. B. Mawuli, J. Kumar, E. Nanor, S. Fu, L. Pan, Q. Yang, W. Zhang, and J. Shao. Semi-supervised federated learning on evolving data streams. *Information Sciences*, 643:119235, 2023.

[110] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, volume 54, pages 1273–1282, Apr 2017.

[111] S. McQuade and C. Monteleoni. Global climate model tracking using geospatial neighborhoods. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pages 335–341, Jul 2012.

[112] A. Mitra, H. Hassani, and G. J. Pappas. Online federated learning. In *IEEE Conference on Decision and Control (CDC)*, pages 4083–4090, Dec 2021.

[113] M. Mohri, G. Sivek, and A. T. Suresh. Agnostic federated learning. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the International Conference on Machine Learning*, volume 97, pages 4615–4625, Jun 2019.

[114] V. Muthukumar, M. Ray, A. Sahai, and P. Bartlett. Best of many worlds: Robust model selection for online supervised learning. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, volume 89, pages 3177–3186, Apr 2019.

[115] V. K. Muthukumar and A. Krishnamurthy. Universal and data-adaptive algorithms for model selection in linear contextual bandits. In *International Conference on Machine Learning*, pages 16197–16222, 2022.

[116] G. D. Németh, M. A. Lozano, N. Quadrianto, and N. M. Oliver. A snapshot of the frontiers of client selection in federated learning. *Transactions on Machine Learning Research*, 2022. Survey Certification.

[117] M. Neshat, B. Alexander, M. Wagner, and Y. Xia. A detailed comparison of meta-heuristic methods for optimising wave energy converter placements. In *Proceedings of the Genetic and Evolutionary Computation Conference*, page 1318–1325, Jul 2018.

[118] A. Pacchiano, C. Dann, and C. Gentile. Best of both worlds model selection. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[119] A. Pacchiano, M. Phan, Y. Abbasi Yadkori, A. Rao, J. Zimmert, T. Lattimore, and C. Szepesvari. Model selection in contextual stochastic bandit problems. In *Advances in Neural Information Processing Systems*, volume 33, pages 10328–10337, 2020.

[120] A. Papoulis and S. U. Pillai. *Probability, random variables, and stochastic processes.* McGraw-Hill, 4th edition, 2002.

[121] J. Park, D. Kwon, and S. Hong. Ofedqit: Communication-efficient online federated learning via quantization and intermittent transmission. *arXiv preprint arXiv:2205.06491*, 2022.

[122] K. K. Patel, L. Wang, A. Saha, and N. Srebro. Federated online and bandit convex optimization. In *Proceedings of the International Conference on Machine Learning*, volume 202, pages 27439–27460, Jul 2023.

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

[124] A. Rakotomamonjy, F. R. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 9(83):2491–2521, 2008.

[125] V. V. Ramasesh, A. Lewkowycz, and E. Dyer. Effect of scale on catastrophic forgetting in neural networks. In *International Conference on Learning Representations*, 2022.

[126] A. Rangi and M. Franceschetti. Online learning with feedback graphs and switching costs. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, pages 2435–2444, Apr 2019.

[127] A. Resler and Y. Mansour. Adversarial online learning with noise. In *Proceedings of International Conference on Machine Learning*, pages 5429–5437, Jun 2019.

[128] M. Reza Karimi, N. Merve Gürel, B. Karlaš, J. Rausch, C. Zhang, and A. Krause. Online active model selection for pre-trained classifiers. In *Proceedings of The International Conference on Artificial Intelligence and Statistics*, volume 130, pages 307–315, Apr 2021.

[129] D. Rothchild, A. Panda, E. Ullah, N. Ivkin, I. Stoica, V. Braverman, J. Gonzalez, and R. Arora. FetchSGD: Communication-efficient federated learning with sketching. In *Proceedings of International Conference on Machine Learning*, volume 119, pages 8253–8265, Jul 2020.

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

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

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

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

[134] S. Shahrampour and V. Tarokh. Learning bounds for greedy approximation with explicit feature maps from multiple kernels. In *Proceedings of the International Conference on Neural Information Processing Systems*, page 4695–4706, Dec 2018.

[135] A. Shamsian, A. Navon, E. Fetaya, and G. Chechik. Personalized federated learning using hypernetworks. In *Proceedings of International Conference on Machine Learning*, volume 139, pages 9489–9502, Jul 2021.

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

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

[138] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, May 2015.

[139] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar. Federated multi-task learning. In *Advances in Neural Information Processing Systems*, volume 30, pages 4424–4434, Dec 2017.

[140] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large scale multiple kernel learning. *Journal of Machine Learning Research*, 7:1531–1565, Dec 2006.

[141] B. K. Sriperumbudur and Z. Szabó. Optimal rates for random fourier features. In *Proceedings of the International Conference on Neural Information Processing Systems*, page 1144–1152, Dec 2015.

[142] B. Sun, H. Huo, Y. YANG, and B. Bai. Partialfed: Cross-domain personalized federated learning via partial initialization. In *Proceedings of International Conference on Neural Information Processing Systems*, volume 34, pages 23309–23320, Dec 2021.

[143] Y. Tan, G. Long, J. Ma, L. Liu, T. Zhou, and J. Jiang. Federated learning from pre-trained models: A contrastive learning approach. *Advances in neural information processing systems*, 35:19332–19344, 2022.

[144] M. Toneva, A. Sordoni, R. T. d. Combes, A. Trischler, Y. Bengio, and G. J. Gordon. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019.

[145] J. Torres-Sospedra, R. Montoliu, A. Martínez-Usó, J. P. Avariento, T. J. Arnau, M. Benedito-Bordonau, and J. Huerta. Ujiindoorloc: A new multi-building and multi-floor database for wlan fingerprint-based indoor localization problems. In *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 261–270, Oct 2014.

[146] P. Tüfekci. Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods. *International Journal of Electrical Power and Energy Systems*, 60:126 – 140, 2014.

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

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

[149] Y. Wang, L. Lin, and J. Chen. Communication-efficient adaptive federated learning. In *Proceedings of the International Conference on Machine Learning*, volume 162, pages 22802–22838, Jul 2022.

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

[151] Y. Wu, S. Zhang, W. Yu, Y. Liu, Q. Gu, D. Zhou, H. Chen, and W. Cheng. Personalized federated learning under mixture of distributions. In *International Conference on Machine Learning*, pages 37860–37879, 2023.

[152] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[153] S. Yang and Y. Gao. An optimal algorithm for the stochastic bandits while knowing the near-optimal mean reward. *IEEE Transactions on Neural Networks and Learning Systems*, 32(5):2285–2291, Jun. 2021.

[154] T. Yang, M. Mahdavi, R. Jin, J. Yi, and S. C. H. Hoi. Online kernel selection: Algorithms and evaluations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, page 1197–1203, 2012.

[155] R. Ye, Z. Ni, F. Wu, S. Chen, and Y. Wang. Personalized federated learning with inferred collaboration graphs. In *International Conference on Machine Learning*, pages 39801–39817, 2023.

[156] I.-C. Yeh. Modeling of strength of high-performance concrete using artificial neural networks. *Cement and Concrete Research*, 28(12):1797 – 1808, Dec 1998.

[157] V. Yurinskiĭ. Exponential inequalities for sums of random vectors. *Journal of Multivariate Analysis*, 6(4):473 – 499, Dec 1976.

[158] J. Zhang, S. Guo, X. Ma, H. Wang, W. Xu, and F. Wu. Parameterized knowledge transfer for personalized federated learning. In *Proceedings of International Conference on Neural Information Processing Systems*, volume 34, pages 10092–10104, Dec 2021.

[159] J. Zhang, Y. Hua, H. Wang, T. Song, Z. Xue, R. Ma, and H. Guan. Fedala: Adaptive local aggregation for personalized federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 11237–11244, 2023.

[160] M. Zhang, K. Sapra, S. Fidler, S. Yeung, and J. M. Alvarez. Personalized federated learning with first order model optimization. In *International Conference on Learning Representations*, 2021.

[161] S. Zhang, B. Guo, A. Dong, J. He, Z. Xu, and S. X. Chen. Cautionary tales on air-quality improvement in beijing. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473(2205):20170457, 2017.

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

[163] X. Zhang, S. Liao, J. Xu, and J.-R. Wen. Regret bounds for online kernel selection in continuous kernel space. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):10931–10938, May 2021.

[164] L. Zhu, Z. Liu, and S. Han. Deep leakage from gradients. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Appendix A

Supplementary Proofs and Experiments for Chapter 1

A.1 Supplementary Experimental Results

This section presents more detailed results on MSE and Run time of proposed algorithms with the change in the number of chosen kernels. Table A.1 lists the MSE and run time of OMKL-GF with the change in the value of M which is the maximum number of kernels selected at each time instant by OMKL-GF. For OMKL-SFG and OMKL-SFG-R, the value of γ_i is chosen such that for each vertex $v_i \in \mathcal{V}$, the number of out-neighbors for each node is M. Tables A.2 and A.3 show both MSE and run time of OMKL-SFG and OMKL-SFG-R respectively with different values of M.

A.2 Proof of Lemma 1.1

In order to prove Lemma 1.1, we first establish the following lemma as a step stone.

Table A.1: MSE($\times 10^{-3}$) and run time of OMKL-GF with different M on real datasets.

	MSE($\times 10^{-3}$)					Run time (s)				
Datasets	$M=1$	$M=5$	$M=10$	$M=15$	$M=20$	$M=1$	$M=5$	$M=10$	$M=15$	$M=20$
Airfoil	49.53	24.95	25.73	26.47	26.91	0.70	1.53	2.56	3.40	4.30
Bias	12.56	7.11	8.15	8.93	9.79	5.44	11.24	17.01	23.17	27.74
Concrete	79.27	34.58	34.45	34.04	33.84	0.55	1.27	2.06	2.82	3.65
Naval	14.60	4.78	5.11	5.69	6.06	3.58	7.61	10.63	14.16	18.11

Table A.2: MSE($\times 10^{-3}$) and run time of OMKL-SFG with different M on real datasets.

	MSE($\times 10^{-3}$)					Run time (s)				
Datasets	$M=1$	$M=5$	$M=10$	$M=15$	$M=20$	$M=1$	$M=5$	$M=10$	$M=15$	$M=20$
Airfoil	51.03	33.49	32.49	31.82	30.89	1.17	1.36	1.65	1.98	2.27
Bias	12.35	11.78	12.06	12.11	12.13	6.58	9.94	13.57	17.49	21.26
Concrete	79.60	48.03	37.75	33.83	32.59	1.03	1.15	1.34	1.60	1.79
Naval	8.15	6.18	5.35	4.85	4.68	4.45	6.22	9.00	11.69	14.09

Lemma A.1. *Let $\hat{f}_{RF,i}(.)$ denote the sequence of estimates generated with a preselected kernel κ_i where $\mathcal{F}_i = \{\hat{f}_i | \hat{f}_i(\boldsymbol{x}) = \boldsymbol{\theta}^\top \mathbf{z}_i(\boldsymbol{x}), \forall \boldsymbol{\theta} \in \mathbb{R}^{2D}\}$. Then, under assumptions (A1) and (A2) the following bound holds*

$$\sum_{t=1}^{T} \mathcal{L}(\hat{f}_{RF,i}(\boldsymbol{x}_t), y_t) - \sum_{t=1}^{T} \mathcal{L}(\hat{f}_i^*(\boldsymbol{x}_t), y_t) \leq \frac{\|\boldsymbol{\theta}_i^*\|^2}{2\eta} + \sum_{t=1}^{T} \frac{\eta L^2}{2q_{i,t}} \tag{A.1}$$

where L is the Lipschitz constant in (A2) and $\boldsymbol{\theta}_i^*$ is the parameter vector associated with the best estimator $\hat{f}_i^*(\boldsymbol{x}) = (\boldsymbol{\theta}_i^*)^\top \mathbf{z}_i(\boldsymbol{x})$.

Proof. For $\boldsymbol{\theta}_{i,t+1}$ and any fixed $\boldsymbol{\theta}$, it can be written that

$$\begin{aligned}\|\boldsymbol{\theta}_{i,t+1} - \boldsymbol{\theta}\|^2 &= \|\boldsymbol{\theta}_{i,t} - \eta \nabla \ell_{i,t} - \boldsymbol{\theta}\|^2 \\ &= \|\boldsymbol{\theta}_{i,t} - \boldsymbol{\theta}\|^2 - 2\eta \nabla^\top \ell_{i,t}(\boldsymbol{\theta}_{i,t} - \boldsymbol{\theta}) + \|\eta \nabla \ell_{i,t}\|^2. \end{aligned} \tag{A.2}$$

Table A.3: MSE($\times 10^{-3}$) and run time of OMKL-SFG-R with different M on real datasets.

	MSE($\times 10^{-3}$)					Run time (s)				
Datasets	$M=1$	$M=5$	$M=10$	$M=15$	$M=20$	$M=1$	$M=5$	$M=10$	$M=15$	$M=20$
Airfoil	106.66	38.03	33.99	32.42	31.21	2.28	2.32	2.81	2.97	3.18
Bias	48.81	14.77	13.24	12.78	12.57	10.87	13.98	17.86	21.13	25.13
Concrete	110.08	50.96	38.58	34.32	32.85	1.91	2.06	2.24	2.51	2.63
Naval	95.37	14.32	7.89	6.03	5.33	6.77	8.55	11.45	13.65	15.84

Furthermore, from the convexity of the loss function with respect to $\boldsymbol{\theta}$ in (A1), we can conclude that

$$\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t) - \mathcal{L}(\boldsymbol{\theta}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t) \leq \nabla^\top \mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t)(\boldsymbol{\theta}_{i,t} - \boldsymbol{\theta}). \tag{A.3}$$

Therefore, from (A.3), it can be inferred that

$$\left(\frac{\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t)}{q_{i,t}} - \frac{\mathcal{L}(\boldsymbol{\theta}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t)}{q_{i,t}}\right)\mathbf{1}_{\mathrm{i}\in\mathbb{S}_\mathrm{t}} \leq \frac{\nabla^\top \mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t)}{q_{i,t}}(\boldsymbol{\theta}_{i,t} - \boldsymbol{\theta})\mathbf{1}_{\mathrm{i}\in\mathbb{S}_\mathrm{t}}. \tag{A.4}$$

Based on (1.35), (A.4) is equivalent to

$$\ell_{i,t} - \frac{\mathcal{L}(\boldsymbol{\theta}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t)}{q_{i,t}}\mathbf{1}_{\mathrm{i}\in\mathbb{S}_\mathrm{t}} \leq \nabla^\top \ell_{i,t}(\boldsymbol{\theta}_{i,t} - \boldsymbol{\theta}). \tag{A.5}$$

Combining (A.2) with (A.5), we get

$$\ell_{i,t} - \frac{\mathcal{L}(\boldsymbol{\theta}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t)}{q_{i,t}}\mathbf{1}_{\mathrm{i}\in\mathbb{S}_\mathrm{t}} \leq \frac{\|\boldsymbol{\theta}_{i,t} - \boldsymbol{\theta}\|^2 - \|\boldsymbol{\theta}_{i,t+1} - \boldsymbol{\theta}\|^2}{2\eta} + \frac{\eta}{2}\|\nabla \ell_{i,t}\|^2. \tag{A.6}$$

Taking the expectation of $\ell_{i,t}$ and $\|\nabla \ell_{i,t}\|^2$ with respect to $\mathbf{1}_{\mathrm{i}\in\mathbb{S}_\mathrm{t}}$, we arrive at

$$\mathbb{E}_t[\ell_{i,t}] = \sum_{j\in\mathbb{N}_i^{\text{in}}} p_{j,t} \frac{\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t)}{q_{i,t}} = \mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t) \tag{A.7a}$$

$$\mathbb{E}_t[\|\nabla \ell_{i,t}\|^2] = \sum_{j\in\mathbb{N}_i^{\text{in}}} p_{j,t} \frac{\|\nabla\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t)\|^2}{q_{i,t}^2} = \frac{\|\nabla\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t)\|^2}{q_{i,t}}. \tag{A.7b}$$

Therefore, taking the expectation with respect to $\mathbf{1}_{\mathrm{i}\in\mathbb{S}_\mathrm{t}}$ from both sides of (A.6), we obtain

$$\begin{aligned}&\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t) - \mathcal{L}(\boldsymbol{\theta}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t)\\ \leq &\frac{\|\boldsymbol{\theta}_{i,t} - \boldsymbol{\theta}\|^2 - \|\boldsymbol{\theta}_{i,t+1} - \boldsymbol{\theta}\|^2}{2\eta} + \frac{\eta\|\nabla\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t)\|^2}{2q_{i,t}}.\end{aligned} \tag{A.8}$$

Based on (A2), we have $\|\nabla\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t)\|^2 \leq L^2$. Therefore, summing (A.8) over time from $t = 1$ to $t = T$ it can be concluded that

$$\sum_{t=1}^{T} \mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t) - \sum_{t=1}^{T} \mathcal{L}(\boldsymbol{\theta}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t) \leq \frac{\|\boldsymbol{\theta}\|^2 - \|\boldsymbol{\theta}_{i,T+1} - \boldsymbol{\theta}\|^2}{2\eta} + \sum_{t=1}^{T} \frac{\eta L^2}{2q_{i,t}}. \tag{A.9}$$

Putting $\boldsymbol{\theta} = \boldsymbol{\theta}_i^*$ in (A.9) and taking into account that $\|\boldsymbol{\theta}_{i,T+1} - \boldsymbol{\theta}\|^2 \geq 0$, we can write

$$\sum_{t=1}^{T} \mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t) - \sum_{t=1}^{T} \mathcal{L}(\boldsymbol{\theta}^\top \mathbf{z}_i(\boldsymbol{x}_t), y_t) \leq \frac{\|\boldsymbol{\theta}_i^*\|^2}{2\eta} + \sum_{t=1}^{T} \frac{\eta L^2}{2q_{i,t}} \tag{A.10}$$

which completes the proof of Lemma A.1. □

Lemma A.2. *Under (A1) and (A2), the following holds*

$$\sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(\hat{f}_{RF}(\boldsymbol{x}_t), y_t)] - \sum_{t=1}^{T} \mathcal{L}(\hat{f}_{RF,i}(\boldsymbol{x}_t), y_t) \leq \frac{\ln N}{\eta} + \eta_e JT + \frac{\eta NT}{2(1-\eta_e)} \tag{A.11}$$

where η is the learning rate, η_e is the exploration rate, $q_{i,t} = \sum_{j=1}^{J} p_{j,t}\left(1-(1-\pi_{ij,t})^M\right)$ and N denotes the number of kernels.

Proof. Let $W_t = \sum_{n=1}^{N} w_{n,t}$. For any t we find

$$\frac{W_{t+1}}{W_t} = \sum_{j=1}^{J} p_{j,t} \frac{W_{t+1}}{W_t} = \sum_{j=1}^{J} p_{j,t} \sum_{i=1}^{N} \frac{w_{i,t+1}}{W_t} = \sum_{j=1}^{J} p_{j,t} \sum_{i=1}^{N} \frac{w_{i,t}}{W_t} \exp(-\eta \ell_{i,t}). \tag{A.12}$$

Based on (1.3), we have

$$\frac{w_{i,t}}{W_t} = \frac{\pi_{ij,t} - \frac{\eta_e^j}{N}}{1 - \eta_e^j}, \forall j \in \{1, ..., J\}. \tag{A.13}$$

Combining (A.12) with (A.13) obtains

$$\frac{W_{t+1}}{W_t} = \sum_{j=1}^{J} p_{j,t} \sum_{i=1}^{N} \frac{\pi_{ij,t} - \frac{\eta_e^j}{N}}{1 - \eta_e^j} \exp(-\eta \ell_{i,t}). \tag{A.14}$$

Using the inequality $e^{-x} \leq 1 - x + \frac{1}{2}x^2, \forall x \geq 0$, (A.14) leads to

$$\frac{W_{t+1}}{W_t} \leq \sum_{j=1}^{J} p_{j,t} \sum_{i=1}^{N} \frac{\pi_{ij,t} - \frac{\eta_e^j}{N}}{1 - \eta_e^j} \bigg(1 - \eta \ell_{i,t} + \frac{1}{2}(\eta \ell_{i,t})^2\bigg). \tag{A.15}$$

Taking logarithm from both sides of inequality (A.15), and use the fact that $1 + x \leq e^x$, we have

$$\ln \frac{W_{t+1}}{W_t} \leq \sum_{j=1}^{J} p_{j,t} \sum_{i=1}^{N} \frac{\pi_{ij,t} - \frac{\eta_e^j}{N}}{1 - \eta_e^j} \bigg(-\eta \ell_{i,t} + \frac{1}{2}(\eta \ell_{i,t})^2\bigg). \tag{A.16}$$

Summing (A.16) over t from 1 to T results in

$$\ln \frac{W_{T+1}}{W_1} \leq \sum_{t=1}^{T} \sum_{j=1}^{J} p_{j,t} \sum_{i=1}^{N} \frac{\pi_{ij,t} - \frac{\eta_e^j}{N}}{1 - \eta_e^j} \bigg(-\eta \ell_{i,t} + \frac{1}{2}(\eta \ell_{i,t})^2\bigg). \tag{A.17}$$

Furthermore, recall the updating rule of $w_{i,T+1}$ in (1.37), for any i we have

$$\ln \frac{W_{T+1}}{W_1} \geq \ln \frac{w_{i,T+1}}{W_1} = -\ln N - \sum_{t=1}^{T} \eta \ell_{i,t}. \tag{A.18}$$

Combining (A.17) with (A.18) results in

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t}\sum_{i=1}^{N}\frac{\pi_{ij,t}}{1-\eta_e^j}(\eta\ell_{i,t}) - \sum_{t=1}^{T}\eta\ell_{i,t} \\
\leq &\ln N + \sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t}\sum_{i=1}^{N}\frac{\frac{\eta_e^j}{N}}{1-\eta_e^j}(\eta\ell_{i,t}) + \sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t}\sum_{i=1}^{N}\frac{\pi_{ij,t}-\frac{\eta_e^j}{N}}{1-\eta_e^j}\left(\frac{1}{2}(\eta\ell_{n,t})^2\right). \qquad \text{(A.19)}
\end{aligned}$$

Multiplying both sides by $\frac{1-\eta_e^J}{\eta}$, we arrive at

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t}\sum_{i=1}^{N}\pi_{ij,t}\frac{1-\eta_e^J}{1-\eta_e^j}\ell_{i,t} - \sum_{t=1}^{T}(1-\eta_e^J)\ell_{i,t} \\
\leq &\frac{1-\eta_e^J}{\eta}\ln N + \sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t}\sum_{i=1}^{N}\frac{\eta_e^j(1-\eta_e^J)}{N(1-\eta_e^j)}\ell_{i,t} \\
&+ \sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t}\sum_{i=1}^{N}\frac{(1-\eta_e^J)(\pi_{ij,t}-\frac{\eta_e^j}{N})}{1-\eta_e^j}(\frac{\eta}{2}\ell_{i,t}^2). \qquad \text{(A.20)}
\end{aligned}$$

Also, using the fact that $0 < \eta_e \leq 1$ we can conclude that $1-\eta_e^J < 1$ and for all $j \geq 1$, $\eta_e^j \leq \eta_e$, the RHS of (A.20) can be upper bounded by

$$\begin{aligned}
&\frac{1-\eta_e^J}{\eta}\ln N + \sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t}\sum_{i=1}^{N}\frac{\eta_e^j(1-\eta_e^J)}{N(1-\eta_e^j)}\ell_{i,t} + \sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t}\sum_{i=1}^{N}\frac{(1-\eta_e^J)(\pi_{ij,t}-\frac{\eta_e^j}{N})}{1-\eta_e^j}(\frac{\eta}{2}\ell_{i,t}^2) \\
\leq &\frac{\ln N}{\eta} + \sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t}\sum_{i=1}^{N}\frac{\eta_e(1-\eta_e^J)}{N(1-\eta_e)}\ell_{i,t} + \sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t}\sum_{i=1}^{N}\frac{\pi_{ij,t}-\frac{\eta_e^j}{N}}{1-\eta_e^j}(\frac{\eta}{2}\ell_{i,t}^2). \qquad \text{(A.21)}
\end{aligned}$$

Since $1-\eta_e^J = (1-\eta_e)(1+\ldots+\eta_e^{J-1})$ and $\eta_e \leq 1$, the following bound holds for the second term on the RHS of (A.21)

$$\begin{aligned}
\sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t}\sum_{i=1}^{N}\frac{\eta_e(1-\eta_e^J)}{N(1-\eta_e)}\ell_{i,t} =&\sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t}\sum_{i=1}^{N}\frac{\eta_e(1+\ldots+\eta_e^{J-1})}{N}\ell_{i,t} \\
\leq&\sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t}\sum_{i=1}^{N}\frac{\eta_e J}{N}\ell_{i,t}. \qquad \text{(A.22)}
\end{aligned}$$

Meanwhile, as $\eta_e^J \leq \eta_e^j$ for all j, $1 \leq j \leq J$, the LHS of (A.20) can be bounded by

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t} \sum_{i=1}^{N} \pi_{ij,t} \frac{1-\eta_e^J}{1-\eta_e^j} \ell_{i,t} - \sum_{t=1}^{T} (1-\eta_e^J)\ell_{i,t} \\
\geq &\sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t} \sum_{i=1}^{N} \pi_{ij,t}\ell_{i,t} - \sum_{t=1}^{T} \ell_{i,t}. \qquad \text{(A.23)}
\end{aligned}$$

Combining (A.20), (A.21), (A.22) and (A.23), we can conclude that

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t} \sum_{i=1}^{N} \pi_{ij,t}\ell_{i,t} - \sum_{t=1}^{T} \ell_{i,t} \\
\leq &\frac{\ln N}{\eta} + \sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t} \sum_{i=1}^{N} \frac{\eta_e J}{N} \ell_{i,t} + \sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t} \sum_{i=1}^{N} \frac{\pi_{ij,t} - \frac{\eta_e^j}{N}}{1-\eta_e^j} (\frac{\eta}{2}\ell_{i,t}^2). \qquad \text{(A.24)}
\end{aligned}$$

Recall the probability of observing the loss of the i-th kernel at time t given in (1.21), the expected first and second moments of $\ell_{i,t}$ in (1.16) given the losses incurred up to time instant $t-1$, i.e., $\{\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_\tau), y_\tau)\}_{\tau=1}^{t-1}$ can be written as

$$\mathbb{E}[\ell_{i,t}] = \sum_{j=1}^{J} p_{j,t} \left(1-(1-\pi_{ij,t})^M\right) \frac{\mathcal{L}(\hat{f}_{RF,i}(\boldsymbol{x}_t), y_t)}{q_{i,t}} = \mathcal{L}(\hat{f}_{RF,i}(\boldsymbol{x}_t), y_t) \qquad \text{(A.25a)}$$

$$\mathbb{E}[\ell_{n,t}^2] = \sum_{j=1}^{J} p_{j,t} \left(1-(1-\pi_{ij,t})^M\right) \frac{\mathcal{L}^2(\hat{f}_{RF,i}(\boldsymbol{x}_t), y_t)}{q_{i,t}^2} = \frac{\mathcal{L}^2(\hat{f}_{RF,i}(\boldsymbol{x}_t), y_t)}{q_{i,t}} \leq \frac{1}{q_{i,t}}. \qquad \text{(A.25b)}$$

Based on (A.25b), the third term in the right hand side of (A.24) can be bounded as follows

$$\sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t} \sum_{i=1}^{N} \frac{\pi_{ij,t} - \frac{\eta_e^j}{N}}{1-\eta_e^j} (\frac{\eta}{2}\ell_{n,t}^2) \leq \sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t} \sum_{i=1}^{N} \frac{\pi_{ij,t} - \frac{\eta_e^j}{N}}{1-\eta_e^j} (\frac{\eta}{2q_{i,t}}). \qquad \text{(A.26)}$$

Taking the expected value of (A.24) at each time t given$\{\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_\tau), y_\tau)\}_{\tau=1}^{t-1}$ and combining with (A.25a) and (A.26) we have

$$\begin{aligned}&\sum_{t=1}^{T}\sum_{j=1}^{J}p_{j,t}\sum_{i=1}^{N}\pi_{ij,t}\mathcal{L}(\hat{f}_{RF,i}(\boldsymbol{x}_t),y_t)-\sum_{t=1}^{T}\mathcal{L}(\hat{f}_{RF,i}(\boldsymbol{x}_t),y_t)\\ \le&\frac{\ln N}{\eta}+\sum_{t=1}^{T}\sum_{j=1}^{J}p_{j,t}\sum_{i=1}^{N}\frac{\eta_e J}{N}\mathcal{L}(\hat{f}_{RF,i}(\boldsymbol{x}_t),y_t)+\sum_{t=1}^{T}\sum_{j=1}^{J}p_{j,t}\sum_{i=1}^{N}\frac{\pi_{ij,t}-\frac{\eta_e^j}{N}}{1-\eta_e^j}(\frac{\eta}{2q_{i,t}}).\end{aligned} \tag{A.27}$$

Since $\frac{\pi_{ij,t}-\frac{\eta_e^j}{N}}{(1-\eta_e^j)q_{i,t}} \le \frac{\pi_{ij,t}}{(1-\eta_e)q_{i,t}}$, replace $\frac{\pi_{ij,t}-\frac{\eta_e^j}{N}}{q_{i,t}(1-\eta_e^j)}$ by $\frac{\pi_{ij,t}}{(1-\eta_e)q_{i,t}}$, the inequality in (A.27) still holds

$$\begin{aligned}&\sum_{t=1}^{T}\sum_{j=1}^{J}p_{j,t}\sum_{i=1}^{N}\pi_{ij,t}\mathcal{L}(\hat{f}_{RF,i}(\boldsymbol{x}_t),y_t)-\sum_{t=1}^{T}\mathcal{L}(\hat{f}_{RF,i}(\boldsymbol{x}_t),y_t)\\ \le&\frac{\ln N}{\eta}+\sum_{t=1}^{T}\sum_{j=1}^{J}p_{j,t}\sum_{i=1}^{N}\frac{\eta_e J}{N}\mathcal{L}(\hat{f}_{RF,i}(\boldsymbol{x}_t),y_t)+\frac{\eta}{2(1-\eta_e)}\sum_{t=1}^{T}\sum_{j=1}^{J}p_{j,t}\sum_{i=1}^{N}\frac{\pi_{ij,t}}{q_{i,t}}.\end{aligned} \tag{A.28}$$

Moreover, based on (1.21), the probability $q_{i,t}$ can be bounded from below as

$$\begin{aligned}q_{i,t}&=\sum_{j=1}^{J}p_{j,t}\left(1-(1-\pi_{ij,t})^M\right)\\ &=\sum_{j=1}^{J}p_{j,t}\pi_{ij,t}\left(1+(1-\pi_{ij,t})+\ldots+(1-\pi_{ij,t})^{M-1}\right)>\sum_{j=1}^{J}p_{j,t}\pi_{ij,t}\end{aligned} \tag{A.29}$$

Therefore, according to (A.29), for the third term in the right hand side of (A.28) we have

$$\frac{\eta}{2(1-\eta_e)}\sum_{t=1}^{T}\sum_{j=1}^{J}p_{j,t}\sum_{i=1}^{N}\frac{\pi_{ij,t}}{q_{i,t}}<\frac{\eta NT}{2(1-\eta_e)}. \tag{A.30}$$

Furthermore, based on that $\mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t),y_t)\le 1$ in (A2), the following inequality holds

$$\sum_{t=1}^{T}\sum_{j=1}^{J}p_{j,t}\sum_{i=1}^{N}\frac{\eta_e J}{N}\mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t),y_t)\le\sum_{t=1}^{T}\sum_{j=1}^{J}p_{j,t}\sum_{i=1}^{N}\frac{\eta_e J}{N}=\eta_e JT. \tag{A.31}$$

From (A.28), (A.30) and (A.31), we can conclude that

$$\sum_{t=1}^{T}\sum_{j=1}^{J} p_{j,t}\sum_{n=1}^{N}\pi_{ij,t}\mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t),y_t)-\sum_{t=1}^{T}\mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t),y_t)$$
$$\le\frac{\ln N}{\eta}+\eta_e JT+\frac{\eta NT}{2(1-\eta_e)}. \tag{A.32}$$

According to the procedure of generating the graph $\mathcal{B}_t$ which is presented in Algorithm 2, for each selective node $v_{s,j}$ a subset of kernels is chosen using PMF $\pi_{ij,t}$ in M independent trials. In fact, a subset of kernels is assigned to each node $v_{s,j}$ in M independent trials and in each trial one kernel is assigned and its associated entry in the sub-adjacency matrix A becomes 1. Now, let b_i represents the frequency that the i-th kernel is chosen in M independent trials. Thus, $\{b_i\}_{i=1}^N$ can be viewed as the solution to the following linear equation

$$b_1+\ldots+b_N=M,\quad \text{s.t.}\;\; b_i\ge 0, b_i\in\mathbb{N} \tag{A.33}$$

where $\mathbb{N}$ denotes the set of natural numbers. There are $\binom{N+M-1}{N}$ different solutions for (A.33). Let, $\{b_{i,k}\}_{i=1}^N$ denotes the k-th set of solution for (A.33). Based on Jensen's inequality, for the expected value of $\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t),y_t)$ we have

$$\begin{aligned}\mathbb{E}_t[\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t),y_t)]&=\sum_{j=1}^{J}p_{j,t}\sum_{k=1}^{\binom{N+M-1}{N}}\left(\prod_{i=1}^{N}(\pi_{ij,t})^{b_{i,k}}\right)\mathcal{L}(\sum_{i\in\mathcal{S}_t}\bar{w}_{i,t}\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t),y_t)\\&\le\sum_{j=1}^{J}p_{j,t}\sum_{k=1}^{\binom{N+M-1}{N}}\left(\prod_{i=1}^{N}(\pi_{ij,t})^{b_{i,k}}\right)\sum_{i\in\mathcal{S}_t}\bar{w}_{i,t}\mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t),y_t).\end{aligned} \tag{A.34}$$

Also, considering (A.34) and the fact that $\bar{w}_{i,t} \leq 1$, we can conclude that

$$\begin{aligned}\mathbb{E}_t[\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t)] &\leq \sum_{j=1}^{J} p_{j,t} \sum_{k=1}^{\binom{N+M-1}{N}} \left(\prod_{i=1}^{N} (\pi_{ij,t})^{b_{i,k}}\right) \sum_{i \in \mathcal{S}_t} \bar{w}_{i,t} \mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t) \\ &\leq \sum_{j=1}^{J} p_{j,t} \sum_{k=1}^{\binom{N+M-1}{N}} \left(\prod_{i=1}^{N} (\pi_{ij,t})^{b_{i,k}}\right) \sum_{i \in \mathcal{S}_t} \mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t). \end{aligned} \tag{A.35}$$

Note that the number of ways to solve (A.33) when the i-th kernel is chosen for at least one time equals to the number of ways to solve the following problem

$$\tilde{b}_{1,i} + \ldots + \tilde{b}_{N,i} = M - 1, \quad \text{s.t. } \tilde{b}_{m,i} \geq 0, \; \tilde{b}_{m,i} \in \mathbb{N}. \tag{A.36}$$

There are $\binom{N+M-2}{N}$ different solutions for (A.36). Let $\{\tilde{b}_{m,i}^{(k)}\}_{i=1}^{N}$ denotes k-th set of solution for (A.36). Therefore, based on this, from (A.35) we can conclude the following equality

$$\begin{aligned}&\sum_{j=1}^{J} p_{j,t} \sum_{k=1}^{\binom{N+M-1}{N}} \left(\prod_{i=1}^{N} (\pi_{ij,t})^{b_{i,k}}\right) \sum_{i \in \mathcal{S}_t} \mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t) \\ =&\sum_{j=1}^{J} p_{j,t} \sum_{i=1}^{N} \pi_{ij,t} \sum_{k=1}^{\binom{N+M-2}{N}} \left(\prod_{m=1}^{N} (\pi_{mj,t})^{\tilde{b}_{m,i}^{(k)}}\right) \mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t)\end{aligned} \tag{A.37}$$

where $\sum_{k=1}^{\binom{N+M-2}{N}} \left(\prod_{m=1}^{N} (\pi_{mj,t})^{\tilde{b}_{m,i}^{(k)}}\right)$ is the total probability of all $\binom{N+M-2}{N}$ possible solutions of (A.36). Therefore, $\sum_{k=1}^{\binom{N+M-2}{N}} \left(\prod_{m=1}^{N} (\pi_{mj,t})^{\tilde{b}_{m,n}^{(k)}}\right) = 1$. Substituting (A.37) into (A.34), we obtain

$$\mathbb{E}_t[\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t)] \leq \sum_{j=1}^{J} p_{j,t} \sum_{i=1}^{N} \pi_{ij,t} \mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t). \tag{A.38}$$

Combining (A.32) with (A.38) leads to

$$\sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t)] - \sum_{t=1}^{T} \mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t) \leq \frac{\ln N}{\eta} + \eta_e JT + \frac{\eta NT}{2(1-\eta_e)} \tag{A.39}$$

which concludes to proof of Lemma A.2. □

From Lemma A.1 and Lemma A.2, we conclude that for any $i : 1 \leq i \leq N$ we have

$$\begin{aligned}&\sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t)] - \sum_{t=1}^{T} \mathcal{L}(\hat{f}_i^*(\boldsymbol{x}_t), y_t)\\ \leq &\frac{\ln N}{\eta} + \frac{\|\boldsymbol{\theta}_i^*\|^2}{2\eta} + \eta_e JT + \frac{\eta NT}{2(1-\eta_e)} + \frac{\eta}{2}\sum_{t=1}^{T}\frac{L^2}{q_{i,t}}.\end{aligned} \tag{A.40}$$

From (A.29) and the facts that $p_{j,t} > \frac{\eta_e}{J}$ and $\pi_{ij,t} > \frac{\eta_e^j}{N}$, the following inequality can be concluded

$$q_{i,t} \geq \sum_{j=1}^{J} p_{j,t}\pi_{ij,t} > p_{1,t}\pi_{i1,t} > \frac{\eta_e^2}{NJ}. \tag{A.41}$$

Therefore, we find $q_{i,t} > \frac{\eta_e^2}{NJ}$, $\forall i : 1 \leq i \leq N$, $\forall t : 1 \leq t \leq T$. Combining (A.40) and (A.41) we can conclude that

$$\begin{aligned}&\sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t)] - \sum_{t=1}^{T} \mathcal{L}(\hat{f}_i^*(\boldsymbol{x}_t), y_t)\\ < &\frac{\ln N}{\eta} + \frac{\|\boldsymbol{\theta}_i^*\|^2}{2\eta} + \eta_e JT + \frac{\eta NT}{2(1-\eta_e)} + \frac{\eta L^2 NJT}{2\eta_e^2}.\end{aligned} \tag{A.42}$$

Hence, Lemma 1.1 is proved.

A.3 Proof of Theorem 1.1

To prove Theorem 1.1, the following lemma is exploited.

Lemma A.3. *For the optimal function estimator in $\mathbb{H}_i$ expressed as $f_i^*(\boldsymbol{x}) := \sum_{t=1}^{T} \alpha_{i,t}^* \kappa_i(\boldsymbol{x}, \boldsymbol{x}_t)$ and its RF-based approximant $\hat{f}_i^*(\boldsymbol{x}, \boldsymbol{x}_t) = \sum_{t=1}^{T} \alpha_{i,t}^* \mathbf{z}_i^\top(\boldsymbol{x})\mathbf{z}_i(\boldsymbol{x}_t)$, the following bound holds* *with probability at least* $1 - 2^8(\frac{\sigma_i}{\epsilon})^2 \exp(-\frac{D\epsilon^2}{4d+8})$

$$\left|\sum_{t=1}^{T} \mathcal{L}(\hat{f}_i^*(\boldsymbol{x}_t), y_t) - \sum_{t=1}^{T} \mathcal{L}(f_i^*(\boldsymbol{x}_t), y_t)\right| \leq \epsilon LTC \tag{A.43}$$

where the equality happens if we have $C := \max_i \sum_{t=1}^{T} |\alpha_{i,t}^*|$.

Proof. For a given shift invariant kernel κ_i, the maximum point-wise error of the random feature kernel approximant is uniformly bounded with probability at least $1-2^8(\frac{\sigma_i}{\epsilon})^2 \exp(-\frac{D\epsilon^2}{4d+8})$, by [123]

$$\sup_{\boldsymbol{x}_j, \boldsymbol{x}_k \in \mathcal{X}} |\mathbf{z}_i^\top(\boldsymbol{x}_j)\mathbf{z}_i(\boldsymbol{x}_k) - \kappa_i(\boldsymbol{x}_j, \boldsymbol{x}_k)| < \epsilon \tag{A.44}$$

where σ_i^2 is the second moment of the Fourier transform of κ_i. Therefore, under (A3) this implies that $\sup_{\boldsymbol{x}_\tau, \boldsymbol{x}_t \in \mathcal{X}} \mathbf{z}_i^\top(\boldsymbol{x}_\tau)\mathbf{z}_i(\boldsymbol{x}_t) \leq 1+\epsilon$ holds with probability at least $1-2^8(\frac{\sigma_i}{\epsilon})^2 \exp(-\frac{D\epsilon^2}{4d+8})$. Let $C := \max_n \sum_{t=1}^{T} |\alpha_{n,t}^*|$. Hence, $\|\boldsymbol{\theta}_j^*\|^2$ can be bounded from above as

$$\|\boldsymbol{\theta}_j^*\|^2 \leq \| \sum_{t=1}^{T} \alpha_{j,t}^* \mathbf{z}_j(\boldsymbol{x}_t)\|^2 \leq | \sum_{t=1}^{T}\sum_{\tau=1}^{T} \alpha_{j,t}^* \alpha_{j,\tau}^* \mathbf{z}_j^\top(\boldsymbol{x}_t)\mathbf{z}_j(\boldsymbol{x}_\tau)| \leq (1+\epsilon)C^2 \tag{A.45}$$

with probability at least $1-2^8(\frac{\sigma_i}{\epsilon})^2 \exp(-\frac{D\epsilon^2}{4d+8})$. Moreover, using the triangle inequality yields

$$\left|\sum_{t=1}^{T} \mathcal{L}(\hat{f}_j^*(\boldsymbol{x}_t), y_t) - \sum_{t=1}^{T} \mathcal{L}(f_j^*(\boldsymbol{x}_t), y_t)\right| \leq \sum_{t=1}^{T} \left|\mathcal{L}(\hat{f}_j^*(\boldsymbol{x}_t), y_t) - \mathcal{L}(f_j^*(\boldsymbol{x}_t), y_t)\right|. \tag{A.46}$$

According to Lipschitz continuity of the loss function, it can be concluded that

$$\begin{aligned}&\sum_{t=1}^{T} \left|\mathcal{L}(\hat{f}_j^*(\boldsymbol{x}_t), y_t) - \mathcal{L}(f_j^*(\boldsymbol{x}_t), y_t)\right| \\ &\leq \sum_{t=1}^{T} L \left|\sum_{\tau=1}^{T} \alpha_{j,\tau}^* \mathbf{z}_j^\top(\boldsymbol{x}_\tau)\mathbf{z}_n(\boldsymbol{x}_t) - \sum_{\tau=1}^{T} \alpha_{j,\tau}^* \kappa_j(\boldsymbol{x}_\tau, \boldsymbol{x}_t)\right|.\end{aligned} \tag{A.47}$$

Using the Cauchy-Schwartz inequality, we can write

$$
\begin{aligned}
&\sum_{t=1}^{T} L \left| \sum_{\tau=1}^{T} \alpha_{j,\tau}^{*} \mathbf{z}_j^{\top}(\boldsymbol{x}_\tau)\mathbf{z}_n(\boldsymbol{x}_t) - \sum_{\tau=1}^{T} \alpha_{j,\tau}^{*} \kappa_j(\boldsymbol{x}_\tau, \boldsymbol{x}_t) \right| \\
&\leq \sum_{t=1}^{T} L \sum_{\tau=1}^{T} |\alpha_{j,\tau}^{*}| \left| \mathbf{z}_j^{\top}(\boldsymbol{x}_\tau)\mathbf{z}_j(\boldsymbol{x}_t) - \kappa_j(\boldsymbol{x}_\tau, \boldsymbol{x}_t) \right|. \qquad \text{(A.48)}
\end{aligned}
$$

Using (A.44) and (A.48), we can conclude that the inequality

$$
\left| \sum_{t=1}^{T} \mathcal{L}(\hat{f}_j^{*}(\boldsymbol{x}_t), y_t) - \sum_{t=1}^{T} \mathcal{L}(f_j^{*}(\boldsymbol{x}_t), y_t) \right| \leq \epsilon LTC \qquad \text{(A.49)}
$$

holds with probability at least $1 - 2^8 (\frac{\sigma_i}{\epsilon})^2 \exp(-\frac{D\epsilon^2}{4d+8})$. □

Combining Lemma 1.1 with Lemma A.3 and (A.45), it can be concluded that the following bound holds with probability at least $1 - 2^8 (\frac{\sigma_n}{\epsilon})^2 \exp(-\frac{D\epsilon^2}{4d+8})$,

$$
\begin{aligned}
&\sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t)] - \sum_{t=1}^{T} \mathcal{L}(f_i^{*}(\boldsymbol{x}_t), y_t) \\
=&\sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t)] - \sum_{t=1}^{T} \mathcal{L}(\hat{f}_i^{*}(\boldsymbol{x}_t), y_t) + \sum_{t=1}^{T} \mathcal{L}(\hat{f}_i^{*}(\boldsymbol{x}_t), y_t) - \sum_{t=1}^{T} \mathcal{L}(f_i^{*}(\boldsymbol{x}_t), y_t) \\
<&\frac{\ln N}{\eta} + \frac{\|\boldsymbol{\theta}_i^{*}\|^2}{2\eta} + \eta_e JT + \epsilon LTC + \frac{\eta NT}{2(1-\eta_e)} + \frac{\eta L^2 NJT}{2\eta_e^2} \qquad \text{(A.50)}
\end{aligned}
$$

which completes the proof of Theorem 1.1.

A.4 Proof of Lemma 1.3

According to (1.2), we obtain

$$
\frac{1}{\mathcal{U}_d} \int |\hat{f}_i(\boldsymbol{x}) - \hat{f}_j(\boldsymbol{x})|^2 d\boldsymbol{x} = \frac{1}{\mathcal{U}_d} \int |\sum_{t=1}^{T} \alpha_{i,t}\kappa_i(\boldsymbol{x}, \boldsymbol{x}_t) - \sum_{t=1}^{T} \alpha_{j,t}\kappa_j(\boldsymbol{x}, \boldsymbol{x}_t)|^2 d\boldsymbol{x}. \qquad \text{(A.51)}
$$

Applying Arithmetic Mean-Geometric Mean inequality on the right hand side of (A.51), we find

$$\begin{aligned}
&\frac{1}{\mathcal{U}_d}\int |\hat{f}_i(\boldsymbol{x}) - \hat{f}_j(\boldsymbol{x})|^2 d\boldsymbol{x} \\
\leq &\frac{2}{\mathcal{U}_d}\int \left(|\sum_{t=1}^{T} \alpha_{i,t}\left(\kappa_i(\boldsymbol{x}, \boldsymbol{x}_t) - \kappa_j(\boldsymbol{x}, \boldsymbol{x}_t)\right)|^2 + |\sum_{t=1}^{T} (\alpha_{j,t} - \alpha_{i,t})\kappa_j(\boldsymbol{x}, \boldsymbol{x}_t)|^2 \right) d\boldsymbol{x}. \qquad \text{(A.52)}
\end{aligned}$$

Using Cauchy-Schwartz inequality, (A.52) can be further relaxed to

$$\begin{aligned}
&\frac{1}{\mathcal{U}_d}\int |\hat{f}_i(\boldsymbol{x}) - \hat{f}_j(\boldsymbol{x})|^2 d\boldsymbol{x} \\
\leq &\frac{2}{\mathcal{U}_d}\int (\sum_{t=1}^{T} |\alpha_{i,t}|^2)(\sum_{t=1}^{T} |\kappa_i(\boldsymbol{x}, \boldsymbol{x}_t) - \kappa_j(\boldsymbol{x}, \boldsymbol{x}_t)|^2) d\boldsymbol{x} \\
&+ \frac{2}{\mathcal{U}_d}\int (\sum_{t=1}^{T} |\alpha_{j,t} - \alpha_{i,t}|^2)(\sum_{t=1}^{T} |\kappa_j(\boldsymbol{x}, \boldsymbol{x}_t)|^2) d\boldsymbol{x}. \qquad \text{(A.53)}
\end{aligned}$$

Considering the fact that $C_m := \max_i \sum_{t=1}^{T} |\alpha_{i,t}|^2$, from (A.53) it can be written that

$$\begin{aligned}
&\frac{1}{\mathcal{U}_d}\int |\hat{f}_i(\boldsymbol{x}) - \hat{f}_j(\boldsymbol{x})|^2 d\boldsymbol{x} \\
\leq &\frac{2C_m}{\mathcal{U}_d}\sum_{t=1}^{T}\int |\kappa_i(\boldsymbol{x}, \boldsymbol{x}_t) - \kappa_j(\boldsymbol{x}, \boldsymbol{x}_t)|^2 d\boldsymbol{x} + \frac{4C_m}{\mathcal{U}_d}\sum_{t=1}^{T}\int |\kappa_j(\boldsymbol{x}, \boldsymbol{x}_t)|^2 d\boldsymbol{x}. \qquad \text{(A.54)}
\end{aligned}$$

Furthermore, based on (A.54) and the fact that $|\kappa_j(\boldsymbol{x}, \boldsymbol{x}_t)|^2 \leq 1$, we can infer that

$$\frac{1}{\mathcal{U}_d}\int |\hat{f}_i(\boldsymbol{x}) - \hat{f}_j(\boldsymbol{x})|^2 d\boldsymbol{x} \leq \frac{2C_m}{\mathcal{U}_d}\sum_{t=1}^{T} (\Delta_S(\kappa_i, \kappa_j) + 2\mathcal{U}_d) \qquad \text{(A.55)}$$

which proves Lemma 1.3.

A.5 Proof of Theorem 1.2

Furthermore, in order to prove Theorem 1.2, the following intermediate Lemma is also proved.

Lemma A.4. *The following inequality holds under (A1) and (A2)*

$$\sum_{t=1}^{T}\mathbb{E}_t[\mathcal{L}(\hat{f}_{RF}(\boldsymbol{x}_t),y_t)]-\sum_{t=1}^{T}\mathcal{L}(\hat{f}_{RF,i}(\boldsymbol{x}_t),y_t)\leq\frac{\ln N}{\eta}+\eta(1+\frac{N}{2}-\frac{\eta}{2})T \tag{A.56}$$

where $\mathcal{L}_i(\hat{f}_{RF}(\boldsymbol{x}_t),y_t)$ denote the loss of function approximation when v_i is drawn.

Proof. For any t, we can write

$$\frac{U_{t+1}}{U_t}=\sum_{i=1}^{N}\frac{u_{i,t+1}}{U_t}=\sum_{i=1}^{N}\frac{u_{i,t}}{U_t}\exp(-\eta\hat{\ell}_{i,t}). \tag{A.57}$$

Based on (1.33), we have $\frac{u_{i,t}}{U_t}=\frac{p_{i,t}-\frac{\xi}{|\mathbb{D}|}\mathbf{1}_{\mathrm{i}\in\mathbb{D}}}{1-\xi}$ and as a result (A.57) can be rewritten as

$$\frac{U_{t+1}}{U_t}=\sum_{i=1}^{N}\frac{p_{i,t}-\frac{\xi}{|\mathbb{D}|}\mathbf{1}_{\mathrm{i}\in\mathbb{D}}}{1-\xi}\exp(-\eta\hat{\ell}_{i,t}). \tag{A.58}$$

Using the inequality $e^{-x}\leq 1-x+\frac{1}{2}x^2,\forall x\geq 0$ and (A.58), it can be concluded that

$$\frac{U_{t+1}}{U_t}\leq\sum_{i=1}^{N}\frac{p_{i,t}-\frac{\xi}{|\mathbb{D}|}\mathbf{1}_{\mathrm{i}\in\mathbb{D}}}{1-\xi}\left(1-\eta\hat{\ell}_{i,t}+\frac{1}{2}(\eta\hat{\ell}_{i,t})^2\right). \tag{A.59}$$

Taking logarithm from both sides of inequality (A.59), and use the fact that $1+x\leq e^x$, we arrive at

$$\ln\frac{U_{t+1}}{U_t}\leq\sum_{i=1}^{N}\frac{p_{i,t}-\frac{\xi}{|\mathbb{D}|}\mathbf{1}_{\mathrm{i}\in\mathbb{D}}}{1-\xi}\left(-\eta\hat{\ell}_{i,t}+\frac{1}{2}(\eta\hat{\ell}_{i,t})^2\right). \tag{A.60}$$

Summing (A.60) over t leads to

$$\ln \frac{U_{T+1}}{U_1} \leq \sum_{t=1}^{T}\sum_{i=1}^{N} \frac{p_{i,t} - \frac{\xi}{|\mathbb{D}|}\mathbf{1}_{\mathrm{i}\in\mathbb{D}}}{1-\xi}\left(-\eta\hat{\ell}_{i,t} + \frac{1}{2}(\eta\hat{\ell}_{i,t})^2\right). \tag{A.61}$$

Furthermore, $\ln \frac{U_{T+1}}{U_1}$ can be bounded from below as

$$\ln \frac{U_{T+1}}{U_1} \geq \ln \frac{u_{i,T+1}}{U_1} = -\sum_{t=1}^{T}\eta\hat{\ell}_{i,t} - \ln N \tag{A.62}$$

for any i such that $1 \leq i \leq N$. Combining (A.61) with (A.62), we obtain

$$\begin{aligned}&\sum_{t=1}^{T}\sum_{i=1}^{N}\frac{p_{i,t}\eta}{1-\xi}\hat{\ell}_{i,t} - \sum_{t=1}^{T}\eta\hat{\ell}_{i,t}\\ &\leq \ln N + \sum_{t=1}^{T}\sum_{i=1}^{N}\frac{\eta\xi\mathbf{1}_{\mathrm{i}\in\mathbb{D}}}{(1-\xi)|\mathbb{D}|}\hat{\ell}_{i,t} + \sum_{t=1}^{T}\sum_{i=1}^{N}\frac{p_{i,t} - \frac{\xi}{|\mathbb{D}|}\mathbf{1}_{\mathrm{i}\in\mathbb{D}}}{1-\xi}\left(\frac{1}{2}(\eta\hat{\ell}_{i,t})^2\right).\end{aligned} \tag{A.63}$$

Multiplying both sides by $\frac{1-\xi}{\eta}$ it can be concluded that

$$\begin{aligned}&\sum_{t=1}^{T}\sum_{i=1}^{N}p_{i,t}\hat{\ell}_{i,t} - \sum_{t=1}^{T}\hat{\ell}_{i,t}\\ \leq&\frac{\ln N}{\eta} + \sum_{t=1}^{T}\sum_{i=1}^{N}\frac{\xi\mathbf{1}_{\mathrm{i}\in\mathbb{D}}}{|\mathbb{D}|}\hat{\ell}_{i,t} + \sum_{t=1}^{T}\sum_{i=1}^{N}\frac{\eta(p_{i,t} - \frac{\xi}{|\mathbb{D}|}\mathbf{1}_{\mathrm{i}\in\mathbb{D}})}{2}\hat{\ell}_{i,t}^2.\end{aligned} \tag{A.64}$$

In addition, taking the expectation of $\hat{\ell}_{i,t}$ and $\hat{\ell}_{i,t}^2$, we get

$$\mathbb{E}_t[\hat{\ell}_{i,t}] = p_{i,t}\frac{\mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t)}{p_{i,t}} = \mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t) \tag{A.65a}$$

$$\mathbb{E}_t[\hat{\ell}_{i,t}^2] = p_{i,t}\frac{\mathcal{L}^2(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t)}{p_{i,t}^2} = \frac{\mathcal{L}^2(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t)}{p_{i,t}} \leq \frac{1}{p_{i,t}} \tag{A.65b}$$

Thus, taking the expectation from both sides of (A.64) leads to

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{i=1}^{N} p_{i,t}\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t) - \sum_{t=1}^{T}\mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t) \\
\leq &\frac{\ln N}{\eta} + \sum_{t=1}^{T}\sum_{i=1}^{N} \frac{\xi \mathbf{1}_{\mathrm{i}\in\mathbb{D}}}{|\mathbb{D}|}\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t) + \sum_{t=1}^{T}\sum_{i=1}^{N} \frac{\eta(p_{i,t} - \frac{\xi}{|\mathbb{D}|}\mathbf{1}_{\mathrm{i}\in\mathbb{D}})}{2p_{i,t}}. \qquad (A.66)
\end{aligned}$$

Taking into account that $\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t) \leq 1$ and based on (A.66) we can write

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{i=1}^{N} p_{i,t}\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t) - \sum_{t=1}^{T}\mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t) \\
\leq &\frac{\ln N}{\eta} + \xi T + \sum_{t=1}^{T}\sum_{i=1}^{N} \frac{\eta(p_{i,t} - \frac{\xi}{|\mathbb{D}|}\mathbf{1}_{\mathrm{i}\in\mathbb{D}})}{2p_{i,t}}. \qquad (A.67)
\end{aligned}$$

Moreover, using (A.67) and the fact that $p_{i,t} \leq 1$, it can be concluded that

$$\sum_{t=1}^{T}\sum_{i=1}^{N} p_{i,t}\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t) - \sum_{t=1}^{T}\mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t) \leq \frac{\ln N}{\eta} + (\xi + \frac{\eta N}{2} - \frac{\eta\xi}{2})T. \qquad (A.68)$$

Furthermore, the expected loss incurred by OMKL-SFG given observed losses in prior time instants can be expressed as

$$\begin{aligned}
\mathbb{E}_t[\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t)] &= \sum_{i=1}^{N} p_{i,t}\mathcal{L}(\sum_{j\in\mathbb{N}_{i,t}^{\mathrm{out}}} \frac{w_{j,t}}{\sum_{k\in\mathbb{N}_{i,t}^{\mathrm{out}}} w_{k,t}} \hat{f}_{\mathrm{RF},j}(\boldsymbol{x}_t), y_t) \\
&= \sum_{i=1}^{N} p_{i,t}\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t). \qquad (A.69)
\end{aligned}$$

Therefore, from (A.68) and (A.69), it can be inferred that

$$\sum_{t=1}^{T}\mathbb{E}_t[\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t)] - \sum_{t=1}^{T}\mathcal{L}(\hat{f}_{\mathrm{RF},i}(\boldsymbol{x}_t), y_t) \leq \frac{\ln N}{\eta} + (\xi + \frac{\eta N}{2} - \frac{\eta\xi}{2})T \qquad (A.70)$$

which establishes the Lemma A.4. □

Furthermore, to prove Theorem 1.2, we prove the following Lemma.

Lemma A.5. *For any $v_i \in \mathcal{V}$ and any $j \in \mathbb{N}_i^{out}$, it can be written that*

$$\sum_{t=1}^{T} \mathcal{L}_i(\hat{f}_{RF}(\boldsymbol{x}_t), y_t) - \sum_{t=1}^{T} \mathcal{L}(\hat{f}_{RF,j}(\boldsymbol{x}_t), y_t) \leq \frac{\ln |\mathbb{N}_i^{out}|}{\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \frac{1}{\bar{q}_{i,t}} \tag{A.71}$$

where $\frac{1}{\bar{q}_{i,t}} = \sum_{j \in \mathbb{N}_i^{out}} \frac{w_{j,t}}{q_{j,t} W_{i,t}}$.

Proof. Let $W_{i,t} = \sum_{j \in \mathbb{N}_i^{\text{out}}} w_{j,t}$. For $v_i \in \mathcal{V}$ we find

$$\frac{W_{i,t+1}}{W_{i,t}} = \sum_{j \in \mathbb{N}_i^{\text{out}}} \frac{w_{j,t+1}}{W_{i,t}} = \sum_{j \in \mathbb{N}_i^{\text{out}}} \frac{w_{j,t}}{W_{i,t}} \exp(-\eta \ell_{j,t}). \tag{A.72}$$

The following inequality can be obtained using the inequality $e^{-x} \leq 1 - x + \frac{1}{2}x^2, \forall x \geq 0$ as follows

$$\frac{W_{i,t+1}}{W_{i,t}} \leq \sum_{j \in \mathbb{N}_i^{\text{out}}} \frac{w_{j,t}}{W_{i,t}} \left(1 - \eta \ell_{j,t} + \frac{1}{2}(\eta \ell_{j,t})^2\right). \tag{A.73}$$

Taking the logarithm from both sides of (A.73) and using the inequality $1 + x \leq e^x$, we get

$$\ln \frac{W_{i,t+1}}{W_{i,t}} \leq \sum_{j \in \mathbb{N}_i^{\text{out}}} \frac{w_{j,t}}{W_{i,t}} \left(-\eta \ell_{j,t} + \frac{1}{2}(\eta \ell_{j,t})^2\right). \tag{A.74}$$

Summing (A.74) over time, we obtain

$$\ln \frac{W_{i,T+1}}{W_{i,1}} \leq \sum_{t=1}^{T} \sum_{j \in \mathbb{N}_i^{\text{out}}} \frac{w_{j,t}}{W_{i,t}} \left(-\eta \ell_{j,t} + \frac{1}{2}(\eta \ell_{j,t})^2\right). \tag{A.75}$$

Moreover, for any $j \in \mathbb{N}_i^{\text{out}}$, $\ln \frac{W_{i,T+1}}{W_{i,1}}$ can be bounded from below as

$$\ln \frac{W_{i,T+1}}{W_{i,1}} \geq \ln \frac{w_{j,T+1}}{W_{i,1}} = -\sum_{t=1}^{T} \eta \ell_{j,t} - \ln |\mathbb{N}_i^{\text{out}}| \tag{A.76}$$

Combining (A.75) with (A.76), it can concluded that

$$\sum_{t=1}^{T}\sum_{j\in\mathbb{N}_i^{\text{out}}}\frac{w_{j,t}}{W_{i,t}}\ell_{j,t}-\sum_{t=1}^{T}\ell_{j,t}\leq\frac{\ln|\mathbb{N}_i^{\text{out}}|}{\eta}+\frac{\eta}{2}\sum_{t=1}^{T}\sum_{j\in\mathbb{N}_i^{\text{out}}}\frac{w_{j,t}}{W_{i,t}}\ell_{j,t}^2. \tag{A.77}$$

For the expected value of $\ell_{i,t}$ and $\ell_{i,t}^2$, we have

$$\mathbb{E}_t[\ell_{j,t}]=\sum_{k\in\mathbb{N}_i^{\text{out}}}p_{k,t}\frac{\mathcal{L}(\boldsymbol{\theta}_{j,t}^\top\mathbf{z}_j(\boldsymbol{x}_t),y_t)}{q_{j,t}}=\mathcal{L}(\boldsymbol{\theta}_{j,t}^\top\mathbf{z}_j(\boldsymbol{x}_t),y_t) \tag{A.78a}$$

$$\mathbb{E}_t[\ell_{j,t}^2]=\sum_{k\in\mathbb{N}_i^{\text{out}}}p_{k,t}\frac{\mathcal{L}^2(\boldsymbol{\theta}_{j,t}^\top\mathbf{z}_j(\boldsymbol{x}_t),y_t)}{q_{j,t}^2}=\frac{\mathcal{L}^2(\boldsymbol{\theta}_{j,t}^\top\mathbf{z}_j(\boldsymbol{x}_t),y_t)}{q_{j,t}}\leq\frac{1}{q_{j,t}} \tag{A.78b}$$

Taking the expectation from (A.77), we get

$$\begin{aligned}&\sum_{t=1}^{T}\sum_{j\in\mathbb{N}_i^{\text{out}}}\frac{w_{j,t}}{W_{i,t}}\mathcal{L}(\boldsymbol{\theta}_{j,t}^\top\mathbf{z}_j(\boldsymbol{x}_t),y_t)-\sum_{t=1}^{T}\mathcal{L}(\boldsymbol{\theta}_{j,t}^\top\mathbf{z}_j(\boldsymbol{x}_t),y_t)\\ \leq&\frac{\ln|\mathbb{N}_i^{\text{out}}|}{\eta}+\frac{\eta}{2}\sum_{t=1}^{T}\sum_{j\in\mathbb{N}_i^{\text{out}}}\frac{w_{j,t}}{q_{j,t}W_{i,t}}.\end{aligned} \tag{A.79}$$

Let $\frac{1}{\bar{q}_{i,t}}=\sum_{j\in\mathbb{N}_i^{\text{out}}}\frac{w_{j,t}}{q_{j,t}W_{i,t}}$ which is the weighted sum of $\frac{1}{q_{j,t}}$ such that $j\in\mathbb{N}_i^{\text{out}}$. Furthermore, according to (1.34), the loss $\mathcal{L}_i(\hat{f}_{\text{RF}}(\boldsymbol{x}_t),y_t)$ can be written as

$$\mathcal{L}_i(\hat{f}_{\text{RF}}(\boldsymbol{x}_t),y_t)=\mathcal{L}(\sum_{j\in\mathbb{N}_i^{\text{out}}}\frac{w_{j,t}}{W_{i,t}}\hat{f}_{\text{RF},j}(\boldsymbol{x}_t),y_t). \tag{A.80}$$

Based on the Jensen's inequality $\mathcal{L}_i(\hat{f}_{\text{RF}}(\boldsymbol{x}_t),y_t)$ can be bounded from above as

$$\mathcal{L}_i(\hat{f}_{\text{RF}}(\boldsymbol{x}_t),y_t)\leq\sum_{j\in\mathbb{N}_i^{\text{out}}}\frac{w_{j,t}}{W_{i,t}}\mathcal{L}(\hat{f}_{\text{RF},j}(\boldsymbol{x}_t),y_t). \tag{A.81}$$

Using (A.79) and (A.81), we can conclude that

$$\sum_{t=1}^{T} \mathcal{L}_i(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t) - \sum_{t=1}^{T} \mathcal{L}(\boldsymbol{\theta}_{j,t}^{\top} \mathbf{z}_j(\boldsymbol{x}_t), y_t) \leq \frac{\ln |\mathbb{N}_i^{\mathrm{out}}|}{\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \frac{1}{\bar{q}_{i,t}} \tag{A.82}$$

which proves the Lemma A.5. $\square$

Combining Lemma A.4 with Lemma A.5, for any $v_j \in \mathcal{V}$ and any $i \in \mathbb{N}_j^{\mathrm{in}}$ we obtain

$$\begin{aligned} &\sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t)] - \sum_{t=1}^{T} \mathcal{L}(\hat{f}_{\mathrm{RF},j}(\boldsymbol{x}_t), y_t) \\ \leq & \frac{\ln N |\mathbb{N}_i^{\mathrm{out}}|}{\eta} + (\xi + \frac{\eta N}{2} - \frac{\eta \xi}{2})T + \frac{\eta}{2} \sum_{t=1}^{T} \frac{1}{\bar{q}_{i,t}}. \end{aligned} \tag{A.83}$$

In addition, combining Lemma A.1 with (A.83), for any $v_j \in \mathcal{V}$ and any $i \in \mathbb{N}_j^{\mathrm{in}}$ we can write

$$\begin{aligned} &\sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t)] - \sum_{t=1}^{T} \mathcal{L}(\hat{f}_j^*(\boldsymbol{x}_t), y_t) \\ \leq & \frac{\ln N |\mathbb{N}_i^{\mathrm{out}}|}{\eta} + \frac{\|\boldsymbol{\theta}_j^*\|^2}{2\eta} + (\xi + \frac{\eta N}{2} - \frac{\eta \xi}{2})T + \frac{\eta}{2} \sum_{t=1}^{T} (\frac{1}{\bar{q}_{i,t}} + \frac{L^2}{q_{j,t}}) \end{aligned} \tag{A.84}$$

We use the above inequality as a step-stone to prove Theorem 1.2.

Therefore, combining (A.84) with (A.45) and (A.49), it can be inferred that the following inequality

$$\begin{aligned} &\sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t)] - \sum_{t=1}^{T} \mathcal{L}(f_j^*(\boldsymbol{x}_t), y_t) \\ \leq & \frac{\ln N |\mathbb{N}_i^{\mathrm{out}}|}{\eta} + \frac{(1+\epsilon)C^2}{2\eta} + \epsilon LTC + (\xi + \frac{\eta N}{2} - \frac{\eta \xi}{2})T + \frac{\eta}{2} \sum_{t=1}^{T} (\frac{1}{\bar{q}_{i,t}} + \frac{L^2}{q_{j,t}}) \end{aligned} \tag{A.85}$$

holds for any $v_j \in \mathcal{V}$ and any $i \in \mathbb{N}_j^{\mathrm{in}}$ with probability at least $1 - 2^8 (\frac{\sigma_i}{\epsilon})^2 \exp(-\frac{D\epsilon^2}{4d+8})$. This completes the proof of Theorem 1.2.

A.6 Proof of Theorem 1.3

According to Theorem 1.2, the following inequality

$$
\begin{aligned}
&\sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t)] - \sum_{t=1}^{T} \mathcal{L}(f^*(\boldsymbol{x}_t), y_t) \\
&\leq \frac{\ln N|\mathbb{N}_i^{\mathrm{out}}|}{\eta} + \frac{(1+\epsilon)C^2}{2\eta} + \epsilon LTC + (\xi + \frac{\eta N}{2} - \frac{\eta\xi}{2})T + \frac{\eta}{2}\sum_{t=1}^{T}(\frac{1}{\bar{q}_{i,t}} + \frac{L^2}{q_{j^*,t}})
\end{aligned} \tag{A.86}
$$

holds with probability at least $1-2^8(\frac{\sigma_{j^*}}{\epsilon})^2\exp(-\frac{D\epsilon^2}{4d+8})$ for any $\epsilon > 0$ and any $i \in \mathbb{N}_{j^*}^{\mathrm{in}}$. When $\mathcal{G}'_t$ is generated by Algorithm 6 as the feedback graph, $\mathbb{D}'_t$ is a dominating set of $\mathcal{G}'_t$. Furthermore, $k \in \mathbb{D}'_t$ if $p_{k,t} \geq \beta$. Based on (1.36), if $i \in \mathbb{N}_{k,t}^{\mathrm{in}}$ (i.e. in-neighborhood of v_k in $\mathcal{G}'_t$), $q_{k,t} \geq p_{i,t}$. Also, considering the condition $\beta \leq \frac{1}{N}$, it is ensured that $\mathbb{D}'_t$ is not an empty set. Moreover, each node v_k in $\mathcal{G}'_t$ is out-neighbor of at least one node in $\mathbb{D}'_t$. Thus, we can conclude that $q_{k,t} \geq \beta$, $\forall v_k \in \mathcal{V}$. Hence, it can be written that

$$
\sum_{t=1}^{T}(\frac{1}{\bar{q}_{i,t}} + \frac{L^2}{q_{j^*,t}}) \geq \sum_{t=1}^{T}(\sum_{j\in\mathbb{N}_i^{\mathrm{out}}} \frac{w_{j,t}}{W_{i,t}\beta} + \frac{L^2}{\beta}) = \frac{(L^2+1)T}{\beta}. \tag{A.87}
$$

Furthermore, since $|\mathbb{N}_i^{\mathrm{out}}| \leq N$, we have $N|\mathbb{N}_i^{\mathrm{out}}| \leq N^2$. Combining (A.86) with (A.87), it can be inferred that in this case the stochastic regret of OMKL-SFG-R satisfies

$$
\begin{aligned}
&\sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(\hat{f}_{\mathrm{RF}}(\boldsymbol{x}_t), y_t)] - \sum_{t=1}^{T} \mathcal{L}(f^*(\boldsymbol{x}_t), y_t) \\
&\leq \frac{2\ln N}{\eta} + \frac{(1+\epsilon)C^2}{2\eta} + \epsilon LTC + (\xi + \frac{\eta}{2}\frac{L^2+N\beta+1}{\beta} - \frac{\eta\xi}{2})T
\end{aligned} \tag{A.88}
$$

with probability at least $1-2^8(\frac{\sigma_{j^*}}{\epsilon})^2\exp(-\frac{D\epsilon^2}{4d+8})$ under (A1)-(A3) for any $\epsilon > 0$ and any $\beta \leq (1-\xi)\max_k \frac{u_{k,t}}{U_t} + \frac{\xi}{N}$. This completes the proof of Theorem 1.3.

Appendix B

Supplementary Proofs for Chapter 2

B.1 Proof of Theorem 2.1

Recall that $W_t = \sum_{i=1}^{K} w_{i,t}$ (below (1.35)), we have

$$\frac{W_{t+1}}{W_t} = \sum_{i=1}^{K} \frac{w_{i,t+1}}{W_t} = \sum_{i=1}^{K} \frac{w_{i,t}}{W_t} \exp\Big(-\eta \hat{\ell}_t(v_i)\Big). \tag{B.1}$$

According to (1.35), we can write

$$\frac{w_{i,t}}{W_t} = \frac{\pi_{i,t} - \eta \bar{F}_{i,t}}{1-\eta} \tag{B.2}$$

where $\bar{F}_{i,t} = \frac{F_{i,t}}{\sum_{j \in \mathcal{D}_t} F_{j,t}} \mathcal{I}(v_i \in \mathcal{D}_t)$. Substituting (B.2) into (B.1) obtains

$$\frac{W_{t+1}}{W_t} = \sum_{i=1}^{K} \frac{\pi_{i,t} - \eta \bar{F}_{i,t}}{1-\eta} \exp\Big(-\eta \hat{\ell}_t(v_i)\Big). \tag{B.3}$$

Using the inequality $e^{-x} \leq 1 - x + \frac{1}{2}x^2, \forall x \geq 0$, the following inequality holds

$$\frac{W_{t+1}}{W_t} \leq \sum_{i=1}^{K} \frac{\pi_{i,t} - \eta \bar{F}_{i,t}}{1-\eta} \left(1 - \eta \hat{\ell}_t(v_i) + \frac{1}{2}(\eta \hat{\ell}_t(v_i))^2\right). \tag{B.4}$$

Taking logarithm of both sides of (B.4) and using the fact that $1 + x \leq e^x$, we have

$$\ln \frac{W_{t+1}}{W_t} \leq \sum_{i=1}^{K} \frac{\pi_{i,t} - \eta \bar{F}_{i,t}}{1-\eta} \left(-\eta \hat{\ell}_t(v_i) + \frac{1}{2}(\eta \hat{\ell}_t(v_i))^2\right). \tag{B.5}$$

Summing (B.5) over time obtains

$$\ln \frac{W_{T+1}}{W_1} \leq \sum_{t=1}^{T} \sum_{i=1}^{K} \frac{\pi_{i,t} - \eta \bar{F}_{i,t}}{1-\eta} \left(-\eta \hat{\ell}_t(v_i) + \frac{1}{2}(\eta \hat{\ell}_t(v_i))^2\right). \tag{B.6}$$

Furthermore, the left hand side of (B.5) can be bounded from below as

$$\ln \frac{W_{T+1}}{W_1} \geq \ln \frac{w_{i,T+1}}{W_1} = -\eta \sum_{t=1}^{T} \hat{\ell}_t(v_i) - \ln K \tag{B.7}$$

where the equality holds due to the fact that $W_1 = \sum_{j=1}^{K} w_{j,1} = K$. Then, (B.6) and (B.7) lead to

$$\begin{aligned} \sum_{t=1}^{T} \sum_{i=1}^{K} \frac{\eta \pi_{i,t}}{(1-\eta)} \hat{\ell}_t(v_i) - \eta \sum_{t=1}^{T} \hat{\ell}_t(v_i) \leq \ln K + \sum_{t=1}^{T} \sum_{i \in \mathcal{D}_t} \frac{\eta^2 \bar{F}_{i,t}}{(1-\eta)} \hat{\ell}_t(v_i) \\ + \sum_{t=1}^{T} \sum_{i=1}^{K} \eta^2 \frac{\pi_{i,t} - \eta \bar{F}_{i,t}}{2(1-\eta)} \hat{\ell}_t(v_i)^2. \end{aligned} \tag{B.8}$$

Multiplying both sides of (B.8) by $\frac{(1-\eta)}{\eta}$

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{i=1}^{K}\pi_{i,t}\hat{\ell}_t(v_i) - \sum_{t=1}^{T}\hat{\ell}_t(v_i) \\
\leq & \frac{\ln K}{\eta} + \sum_{t=1}^{T}\sum_{i\in\mathcal{D}_t}\eta\bar{F}_{i,t}\hat{\ell}_t(v_i) \\
&+ \sum_{t=1}^{T}\sum_{i=1}^{K}\frac{\eta}{2}(\pi_{i,t}-\eta\bar{F}_{i,t})\hat{\ell}_t(v_i)^2.
\end{aligned} \tag{B.9}$$

Furthermore, the expected values of $\hat{\ell}_t(v_i)$ and $\hat{\ell}_t(v_i)^2$ can be written as

$$\mathbb{E}_t[\hat{\ell}_t(v_i)] = \sum_{j=1}^{K}\pi_{j,t}p_{ji,t}\frac{\ell_t(v_i)}{q_{i,t}} = \ell_t(v_i) \tag{B.10a}$$

$$\mathbb{E}_t[\hat{\ell}_t(v_i)^2] = \sum_{j=1}^{K}\pi_{j,t}p_{ji,t}\frac{\ell_t(v_i)^2}{q_{i,t}^2} = \frac{\ell_t(v_i)^2}{q_{i,t}} \leq \frac{1}{q_{i,t}} \tag{B.10b}$$

where the inequality in (B.10b) holds because of (A1) which implies $\ell_t(v_i) \leq 1$. Taking the expectation of both sides of (B.9), we arrive at

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{i=1}^{K}\pi_{i,t}\ell_t(v_i) - \sum_{t=1}^{T}\ell_t(v_i) \\
\leq & \frac{\ln K}{\eta} + \sum_{t=1}^{T}\sum_{i=1}^{K}\eta\bar{F}_{i,t}\ell_t(v_i) \\
&+ \sum_{t=1}^{T}\sum_{i=1}^{K}\frac{\eta}{2}(\pi_{i,t}-\eta\bar{F}_{i,t})\frac{1}{q_{i,t}}.
\end{aligned} \tag{B.11}$$

Moreover, using the fact that $q_{i,t} \leq 1$ we have

$$\frac{\eta^2}{2}\sum_{t=1}^{T}\sum_{i=1}^{K}\frac{\bar{F}_{i,t}}{q_{i,t}} \geq \frac{\eta^2}{2}\sum_{t=1}^{T}\sum_{i=1}^{K}\bar{F}_{i,t} = \frac{\eta^2}{2}\sum_{t=1}^{T}1 = \frac{\eta^2 T}{2}. \tag{B.12}$$

Furthermore, since based on (A1) $\ell_t(v_i) \leq 1$, the second term on the RHS of (B.11) can be bounded by

$$\eta \sum_{t=1}^{T} \sum_{i=1}^{K} \bar{F}_{i,t} \ell_t(v_i) \leq \eta \sum_{t=1}^{T} \sum_{i=1}^{K} \bar{F}_{i,t} = \eta \sum_{t=1}^{T} 1 = \eta T. \tag{B.13}$$

Combining (B.12), (B.13) with (B.11) we have

$$\begin{aligned} & \sum_{t=1}^{T} \sum_{i=1}^{K} \pi_{i,t} \ell_t(v_i) - \sum_{t=1}^{T} \ell_t(v_i) \\ \leq & \frac{\ln K}{\eta} + \eta T - \frac{\eta^2 T}{2} + \frac{\eta}{2} \sum_{t=1}^{T} \sum_{i=1}^{K} \frac{\pi_{i,t}}{q_{i,t}}. \end{aligned} \tag{B.14}$$

By definition, the first term on the RHS of (B.14) equals to $\mathbb{E}_t[\ell_t(v_{I_t})]$. In addition, note that (B.14) holds for all $v_i \in \mathcal{V}$, hence the following inequality holds

$$\begin{aligned} & \sum_{t=1}^{T} \mathbb{E}_t[\ell_t(v_{I_t})] - \min_{v_i \in \mathcal{V}} \sum_{t=1}^{T} \ell_t(v_i) \\ \leq & \frac{\ln K}{\eta} + \eta(1 - \frac{\eta}{2})T + \frac{\eta}{2} \sum_{t=1}^{T} \sum_{i=1}^{K} \frac{\pi_{i,t}}{q_{i,t}} \end{aligned} \tag{B.15}$$

which completes the proof of Theorem 1.2.

B.2 Proof of Lemma 2.1

Based on Theorem 1.2, the upper bound of the expected regret of Exp3-IP is

$$\sum_{t=1}^{T} \mathbb{E}_t[\ell_t(v_{I_t})] - \min_{v_i \in \mathcal{V}} \sum_{t=1}^{T} \ell_t(v_i) \leq \frac{\ln K}{\eta} + \eta(1 - \frac{\eta}{2})T + \frac{\eta}{2} \sum_{t=1}^{T} \sum_{i=1}^{K} \frac{\pi_{i,t}}{q_{i,t}}. \tag{B.16}$$

Let at each time instant t, Q_t is defined as

$$Q_t = 1 + \frac{1}{2}\sum_{i=1}^{K}\frac{\pi_{i,t}}{q_{i,t}}. \tag{B.17}$$

Furthermore, let τ_r be the largest time instant satisfying $\sum_{t=1}^{\tau_r} Q_t \leq 2^r$. According to the doubling trick, at $\tau_{r-1}+1$, such that $\sum_{t=1}^{\tau_{r-1}+1} Q_t > 2^{r-1}$, the algorithm restarts with

$$\eta_r = \sqrt{\frac{\ln K}{2^r}}. \tag{B.18}$$

Also, the algorithm starts with $r = 0$. Therefore, based on (B.16) and (B.18), it can be concluded that

$$\sum_{t=1}^{\tau_r}\pi_{i,t}\ell_t(v_i) - \min_{v_i\in\mathcal{V}}\sum_{t=1}^{\tau_r}\ell_t(v_i) \leq 2\sqrt{2^r\ln K} - \frac{\ln K}{2^{r+1}}\tau_r \tag{B.19}$$

when $2^{r-1} < \sum_{t=1}^{\tau_r} Q_t \leq 2^r$. The maximum number of restarts required is $\left\lceil \log_2 \sum_{t=1}^{T} Q_t \right\rceil$. Moreover, it can be written that

$$\sum_{r=0}^{\left\lceil \log_2 \sum_{t=1}^{T} Q_t \right\rceil} 2\sqrt{2^r \ln K} < \frac{4\sqrt{\ln K}}{\sqrt{2}-1}\sqrt{\sum_{t=1}^{T} Q_t}. \tag{B.20}$$

Therefore, based on (B.16) and considering the fact that the maximum possible value for incurred loss at each restart is 1, combining (B.19) with (B.20) leads to

$$\begin{aligned}
&\sum_{t=1}^{T}\mathbb{E}_t[\ell_t(v_{I_t})] - \min_{v_i\in\mathcal{V}}\sum_{t=1}^{T}\ell_t(v_i) \\
\leq&\mathcal{O}\left(\sqrt{(\ln K)\sum_{t=1}^{T} Q_t} + \left\lceil \log_2 \sum_{t=1}^{T} Q_t \right\rceil\right) \\
=&\mathcal{O}\left(\sqrt{\ln K \sum_{t=1}^{T}(1 + \frac{1}{2}\sum_{i=1}^{K}\frac{\pi_{i,t}}{q_{i,t}})} + \left\lceil \log_2 \sum_{t=1}^{T} Q_t \right\rceil\right)
\end{aligned} \tag{B.21}$$

Based on (A2), we can write $p_{ij} \geq \epsilon > 0$ if $(i,j) \in \mathcal{E}_t$. According to (1.36) and the fact that the i-th expert is chosen by the learner with probability of $\pi_{i,t}$, based on (A2) the inequality $q_{i,t} \geq \pi_{i,t}\epsilon$ holds. Thus, we have

$$\left\lceil \log_2 \sum_{t=1}^{T} Q_t \right\rceil = \mathcal{O}\left(\ln(\frac{K}{\epsilon}T)\right). \tag{B.22}$$

Combining (B.21) with (B.22) obtains

$$\sum_{t=1}^{T} \mathbb{E}_t[\ell_t(v_{I_t})] - \min_{v_i \in \mathcal{V}} \sum_{t=1}^{T} \ell_t(v_i) \leq \mathcal{O}\left(\sqrt{\ln K \sum_{t=1}^{T}(1 + \frac{1}{2}\sum_{i=1}^{K}\frac{\pi_{i,t}}{q_{i,t}})} + \ln(\frac{K}{\epsilon}T)\right) \tag{B.23}$$

In addition, the following Lemma is used as a step stone [4].

Lemma B.1. *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a directed graph with a set of vertices $\mathcal{V}$ and a set of edges $\mathcal{E}$. Let $\mathcal{D} \subseteq \mathcal{V}$ be a dominating set for $\mathcal{G}$ and $p_1, \ldots, p_K$ be a probability distribution defined over $\mathcal{V}$, such that $p_i \geq \beta > 0$, for $i \in \mathcal{D}$. Then*

$$\sum_{i=1}^{K} \frac{p_i}{\sum_{j:j\to i} p_j} \leq 2\alpha(\mathcal{G}) \ln(1 + \frac{\left\lceil \frac{K^2}{\beta|\mathcal{D}|} \right\rceil + K}{\alpha(\mathcal{G})}) + 2|\mathcal{D}| \tag{B.24}$$

where $\alpha(\mathcal{G})$ represents independence number for the graph $\mathcal{G}$.

Based on Lemma B.1 and (A2), we get

$$\sum_{i=1}^{K} \frac{\pi_{i,t}}{\sum_{\forall j: j \in \mathcal{N}_{i,t}^{\text{in}}} \pi_{j,t}} < 2\alpha(\mathcal{G}_t) \ln(1 + \frac{\left\lceil \frac{K^3}{\eta\epsilon} \right\rceil + K}{\alpha(\mathcal{G}_t)}) + 2|\mathcal{D}_t|. \tag{B.25}$$

Considering the fact that $q_{i,t} \geq \epsilon \sum_{\forall j: j \in \mathcal{N}_{i,t}^{\text{in}}} \pi_{j,t}$ which is induced by (A2), from (B.25), it can be inferred that

$$\sum_{i=1}^{K} \frac{\pi_{i,t}}{q_{i,t}} < \frac{2\alpha(\mathcal{G}_t)}{\epsilon} \ln(1 + \frac{\left\lceil \frac{K^3}{\eta\epsilon} \right\rceil + K}{\alpha(\mathcal{G}_t)}) + \frac{2|\mathcal{D}_t|}{\epsilon}. \tag{B.26}$$

Furthermore, if greedy set cover algorithm by [28] is employed to obtain the dominating set $|\mathcal{D}_t|$, it can be written that [4]

$$|\mathcal{D}_t| = \mathcal{O}(\alpha(\mathcal{G}_t) \ln K). \tag{B.27}$$

Therefore, from (B.26) we can conclude that

$$\sum_{i=1}^{K} \frac{\pi_{i,t}}{q_{i,t}} \leq \mathcal{O}\left(\frac{\alpha(\mathcal{G}_t)}{\epsilon} \ln(\frac{KT}{\epsilon})\right) \tag{B.28}$$

Combining (B.23) with (B.27) and (B.28), we arrive at

$$\sum_{t=1}^{T} \mathbb{E}_t[\ell_t(v_{I_t})] - \min_{v_i \in \mathcal{V}} \sum_{t=1}^{T} \ell_t(v_i) \leq \mathcal{O}\left(\sqrt{\ln K \ln(\frac{K}{\epsilon}T) \sum_{t=1}^{T} \frac{\alpha(\mathcal{G}_t)}{\epsilon}} + \ln(\frac{K}{\epsilon}T)\right) \tag{B.29}$$

which completes the proof of Lemma 1.3.

B.3 Proof of Theorem 2.2

In order to prove Theorem 2.2, let's first consider when $t \leq KM$. When the learner chooses among experts in a deterministic fashion. The (expected) loss can be written as $\mathbb{E}_t[\ell_t(v_i)] = \ell_t(v_k)$. Since $\ell_t(v_i) \leq 1$, we have

$$\sum_{t=1}^{KM} \mathbb{E}_t[\ell_t(v_i)] - \sum_{t=1}^{KM} \ell_t(v_i) \leq (K-1)M. \tag{B.30}$$

On the other hand, for any $t > KM$, the following equality holds

$$\frac{W_{t+1}}{W_t} = \sum_{i=1}^{K} \frac{w_{i,t+1}}{W_t} = \sum_{i=1}^{K} \frac{w_{i,t}}{W_t} \exp\left(-\eta \tilde{\ell}_t(v_i)\right). \tag{B.31}$$

Recall (1.10), we have

$$\frac{w_{i,t}}{W_t} = \frac{\pi_{i,t} - \eta \hat{\tilde{F}}_{i,t}}{1-\eta} \tag{B.32}$$

where $\hat{\tilde{F}}_{i,t} = \frac{\eta}{|\mathcal{D}|}\mathcal{I}(v_i \in \mathcal{D}_t)$. Following similar steps from (B.3) to (B.8), and from (B.31) and (B.32) we obtain

$$\begin{aligned}\sum_{t=t'}^{T}\sum_{i=1}^{K}\pi_{i,t}\tilde{\ell}_t(v_i) - \sum_{t=t'}^{T}\tilde{\ell}_t(v_i) \leq & \frac{\ln K}{\eta} + \sum_{t=t'}^{T}\sum_{i=1}^{K}\eta\hat{\tilde{F}}_{i,t}\tilde{\ell}_t(v_i) \\ & + \sum_{t=t'}^{T}\sum_{i=1}^{K}\frac{\eta}{2}(\pi_{i,t} - \eta\hat{\tilde{F}}_{i,t})\tilde{\ell}_t(v_i)^2\end{aligned} \tag{B.33}$$

where $t' = KM + 1$. In addition, the expected value of $\tilde{\ell}_t(v_i)$ and $\tilde{\ell}_t(v_i)^2$ at time instant t can be written as

$$\mathbb{E}_t[\tilde{\ell}_t(v_i)] = \sum_{\forall j: v_j \in \mathcal{N}_{i,t}^{\text{in}}} \pi_{j,t} p_{ji} \frac{1}{\hat{q}_{i,t}} \ell_t(v_i) = \frac{q_{i,t}}{\hat{q}_{i,t}} \ell_t(v_i) \tag{B.34a}$$

$$\mathbb{E}_t[\tilde{\ell}_t(v_i)^2] = \sum_{\forall j: v_j \in \mathcal{N}_{i,t}^{\text{in}}} \pi_{j,t} p_{ji} \frac{1}{\hat{q}_{i,t}^2} \ell_t(v_i)^2 = \frac{q_{i,t}}{\hat{q}_{i,t}^2} \ell_t(v_i)^2 \leq \frac{q_{i,t}}{\hat{q}_{i,t}^2}. \tag{B.34b}$$

Let $e_{ij,t} := |\hat{p}_{ij,t} - p_{ij}|$. According to (1.11), the probability that $\hat{q}_{i,t} \geq q_{i,t}$ is at least $\prod_{\forall j: v_j \in \mathcal{N}_{i,t}^{\text{in}}} \Pr(e_{ij,t} \leq \xi/\sqrt{M})$ since the incidents $\{e_{ij,t} \leq \xi/\sqrt{M}, \forall (i,j) \in \mathcal{E}\}$ are independent from each other. Let ε denote $\xi/\sqrt{M}$ and $\mu_{i,t} := \frac{1}{\hat{q}_{i,t}} - \frac{1}{q_{i,t}}$, we have

$$\mu_{i,t} = \frac{q_{i,t} - \hat{q}_{i,t}}{\hat{q}_{i,t} q_{i,t}} = \frac{\sum_{\forall j: v_j \in \mathcal{N}_{i,t}^{\text{in}}} \pi_{j,t}(p_{ji} - \hat{p}_{ji,t} - \varepsilon)}{\hat{q}_{i,t} q_{i,t}} \geq -\frac{\sum_{\forall j: v_j \in \mathcal{N}_{i,t}^{\text{in}}} 2\pi_{j,t}\varepsilon}{q_{i,t}^2} \tag{B.35}$$

where the last inequality holds with probability $\prod_{\forall j: v_j \in \mathcal{N}_{i,t}^{\text{in}}} \Pr(e_{ij,t} \leq \varepsilon)$. Therefore, the following inequalities hold with the probability $\prod_{\forall j: v_j \in \mathcal{N}_{i,t}^{\text{in}}} \Pr(e_{ij,t} \leq \varepsilon)$

$$\ell_t(v_i) - \sum_{\forall j: v_j \in \mathcal{N}_{i,t}^{\text{in}}} \frac{2\pi_{j,t}\varepsilon}{q_{i,t}} \ell_t(v_i) \leq \mathbb{E}_t[\tilde{\ell}_t(v_i)] = \ell_t(v_i) + q_{i,t}\mu_{i,t}\ell_t(v_i) \leq \ell_t(v_i) \tag{B.36a}$$

$$\mathbb{E}_t[\tilde{\ell}_t(v_i)^2] \leq \frac{1}{q_{i,t}}. \tag{B.36b}$$

Taking expectation of both sides of (B.33) and combining with (B.36), we obtain the following inequality

$$\begin{aligned} &\sum_{t=t'}^{T}\sum_{i=1}^{K} \pi_{i,t}\ell_t(v_i) - \sum_{t=t'}^{T} \ell_t(v_i) - \sum_{t=t'}^{T}\sum_{i=1}^{K} \pi_{i,t} \sum_{\forall j: v_j \in \mathcal{N}_{i,t}^{\text{in}}} \frac{2\pi_{j,t}\varepsilon}{q_{i,t}} \ell_t(v_i) \\ \leq &\frac{\ln K}{\eta} + \sum_{t=t'}^{T}\sum_{i=1}^{K} \eta \hat{\tilde{F}}_{i,t}\ell_t(v_i) + \sum_{t=t'}\sum_{i=1}^{K} \frac{\eta}{2}(\pi_{i,t} - \eta\hat{\tilde{F}}_{i,t})\frac{1}{q_{i,t}} \\ \leq &\frac{\ln K}{\eta} + \sum_{t=t'}^{T}\sum_{i=1}^{K} \eta \hat{\tilde{F}}_{i,t} + \sum_{t=t'}^{T}\sum_{i=1}^{K} \frac{\eta}{2}(\pi_{i,t} - \eta\hat{\tilde{F}}_{i,t})\frac{1}{q_{i,t}} \end{aligned} \tag{B.37}$$

which holds with probability at least $\prod_{(i,j)\in\mathcal{E}_t} \Pr(e_{ij,t'} \leq \varepsilon, \ldots, e_{ij,T} \leq \varepsilon)$. Applying the chain rule for one term in the product, we have

$$\begin{aligned} \Pr(e_{ij,t'} \leq \varepsilon, \ldots, e_{ij,T} \leq \varepsilon) &= \Pr(e_{ij,t'} \leq \varepsilon) \prod_{t=t'+1}^{T} \Pr(e_{ij,t} \leq \varepsilon \mid e_{ij,t-1} \leq \varepsilon, \ldots, e_{ij,t'} \leq \varepsilon) \\ &\geq \prod_{t=t'}^{T} \Pr(e_{ij,t} \leq \varepsilon). \end{aligned} \tag{B.38}$$

In order to obtain the lower bound of the probability $\Pr(e_{ij,t} \leq \varepsilon_{ij,t})$, the Bernstein inequality is employed. To this end consider the following lemma [157].

Lemma B.2. *Let $\zeta_1 \ldots \zeta_n$ be independent random variables with*

$$\mathbb{E}[\zeta_i] = 0, \forall i : 1 \leq i \leq n \tag{B.39a}$$

$$|\mathbb{E}[\zeta_i^m]| \leq \frac{m!}{2} b_i^2 H^{m-2}, m = 2, 3, \ldots, \forall i : 1 \leq i \leq n. \tag{B.39b}$$

Then for $x \geq 0$, we have

$$\Pr(|\zeta_1 + \ldots + \zeta_n| \geq xB_n) \leq 2\exp(-\frac{\frac{x^2}{2}}{1 + \frac{xH}{B_n}}) \tag{B.40}$$

where $B_n^2 = b_1^2 + \ldots + b_n^2$.

Let $\theta_{ij}(t) := X_{ij}(t) - p_{ij}$, $\forall (i,j) \in \mathcal{E}_t$. Since $X_{ij}(t)$ follows Bernoulli distribution with the parameter p_{ij}, it can be readily obtained that $\mathbb{E}[\theta_{ij}(t)] = 0$. Furthermore, for the moment generating function of $\theta_{ij}(t)$, we have

$$M_{\theta_{ij}(t)}(z) = (1 - p_{ij})e^{-p_{ij}z} + p_{ij}e^{(1-p_{ij})z}. \tag{B.41}$$

Therefore, the expected value of $\theta_{ij}^m(t)$, $m = 2, 3, \ldots$ satisfies

$$\mathbb{E}[\theta_{ij}^m(t)] = \frac{d^m M_{\theta_{ij}(t)}(z)}{dz^m} \mid_{z=0} = (-p_{ij})^m(1 - p_{ij}) + (1 - p_{ij})^m p_{ij}. \tag{B.42}$$

From (B.42), we can conclude that

$$|\mathbb{E}[\theta_{ij}^m(t)]| \leq p_{ij}(1 - p_{ij}) \leq \frac{1}{4} \leq \frac{m!}{8} = \frac{m!}{2}\left(\frac{1}{2}\right)^2 \times 1^{m-2}, m = 2, 3, \ldots \tag{B.43}$$

Thus, letting $b_i = \frac{1}{2}$, $H = 1$ in Lemma B.2 and combining with (B.43), the following inequality can be obtained

$$\begin{aligned}
&\Pr\left(|\sum_{\tau \in \mathcal{T}_{ij,t}} \theta_{ij}(\tau)| \geq \frac{\xi C_{ij,t}}{\sqrt{M}}\right) \\
=&\Pr\left(|\sum_{\tau \in \mathcal{T}_{ij,t}} X_{ij}(\tau) - p_{ij}| \geq \frac{\xi C_{ij,t}}{\sqrt{M}}\right) \\
\leq&2\exp\left(-\frac{2\xi^2 \frac{C_{ij,t}}{M}}{1 + \frac{4\xi}{\sqrt{M}}}\right) = 2\exp\left(-\frac{2\xi^2 C_{ij,t}}{M + 4\xi\sqrt{M}}\right)
\end{aligned} \tag{B.44}$$

which leads to

$$\Pr(e_{ij,t} \geq \varepsilon) \leq 2\exp\left(-\frac{2\xi^2 C_{ij,t}}{M + 4\xi\sqrt{M}}\right) \tag{B.45}$$

Therefore, (B.37) holds with probability at least

$$\delta_\xi = \prod_{t=t'}^{T} \prod_{(i,j)\in\mathcal{E}_t} \left(1 - 2\exp(-\frac{2\xi^2 C_{ij,t}}{M + 4\xi\sqrt{M}})\right). \tag{B.46}$$

Since $\sum_{\forall j: v_j \in \mathcal{N}_{i,t}^{\text{in}}} \pi_{j,t} \leq 1$ and $\varepsilon = \frac{\xi}{\sqrt{M}}$, the following inequality holds

$$\sum_{t=t'}^{T}\sum_{i=1}^{K} \pi_{i,t} \sum_{\forall j: v_j \in \mathcal{N}_{i,t}^{\text{in}}} \frac{2\pi_{j,t}\varepsilon}{q_{i,t}} = \sum_{t=t'}^{T}\sum_{i=1}^{K} \pi_{i,t} \sum_{\forall j: v_j \in \mathcal{N}_{i,t}^{\text{in}}} \frac{2\pi_{j,t}\frac{\xi}{\sqrt{M}}}{q_{i,t}} \leq \sum_{t=t'}^{T}\sum_{i=1}^{K} \frac{2\pi_{i,t}\xi}{q_{i,t}\sqrt{M}}. \tag{B.47}$$

Using (B.47) and the fact that $\frac{1}{q_{i,t}} \geq 1$, (B.37) can be rewritten as

$$\sum_{t=t'}^{T}\sum_{i=1}^{K} \pi_{i,t}\ell_t(v_i) - \sum_{t=t'}^{T} \ell_t(v_i) \leq \frac{\ln K}{\eta} + \sum_{t=t'}^{T} \eta(1 - \frac{\eta}{2}) + \sum_{t=t'}^{T}\sum_{i=1}^{K} \frac{\pi_{i,t}}{q_{i,t}}(\frac{2\xi}{\sqrt{M}} + \frac{\eta}{2}) \tag{B.48}$$

Combining (B.48) with (B.30) results in following inequality

$$\begin{aligned} &\sum_{t=1}^{T} \mathbb{E}_t[\ell_t(v_{I_t})] - \sum_{t=1}^{T} \ell_t(v_i) \\ \leq &\frac{\ln K}{\eta} + (K-1)M + \eta(1 - \frac{\eta}{2})(T - KM) + \sum_{t=t'}^{T}\sum_{i=1}^{K} \frac{\pi_{i,t}}{q_{i,t}}(\frac{2\xi}{\sqrt{M}} + \frac{\eta}{2}) \end{aligned} \tag{B.49}$$

which holds with probability at least δ_ξ and the proof of Theorem 2.2 is completed.

B.4 Proof of Corollary 2.1

The proof of Corollary 2.1 will be built upon the following Lemma.

Lemma B.3. *Let $\zeta_1, \ldots, \zeta_N$ $(N > 1)$ be a sequence of real positive numbers such that $\forall i : 1 \leq i \leq N$, $0 < \zeta_i < 1$ and $\forall n : 1 \leq n \leq N$, $\sum_{i=1}^{n} \zeta_i < 1$. Then, it can be written that*

$$\prod_{i=1}^{N}(1-\zeta_i) > 1 - \sum_{i=1}^{N} \zeta_i \tag{B.50}$$

Proof. We prove this Lemma using mathematical induction. Firstly, Consider (B.50) for $N = 2$

$$(1-\zeta_1)(1-\zeta_2) = 1 - \zeta_1 - \zeta_2 + \zeta_1\zeta_2 > 1 - \zeta_1 - \zeta_2. \tag{B.51}$$

Assuming that (B.50) holds for $N = n$. Then, based on (B.51) we have for $N = n + 1$

$$\prod_{i=1}^{n+1}(1-\zeta_i) = \left(\prod_{i=1}^{n}(1-\zeta_i)\right) \times (1-\zeta_{n+1}) > (1 - \sum_{i=1}^{n}\zeta_i)(1-\zeta_{n+1}) > 1 - \sum_{i=1}^{n+1}\zeta_i. \tag{B.52}$$

Hence, (B.50) also holds for $N = n + 1$, and Lemma A.4 is proved by induction. □

Assuming M satisfies

$$M \geq \left(\frac{4\xi \ln(KT)}{\xi^2 - \ln(KT)}\right)^2. \tag{B.53}$$

Hence, (B.53) can be re-written as

$$\frac{1}{K^2T^2} \geq \exp(-\frac{2\xi^2\sqrt{M}}{\sqrt{M} + 4\xi}) \geq \exp(-\frac{2\xi^2 C_{ij,t}}{M + 4\xi\sqrt{M}}) \tag{B.54}$$

where the second inequality holds since $C_{ij,t} \geq M$. Let $t' = KM + 1$. Note that the regret bound in (1.47) holds with probability at least δ_ξ in (B.46). According to Lemma A.4, we can obtain the following inequality

$$\begin{aligned}\delta_\xi =& \prod_{t=t'}^{T} \prod_{(i,j)\in\mathcal{E}_t} \left(1 - 2\exp(-\frac{2\xi^2 C_{ij,t}}{M + 4\xi\sqrt{M}})\right) \\ >& 1 - \sum_{(i,j)\in\mathcal{E}_t} \sum_{t=t'}^{T} 2\exp(-\frac{2\xi^2 C_{ij,t}}{M + 4\xi\sqrt{M}}).\end{aligned} \tag{B.55}$$

Combining (B.54) with (B.55) obtains

$$\delta_\xi \geq 1 - \frac{2(T - KM)|\mathcal{E}|}{K^2T^2} \tag{B.56}$$

where $|\mathcal{E}|$ denotes the cardinality of the $\mathcal{E}$. Since $\mathcal{G}$ does not change over time, $|\mathcal{E}|$ is a constant. According to (B.56), it can be readily obtained that when (B.53) holds, the regret bound in (1.47) holds with probability at least of order $1 - \mathcal{O}(\frac{1}{T})$. Consider the case where the learner sets η, M and ξ as follows

$$\eta = \mathcal{O}(\sqrt{\frac{\ln K}{T}}) \tag{B.57a}$$

$$M = \mathcal{O}(\frac{1}{\sqrt{K}} T^{\frac{2}{3}}) \tag{B.57b}$$

$$\xi = \mathcal{O}(K^{\frac{1}{4}} \sqrt{\ln(KT)}). \tag{B.57c}$$

Putting η, M and ξ in (B.57) into (1.47) and based on Lemma A.5, it can be concluded that the expected regret of Exp3-UP satisfies

$$\sum_{t=1}^{T} \mathbb{E}_t[\ell_t(v_{I_t})] - \sum_{t=1}^{T} \ell_t(v_i) \leq \mathcal{O}\left(\frac{\alpha(\mathcal{G})}{\epsilon} \ln(KT)(\sqrt{T\ln K} + \sqrt{K\ln(KT)}T^{\frac{2}{3}})\right) \tag{B.58}$$

with probability at least $1 - \mathcal{O}(\frac{1}{T})$.

B.5 Proof of Lemma 2.2

In this section, doubling trick technique is employed such that Exp3-UP can achieve sub-linear regret. If $2^b < t \leq 2^{b+1}$, the value of the learning rate η_b, M_b and ξ_b are

$$\eta_b = \sqrt{\frac{\ln K}{2^{b+1}}} \tag{B.59a}$$

$$M_b = \left\lceil 2^{\frac{2(b+1)}{3}} \frac{1}{\sqrt{K}} + \ln 4K \right\rceil \tag{B.59b}$$

$$\xi_b = \left(2K^{\frac{1}{4}} + \sqrt{4\sqrt{K}+1}\right)\sqrt{\ln(K2^{b+3})} \tag{B.59c}$$

When the learner realizes that $t > 2^{b+1}$, the algorithm restarts with η_{b+1}, M_{b+1} and ξ_{b+1}. The algorithm starts with $b = \lceil \log_2 K \rceil$. Therefore, when $t < 2^{\lceil \log_2 K \rceil}$, the value of η_b, M_b and ξ_b are set with respect to $b = \lceil \log_2 K \rceil$. Let $\mathcal{M}_i$ denotes a set which includes the time instants when the learner chooses the i-th expert in a deterministic fashion for exploration. Specifically, when at time instant τ, the learner chooses the i-th expert for exploration without using the PMF in (1.10), the time instant τ is appended to $\mathcal{M}_i$. At each restart the learner chooses the experts one by one for the exploration until the condition $|\mathcal{M}_i| \geq M_b$, $\forall i \in [K]$ is satisfied. Then, the learner chooses among experts according to PMF in (2.11) using the learning rate η_b. Therefore, based on Theorem 2.2, for each b, the algorithm satisfies

$$\begin{aligned}
&\sum_{t=2^b+1}^{T_b} \mathbb{E}_t[\ell_t(v_{I_t})] - \sum_{t=2^b+1}^{T_b} \ell_t(v_i) \\
\leq &\frac{\ln K}{\eta_b} + (K-1)(M_b - M_{b-1}) + \eta_b(1 - \frac{\eta_b}{2})(T_b - 2^b - K(M_b - M_{b-1})) \\
&+ \sum_{t=2^b+1}^{T_b} \sum_{i=1}^{K} \frac{\pi_{i,t}}{q_{i,t}} (\frac{2\xi_b}{\sqrt{M_b}} + \frac{\eta_b}{2})
\end{aligned} \tag{B.60}$$

with probability at least δ_b where it can be expressed as

$$\delta_b = \prod_{t=t'_b}^{T_b} \prod_{(i,j)\in\mathcal{E}_t} \left(1 - 2\exp(-\frac{2\xi_b^2 C_{ij,t}}{M_b + 4\xi_b\sqrt{M_b}})\right) \tag{B.61}$$

where T_b denote the greatest time instant which satisfies $2^b < T_b \leq 2^{b+1}$ and t'_b can be written as

$$t'_b = \min(T_{b-1} + K(M_b - M_{b-1}) + 1, T_b). \tag{B.62}$$

Note that $M_{b-1} = 0$ when $b = \lceil \log_2 K \rceil$. Since for each b, M_b and ξ_b in (B.59) meet the following condition

$$M_b \geq \left(\frac{4\xi_b \ln(4KT_b)}{\xi_b^2 - \ln(4KT_b)}\right)^2, \tag{B.63}$$

it can be concluded that the following inequality holds true

$$\frac{1}{16K^2T_b^2} \geq \exp(-\frac{2\xi_b^2\sqrt{M_b}}{\sqrt{M_b} + 4\xi_b}) \geq \exp(-\frac{2\xi_b^2 C_{ij,t}}{M_b + 4\xi_b\sqrt{M_b}}), \tag{B.64}$$

and as a result according to Lemma A.4 we can write

$$\delta_b > 1 - \sum_{(i,j)\in\mathcal{E}_t} \sum_{t=t'_b}^{T_b} 2\exp(-\frac{2\xi_b^2 C_{ij,t}}{M_b + 4\xi_b\sqrt{M_b}}). \tag{B.65}$$

Combining (B.64) with (B.65), it can be concluded that

$$\delta_b > 1 - \max(0, \frac{(T_b - t'_b)|\mathcal{E}_{T_b}|}{8K^2T_b^2}). \tag{B.66}$$

Therefore, for each b from (B.60), (B.66) and Lemma A.5 it can be inferred that

$$\sum_{t=2^b+1}^{T_b} \mathbb{E}_t[\ell_t(v_{I_t})] - \sum_{t=2^b+1}^{T_b} \ell_t(v_i) \leq \mathcal{O}(\omega) \tag{B.67}$$

where $\omega := \frac{\alpha(\mathcal{G})}{\epsilon}\ln(KT_b)(\sqrt{T_b \ln K} + \sqrt{K\ln(KT_b)}T_b^{\frac{2}{3}})$ holds with probability at least $1-\mathcal{O}(\frac{1}{T_b})$. Summing (B.67) over all possible values of b, from $b := \lceil \log_2 K \rceil$ to $\lceil \log_2 T \rceil$ and taking into account that the maximum value of the loss at each restart is 1, we arrive at

$$\begin{aligned}\sum_{t=1}^{T} \mathbb{E}_t[\ell_t(v_{I_t})] - \sum_{t=1}^{T} \ell_t(v_i) &\leq \sum_{b=\lceil \log_2 K \rceil}^{\lceil \log_2 T \rceil} \mathcal{O}\left(\omega\right) + \lceil \log_2 T \rceil - \lceil \log_2 K \rceil \\ &\leq \mathcal{O}\left(\omega \ln T\right) + \mathcal{O}(\ln T)\end{aligned} \tag{B.68}$$

which holds with probability at least

$$\Delta = \prod_{b=\lceil \log_2 K \rceil}^{\lceil \log_2 T \rceil} \left(1 - \max(0, \frac{(T_b - t'_b)|\mathcal{E}_{T_b}|}{8K^2T_b^2})\right). \tag{B.69}$$

When $b = \lceil \log_2 K \rceil$, we have $T_b \geq 2K$. Furthermore, when $\lceil \log_2 K \rceil < b \leq \lfloor \log_2 T \rfloor$, it can be concluded that $T_b = 2T_{b-1}$. Therefore, we can write

$$\begin{aligned}&\sum_{b=\lceil \log_2 K \rceil}^{\lfloor \log_2 T \rfloor} \max(0, \frac{(T_b - t'_b)|\mathcal{E}_{T_b}|}{8K^2T_b^2}) \\ <&\sum_{b=\lceil \log_2 K \rceil}^{\lfloor \log_2 T \rfloor} \frac{1}{8T_b} \leq \frac{1}{8K}(\sum_{b=\lceil \log_2 K \rceil}^{\lfloor \log_2 T \rfloor} (\frac{1}{2})^{b-\lceil \log_2 K \rceil}) \\ =&\frac{1}{8K}(2 - (\frac{1}{2})^{\lfloor \log_2 T \rfloor - \lceil \log_2 K \rceil}).\end{aligned} \tag{B.70}$$

Based on (B.70) and under the assumption that $T > K$, we find

$$\sum_{b=\lceil \log_2 K \rceil}^{\lceil \log_2 T \rceil} \max(0, \frac{(T_b - t'_b)|\mathcal{E}_{T_b}|}{8K^2T_b^2}) < \frac{1}{8K}(2 - (\frac{1}{2})^{\lfloor \log_2 T \rfloor - \lceil \log_2 K \rceil}) + \frac{1}{8T} < \frac{3}{8K}. \tag{B.71}$$

Thus, Δ meet the conditions in the Lemma A.4 and it can be inferred that

$$\Delta > 1 - \sum_{b=\lceil \log_2 K \rceil}^{\lceil \log_2 T \rceil} \max(0, \frac{(T_b - t'_b)|\mathcal{E}_{T_b}|}{8K^2 T_b^2}) \geq 1 - \mathcal{O}(\frac{1}{K}).$$

Therefore, in this case, Exp3-UP satisfies

$$\begin{aligned} &\sum_{t=1}^{T} \mathbb{E}_t[\ell_t(v_{I_t})] - \sum_{t=1}^{T} \ell_t(v_i) \\ \leq &\mathcal{O}\left(\frac{\alpha(\mathcal{G})}{\epsilon} \ln(T)\ln(KT)(\sqrt{T \ln K} + \sqrt{K \ln(KT)} T^{\frac{2}{3}})\right) \end{aligned} \tag{B.72}$$

with probability at least $1 - \mathcal{O}(\frac{1}{K})$. This completes the proof of Lemma 1.2.

B.6 Proof of Theorem 2.3

Since Exp3-GR chooses the experts one by one for the exploration at the first KM time instants, $\mathbb{E}_t[\ell_t(v_i)] = \ell_t(v_k)$ and (B.30) hold true. In addition, for $t > KM$ we have

$$\frac{W_{t+1}}{W_t} = \sum_{i=1}^{K} \frac{w_{i,t+1}}{W_t} = \sum_{i=1}^{K} \frac{w_{i,t}}{W_t} \exp\left(-\eta \tilde{\ell}_t(v_i)\right). \tag{B.73}$$

According to (1.28), $\frac{w_{i,t}}{W_t}$ can be expressed as

$$\frac{w_{i,t}}{W_t} = \frac{\pi_{i,t} - \frac{\eta}{|\mathcal{D}|}\mathcal{I}(v_i \in \mathcal{D})}{1 - \eta}. \tag{B.74}$$

Following similar steps performed to obtain (B.9) from (B.1) and (B.2), given (B.73) and (B.74) we get

$$\begin{aligned}&\sum_{t=t'}^{T}\sum_{i=1}^{K}\pi_{i,t}\tilde{\ell}_t(v_i)-\sum_{t=t'}^{T}\tilde{\ell}_t(v_i)\\ \leq&\frac{\ln K}{\eta}+\sum_{t=t'}^{T}\sum_{i\in\mathcal{D}}\frac{\eta}{|\mathcal{D}|}\tilde{\ell}_t(v_i)+\sum_{t=t'}^{T}\sum_{i=1}^{K}\frac{\eta}{2}(\pi_{i,t}-\frac{\eta}{|\mathcal{D}|}\mathcal{I}(v_i\in\mathcal{D}))\tilde{\ell}_t(v_i)^2\end{aligned}\tag{B.75}$$

where $t'=KM+1$. According to (1.1), expected value of loss estimate $\tilde{\ell}_t(v_i)$ can be expressed as

$$\mathbb{E}_t[\tilde{\ell}_t(v_i)]=\sum_{\forall j:v_j\in\mathcal{N}_{i,t}^{\text{in}}}\pi_{j,t}p_{ji}\mathbb{E}_t[Q_{i,t}]\ell_t(v_i)=q_{i,t}\mathbb{E}_t[Q_{i,t}]\ell_t(v_i)\tag{B.76a}$$

$$\mathbb{E}_t[\tilde{\ell}_t(v_i)^2]=\sum_{\forall j:v_j\in\mathcal{N}_{i,t}^{\text{in}}}\pi_{j,t}p_{ji}\mathbb{E}_t[Q_{i,t}^2]\ell_t(v_i)^2=q_{i,t}\mathbb{E}_t[Q_{i,t}^2]\ell_t(v_i)^2.\tag{B.76b}$$

Note that the expected values depend on random variable $\{Z_{i,u}(t)\}_{u=1}^{M}$ in (1.2), where $P_{i,u}(t)$ and $Y_{ij,u}(t)$, $\forall i\in[K]$, $\forall(i,j)\in\mathcal{E}_t$ are independent Bernoulli random variables with parameters $\pi_{i,t}$ and p_{ij}, respectively. Therefore, $\{Z_{i,u}(t)\}_{u=1}^{M}$ are also Bernoulli random variables with expected value

$$\begin{aligned}\mathbb{E}_t[Z_{i,u}(t)]&=\mathbb{E}_t\left[\sum_{\forall j:v_j\in\mathcal{N}_{i,t}^{\text{in}}}P_{j,u}(t)Y_{ji,u}(t)\right]=\sum_{\forall j:v_j\in\mathcal{N}_{i,t}^{\text{in}}}\mathbb{E}_t[P_{j,u}(t)]\mathbb{E}_t[Y_{ji,u}(t)]\\&=\sum_{\forall j:v_j\in\mathcal{N}_{i,t}^{\text{in}}}\pi_{j,t}p_{ji}=q_{i,t}.\end{aligned}\tag{B.77}$$

In other words, $Z_{i,u}(t)$ is a Bernoulli random variable whose value is 1 with probability $q_{i,t}$. The expected value of $Q_{i,t}$ and $Q_{i,t}^2$ can henceforth be written as

$$\begin{aligned}
\mathbb{E}_t[Q_{i,t}] &= \sum_{u=1}^{M} u q_{i,t}(1-q_{i,t})^{u-1} + M(1-q_{i,t})^M \\
&= \frac{1-(Mq_{i,t}+1)(1-q_{i,t})^M}{q_{i,t}} + M(1-q_{i,t})^M = \frac{1-(1-q_{i,t})^M}{q_{i,t}} \qquad \text{(B.78a)}\\
\mathbb{E}_t[Q_{i,t}^2] &= \sum_{u=1}^{M} u^2 q_{i,t}(1-q_{i,t})^{u-1} + M^2(1-q_{i,t})^M \\
&= \frac{2-2(1-q_{i,t}^{M+2})}{q_{i,t}^2} - \frac{1+(2M+3)(1-q_{i,t})^{M+1}}{q_{i,t}} \\
&\quad - (M+1)^2(1-q_{i,t})^M + M^2(1-q_{i,t})^M \\
&= \frac{2-2(1-q_{i,t}^{M+2})}{q_{i,t}^2} - \frac{1+(2M+3)(1-q_{i,t})^{M+1}}{q_{i,t}} \\
&\quad - (2M+1)(1-q_{i,t})^M. \qquad \text{(B.78b)}
\end{aligned}$$

Combining (B.76) with (B.78), we arrive at

$$\begin{aligned}
\mathbb{E}_t[\tilde{\ell}_t(v_i)] =& q_{i,t}\frac{1-(1-q_{i,t})^M}{q_{i,t}}\ell_t(v_i) = \left(1-(1-q_{i,t})^M\right)\ell_t(v_i) \le \ell_t(v_i) \qquad \text{(B.79a)}\\
\mathbb{E}_t[\tilde{\ell}_t(v_i)^2] =& \left(\frac{2-2(1-q_{i,t}^{M+2})}{q_{i,t}} - 1\right)\ell_t(v_i)^2 + (2M+3)(1-q_{i,t})^{M+1}\ell_t(v_i)^2 \\
& - q_{i,t}(2M+1)(1-q_{i,t})^M\ell_t(v_i)^2 \le \frac{2}{q_{i,t}}. \qquad \text{(B.79b)}
\end{aligned}$$

Combining (B.75) and (B.79), it can be concluded that

$$\begin{aligned}
&\sum_{t=t'}^{T}\sum_{i=1}^{K}\pi_{i,t}\ell_t(v_i) - \sum_{t=t'}^{T}\ell_t(v_i) - \sum_{t=t'}^{T}\sum_{i=1}^{K}\pi_{i,t}(1-q_{i,t})^M\ell_t(v_i) \\
\le& \frac{\ln K}{\eta} + \sum_{t=t'}^{T}\sum_{i=1}^{K}\frac{\eta}{2}(\pi_{i,t} - \frac{\eta}{|\mathcal{D}|}\mathcal{I}(v_i\in\mathcal{D}))\frac{2}{q_{i,t}} + \sum_{t=t'}^{T}\sum_{i\in\mathcal{D}}\frac{\eta}{|\mathcal{D}|}\ell_t(v_i). \qquad \text{(B.80)}
\end{aligned}$$

According to (A1) $\ell_t(v_i) \leq 1$ and using the fact that $\frac{2}{q_{i,t}} \geq 2$, (B.80) can be further bounded by

$$\begin{aligned}
&\sum_{t=t'}^{T}\sum_{i=1}^{K}\pi_{i,t}\ell_t(v_i) - \sum_{t=t'}^{T}\ell_t(v_i) \qquad \text{(B.81)}\\
\leq&\frac{\ln K}{\eta} + \sum_{t=t'}^{T}(1-q_{i,t})^M + \sum_{t=t'}^{T}\sum_{i\in\mathcal{D}}\frac{\eta-\eta^2}{|\mathcal{D}|} + \sum_{t=t'}^{T}\sum_{i=1}^{K}\eta\frac{\pi_{i,t}}{q_{i,t}}\\
=&\frac{\ln K}{\eta}+\sum_{t=t'}^{T}(1-q_{i,t})^M+\eta(1-\eta)(T-KM)+\eta\sum_{t=t'}^{T}\sum_{i=1}^{K}\frac{\pi_{i,t}}{q_{i,t}}.
\end{aligned}$$

Note that when $t > t'$, we have $\mathbb{E}_t[\ell_t(v_{I_t})] = \sum_{i=1}^{K}\pi_{i,t}\ell_t(v_i)$. Combining (B.30) with (B.81) leads to

$$\begin{aligned}
&\sum_{t=1}^{T}\mathbb{E}_t[\ell_t(v_{I_t})] - \sum_{t=1}^{T}\ell_t(v_i)\\
\leq&\frac{\ln K}{\eta} + (K-1)M + \sum_{t=t'}^{T}(1-q_{i,t})^M + \eta(1-\eta)(T-KM) + \eta\sum_{t=t'}^{T}\sum_{i=1}^{K}\frac{\pi_{i,t}}{q_{i,t}} \qquad \text{(B.82)}
\end{aligned}$$

which completes the proof of Theorem 2.3.

B.7 Proof of Lemma 2.3

In this section, the doubling trick is employed to choose η and M when the learner does not know the time horizon T beforehand. At time instant t, when $2^b < t \leq 2^{b+1}$, η_b and M_b the are chosen as

$$\eta_b = \sqrt{\frac{K\ln K}{2^{b+1}}}, M_b = \left\lceil\frac{(b+1)\sqrt{2^{b-1}}|\mathcal{D}|\ln 2}{\epsilon\sqrt{K\ln K}}\right\rceil. \qquad \text{(B.83)}$$

When $t > 2^{b+1}$ holds true, the algorithm restarts with η_{b+1} and M_{b+1}. The algorithm starts with $b = 0$. At each restart, the algorithm chooses the experts one by one for exploration until the condition that each expert is chosen at least M_b times is met. Then, the learner uses the last M_b observed samples from each expert to perform geometric resampling. In this case, for each b, Exp3-GR satisfies

$$
\begin{aligned}
&\sum_{t=2^b+1}^{T_b} \mathbb{E}_t[\ell_t(v_{I_t})] - \min_{v_i \in \mathcal{V}} \sum_{t=2^b+1}^{T_b} \ell_t(v_i) \\
\leq & \frac{\ln K}{\eta_b} + (K-1)(M_b - M_{b-1}) + \sum_{t=t'_b}^{T_b} (1 - q_{i,t})^{M_b} \\
&+ \eta_b(1-\eta_b)(T_b - 2^b - K(M_b - M_{b-1})) + \eta_b \sum_{t=t'_b}^{T_b} \sum_{i=1}^{K} \frac{\pi_{i,t}}{q_{i,t}}
\end{aligned}
\tag{B.84}
$$

where T_b denote the greatest time instant which satisfies $2^b < T_b \leq 2^{b+1}$ and t'_b can be expressed as in (B.62). Note that when $b = 0$, we have $M_{b-1} = 0$. Taking into account that the maximum loss at each restart is 1, summing (B.84) over all possible values for b obtains

$$
\begin{aligned}
&\sum_{t=1}^{T} \mathbb{E}_t[\ell_t(v_{I_t})] - \min_{v_i \in \mathcal{V}} \sum_{t=1}^{T} \ell_t(v_i) \\
\leq & \lceil \log_2 T \rceil + \sum_{b=0}^{\lfloor \log_2 T \rfloor} \frac{\ln K}{\eta_b} + (K-1)M \\
&+ \sum_{b=0}^{\lfloor \log_2 T \rfloor} \eta_b(1-\eta_b)(T_b - 2^b - K(M_b - M_{b-1})) \\
&+ \sum_{b=0}^{\lfloor \log_2 T \rfloor} \eta_b \sum_{t=t'_b}^{T_b} \sum_{i=1}^{K} \frac{\pi_{i,t}}{q_{i,t}} + \sum_{b=0}^{\lfloor \log_2 T \rfloor} \sum_{t=t'_b}^{T_b} (1 - q_{i,t})^{M_b}
\end{aligned}
\tag{B.85}
$$

where M is the number of samples for each expert when $b = \lfloor \log_2 T \rfloor$ which are used for geometric resampling. According to (B.83) and the fact that $\mathcal{D}$ is obtained using the greedy set cover algorithm, we have

$$M = \mathcal{O}\left(\frac{\alpha(\mathcal{G})}{\epsilon\sqrt{K}} \ln T\sqrt{T \ln K}\right). \tag{B.86}$$

Furthermore, for each b, the inequality $q_{i,t} > \frac{\eta_b \epsilon}{|\mathcal{D}|}$ holds. Therefore, according to (B.83), we can write $M_b q_{i,t} > \frac{b+1}{2}\ln 2$. Thus, it can be concluded that $(1-q_{i,t})^{M_b} \leq e^{-M_b q_{i,t}} < \frac{1}{\sqrt{2^{b+1}}}$ from which we obtain

$$\sum_{b=0}^{\lfloor \log_2 T\rfloor} \sum_{t=t'_b}^{T_b} (1-q_{i,t})^{M_b} < \sum_{b=0}^{\lfloor \log_2 T\rfloor} \frac{T_b - 2^b}{\sqrt{2^{b+1}}} \leq \sum_{b=0}^{\lfloor \log_2 T\rfloor} \sqrt{2^{b-1}} \leq \frac{\sqrt{2T}-1}{2-\sqrt{2}}. \tag{B.87}$$

In addition, based on the Lemma A.5, it can be written that

$$\begin{aligned}
&\sum_{b=0}^{\lfloor \log_2 T\rfloor} \eta_b \sum_{t=t'_b}^{T_b} \sum_{i=1}^{K} \frac{\pi_{i,t}}{q_{i,t}} \\
\leq &\sum_{b=0}^{\lfloor \log_2 T\rfloor} \sqrt{\frac{K\ln K}{2^{b+1}}}(T_b - 2^b)\mathcal{O}\left(\frac{\alpha(\mathcal{G})}{\epsilon}\ln(KT)\right) \\
\leq &\lceil \log_2 T\rceil \sqrt{2^{b-1}K\ln K}\mathcal{O}\left(\frac{\alpha(\mathcal{G})}{\epsilon}\ln(KT)\right) \\
= &\mathcal{O}\left(\frac{\alpha(\mathcal{G})}{\epsilon}(\ln T)\ln(KT)\sqrt{KT\ln K}\right).
\end{aligned} \tag{B.88}$$

Therefore, combining (B.85) with (B.86), (B.87) and (B.88), it can be inferred that Exp3-GR satisfies

$$\sum_{t=1}^{T} \mathbb{E}_t[\ell_t(v_{I_t})] - \min_{v_i \in \mathcal{V}} \sum_{t=1}^{T} \ell_t(v_i) \leq \mathcal{O}\left(\frac{\alpha(\mathcal{G})\ln T}{\epsilon}\ln(KT)\sqrt{KT\ln K}\right) \tag{B.89}$$

which completes the proof of Lemma 1.3.

Appendix C

Supplementary Proofs and Experiments for Chapter 3

C.1 Proof of Theorem 3.1

Since $W_{k,t} = \sum_{i=1}^{N} w_{ik,t}$ and according to (3.7), we can write

$$\frac{W_{k,t+1}}{W_{k,t}} = \sum_{i=1}^{N} \frac{w_{k,t+1}}{W_{k,t}} = \sum_{i=1}^{N} \frac{w_{ik,t}}{W_{k,t}} \exp\Big(-\eta_k \mathcal{L}(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t};\boldsymbol{\theta}_{i,t}), y_{k,t})\Big). \tag{C.1}$$

Using the inequality $e^{-x} \leq 1 - x + \frac{1}{2}x^2, \forall x \geq 0$, the upper bound of (C.1) can be obtained as

$$\frac{W_{k,t+1}}{W_{k,t}} \leq \sum_{i=1}^{N} \frac{w_{ik,t}}{W_{k,t}} \Big(1 - \eta_k \mathcal{L}(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t};\boldsymbol{\theta}_{i,t}), y_{k,t}) + \frac{\eta_k^2}{2}\mathcal{L}^2(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t};\boldsymbol{\theta}_{i,t}), y_{k,t})\Big). \tag{C.2}$$

Using the inequality $1 + x \leq e^x$ and taking the logarithm from both sides of (C.2), we get

$$\ln \frac{W_{k,t+1}}{W_{k,t}} \leq \sum_{i=1}^{N} \frac{w_{ik,t}}{W_{k,t}} \Big(-\eta_k \mathcal{L}(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t};\boldsymbol{\theta}_{i,t}), y_{k,t}) + \frac{\eta_k^2}{2}\mathcal{L}^2(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t};\boldsymbol{\theta}_{i,t}), y_{k,t})\Big). \tag{C.3}$$

According to (A2), $\mathcal{L}^2(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t};\boldsymbol{\theta}_{i,t}),y_{k,t}) \leq 1$. Therefore, from (C.3), we can conclude that

$$\ln\frac{W_{k,t+1}}{W_{k,t}} \leq \sum_{i=1}^{N}\frac{w_{ik,t}}{W_{k,t}}\left(\frac{\eta_k^2}{2} - \eta_k\mathcal{L}(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t};\boldsymbol{\theta}_{i,t}),y_{k,t})\right). \tag{C.4}$$

Summing (C.4) over time, we arrive at

$$\ln\frac{W_{k,T+1}}{W_{k,1}} \leq \sum_{t=1}^{T}\sum_{i=1}^{N}\frac{w_{ik,t}}{W_{k,t}}\left(\frac{\eta_k^2}{2} - \eta_k\mathcal{L}(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t};\boldsymbol{\theta}_{i,t}),y_{k,t})\right). \tag{C.5}$$

Moreover, for any $i \in [N]$, $\ln\frac{W_{k,T+1}}{W_{k,1}}$ can be lower bounded as

$$\ln\frac{W_{k,T+1}}{W_{k,1}} \geq \ln\frac{w_{ik,T+1}}{W_{k,1}} = -\eta_k\sum_{t=1}^{T}\mathcal{L}(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t};\boldsymbol{\theta}_{i,t}),y_{k,t}) - \ln N. \tag{C.6}$$

Combining (C.5) with (C.6), we obtain

$$\sum_{t=1}^{T}\sum_{i=1}^{N}\frac{w_{ik,t}}{W_{k,t}}\mathcal{L}(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t};\boldsymbol{\theta}_{i,t}),y_{k,t}) - \sum_{t=1}^{T}\mathcal{L}(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t};\boldsymbol{\theta}_{i,t}),y_{k,t}) \leq \frac{\ln N}{\eta_k} + \frac{\eta_k}{2}T. \tag{C.7}$$

Since the loss function $\mathcal{L}(\cdot,\cdot)$ is convex, using (C.7) and Jensen inequality we can write

$$\sum_{t=1}^{T}\mathcal{L}\left(\sum_{i=1}^{N}\frac{w_{ik,t}}{W_{k,t}}\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t};\boldsymbol{\theta}_{i,t}),y_{k,t}\right) - \sum_{t=1}^{T}\mathcal{L}(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t};\boldsymbol{\theta}_{i,t}),y_{k,t}) \leq \frac{\ln N}{\eta_k} + \frac{\eta_k}{2}T \tag{C.8}$$

which proves the Theorem 3.1.

C.2 Proof of Theorem 3.2

In order to prove Theorem 3.2, the following Lemma is used as a stepstone.

Lemma C.1. *Let $\alpha^*_{ik,t}$, $\forall t \in [T]$, $\forall k \in [K]$ represents the optimal coefficients associ-*

ated with the i-th kernel such that $f_i^*(\boldsymbol{x}) = \sum_{t=1}^{T}\sum_{k=1}^{K}\alpha_{ik,t}^*\kappa_i(\boldsymbol{x},\boldsymbol{x}_{k,t})$*. And* $\hat{f}_i^*(\boldsymbol{x}) = (\boldsymbol{\theta}_i^*)^\top \boldsymbol{z}_i(\boldsymbol{x})$ *denotes the best RF-based estimator associated with the i-th kernel such that* $\boldsymbol{\theta}_i^* = \sum_{t=1}^{T}\sum_{k=1}^{K}\alpha_{ik,t}^*\boldsymbol{z}_i(\boldsymbol{x}_{k,t})$*. Under assumptions (A1)–(A3), using POF-MKL, the RF approximation of the i-th kernel satisfies*

$$\begin{aligned}&\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}(\hat{f}_{RF,it}(\boldsymbol{x}_{k,t};\boldsymbol{\theta}_{i,t}),y_{k,t}) - \sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}(\hat{f}_i^*(\boldsymbol{x}_{k,t}),y_{k,t})\\ \le&\frac{K\|\boldsymbol{\theta}_i^*\|^2}{2\eta} + \frac{\eta}{2}\sum_{t=1}^{T}\sum_{k=1}^{K}\frac{L^2}{p_{ik,t}}.\end{aligned} \tag{C.9}$$

Proof. Let $\ell_{ik,t}$ be the importance sampling loss estimate defined as

$$\ell_{ik,t} = \frac{\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \boldsymbol{z}_i(\boldsymbol{x}_{k,t}),y_{k,t})}{p_{ik,t}}\mathbf{1}_{i\in\mathcal{S}_{k,t}}. \tag{C.10}$$

Then according to (1.11), for any fixed $\boldsymbol{\theta}$, it can be written that

$$\begin{aligned}\left\|\frac{1}{K}\sum_{k=1}^{K}\boldsymbol{\theta}_{ik,t+1} - \boldsymbol{\theta}\right\|^2 &= \left\|\frac{1}{K}\sum_{k=1}^{K}(\boldsymbol{\theta}_{i,t} - \eta\nabla\ell_{ik,t}) - \boldsymbol{\theta}\right\|^2 = \left\|\boldsymbol{\theta}_{i,t} - \boldsymbol{\theta} - \frac{\eta}{K}\sum_{k=1}^{K}\nabla\ell_{ik,t}\right\|^2\\ &= \|\boldsymbol{\theta}_{i,t}-\boldsymbol{\theta}\|^2 - \frac{2\eta}{K}\left(\sum_{k=1}^{K}\nabla^\top\ell_{ik,t}\right)(\boldsymbol{\theta}_{i,t}-\boldsymbol{\theta}) + \left\|\frac{\eta}{K}\sum_{k=1}^{K}\nabla\ell_{ik,t}\right\|^2\end{aligned} \tag{C.11}$$

According to the convexity of the loss function $\mathcal{L}(\boldsymbol{\theta}^\top\boldsymbol{x},y)$ with respect to $\boldsymbol{\theta}$ as stated in (A1), we find

$$\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top\boldsymbol{z}_i(\boldsymbol{x}_{k,t}),y_{k,t}) - \mathcal{L}(\boldsymbol{\theta}^\top\boldsymbol{z}_i(\boldsymbol{x}_{k,t}),y_{k,t}) \le \nabla^\top\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top\boldsymbol{z}_i(\boldsymbol{x}_{k,t}),y_{k,t})(\boldsymbol{\theta}_{i,t}-\boldsymbol{\theta}). \tag{C.12}$$

Multiplying both sides of (C.12) by $\frac{\mathbf{1}_{i\in\mathcal{S}_{k,t}}}{p_{ik,t}}$, we get

$$\begin{aligned}&\left(\frac{\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})}{p_{ik,t}} - \frac{\mathcal{L}(\boldsymbol{\theta}^\top \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})}{p_{ik,t}}\right)\mathbf{1}_{i\in\mathcal{S}_{k,t}} \\ \le& \frac{\nabla^\top \mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})}{p_{ik,t}}(\boldsymbol{\theta}_{i,t} - \boldsymbol{\theta})\mathbf{1}_{i\in\mathcal{S}_{k,t}}. \end{aligned} \tag{C.13}$$

Summing (C.13) over k, $\forall k \in [K]$, we arrive at

$$\begin{aligned}&\sum_{k=1}^{K}\left(\frac{\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})}{p_{ik,t}} - \frac{\mathcal{L}(\boldsymbol{\theta}^\top \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})}{p_{ik,t}}\right)\mathbf{1}_{i\in\mathcal{S}_{k,t}} \\ \le& \left(\sum_{k=1}^{K}\frac{\nabla^\top \mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})}{p_{ik,t}}\mathbf{1}_{i\in\mathcal{S}_{k,t}}\right)(\boldsymbol{\theta}_{i,t} - \boldsymbol{\theta}). \end{aligned} \tag{C.14}$$

Based on the definition of $\ell_{ik,t}$, (C.14) can be rewritten as

$$\sum_{k=1}^{K}\ell_{ik,t} - \sum_{k=1}^{K}\frac{\mathcal{L}(\boldsymbol{\theta}^\top \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})}{p_{ik,t}}\mathbf{1}_{i\in\mathcal{S}_{k,t}} \le \left(\sum_{k=1}^{K}\nabla^\top \ell_{ik,t}\right)(\boldsymbol{\theta}_{i,t} - \boldsymbol{\theta}). \tag{C.15}$$

According to (C.11), (C.15) is equivalent to

$$\begin{aligned}&\sum_{k=1}^{K}\ell_{ik,t} - \sum_{k=1}^{K}\frac{\mathcal{L}(\boldsymbol{\theta}^\top \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})}{p_{ik,t}}\mathbf{1}_{i\in\mathcal{S}_{k,t}} \\ \le& \frac{K}{2\eta}\left(\|\boldsymbol{\theta}_{i,t} - \boldsymbol{\theta}\|^2 - \left\|\frac{1}{K}\sum_{k=1}^{K}\boldsymbol{\theta}_{ik,t+1} - \boldsymbol{\theta}\right\|^2 + \left\|\frac{\eta}{K}\sum_{k=1}^{K}\nabla \ell_{ik,t}\right\|^2\right) \end{aligned} \tag{C.16}$$

Expectations of $\ell_{ik,t}$ and $\|\nabla \ell_{ik,t}\|^2$ with respect to $\mathbf{1}_{i\in\mathcal{S}_{k,t}}$ can be calculated as

$$\mathbb{E}_t[\ell_{ik,t}] = \frac{\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})}{p_{ik,t}}p_{ik,t} = \mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t}) \tag{C.17a}$$

$$\mathbb{E}_t[\|\nabla \ell_{ik,t}\|^2] = \frac{\|\nabla\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})\|^2}{p_{ik,t}^2}p_{ik,t} = \frac{\|\nabla\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top \boldsymbol{z}_i(\boldsymbol{x}_{k,t}), y_{k,t})\|^2}{p_{ik,t}}. \tag{C.17b}$$

Furthermore, using AM-GM inequality and (C.17b), it can be concluded that

$$\mathbb{E}_t\left[\|\sum_{k=1}^{K}\nabla\ell_{ik,t}\|^2\right] \leq \mathbb{E}_t\left[K\sum_{k=1}^{K}\|\nabla\ell_{ik,t}\|^2\right] \leq K\sum_{k=1}^{K}\frac{\|\nabla\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top\boldsymbol{z}_i(\boldsymbol{x}_{k,t}),y_{k,t})\|^2}{p_{ik,t}}. \tag{C.18}$$

According to (3.10), considering the fact that

$$\|\boldsymbol{\theta}_{i,t+1}-\boldsymbol{\theta}\|^2 = \left\|\frac{1}{K}\sum_{k=1}^{K}\boldsymbol{\theta}_{ik,t+1}-\boldsymbol{\theta}\right\|^2$$

taking the expectation from (C.16) with respect to $\mathbf{1}_{i\in\mathcal{S}_{k,t}}$, $\forall k\in[K]$ leads to

$$\begin{aligned}&\sum_{k=1}^{K}\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top\boldsymbol{z}_i(\boldsymbol{x}_{k,t}),y_{k,t}) - \sum_{k=1}^{K}\mathcal{L}(\boldsymbol{\theta}^\top\boldsymbol{z}_i(\boldsymbol{x}_{k,t}),y_{k,t})\\ \leq&\frac{K}{2\eta}\left(\|\boldsymbol{\theta}_{i,t}-\boldsymbol{\theta}\|^2-\|\boldsymbol{\theta}_{i,t+1}-\boldsymbol{\theta}\|^2\right)+\frac{\eta}{2}\sum_{k=1}^{K}\frac{\|\nabla\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top\boldsymbol{z}_i(\boldsymbol{x}_{k,t}),y_{k,t})\|^2}{p_{ik,t}}.\end{aligned} \tag{C.19}$$

According to (A2), we can conclude that $\|\nabla\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top\boldsymbol{z}_i(\boldsymbol{x}_{k,t}),y_{k,t})\|^2 \leq L^2$. Hence, summing (C.19) over time, given the fact that $\boldsymbol{\theta}_{i,1}=\mathbf{0}$, $\forall i\in[N]$, we get

$$\begin{aligned}&\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top\boldsymbol{z}_i(\boldsymbol{x}_{k,t}),y_{k,t}) - \sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}(\boldsymbol{\theta}^\top\boldsymbol{z}_i(\boldsymbol{x}_{k,t}),y_{k,t})\\ \leq&\frac{K}{2\eta}\left(\|\boldsymbol{\theta}\|^2-\|\boldsymbol{\theta}_{i,T+1}-\boldsymbol{\theta}\|^2\right)+\frac{\eta}{2}\sum_{t=1}^{T}\sum_{k=1}^{K}\frac{L^2}{p_{ik,t}}.\end{aligned} \tag{C.20}$$

Replacing $\boldsymbol{\theta}$ with $\boldsymbol{\theta}_i^*$ and considering the fact that $\|\boldsymbol{\theta}_{i,T+1}-\boldsymbol{\theta}\|^2\geq 0$, we obtain

$$\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}(\boldsymbol{\theta}_{i,t}^\top\boldsymbol{z}_i(\boldsymbol{x}_{k,t}),y_{k,t}) - \sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}((\boldsymbol{\theta}_i^*)^\top\boldsymbol{z}_i(\boldsymbol{x}_{k,t}),y_{k,t}) \leq \frac{K\|\boldsymbol{\theta}_i^*\|^2}{2\eta}+\frac{\eta}{2}\sum_{t=1}^{T}\sum_{k=1}^{K}\frac{L^2}{p_{ik,t}}$$

which proves the Lemma C.1. □

In order to proof Theorem 3.2, we leverage the results obtained in the proofs of Lemma 1.2 and Theorem 3.1. Since (C.8) holds true for any i, summing (C.8) over all $k\in[K]$, for any i

we can write

$$\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}(\hat{f}(\boldsymbol{x}_{k,t};\hat{\boldsymbol{\Theta}}_t,\boldsymbol{w}_{k,t}),y_{k,t})-\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}(\hat{f}_{\mathrm{RF},it}(\boldsymbol{x}_{k,t};\boldsymbol{\theta}_{i,t}),y_{k,t})$$
$$\leq\sum_{k=1}^{K}\left(\frac{\ln N}{\eta_k}+\frac{\eta_k}{2}T\right). \tag{C.21}$$

Combining (C.21) with (1.44), we arrive at

$$\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}(\hat{f}(\boldsymbol{x}_{k,t};\hat{\boldsymbol{\Theta}}_t,\boldsymbol{w}_{k,t}),y_{k,t})-\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}(\hat{f}_i^*(\boldsymbol{x}_{k,t}),y_{k,t})$$
$$\leq\frac{K\|\boldsymbol{\theta}_i^*\|^2}{2\eta}+\frac{\eta}{2}\sum_{t=1}^{T}\sum_{k=1}^{K}\frac{L^2}{p_{ik,t}}+\sum_{k=1}^{K}\left(\frac{\ln N}{\eta_k}+\frac{\eta_k}{2}T\right). \tag{C.22}$$

According to claim 1 in [123], it can be written that $\sup_{\boldsymbol{x},\boldsymbol{x}'}|\boldsymbol{z}_i^\top(\boldsymbol{x})\boldsymbol{z}_i(\boldsymbol{x}')-\kappa_i(\boldsymbol{x},\boldsymbol{x}')|\leq\epsilon$ holds true with probability greater than $1-2^8\left(\frac{\sigma_i}{\epsilon}\right)^2\exp\left(-\frac{D\epsilon^2}{4(d+2)}\right)$ where σ_i is the second Fourier moment of the i-th kernel $\kappa_i(\cdot)$. Furthermore, according to (A3), the loss function is L-Lipschitz continuous and as a result it can be written that

$$\sum_{t=1}^{T}\sum_{k=1}^{K}|\mathcal{L}(\hat{f}_i^*(\boldsymbol{x}_{k,t}),y_{k,t})-\mathcal{L}(f_i^*(\boldsymbol{x}_{k,t}),y_{k,t})|$$
$$\leq\sum_{t=1}^{T}\sum_{k=1}^{K}L\left|\sum_{\tau=1}^{T}\sum_{j=1}^{K}\alpha_{ij,\tau}^*\boldsymbol{z}_i^\top(\boldsymbol{x}_{j,\tau})\boldsymbol{z}_i(\boldsymbol{x}_{k,t})-\sum_{\tau=1}^{T}\sum_{j=1}^{K}\alpha_{ij,\tau}^*\kappa_i(\boldsymbol{x}_{j,\tau},\boldsymbol{x}_{k,t})\right|. \tag{C.23}$$

Applying Cauchy-Schwartz inequality to the right hand side of (C.23), the left hand side of (C.23) can be bounded from above as

$$\sum_{t=1}^{T}\sum_{k=1}^{K}|\mathcal{L}(\hat{f}_i^*(\boldsymbol{x}_{k,t}),y_{k,t})-\mathcal{L}(f_i^*(\boldsymbol{x}_{k,t}),y_{k,t})|$$
$$\leq\sum_{t=1}^{T}\sum_{k=1}^{K}L\sum_{\tau=1}^{T}\sum_{j=1}^{K}|\alpha_{ij,\tau}^*||\boldsymbol{z}_i^\top(\boldsymbol{x}_{j,\tau})\boldsymbol{z}_i(\boldsymbol{x}_{k,t})-\kappa_i(\boldsymbol{x}_{j,\tau},\boldsymbol{x}_{k,t})|. \tag{C.24}$$

Let $C := \max_{i\in[N]} \sum_{t=1}^{T}\sum_{k=1}^{K} \alpha^*_{ik,t}$. Therefore, we can conclude that

$$\sum_{t=1}^{T}\sum_{k=1}^{K} |\mathcal{L}(\hat{f}_i^*(\boldsymbol{x}_{k,t}), y_{k,t}) - \mathcal{L}(f_i^*(\boldsymbol{x}_{k,t}), y_{k,t})| \leq \epsilon LKTC \tag{C.25}$$

with probability at least $1 - 2^8\left(\frac{\sigma_i}{\epsilon}\right)^2 \exp\left(-\frac{D\epsilon^2}{4(d+2)}\right)$. Moreover, using Triangle inequality, we can write

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{k=1}^{K} \mathcal{L}(\hat{f}_i^*(\boldsymbol{x}_{k,t}), y_{k,t}) - \sum_{t=1}^{T}\sum_{k=1}^{K} \mathcal{L}(f_i^*(\boldsymbol{x}_{k,t}), y_{k,t}) \\
\leq &\left|\sum_{t=1}^{T}\sum_{k=1}^{K} \mathcal{L}(\hat{f}_i^*(\boldsymbol{x}_{k,t}), y_{k,t}) - \mathcal{L}(f_i^*(\boldsymbol{x}_{k,t}), y_{k,t})\right| \\
\leq &\sum_{t=1}^{T}\sum_{k=1}^{K} |\mathcal{L}(\hat{f}_i^*(\boldsymbol{x}_{k,t}), y_{k,t}) - \mathcal{L}(f_i^*(\boldsymbol{x}_{k,t}), y_{k,t})| \leq \epsilon LKTC
\end{aligned} \tag{C.26}$$

which holds true with probability at least $1-2^8\left(\frac{\sigma_i}{\epsilon}\right)^2 \exp\left(-\frac{D\epsilon^2}{4(d+2)}\right)$. Moreover, for $\boldsymbol{z}_i^\top(\boldsymbol{x})\boldsymbol{z}_i(\boldsymbol{x}')$ we can write

$$\boldsymbol{z}_i^\top(\boldsymbol{x})\boldsymbol{z}_i(\boldsymbol{x}') = \frac{1}{D}\sum_{j=1}^{D} (\sin(\boldsymbol{\rho}_{i,j}^\top\boldsymbol{x})\sin(\boldsymbol{\rho}_{i,j}^\top\boldsymbol{x}') + \cos(\boldsymbol{\rho}_{i,j}^\top\boldsymbol{x})\cos(\boldsymbol{\rho}_{i,j}^\top\boldsymbol{x}')). \tag{C.27}$$

Based on arithmetic-mean geometric-mean, (C.27) can be relaxed to

$$\boldsymbol{z}_i^\top(\boldsymbol{x})\boldsymbol{z}_i(\boldsymbol{x}') \leq \frac{1}{D}\sum_{j=1}^{D} \frac{1}{2}(\sin^2(\boldsymbol{\rho}_{i,j}^\top\boldsymbol{x}) + \sin^2(\boldsymbol{\rho}_{i,j}^\top\boldsymbol{x}') + \cos^2(\boldsymbol{\rho}_{i,j}^\top\boldsymbol{x}) + \cos^2(\boldsymbol{\rho}_{i,j}^\top\boldsymbol{x}')) = 1. \tag{C.28}$$

Thus, given the fact that $|\boldsymbol{z}_i^\top(\boldsymbol{x})\boldsymbol{z}_i(\boldsymbol{x}')| \leq 1$, $\|\boldsymbol{\theta}_i^*\|^2$ can be bounded as

$$\|\boldsymbol{\theta}_i^*\|^2 \leq \sum_{t=1}^{T}\sum_{k=1}^{K}\sum_{\tau=1}^{T}\sum_{j=1}^{K} |\alpha^*_{ik,t}\alpha^*_{ij,\tau}\boldsymbol{z}_i^\top(\boldsymbol{x}_{j,\tau})\boldsymbol{z}_i(\boldsymbol{x}_{k,t})| \leq C^2. \tag{C.29}$$

Combining (C.26) and (C.29) with (C.22) yields

$$\begin{aligned}&\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}(\hat{f}(\boldsymbol{x}_{k,t};\hat{\boldsymbol{\Theta}}_t,\boldsymbol{w}_{k,t}),y_{k,t})-\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{L}(f_i^*(\boldsymbol{x}_{k,t}),y_{k,t})\\ \le&\frac{KC^2}{2\eta}+\frac{\eta}{2}\sum_{t=1}^{T}\sum_{k=1}^{K}\frac{L^2}{p_{ik,t}}+\sum_{k=1}^{K}\left(\frac{\ln N}{\eta_k}+\frac{\eta_k}{2}T\right)+\epsilon LKTC \end{aligned} \tag{C.30}$$

which holds true for any $i \in [N]$ with probability at least $1-2^8\left(\frac{\sigma_i}{\epsilon}\right)^2\exp\left(-\frac{D\epsilon^2}{4(d+2)}\right)$. Therefore, this proves the Theorem 3.2.

C.3 Supplementary Experimental Results and Details

This section presents further experimental results testing different aspects of the proposed algorithm POF-MKL. Moreover, this section provides more detailed information about experimental setup associated with results in section 3.4. The performance of federated kernel learning algorithms are tested on the following datasets:

- **Naval:** The dataset consists of $11,500$ samples. Each sample has 15 features of a a naval vessel. The goal is to predict lever position [31].
- **UJI:** The dataset consists of $21,000$ data samples. Each data sample has 520 features which are WiFi fingerprints. The goal is to predict the geographical longitude associated with each data sample.
- **Air:** The dataset consists of $120,000$ samples with 14 features including information related to air quality such as concentration of some chemicals in the air. Data samples are collected from 4 different geographical sites. The goal is to predict the concentration of CO in the air [161]. For each site there are $30,000$ samples in the dataset.
- **WEC:** The dataset consists of $280,000$ samples with 48 features of wave energy

converters. Data samples are collected from 4 different geographical sites. The goal is to predict total power output [117]. For each site, there are $70,000$ samples.

Data samples of Naval and UJI datasets are distributed i.i.d among clients. The number of clients for Naval and UJI datasets are 23 and 42, respectively. Data samples in Air and WEC datasets are distributed non-i.i.d among clients. The number of clients for Air and WEC datasets are 240 and 560, respectively. For both Air and WEC datasets, there are 4 different geographical sites that each sample belongs to one of them. Each client observes 350 samples from one site and 50 samples from each of the rest of 3 sites. Moreover, PerFedAvg uses a feedforward neural network model. Each layer is a fully-connected dense layer with at most 20 neurons. Neurons in hidden layers exploit ReLU activation functions. Since each client cannot transmit more than 1000 parameters to the server, the number of hidden layers is determined in a way that the number of the neural network's parameters to be less than 1000. The number of parameters depends on the number of features in data samples. Therefore, the number of hidden layers varies across different datasets. For each dataset, given the number of features, the maximum number of hidden layers with 20 neurons is chosen. All experiments were carried out using Intel(R) Core(TM) i7-10510U CPU @ 1.80 GHz 2.30 GHz processor with a 64-bit Windows operating system.

Table C.1 presents average MSE along with MSE standard deviation calculated over 20 different sets of random feature vectors. As it can be seen from Table C.1, the proposed POF-MKL provides lower standard deviation compared to all other baselines. This shows that the proposed POF-MKL is less sensitive to the choice of random features. Furthermore, Table C.2 reports the average cumulative regret of clients along with the standard deviation of regret among clients. As it can be inferred from Table C.2, the proposed POF-MKL obtains lower regret than other online federated MKL algorithms. Moreover, for Air and WEC datasets, the standard deviation of regret among clients associated with POF-MKL is considerably lower than those of other online federated MKL algorithms. Note that data

Table C.1: MSE($\times 10^{-3}$) and standard deviation($\times 10^{-3}$) of online federated learning algorithms on real datasets.

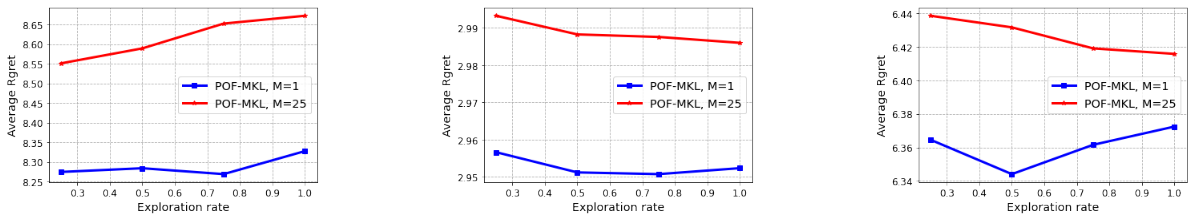
Algorithms	M	D	Naval	UJI	Air	WEC
OFSKL	1	100	77.77 ± 1.04	61.82 ± 2.76	13.65 ± 0.61	87.87 ± 3.93
OFMKL-Avg	51	9	33.25 ± 1.46	55.44 ± 2.48	10.63 ± 0.47	34.01 ± 1.52
vM-KOFL	51	9	26.42 ± 1.16	51.50 ± 2.30	10.58 ± 0.47	25.17 ± 1.12
eM-KOFL	1	100	28.64 ± 1.32	61.08 ± 2.73	21.94 ± 1.16	20.14 ± 0.93
POF-MKL	1	100	$\mathbf{16.16 \pm 0.72}$	$\mathbf{33.02 \pm 1.48}$	$\mathbf{9.27 \pm 0.41}$	$\mathbf{11.44 \pm 0.52}$
POF-MKL	25	20	16.82 ± 0.74	37.34 ± 1.67	9.34 ± 0.42	11.58 ± 0.53
POF-MKL	51	9	16.65 ± 0.74	41.00 ± 1.83	9.38 ± 0.42	11.97 ± 0.55

Table C.2: Average regret and its standard deviation across clients for online federated MKL learning algorithms.

Algorithms	M	D	Naval	UJI	Air	WEC
OFMKL-Avg	51	9	16.95 ± 0.39	24.23 ± 20.33	3.05 ± 2.83	17.07 ± 14.52
vM-KOFL	51	9	13.40 ± 0.39	22.28 ± 15.56	3.02 ± 2.79	12.65 ± 11.53
eM-KOFL	1	100	14.92 ± 0.65	27.26 ± 19.79	8.53 ± 2.51	10.43 ± 8.28
POF-MKL	1	100	$\mathbf{8.33} \pm 0.39$	$\mathbf{13.23 \pm 8.80}$	$\mathbf{2.95 \pm 2.08}$	$\mathbf{6.37 \pm 5.28}$
POF-MKL	25	20	8.67 ± 0.39	15.41 ± 9.58	2.98 ± 2.12	6.41 ± 5.39
POF-MKL	51	9	8.55 ± 0.39	17.23 ± 10.07	3.01 ± 2.15	6.51 ± 5.57

samples in Air and WEC datasets are distributed non-i.i.d among clients. Therefore, the results in Table C.2 confirm that the proposed POF-MKL can better deal with heterogeneous data among clients.

Figure C.1 illustrates the average regret of clients employing POF-MKL with the change in the value of exploration rate ξ_k when the exploration rate of all clients are the same. In particular, Figure C.1 depicts the performance of POF-MKL for $M = 1$ and $M = 25$ with the change in ξ_k. According to the PMF $\boldsymbol{q}_{k,t}$ defined in (1.10), the increase in ξ_k leads to increase in exploration such that if $\xi_k = 1$, the k-th client chooses a subset of kernels uniformly at random. Figure C.1 indicates that the optimal choice of ξ_k in terms of regret depends on the dataset distributed among clients as well as the number of chosen kernels M. Moreover, the



(a) Naval dataset. (b) Air dataset. (c) WEC dataset.

Figure C.1: Average cumulative regret of clients with the change in the value of exploration rate (ξ_k).

choice of ξ_k is related to the computational complexity of executing POF-MKL by clients. Specifically, when $\xi_k < 1$, in order to choose a subset of kernels, the k-th client needs to sort kernels which imposes worst case computational complexity of $\mathcal{O}(N \log N)$. However, when $\xi_k = 1$, according to PMF in (1.10), the k-th client chooses one bin uniformly at random and as a result in this case the k-th client does not need to sort kernels. Also, it is useful to note that as it can be inferred from (1.11), clients can leverage the exploration rate ξ_k to send their updates to the server without revealing both the gradient of loss and the loss of kernels which can promote the privacy of the proposed POF-MKL.

Appendix D

Supplementary Proofs and Experiments for Chapter 4

D.1 First Fit Decreasing Algorithm

In this chapter, first fit decreasing algorithm is employed to split models into clusters. To begin with, order models by decreasing cost. Let $s(1), \dots, s(K-1)$ denote the indices of all models except for the $I_{i,t}$-th model ordered in ascending manner according to models' costs such that if $i \le j$, then $c_{s(i)} \le c_{s(j)}$. At the k-th step of clustering, client i checks whether the $s(k)$-th model can be fit into any currently existing clusters according to budget $B_i - c_{I_{i,t}}$. The $s(k)$-th model is put into the first cluster that it can be fit into. Otherwise, if it cannot be fit into any opened cluster, then it is assigned to a new cluster indexed by $m_{i,t}+1$. This continues until all models except for the $I_{i,t}$-th model are corresponded to a cluster. Algorithm 15 summarizes the clustering procedure performed by the i-th client.

Algorithm 15 Cluster Generation by Client i at Learning Round t

```
1: Input: Chosen model index $I_{i,t}$, costs $c_k$, $\forall k \in [K]$ and budget $B_i$.
2: Initialize: $m_{i,t} = 1$.
3: Order all models except for the $I_{i,t}$-th model by decreasing cost to obtain $s(1), \ldots, s(K-1)$.

4: for $k = 1, \ldots, K-1$ do
5:    Set $j = 1$ and $d = 0$
6:    while $d = 0$ and $j \leq m_{i,t}$ do
7:       if $\sum_{m \in \mathbb{D}_{ij,t}} c_m + c_{s(k)} \leq B_i - c_{I_{i,t}}$ then
8:          Set $d = 1$ and $j = j + 1$
9:       end if
10:   end while
11:   if $j = m_{i,t} + 1$ and $d = 0$ then
12:      Assign the $s(k)$-th model to a new cluster $\mathbb{D}_{i(m_{i,t}+1),t}$.
13:      Update $m_{i,t} = m_{i,t} + 1$.
14:   end if
15: end for
16: Output: $\{\mathbb{D}_{i1,t}, \ldots, \mathbb{D}_{im_{i,t},t}\}$
```

D.2 Proof of Theorem 4.1

In order to prove Theorem 4.1, the following Lemma is used as the step-stone.

Lemma D.1. *Under (A1) and (A2), the regret of the i-th client with respect to any model k is bounded from above as*

$$\sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(f_{I_{i,t}}(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{I_{i,t},t}), y_{i,t})] - \sum_{t=1}^{T} \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t}) \leq \frac{\ln K}{\eta_i} + \eta_i \mu_i T. \tag{D.1}$$

Proof. Recall that $Z_{i,t} = \sum_{k=1}^{K} z_{ik,t}$. Therefore, we can write

$$\frac{Z_{i,t+1}}{Z_{i,t}} = \sum_{k=1}^{K} \frac{z_{ik,t+1}}{Z_{i,t}} = \sum_{k=1}^{K} \frac{z_{ik,t}}{Z_{i,t}} \exp\left(-\eta_i \ell_{ik,t}\right). \tag{D.2}$$

According to (4.4), $\frac{z_{ik,t}}{Z_{i,t}} = p_{ik,t}$ and as a result (D.2) can be rewritten as

$$\frac{Z_{i,t+1}}{Z_{i,t}} = \sum_{k=1}^{K} p_{ik,t} \exp\left(-\eta_i \ell_{ik,t}\right). \tag{D.3}$$

Combining the inequality $e^{-x} \leq 1 - x + \frac{1}{2}x^2, \forall x \geq 0$ with (D.2) we can conclude that

$$\frac{Z_{i,t+1}}{Z_{i,t}} \leq \sum_{k=1}^{K} p_{ik,t} \left(1 - \eta_i \ell_{ik,t} + \frac{1}{2}(\eta_i \ell_{ik,t})^2\right). \tag{D.4}$$

Employing the inequality $1 + x \leq e^x$ and taking logarithm from both sides of (D.4), we arrive at

$$\ln \frac{Z_{i,t+1}}{Z_{i,t}} \leq \sum_{k=1}^{K} p_{ik,t} \left(-\eta_i \ell_{ik,t} + \frac{1}{2}(\eta_i \ell_{ik,t})^2\right). \tag{D.5}$$

Summing (D.5) over learning rounds leads to

$$\ln \frac{Z_{i,T+1}}{Z_{i,1}} \leq \sum_{t=1}^{T} \sum_{k=1}^{K} p_{ik,t} \left(-\eta_i \ell_{ik,t} + \frac{1}{2}(\eta_i \ell_{ik,t})^2\right). \tag{D.6}$$

In addition, $\ln \frac{Z_{i,T+1}}{Z_{i,1}}$ can be bounded from below as

$$\ln \frac{Z_{i,T+1}}{Z_{i,1}} \geq \ln \frac{z_{ik,T+1}}{Z_{i,1}} = -\eta_i \sum_{t=1}^{T} \ell_{ik,t} - \ln K, \tag{D.7}$$

which holds for any $k \in [K]$. Combining (D.6) with (D.7), we get

$$\sum_{t=1}^{T} \sum_{k=1}^{K} p_{ik,t} \ell_{ik,t} - \sum_{t=1}^{T} \ell_{ik,t} \leq \frac{\ln K}{\eta_i} + \frac{\eta_i}{2} \sum_{t=1}^{T} \sum_{k=1}^{K} p_{ik,t} \ell_{ik,t}^2. \tag{D.8}$$

Considering (1.33) and (1.34), given the observed losses in prior rounds the expected value of $\ell_{ik,t}$ can be obtained as

$$\begin{aligned} \mathbb{E}_t[\ell_{ik,t}] &= \frac{\mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t})}{q_{ik,t}} p_{ik,t} + \sum_{\forall j : j \neq k} \frac{\mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t})}{q_{ik,t}} \frac{p_{ij,t}}{m_{ij,t}} \\ &= \frac{\mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t})}{q_{ik,t}} \left(p_{ik,t} + \sum_{\forall j : j \neq k} \frac{p_{ij,t}}{m_{ij,t}}\right) = \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t}). \end{aligned} \tag{D.9}$$

Moreover, based on the assumption that $0 \leq \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t}) \leq 1$, given the observed losses in prior rounds, the expected value of $\ell_{ik,t}^2$ can be bounded from above as

$$\begin{aligned}\mathbb{E}_t[\ell_{ik,t}^2] &= \frac{\mathcal{L}^2(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t})}{q_{ik,t}^2}\left(p_{ik,t} + \sum_{\forall j: j\neq k}\frac{p_{ij,t}}{m_{ij,t}}\right)\\ &= \frac{\mathcal{L}^2(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t})}{q_{ik,t}} \leq \frac{1}{q_{ik,t}}. \end{aligned} \tag{D.10}$$

Taking the expectation from both sides of (D.8), we obtain

$$\sum_{t=1}^{T}\sum_{k=1}^{K} p_{ik,t}\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t}) - \sum_{t=1}^{T}\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t}) \leq \frac{\ln K}{\eta_i} + \frac{\eta_i}{2}\sum_{t=1}^{T}\sum_{k=1}^{K}\frac{p_{ik,t}}{q_{ik,t}}. \tag{D.11}$$

In addition, it can be written that

$$\sum_{t=1}^{T}\mathbb{E}_t[\mathcal{L}(f_{I_{i,t}}(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{I_{i,t},t}), y_{i,t})] = \sum_{t=1}^{T}\sum_{k=1}^{K} p_{ik,t}\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t}). \tag{D.12}$$

Therefore, from (D.11) we arrive at

$$\sum_{t=1}^{T}\mathbb{E}_t[\mathcal{L}(f_{I_{i,t}}(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{I_{i,t},t}), y_{i,t})] - \sum_{t=1}^{T}\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t}) \leq \frac{\ln K}{\eta_i} + \frac{\eta_i}{2}\sum_{t=1}^{T}\sum_{k=1}^{K}\frac{p_{ik,t}}{q_{ik,t}}. \tag{D.13}$$

According to Algorithm 12, at each learning round, client i splits all models except for the chosen model into clusters. Let ν_{ij} be the minimum number of clusters when client i chooses $I_{i,t} = j$ and splits all models except for model j into clusters. If client i employs FFD algorithm to split models, the number of clusters $m_{ij,t}$ when client i chooses $I_{i,t} = j$ satisfies $m_{ij,t} \leq \frac{11}{9}\nu_{ij} + \frac{2}{3}$ [43]. Let μ_i be defined as $\mu_i = \max_j \nu_{ij}$. Therefore, it can be concluded that

$m_{ij,t} \leq \frac{11}{9}\mu_i + \frac{2}{3} \leq 2\mu_i$. Thus, it can be written that

$$q_{ik,t} \geq p_{ik,t} + \frac{1-p_{ik,t}}{2\mu_i} \geq \frac{1}{2\mu_i} \tag{D.14}$$

Combining (D.13) with (D.14) yields

$$\sum_{t=1}^{T} \mathbb{E}_t[\mathcal{L}(f_{I_{i,t}}(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{I_{i,t},t}), y_{i,t})] - \sum_{t=1}^{T} \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t}) \leq \frac{\ln K}{\eta_i} + \eta_i\mu_i T. \tag{D.15}$$

which proves the lemma. □

In what follows the server regret upper bound in fine-tuning model k is obtained. Let $\hat{\ell}_{ik,t}$ denote the fine-tuning importance sampling loss estimate at learning round t associated with the i-th client and the k-th model, defined as

$$\hat{\ell}_{ik,t} = \frac{\alpha}{q_{ik,t}} \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t}) \mathcal{I}(i \in \mathbb{G}_t, k \in \mathbb{S}_{i,t}). \tag{D.16}$$

According to (4.11), for any fixed $\boldsymbol{\theta}$ and $k \in [K]$, it can be written that

$$\begin{aligned} \|\boldsymbol{\theta}_{k,t+1} - \boldsymbol{\theta}\|^2 &= \left\| \boldsymbol{\theta}_{k,t} - \boldsymbol{\theta} - \frac{\eta_f}{N} \sum_{i=1}^{N} \nabla \hat{\ell}_{ik,t} \right\|^2 \\ &= \|\boldsymbol{\theta}_{k,t} - \boldsymbol{\theta}\|^2 - \frac{2\eta_f}{N} \sum_{i=1}^{N} \nabla^\top \hat{\ell}_{ik,t}(\boldsymbol{\theta}_{k,t} - \boldsymbol{\theta}) + \left\| \frac{\eta_f}{N} \sum_{i=1}^{N} \nabla \hat{\ell}_{ik,t} \right\|^2. \end{aligned} \tag{D.17}$$

Moreover, due to the convexity of the loss function $\mathcal{L}(\cdot,\cdot)$, for any learning round t, we find that

$$\nabla^\top \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t})(\boldsymbol{\theta} - \boldsymbol{\theta}_{k,t}) \leq \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}), y_{i,t}) - \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t}) \tag{D.18}$$

Multiplying both sides of (D.18) by $\frac{\alpha \mathcal{I}(i \in \mathbb{G}_t, k \in \mathbb{S}_{i,t})}{q_{ik,t}}$, we get

$$\nabla^\top \hat{\ell}_{ik,t}(\boldsymbol{\theta} - \boldsymbol{\theta}_{k,t}) \leq \frac{\alpha}{q_{ik,t}} \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}), y_{i,t}) \mathcal{I}(i \in \mathbb{G}_t, k \in \mathbb{S}_{i,t}) - \hat{\ell}_{ik,t}. \tag{D.19}$$

Summing (D.19) over clients, we obtain

$$\sum_{i=1}^{N} \hat{\ell}_{ik,t} - \sum_{i=1}^{N} \frac{\alpha}{q_{ik,t}} \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}), y_{i,t}) \mathcal{I}(i \in \mathbb{G}_t, k \in \mathbb{S}_{i,t}) \leq \sum_{i=1}^{N} \nabla^\top \hat{\ell}_{ik,t}(\boldsymbol{\theta}_{k,t} - \boldsymbol{\theta}). \tag{D.20}$$

Combining (D.17) with (D.20) leads to

$$\begin{aligned}
&\sum_{i=1}^{N} \hat{\ell}_{ik,t} - \sum_{i=1}^{N} \frac{\alpha}{q_{ik,t}} \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}), y_{i,t}) \mathcal{I}(i \in \mathbb{G}_t, k \in \mathbb{S}_{i,t}) \\
\leq& \frac{N}{2\eta_f}(\|\boldsymbol{\theta}_{k,t} - \boldsymbol{\theta}\|^2 - \|\boldsymbol{\theta}_{k,t+1} - \boldsymbol{\theta}\|^2) + \frac{\eta_f}{2N} \left\| \sum_{i=1}^{N} \nabla \hat{\ell}_{ik,t} \right\|^2.
\end{aligned} \tag{D.21}$$

Moreover, the expected value of $\hat{\ell}_{ik,t}$ and $\|\nabla \hat{\ell}_{ik,t}\|^2$ with respect to $\mathcal{I}(i \in \mathbb{G}_t, k \in \mathbb{S}_{i,t})$ can be obtained as

$$\begin{aligned}
\mathbb{E}_t[\hat{\ell}_{ik,t}] &= \frac{\alpha}{q_{ik,t}} \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t}) \times \left(\frac{p_{ik,t}}{\alpha} + \sum_{\forall j: j \neq k} \frac{p_{ij,t}}{\alpha m_{ij,t}} \right) \\
&= \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t}) && \text{(D.22a)} \\
\mathbb{E}_t[\|\nabla \hat{\ell}_{ik,t}\|^2] &= \frac{\alpha^2}{q_{ik,t}^2} \|\nabla \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t})\|^2 \times \left(\frac{p_{ik,t}}{\alpha} + \sum_{\forall j: j \neq k} \frac{p_{ij,t}}{\alpha m_{ij,t}} \right) \\
&= \frac{\alpha}{q_{ik,t}} \|\nabla \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t})\|^2 \leq \frac{\alpha G^2}{q_{ik,t}} && \text{(D.22b)}
\end{aligned}$$

where the last inequality in (D.22b) can be concluded from the assumption (A3) where $\|\nabla \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,t}), y_{i,t})\| \leq G$. In addition, using arithmetic mean geometric mean (AM-GM) inequality we find

$$\left\|\sum_{i=1}^{N} \nabla\hat{\ell}_{ik,t}\right\|^2 \leq N\sum_{i=1}^{N} \|\nabla\hat{\ell}_{ik,t}\|^2. \tag{D.23}$$

Therefore, using (D.22) and (D.23), taking the expectation from both sides of (D.21), it can be written that

$$\begin{aligned}&\sum_{i=1}^{N} \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t}) - \sum_{i=1}^{N} \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}), y_{i,t})\\ \leq &\frac{N}{2\eta_f}(\|\boldsymbol{\theta}_{k,t}-\boldsymbol{\theta}\|^2 - \|\boldsymbol{\theta}_{k,t+1}-\boldsymbol{\theta}\|^2) + \frac{\eta_f G^2}{2}\sum_{i=1}^{N}\frac{\alpha}{q_{ik,t}}.\end{aligned} \tag{D.24}$$

Summing (D.24) over learning rounds yields

$$\begin{aligned}&\sum_{i=1}^{N}\sum_{t=1}^{T} \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t}) - \sum_{i=1}^{N}\sum_{t=1}^{T} \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}), y_{i,t})\\ \leq &\frac{N}{2\eta_f}(\|\boldsymbol{\theta}_{k,1}-\boldsymbol{\theta}\|^2 - \|\boldsymbol{\theta}_{k,T+1}-\boldsymbol{\theta}\|^2) + \frac{\eta_f G^2}{2}\sum_{t=1}^{T}\sum_{i=1}^{N}\frac{\alpha}{q_{ik,t}}.\end{aligned} \tag{D.25}$$

Plugging in $\boldsymbol{\theta} = \boldsymbol{\theta}_k^*$ in (D.25) and considering the facts that $\boldsymbol{\theta}_{k,1} = \mathbf{0}$ and $\|\boldsymbol{\theta}_{k,T+1}-\boldsymbol{\theta}\|^2 \geq 0$, we arrive at

$$\begin{aligned}&\sum_{i=1}^{N}\sum_{t=1}^{T} \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t}) - \sum_{i=1}^{N}\sum_{t=1}^{T} \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_k^*), y_{i,t})\\ \leq &\frac{N}{2\eta_f}\|\boldsymbol{\theta}_k^*\|^2 + \frac{\eta_f G^2}{2}\sum_{t=1}^{T}\sum_{i=1}^{N}\frac{\alpha}{q_{ik,t}}\end{aligned} \tag{D.26}$$

According to (D.14), it can be concluded that $\frac{1}{q_{ik,t}} \leq 2\mu_i$. Therefore, considering assumption (A4), we get

$$\sum_{i=1}^{N}\sum_{t=1}^{T} \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,t}), y_{i,t}) - \sum_{i=1}^{N}\sum_{t=1}^{T} \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_k^*), y_{i,t}) \leq \frac{NR}{2\eta_f} + \sum_{i=1}^{N}\mu_i\alpha\eta_f G^2 T \tag{D.27}$$

which proves (4.13) and completes the proof of Theorem 4.1.

D.3 Supplementary Experimental Results and Details

The performance of both the proposed OFMS-FT method and other baseline approaches is evaluated through online image classification and online regression tasks. The image classification experiments involve the utilization of the CIFAR-10 and MNIST datasets. CIFAR-10 and MNIST are well-known computer vision datasets, comprising a total of $60,000$ and $70,000$ color images, respectively, distributed across 10 distinct classes. Each dataset includes $10,000$ test samples, with the remaining samples designated for training. To facilitate model selection, as outlined in Section 4.6, we train a set of 20 models using the training data from CIFAR-10 and MNIST. These models encompass two distinct architectural designs, resulting in ten models trained under each architecture. For each class label within these datasets, two models with differing architectures are trained. These models exhibit a bias towards the specific class label they are trained on, utilizing a portion of the training dataset that contains a greater number of samples from that class compared to the other classes. For the CIFAR-10 dataset, ten CNNs are trained using the VGG architecture [138] with 2 blocks, while the remaining ten are trained using VGG architecture with 3 blocks. The training data for each model is non-i.i.d. sampled from the $50,000$ training samples. Precisely, each CNN is trained on $9,500$ training data samples, consisting of $5,000$ samples from one class and 500 samples drawn from the training set of each of the other nine classes. Similarly, for the MNIST dataset, ten CNNs are trained using VGG with one block, and the other ten are trained using VGG with 2 blocks. To train each model, $6,900$ data samples are drawn from the training set, with $6,000$ samples belonging to one class and 100 samples from each of the other nine classes. Additionally, the testing data samples for CIFAR-10 and MNIST are distributed among clients in a non-i.i.d. manner. For CIFAR-10, each client receives 155 testing data samples from one class and 5 samples from each of the other nine classes. In the case of MNIST, each client receives at least 133 samples from one class and at least 5 samples from the other classes. The 200 testing data samples are randomly shuffled and are sequentially presented to each client over $T = 200$ learning rounds. Moreover, for online regression task, the performance of algorithms are tested on the following datasets [84]:

- **Air:** Each data sample has 14 features including information related to air quality such as concentration of some chemicals in the air. Data samples are collected from different geographical sites. The goal is to predict the concentration of CO in the air [161].
- **WEC:** Each data sample has 48 features of wave energy converters. Data samples are collected from 4 different geographical sites. The goal is to predict total power output [117].

For each regression dataset, 20 fully-connected feedforward neural networks are trained. All neural networks have 5 hidden layers each with 100 hidden neurons. ReLU activation function is employed for all hidden neurons in all networks. In order to train models for Air dataset, 10 neural networks are trained on $30,000$ samples from the site Dongsi with different initialization while other 10 neural networks are trained on $30,000$ samples of Dingling site with different initialization. In the experiments, data samples of Air dataset are distributed non-i.i.d among clients such that 50 clients observe data samples from Aotizhongxin site while other 50 clients observe data samples from Changping site. To train models for WEC dataset, 10 neural networks are trained on $70,000$ samples from the site in Sydney with different initialization. The remaining 10 neural networks are trained on $70,000$ samples from the site in Tasmania with different initialization. Data samples of WEC dataset are distributed non-i.i.d among clients such that 50 clients observe data samples from Adelaide site while other 50 clients observe data samples from Perth site. In the experiments, using Fed-OMD and PerFedAvg, each client performs one epoch of stochastic gradient descent

Table D.1: Average and standard deviation of clients' accuracy using OFMS-FT over CIFAR-10 and MNIST with the change in budget.

Budget	CIFAR-10	MNIST
$B_i = 2, \forall i \in [N]$	$70.01\% \pm 6.96\%$	$89.87\% \pm 3.33\%$
$B_i = 5, \forall i \in [N]$	$76.77\% \pm 4.46\%$	$92.05\% \pm 2.69\%$
$B_i = 10, \forall i \in [N]$	$79.90\% \pm 4.23\%$	$92.93\% \pm 2.58\%$

Table D.2: Average and standard deviation of clients' MSE ($\times 10^{-3}$) using OFMS-FT over Air and WEC with the change in budget.

Budget	Air	WEC
$B_i = 2, \forall i \in [N]$	7.51 ± 4.82	8.27 ± 1.56
$B_i = 5, \forall i \in [N]$	7.46 ± 5.10	7.09 ± 1.67
$B_i = 10, \forall i \in [N]$	7.38 ± 4.86	6.99 ± 1.63

(SGD) with learning rate of 0.001 on its batch of data to fine-tune the model. In order to perform fine-tuning, clients start to update models after 50 learning rounds so that clients can store 50 samples in batch. All experiments were carried out using Intel(R) Core(TM) i7-10510U CPU @ 1.80 GHz 2.30 GHz processor with a 64-bit Windows operating system.

Table D.1 shows the average accuracy of clients along with its standard deviation on CIFAR-10 and MNIST datasets with the change in the memory budget when clients employ OFMS-FT. Moreover, Table D.2 demonstrates the average MSE and its standard deviation across clients for different memory budgets on Air and WEC datasets when clients use OFMS-FT. Results in Tables D.1 and D.2 confirm that if clients have larger memory, the accuracy of OFMS-FT improves.

We report the run times of algorithms in Table D.3. Run time refers to average total run time of clients to perform their prediction task on the entire data samples that they observe up until time horizon T. In Table D.3, OFMS-FT, $B_i = 5$, and OFMS-FT, $B_i = 2$ refer

Table D.3: Average run time (s) of clients on CIFAR-10, MNIST, Air and WEC datasets.

Algorithms	CIFAR-10	MNIST	Air	WEC
MAB	9.43	9.34	8.89	9.51
Non-Fed-OMS	57.38	62.09	57.56	52.70
Fed-OMD	32.80	23.24	15.86	15.64
PerFedAvg	47.61	32.21	19.29	22.03
OFMS-FT, $B_i = 5$, $\forall i$	145.91	99.49	55.59	55.69
OFMS-FT, $B_i = 2$, $\forall i$	64.42	43.06	27.78	28.82

to the proposed algorithm with budgets $B_i = 5$ and $B_i = 2$, respectively. Table D.3 shows that other algorithms run faster than OFMS-FT while OFMS-FT outperforms others in terms of accuracy (see Table 4.1). OFMS-FT runs slower since OFMS-FT evaluates and fine-tunes multiple models at each round. Comparing the run times of OFMS-FT, $B_i = 5$ with OFMS-FT, $B_i = 2$ shows that the time complexity of OFMS-FT can be controlled by budget. In time-sensitive scenarios, the budget can be chosen such that OFMS-FT can fulfill required computations before the start of the next round.

D.4 Supplementary Discussions and Analysis

This section presents extended discussions on performance analysis of OFMS-FT.

D.4.1 Supplementary Analysis

In sections 4.3 and 4.4, it is assumed that at each learning round t, each client observes one data sample and communicate with the server every learning round. This subsection analyzes the regret of OFMS-FT when clients communicate with the server every $n \geq 1$ learning rounds. Every round that clients communicate with the server called *communication round.*

Therefore, the number of communication rounds U is $\lfloor \frac{T}{n} \rfloor$. Let the communication u occurs at learning round τ_u. Without loss of generality, we can assume that $\tau_u = n(u-1)+1$. In this case, at communication round u, client i draws the model index $I_{i,u}$ using the PMF specified in (4.4). Then client i splits all models except for model $I_{i,u}$ into clusters $\mathbb{D}_{i1,u}, \ldots, \mathbb{D}_{im_{i,u},u}$ such that the cumulative cost of models in each cluster does not exceed $B_i - c_{I_{i,u}}$. Then client i draws one of the clusters uniformly at random. Let $J_{i,u}$ denote the index of the selected cluster. Client i downloads all models in $J_{i,u}$-th cluster in addition to model $I_{i,u}$. Upon receiving models, client i computes the importance loss estimate as

$$\ell_{ik,u} = \sum_{t=\tau_u}^{\tau_{u+1}-1} \frac{\mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,u}), y_{i,t})}{q_{ik,u}} \mathcal{I}(k \in \mathbb{S}_{i,u}) \tag{D.28}$$

where $\boldsymbol{\theta}_{k,u}$ denote the parameter of model k between communications rounds u and $u+1$ and $\mathbb{S}_{i,u}$ is a subset of models stored by client i between communications rounds u and $u+1$. Also, $q_{ik,u}$ can be obtained as

$$q_{ik,u} = p_{ik,\tau_u} + \sum_{\forall j: j \neq k} \frac{p_{ij,\tau_u}}{m_{ij,u}} \tag{D.29}$$

where $m_{ij,u}$ denote the number of model clusters at communication round u if $I_{i,u} = j$. Moreover, importance sampling gradient estimate is calculated as follows by client i

$$\nabla \hat{\ell}_{ik,u} = \sum_{t=\tau_u}^{\tau_{u+1}-1} \frac{\alpha}{q_{ik,u}} \nabla \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,u}), y_{i,t}) \mathcal{I}(i \in \mathbb{D}_u, k \in \mathbb{S}_{i,u}) \tag{D.30}$$

where $\mathbb{D}_u$ represents a subset of clients chosen by the server at communication round u to fine-tune models. The rest of procedures and definitions are the same as Algorithm 12 and Section 4.3. Moreover, when clients communicate with the server every n learning rounds, the i-th client regret $\mathcal{R}_{i,T}$ and the server regret $\mathcal{S}_{k,T}$ associated with model k are defined as

$$\mathcal{R}_{i,T} = \sum_{u=1}^{U} \mathbb{E}_u \left[\sum_{t=\tau_u}^{\tau_{u+1}-1} \mathcal{L}(f_{I_{i,u}}(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,u}), y_{i,t}) \right] - \min_{k \in [K]} \sum_{u=1}^{U} \sum_{t=\tau_u}^{\tau_{u+1}-1} \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,u}), y_{i,t}) \tag{D.31a}$$

$$\mathcal{S}_{k,T} = \frac{1}{N} \sum_{i=1}^{N} \sum_{u=1}^{U} \sum_{t=\tau_u}^{\tau_{u+1}-1} \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_{k,u}), y_{i,t}) - \frac{1}{N} \sum_{i=1}^{N} \sum_{u=1}^{U} \sum_{t=\tau_u}^{\tau_{u+1}-1} \mathcal{L}(f_k(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_k^*), y_{i,t}) \tag{D.31b}$$

where $\mathbb{E}_u[\cdot]$ denote the expected value given observed losses up until communication round u. The following Theorem obtains the regret upper bound for OFMS-FT when clients communicate with the server every n learning rounds.

Theorem D.1. *Assume that client i, $\forall i \in [N]$ communicates with the server every n learning rounds. Under (A1) and (A2), the expected cumulative regret of the i-th client using OFMS-FT is bounded by*

$$\mathcal{R}_{i,T} \leq \frac{\ln K}{\eta_i} + \eta_i \mu_i n T. \tag{D.32}$$

which holds for all $i \in [N]$. Under (A1)–(A4), the cumulative regret of the server in fine-tuning model k using OFMS-FT is bounded by

$$\mathcal{S}_{k,T} \leq \frac{R}{2\eta_f} + \frac{1}{N} \sum_{i=1}^{N} \eta_f \alpha \mu_i G^2 n T \tag{D.33}$$

Proof. see subSection D.4.2 □

If client i sets

$$\eta_i = \mathcal{O}\left(\sqrt{\frac{\ln K}{\mu_i n T}} \right), \tag{D.34}$$

then the i-th client achieves sub-linear regret of

$$\mathcal{R}_{i,T} \leq \mathcal{O}\left(\sqrt{(\ln K)\mu_i nT}\right), \tag{D.35}$$

while the server achieves sub-linear regret of

$$\mathcal{S}_{k,T} \leq \mathcal{O}\left(\sqrt{\frac{\alpha nT}{N}\sum_{i=1}^{N}\mu_i}\right) \tag{D.36}$$

by setting

$$\eta_f = \mathcal{O}\left(\frac{1}{\sqrt{\frac{\alpha nT}{N}\sum_{i=1}^{N}\mu_i}}\right). \tag{D.37}$$

As can be inferred from theorem D.1 and regret analysis presented in this subsection, the increase in n, degrades the regret upper bound of both clients and the server. Increase in n causes that clients update their stored models fewer times and this reduces the flexibility of model selection for clients. Also, increase in n leads to fine-tuning models less often which can adversely affect the prediction accuracy of models.

D.4.2 Proof of Theorem D.1

Substituting $\ell_{ik,t}$ with $\ell_{ik,u}$ in (D.2) and following the steps from (D.2) to (D.8), we get

$$\sum_{u=1}^{U}\sum_{k=1}^{K} p_{ik,\tau_u}\ell_{ik,u} - \sum_{u=1}^{U}\ell_{ik,u} \leq \frac{\ln K}{\eta_i} + \frac{\eta_i}{2}\sum_{u=1}^{U}\sum_{k=1}^{K} p_{ik,\tau_u}\ell_{ik,u}^2. \tag{D.38}$$

Moreover, the expected value of $\ell_{ik,u}$ and $\ell_{ik,u}^2$ given observed losses till communication round u, can be obtained as

$$\mathbb{E}_u[\ell_{ik,u}] = \left(\sum_{t=\tau_u}^{\tau_{u+1}-1} \frac{\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}), y_{i,t})}{q_{ik,u}}\right) \times \left(p_{ik,\tau_u} + \sum_{\forall j: j\neq k} \frac{p_{ij,\tau_u}}{m_{ij,u}}\right)$$
$$= \sum_{t=\tau_u}^{\tau_{u+1}-1} \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}), y_{i,t}). \tag{D.39}$$

Furthermore, using arithmetic-mean geometric-mean (AM-GM) inequality, $\ell_{ik,u}^2$ can bounded from above as

$$\ell_{ik,u}^2 \leq n \left(\sum_{t=\tau_u}^{\tau_{u+1}-1} \left(\frac{\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}), y_{i,t})}{q_{ik,u}} \mathcal{I}(k \in \mathbb{S}_{i,u})\right)^2\right). \tag{D.40}$$

Moreover, based on the assumption that $0 \leq \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}), y_{i,t}) \leq 1$, given the observed losses in prior rounds, expected value of $\ell_{ik,u}^2$ can be bounded from above as

$$\mathbb{E}_u\left[\left(\frac{\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}), y_{i,t})}{q_{ik,u}} \mathcal{I}(k \in \mathbb{S}_{i,u})\right)^2\right] = \frac{\mathcal{L}^2(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}), y_{i,t})}{q_{ik,u}^2} \times \left(p_{ik,\tau_u} + \sum_{\forall j: j\neq k} \frac{p_{ij,\tau_u}}{m_{ij,u}}\right)$$
$$= \frac{\mathcal{L}^2(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}), y_{i,t})}{q_{ik,u}} \leq \frac{1}{q_{ik,u}}. \tag{D.41}$$

Combining (D.40) with (D.41), we arrive at

$$\mathbb{E}_u[\ell_{ik,u}^2] \leq \frac{n^2}{q_{ik,u}}. \tag{D.42}$$

Taking the expectation from both sides of (D.38) and considering the fact that $\xi_i \geq 0$, it can be concluded that

$$\sum_{u=1}^{U}\sum_{t=\tau_u}^{\tau_{u+1}}\sum_{k=1}^{K} p_{ik,\tau_u}\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}), y_{i,t}) - \sum_{u=1}^{U}\sum_{t=\tau_u}^{\tau_{u+1}} \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}), y_{i,t})$$
$$\leq \frac{\ln K}{\eta_i} + \frac{\eta_i n^2}{2} \sum_{u=1}^{U}\sum_{k=1}^{K} \frac{p_{ik,\tau_u}}{q_{ik,u}}. \tag{D.43}$$

Moreover, it can be written that

$$\mathbb{E}_u\left[\sum_{t=\tau_u}^{\tau_{u+1}-1}\mathcal{L}(f_{I_{i,u}}(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}),y_{i,t})\right]=\sum_{t=\tau_u}^{\tau_{u+1}}\sum_{k=1}^{K}p_{ik,\tau_u}\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}),y_{i,t}). \tag{D.44}$$

Considering the facts that (D.14) holds true for p_{ik,τ_u} and $q_{ik,u}$, $\forall k\in[K]$ and $nU=T$, we can conclude that

$$\begin{aligned}&\sum_{u=1}^{U}\mathbb{E}_u\left[\sum_{t=\tau_u}^{\tau_{u+1}-1}\mathcal{L}(f_{I_{i,u}}(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}),y_{i,t})\right]-\sum_{u=1}^{U}\sum_{t=\tau_u}^{\tau_{u+1}}\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}),y_{i,t})\\ \leq&\frac{\ln K}{\eta_i}+\eta_i\mu_i nT\end{aligned} \tag{D.45}$$

which obtains the regret of client i using OFMS-FT when the i-th client communicates with the server every n learning rounds. Similar to $\hat{\ell}_{ik,t}$, define $\hat{\ell}_{ik,u}$ as

$$\hat{\ell}_{ik,u}=\sum_{t=\tau_u}^{\tau_{u+1}-1}\frac{\alpha}{q_{ik,u}}\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}),y_{i,t})\mathcal{I}(i\in\mathbb{D}_u,k\in\mathbb{S}_{i,u}) \tag{D.46}$$

Moreover, substituting $\hat{\ell}_{ik,t}$ with $\hat{\ell}_{ik,u}$ in (D.17) and following the derivation steps from (D.17) to (D.21), we obtain

$$\begin{aligned}&\sum_{i=1}^{N}\hat{\ell}_{ik,u}-\sum_{i=1}^{N}\sum_{t=\tau_u}^{\tau_{u+1}-1}\frac{\alpha}{q_{ik,u}}\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}),y_{i,t})\mathcal{I}(i\in\mathbb{D}_u,k\in\mathbb{S}_{i,u})\\ \leq&\frac{N}{2\eta_f}(\|\boldsymbol{\theta}_{k,u}-\boldsymbol{\theta}\|^2-\|\boldsymbol{\theta}_{k,u+1}-\boldsymbol{\theta}\|^2)+\frac{\eta_f}{2N}\left\|\sum_{i=1}^{N}\nabla\hat{\ell}_{ik,u}\right\|^2.\end{aligned} \tag{D.47}$$

Expected value of $\hat{\ell}_{ik,u}$ and $\|\nabla\hat{\ell}_{ik,u}\|^2$ can be obtained as

$$\begin{aligned}
\mathbb{E}_u[\hat{\ell}_{ik,u}] &= \sum_{t=\tau_u}^{\tau_{u+1}-1} \frac{\alpha}{q_{ik,u}}\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}), y_{i,t}) \times \left(\frac{p_{ik,\tau_u}}{\alpha} + \sum_{\forall j: j\neq k} \frac{p_{ij,\tau_u}}{\alpha m_{ij,u}}\right) \\
&= \sum_{t=\tau_u}^{\tau_{u+1}-1} \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}), y_{i,t}) \qquad \text{(D.48a)} \\
\mathbb{E}_u[\|\nabla\hat{\ell}_{ik,u}\|^2] &= \frac{\alpha^2}{q_{ik,u}^2}\left\|\sum_{t=\tau_u}^{\tau_{u+1}-1} \nabla\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}), y_{i,t})\right\|^2 \times \left(\frac{p_{ik,\tau_u}}{\alpha} + \sum_{\forall j: j\neq k} \frac{p_{ij,\tau_u}}{\alpha m_{ij,u}}\right) \\
&= \frac{\alpha}{q_{ik,u}}\left\|\sum_{t=\tau_u}^{\tau_{u+1}-1} \nabla\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}), y_{i,t})\right\|^2 \\
&\le \frac{\alpha n}{q_{ik,u}} \sum_{t=\tau_u}^{\tau_{u+1}-1} \|\nabla\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}), y_{i,t})\|^2 \le \frac{\alpha n^2 G^2}{q_{ik,u}} \qquad \text{(D.48b)}
\end{aligned}$$

where the last two inequalities in (D.48b) obtained using AM-GM inequality and the assumption that $\|\nabla\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}), y_{i,t})\|^2 \le G^2$. Moreover, using AM-GM inequality and (D.48b), we can write that

$$\mathbb{E}_u\left[\left\|\sum_{i=1}^{N}\nabla\hat{\ell}_{ik,u}\right\|^2\right] \le N\sum_{i=1}^{N}\mathbb{E}_u[\|\nabla\hat{\ell}_{ik,u}\|^2] \le N\sum_{i=1}^{N}\frac{\alpha n^2 G^2}{q_{ik,u}}. \qquad \text{(D.49)}$$

Taking the expectation from both sides of (D.47), we get

$$\begin{aligned}
&\sum_{i=1}^{N}\sum_{t=\tau_u}^{\tau_{u+1}-1} \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}), y_{i,t}) - \sum_{i=1}^{N}\sum_{t=\tau_u}^{\tau_{u+1}-1} \mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}), y_{i,t}) \\
\le& \frac{N}{2\eta_f}(\|\boldsymbol{\theta}_{k,u}-\boldsymbol{\theta}\|^2 - \|\boldsymbol{\theta}_{k,u+1}-\boldsymbol{\theta}\|^2) + \frac{\eta_f}{2}\sum_{i=1}^{N}\frac{\alpha n^2 G^2}{q_{ik,u}}. \qquad \text{(D.50)}
\end{aligned}$$

Following derivation steps from (D.24) to (D.26), using (D.50) we can obtain

$$\begin{aligned}&\sum_{i=1}^{N}\sum_{u=1}^{U}\sum_{t=\tau_u}^{\tau_{u+1}-1}\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}),y_{i,t})-\sum_{i=1}^{N}\sum_{u=1}^{U}\sum_{t=\tau_u}^{\tau_{u+1}-1}\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_k^*),y_{i,t})\\&\leq\frac{N\|\boldsymbol{\theta}_k^*\|^2}{2\eta_f}+\frac{\eta_f}{2}\sum_{i=1}^{N}\sum_{u=1}^{U}\frac{\alpha n^2G^2}{q_{ik,u}}.\end{aligned} \tag{D.51}$$

Considering the fact that $q_{ik,u}\geq\frac{1}{2\mu_i}$ (see (D.14)), using (D.51) we arrive at

$$\begin{aligned}&\sum_{i=1}^{N}\sum_{u=1}^{U}\sum_{t=\tau_u}^{\tau_{u+1}-1}\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_{k,u}),y_{i,t})-\sum_{i=1}^{N}\sum_{u=1}^{U}\sum_{t=\tau_u}^{\tau_{u+1}-1}\mathcal{L}(f_k(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_k^*),y_{i,t})\\&\leq\frac{N\|\boldsymbol{\theta}_k^*\|^2}{2\eta_f}+\sum_{i=1}^{N}\eta_f\alpha\mu_iG^2nT\end{aligned} \tag{D.52}$$

which proves the theorem.

Appendix E

Supplementary Proofs and Experiments for Chapter 5

E.1 Proof of Theorem 5.1

This section provides the proof of Theorem 5.1. The proof follows similar steps to those in [112], and it is included here for the sake of completeness and to make the chapter self-contained.

According to (5.3) and (5.4), for any $\boldsymbol{\theta}$ it can be written that

$$\begin{aligned}\|\boldsymbol{\theta}_{t+1}-\boldsymbol{\theta}\|^2 =& \left\|\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{\phi}_{i,t+1}-\boldsymbol{\theta}\right\|^2 = \left\|\boldsymbol{\theta}_t-\frac{\eta}{N}\sum_{i=1}^{N}\nabla\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t),y_{i,t})-\boldsymbol{\theta}\right\|^2\\ =&\|\boldsymbol{\theta}_t-\boldsymbol{\theta}\|^2+\left\|\frac{\eta}{N}\sum_{i=1}^{N}\nabla\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t),y_{i,t})\right\|^2\\ &-\frac{2\eta}{N}\sum_{i=1}^{N}(\boldsymbol{\theta}_t-\boldsymbol{\theta})^\top\nabla\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t),y_{i,t}).\end{aligned} \tag{E.1}$$

Due to convexity of $\mathcal{L}(\cdot, \cdot)$ it can be concluded that

$$\begin{aligned}
&\frac{2\eta}{N}\sum_{i=1}^{N}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t), y_{i,t}) - \mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}), y_{i,t}) \\
\leq &\frac{2\eta}{N}\sum_{i=1}^{N}(\boldsymbol{\theta}_t - \boldsymbol{\theta})^\top \nabla\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t), y_{i,t}).
\end{aligned} \tag{E.2}$$

Combining (E.1) with (E.2), we get

$$\begin{aligned}
&\frac{2\eta}{N}\sum_{i=1}^{N}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t), y_{i,t}) - \frac{2\eta}{N}\sum_{i=1}^{N}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}), y_{i,t}) \\
\leq &\|\boldsymbol{\theta}_t - \boldsymbol{\theta}\|^2 - \|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}\|^2 + \left\|\frac{\eta}{N}\sum_{i=1}^{N}\nabla\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t), y_{i,t})\right\|^2.
\end{aligned} \tag{E.3}$$

Using assumption A2 and Arithmetic Mean Geometric Mean (AM-GM) inequality it can be written that

$$\left\|\sum_{i=1}^{N}\nabla\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t), y_{i,t})\right\|^2 \leq N\sum_{i=1}^{N}\|\nabla\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t), y_{i,t})\|^2 \leq N^2G^2. \tag{E.4}$$

Combining (E.3) with (E.4), we arrive at

$$\begin{aligned}
&\frac{2\eta}{N}\sum_{i=1}^{N}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t), y_{i,t}) - \frac{2\eta}{N}\sum_{i=1}^{N}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}), y_{i,t}) \\
\leq &\|\boldsymbol{\theta}_t - \boldsymbol{\theta}\|^2 - \|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}\|^2 + \eta^2G^2.
\end{aligned} \tag{E.5}$$

Dividing both sides by $\frac{2\eta}{N}$ yields

$$\begin{aligned}
&\sum_{i=1}^{N}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t), y_{i,t}) - \sum_{i=1}^{N}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}), y_{i,t}) \\
\leq &\frac{N(\|\boldsymbol{\theta}_t - \boldsymbol{\theta}\|^2 - \|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}\|^2)}{2\eta} + \frac{\eta N}{2}G^2.
\end{aligned} \tag{E.6}$$

Summing (E.6) over time, we obtain

$$\begin{aligned}&\sum_{t=1}^{T}\sum_{i=1}^{N}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t),y_{i,t})-\sum_{t=1}^{T}\sum_{i=1}^{N}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}),y_{i,t})\\ \leq&\frac{N(\|\boldsymbol{\theta}_0-\boldsymbol{\theta}\|^2-\|\boldsymbol{\theta}_{T+1}-\boldsymbol{\theta}\|^2)}{2\eta}+\frac{\eta N}{2}G^2T.\end{aligned} \tag{E.7}$$

Plugging in $\boldsymbol{\theta}=\boldsymbol{\theta}^*$ and $\boldsymbol{\theta}_0=\mathbf{0}$ into (E.7) and considering the fact that $\|\boldsymbol{\theta}_{T+1}-\boldsymbol{\theta}\|^2\geq 0$, we find

$$\sum_{t=1}^{T}\sum_{i=1}^{N}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t),y_{i,t})-\sum_{t=1}^{T}\sum_{i=1}^{N}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}^*),y_{i,t})\leq\frac{N\|\boldsymbol{\theta}^*\|^2}{2\eta}+\frac{\eta N}{2}G^2T \tag{E.8}$$

which proves the Theorem.

E.2 Proof of Theorem 5.2

According to (5.10), we can write

$$\begin{aligned}\frac{\alpha_{i,t+1}+\beta_{i,t+1}}{\alpha_{i,t}+\beta_{i,t}}=&\frac{\alpha_{i,t}}{\alpha_{i,t}+\beta_{i,t}}\exp\left(-\eta_c\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t),y_{i,t})\right)\\&+\frac{\beta_{i,t}}{\alpha_{i,t}+\beta_{i,t}}\exp\left(-\eta_c\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\phi}_{i,t}),y_{i,t})\right).\end{aligned} \tag{E.9}$$

Using the inequality $e^{-x}\leq 1-x+\frac{1}{2}x^2, \forall x\geq 0$, from (E.9) we arrive at

$$\begin{aligned}\frac{\alpha_{i,t+1}+\beta_{i,t+1}}{\alpha_{i,t}+\beta_{i,t}}\leq&\frac{\alpha_{i,t}}{\alpha_{i,t}+\beta_{i,t}}\left(1-\eta_c\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t),y_{i,t})+\frac{\eta_c^2}{2}\mathcal{L}^2(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t),y_{i,t})\right)\\&+\frac{\beta_{i,t}}{\alpha_{i,t}+\beta_{i,t}}\left(1-\eta_c\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\phi}_{i,t}),y_{i,t})+\frac{\eta_c^2}{2}\mathcal{L}^2(f(\boldsymbol{x}_{i,t};\boldsymbol{\phi}_{i,t}),y_{i,t})\right).\end{aligned} \tag{E.10}$$

Taking the logarithm from both sides of (E.10) and using the inequality $1 + x \leq e^x$, we get

$$\begin{aligned}\ln(\frac{\alpha_{i,t+1} + \beta_{i,t+1}}{\alpha_{i,t} + \beta_{i,t}}) \leq & \frac{\alpha_{i,t}}{\alpha_{i,t} + \beta_{i,t}} \left(-\eta_c \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t), y_{i,t}) + \frac{\eta_c^2}{2} \mathcal{L}^2(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t), y_{i,t})\right) \\ & + \frac{\beta_{i,t}}{\alpha_{i,t} + \beta_{i,t}} \left(-\eta_c \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}_{i,t}), y_{i,t}) + \frac{\eta_c^2}{2} \mathcal{L}^2(f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}_{i,t}), y_{i,t})\right). \end{aligned} \tag{E.11}$$

Considering the assumption that $0 \leq \mathcal{L}(f(\boldsymbol{x}; \boldsymbol{\theta}), y) \leq 1$, $\forall \boldsymbol{x}, \boldsymbol{\theta}$, (E.11) can be relaxed to

$$\begin{aligned}\ln(\frac{\alpha_{i,t+1} + \beta_{i,t+1}}{\alpha_{i,t} + \beta_{i,t}}) \leq & \frac{\alpha_{i,t}}{\alpha_{i,t} + \beta_{i,t}} \left(-\eta_c \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t), y_{i,t})\right) \\ & + \frac{\beta_{i,t}}{\alpha_{i,t} + \beta_{i,t}} \left(-\eta_c \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}_{i,t}), y_{i,t})\right) + \frac{\eta_c^2}{2}. \end{aligned} \tag{E.12}$$

Summing (E.12) over time, we obtain

$$\begin{aligned}\ln(\frac{\alpha_{i,T+1} + \beta_{i,T+1}}{\alpha_{i,1} + \beta_{i,1}}) \leq & \frac{\alpha_{i,t}}{\alpha_{i,t} + \beta_{i,t}} \left(-\eta_c \sum_{t=1}^{T} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t), y_{i,t})\right) \\ & + \frac{\beta_{i,t}}{\alpha_{i,t} + \beta_{i,t}} \left(-\eta_c \sum_{t=1}^{T} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}_{i,t}), y_{i,t})\right) + \frac{\eta_c^2 T}{2}. \end{aligned} \tag{E.13}$$

According to Hölder's inequality, for any positive real numbers p and q satisfying $\frac{1}{p} + \frac{1}{q} = 1$, the following inequality holds:

$$\frac{\alpha_{i,T+1}}{p} + \frac{\beta_{i,T+1}}{q} \geq \alpha_{i,T+1}^{\frac{1}{p}} \beta_{i,T+1}^{\frac{1}{q}}. \tag{E.14}$$

To meet the condition $\frac{1}{p} + \frac{1}{q} = 1$, it is necessary that $p \geq 1$ and $q \geq 1$. Consequently, (E.14) can be modified to:

$$\alpha_{i,T+1} + \beta_{i,T+1} \geq \alpha_{i,T+1}^{\frac{1}{p}} \beta_{i,T+1}^{\frac{1}{q}}. \tag{E.15}$$

Considering the fact that $\alpha_{i,1} = \beta_{i,1} = 1$, based on (E.15) we can write

$$\ln(\frac{\alpha_{i,T+1} + \beta_{i,T+1}}{\alpha_{i,1} + \beta_{i,1}}) \geq \frac{1}{p}\ln(\alpha_{i,T+1}) + \frac{1}{q}\ln(\beta_{i,T+1}) - \ln(2). \tag{E.16}$$

According to the update rule in (E.9), (E.16) is equivalent to

$$\begin{aligned}\ln(\frac{\alpha_{i,T+1} + \beta_{i,T+1}}{\alpha_{i,1} + \beta_{i,1}}) \geq & -\frac{\eta_c}{p}\sum_{t=1}^{T}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t), y_{i,t}) \\ & -\frac{\eta_c}{q}\sum_{t=1}^{T}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\phi}_{i,t}), y_{i,t}) - \ln(2). \end{aligned}\tag{E.17}$$

Combining (E.13) with (E.17) we arrive at

$$\begin{aligned}&\frac{\alpha_{i,t}}{\alpha_{i,t} + \beta_{i,t}}\sum_{t=1}^{T}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t), y_{i,t}) + \frac{\beta_{i,t}}{\alpha_{i,t} + \beta_{i,t}}\sum_{t=1}^{T}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\phi}_{i,t}), y_{i,t}) \\ &-\frac{1}{p}\sum_{t=1}^{T}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t), y_{i,t}) - \frac{1}{q}\sum_{t=1}^{T}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\phi}_{i,t}), y_{i,t}) \leq \frac{\ln(2)}{\eta_c} + \frac{\eta_c T}{2}.\end{aligned}\tag{E.18}$$

Due to the convexity of $\mathcal{L}(\cdot,\cdot)$, we can write

$$\begin{aligned}\mathcal{L}(f_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t}) =& \mathcal{L}\left(\frac{\alpha_{i,t}}{\alpha_{i,t} + \beta_{i,t}}f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t) + \frac{\beta_{i,t}}{\alpha_{i,t} + \beta_{i,t}}f(\boldsymbol{x}_{i,t};\boldsymbol{\phi}_{i,t}), y_{i,t}\right) \\ \leq & \frac{\alpha_{i,t}}{\alpha_{i,t} + \beta_{i,t}}\sum_{t=1}^{T}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t), y_{i,t}) \\ & + \frac{\beta_{i,t}}{\alpha_{i,t} + \beta_{i,t}}\sum_{t=1}^{T}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\phi}_{i,t}), y_{i,t}).\end{aligned}\tag{E.19}$$

Combining (E.18) with (E.19) we get

$$\begin{aligned}&\sum_{t=1}^{T}\mathcal{L}(f_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t}) - \frac{1}{p}\sum_{t=1}^{T}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t), y_{i,t}) - \frac{1}{q}\sum_{t=1}^{T}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\phi}_{i,t}), y_{i,t}) \\ \leq & \frac{\ln(2)}{\eta_c} + \frac{\eta_c T}{2}\end{aligned}\tag{E.20}$$

Substituting $p = \infty$ and $q = 1$ in (E.20), we obtain

$$\sum_{t=1}^{T} \mathcal{L}(f_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t}) - \sum_{t=1}^{T} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}_{i,t}), y_{i,t}) \leq \frac{\ln(2)}{\eta_c} + \frac{\eta_c T}{2}. \tag{E.21}$$

Since Fed-POE updates $\boldsymbol{\phi}_{i,t}$ locally using online gradient descent as outlined in (5.8), according to (5.6), for any $\boldsymbol{\phi}$ we can write

$$\sum_{t=1}^{T} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}_{i,t}), y_{i,t}) - \sum_{t=1}^{T} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}), y_{i,t}) \leq \frac{\|\boldsymbol{\phi}\|^2}{2\eta} + \frac{\eta}{2} G^2 T. \tag{E.22}$$

Substituting $\boldsymbol{\phi}_i^*$ in (E.22) along with combining (E.21) with (E.22), we can conclude that

$$\sum_{t=1}^{T} \mathcal{L}(f_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t}) - \sum_{t=1}^{T} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\phi}_i^*), y_{i,t}) \leq \frac{\|\boldsymbol{\phi}_i^*\|^2}{2\eta} + \frac{\ln(2)}{\eta_c} + \frac{\eta}{2} G^2 T + \frac{\eta_c T}{2} \tag{E.23}$$

which proves the personalized regret upper bound of Fed-POE in (5.12). Furthermore, plunging in $p = 1$ and $q = \infty$ into (E.20), we get

$$\sum_{t=1}^{T} \mathcal{L}(f_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t}) - \sum_{t=1}^{T} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t), y_{i,t}) \leq \frac{\ln(2)}{\eta_c} + \frac{\eta_c T}{2}. \tag{E.24}$$

Summing (E.24) over all clients, we obtain

$$\sum_{t=1}^{T} \sum_{i=1}^{N} \mathcal{L}(f_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t}) - \sum_{t=1}^{T} \sum_{i=1}^{N} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t), y_{i,t}) \leq \frac{N \ln(2)}{\eta_c} + \frac{\eta_c N T}{2}. \tag{E.25}$$

Combining (E.25) with (E.8), we arrive at

$$\begin{aligned} &\sum_{t=1}^{T} \sum_{i=1}^{N} \mathcal{L}(f_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t}) - \sum_{t=1}^{T} \sum_{i=1}^{N} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}^*), y_{i,t}) \\ \leq &\frac{N\|\boldsymbol{\theta}^*\|^2}{2\eta} + \frac{N \ln(2)}{\eta_c} + \frac{\eta N}{2} G^2 T + \frac{\eta_c N T}{2} \end{aligned} \tag{E.26}$$

which proves the Theorem.

E.3 Proof of Theorem 5.3

According to assumption (A3) that $0 \leq \mathcal{L}(f(\boldsymbol{x};\boldsymbol{\theta}), y) \leq 1$, it can be written that

$$\frac{1}{N}\sum_{t=1}^{U}\sum_{i=1}^{N}\mathcal{L}(f_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t}) - \frac{1}{N}\sum_{t=1}^{U}\sum_{i=1}^{N}\mathcal{L}(h^*(\boldsymbol{x}_{i,t}), y_{i,t}) \leq U. \tag{E.27}$$

Let $\ell_{ij,t}$ denote the importance sampling loss estimate, which is expressed as

$$\ell_{ij,t} = \frac{\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\rho}_j), y_{i,t})}{q_{ij,t}}\mathbf{1}_{j\in\mathbb{M}_{i,t}}. \tag{E.28}$$

Let the total number of model parameters stored by the server after time step U is D and $W_{i,t} = \sum_{j=1}^{D} w_{ij,t}$. For any $t > U$, considering (5.17), we can write

$$\frac{W_{i,t+1}}{W_{i,t}} = \sum_{j=1}^{D}\frac{w_{ij,t}}{W_{i,t}}\exp(-\eta_c\ell_{ij,t}) = \sum_{j=1}^{D}p_{ij,t}\exp(-\eta_c\ell_{ij,t}). \tag{E.29}$$

Employing the inequality $e^{-x} \leq 1 - x + \frac{1}{2}x^2, \forall x \geq 0$, from (E.29) we obtain

$$\frac{W_{i,t+1}}{W_{i,t}} \leq \sum_{j=1}^{D}p_{ij,t}(1 - \eta_c\ell_{ij,t} + \frac{\eta_c^2}{2}\ell_{ij,t}^2). \tag{E.30}$$

Taking the logarithm from both sides of (E.30) and using the inequality $1 + x \leq e^x$, we arrive at

$$\ln\frac{W_{i,t+1}}{W_{i,t}} \leq \sum_{j=1}^{D}p_{ij,t}(-\eta_c\ell_{ij,t} + \frac{\eta_c^2}{2}\ell_{ij,t}^2). \tag{E.31}$$

Summing (E.31) over time, we get

$$\ln\frac{W_{i,T+1}}{W_{i,U}} \leq \sum_{t=U}^{T}\sum_{j=1}^{D}p_{ij,t}(-\eta_c\ell_{ij,t} + \frac{\eta_c^2}{2}\ell_{ij,t}^2). \tag{E.32}$$

Considering the fact that the weights $\{w_{ij,t}\}_{j=1}^{D}$ are initialized as $w_{ij,1} = 1$, $\forall j \in [D]$, it can concluded that $W_{i,U} \leq D$. Theefore, for any $j \in [D]$, the left hand side of (E.32) is bounded from below as

$$\ln \frac{W_{i,T+1}}{W_{i,U}} \geq \ln \frac{w_{ij,T+1}}{W_{i,U}} \geq \ln \frac{w_{ij,T+1}}{D} = -\sum_{t=U}^{T} \eta_c \ell_{ij,t} - \ln D. \tag{E.33}$$

Combining (E.33) with (E.32) yields

$$\sum_{t=U}^{T} \sum_{j=1}^{D} p_{ij,t} \ell_{ij,t} - \sum_{t=U}^{T} \ell_{ij,t} \leq \frac{\ln D}{\eta_c} + \frac{\eta_c}{2} \sum_{t=U}^{T} \sum_{j=1}^{D} p_{ij,t} \ell_{ij,t}^2. \tag{E.34}$$

The expected values of $\ell_{ij,t}$ and $\ell_{ij,t}^2$ given observed losses up until time step t can be obtained as

$$\mathbb{E}_t[\ell_{ij,t}] = \frac{\mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\rho}_j), y_{i,t})}{q_{ij,t}} \mathbb{E}_t[\mathbf{1}_{j \in \mathbb{M}_{i,t}}] = \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\rho}_j), y_{i,t}) \tag{E.35a}$$

$$\mathbb{E}_t[\ell_{ij,t}^2] = \frac{\mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\rho}_j), y_{i,t})^2}{q_{ij,t}^2} \mathbb{E}_t[\mathbf{1}_{j \in \mathbb{M}_{i,t}}] = \frac{\mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\rho}_j), y_{i,t})^2}{q_{ij,t}} \leq \frac{1}{q_{ij,t}}. \tag{E.35b}$$

Taking the expectation from (E.34), we arrive at

$$\sum_{t=U}^{T} \sum_{j=1}^{D} p_{ij,t} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\rho}_j), y_{i,t}) - \sum_{t=U}^{T} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\rho}_j), y_{i,t}) \leq \frac{\ln D}{\eta_c} + \frac{\eta_c}{2} \sum_{t=U}^{T} \sum_{j=1}^{D} \frac{p_{ij,t}}{q_{ij,t}}. \tag{E.36}$$

Since $q_{ij,t} = 1 - (1 - p_{ij,t})^M = p_{ij,t}(1 + (1 - p_{ij,t}) + (1 - p_{ij,t})^2 + \ldots + (1 - p_{ij,t})^{M-1})$, it can be concluded that $q_{ij,t} \geq p_{ij,t}$. Therefore, (E.36) can be relaxed to

$$\sum_{t=U}^{T} \sum_{j=1}^{D} p_{ij,t} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\rho}_j), y_{i,t}) - \sum_{t=U}^{T} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\rho}_j), y_{i,t}) \leq \frac{\ln D}{\eta_c} + \frac{\eta_c}{2} D(T - U). \tag{E.37}$$

According to model selection procedure adopted by Fed-POE presented in Algorithm 13, client i chooses a subset of models by sampling them in M rounds with replacement. Let $a_{ij,t} \geq 0$ denote the number of times that the model j in $\mathbb{D}_t$ is chosen by client i at time step t. The number of different situations for selected subset of models $\mathbb{M}_{i,t}$ is equal to the number of solutions for the linear equation

$$a_{i1,t} + \ldots + a_{iD,t} = M, a_{ij,t} \geq 0, \forall j \in [D]. \tag{E.38}$$

Let $\mathbb{A}$ denote the set of all possible solutions for (E.38) such that if $\boldsymbol{a} \in \mathbb{A}$ where $\boldsymbol{a} = [a_1, \ldots, a_D]$, $a_1, \ldots, a_D$ satisfies (E.38). Therefore, for the expected loss of the ensemble $\tilde{f}_{i,t}(\boldsymbol{x}_{i,t})$ in (5.14), we can write

$$\mathbb{E}_t[\mathcal{L}(\tilde{f}_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t})] = \sum_{k=1}^{|\mathbb{A}|} \prod_{j=1}^{D} p_{ij,t}^{a_{j,k}} \mathcal{L}(\tilde{f}_{i,t}^{(k)}(\boldsymbol{x}_{i,t}), y_{i,t}) \tag{E.39}$$

where $\tilde{f}_{i,t}^{(k)}(\boldsymbol{x}_{i,t})$ denote the k-th possible ensemble model generated by client i using Fed-POE. Using the Jensen inequality and convexity of the loss function, we can relax (E.39) to

$$\begin{aligned}\mathbb{E}_t[\mathcal{L}(\tilde{f}_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t})] &= \sum_{k=1}^{|\mathbb{A}|} \left(\prod_{j=1}^{D} p_{ij,t}^{a_{j,k}}\right) \mathcal{L}(\tilde{f}_{i,t}^{(k)}(\boldsymbol{x}_{i,t}), y_{i,t}) \\ &\leq \sum_{k=1}^{|\mathbb{A}|} \left(\prod_{j=1}^{D} p_{ij,t}^{a_{j,k}}\right) \sum_{m \in \mathbb{M}_{i,t}^{(k)}} \frac{w_{im,t}}{W_{i,t}^{(k)}} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\rho}_j), y_{i,t})\end{aligned} \tag{E.40}$$

where $\mathbb{M}_{i,t}^{(k)}$ and $W_{i,t}^{(k)}$ are the k-th possible model subset and weight summations, respectively. Rearranging the right hand side of (E.40), we can write

$$\mathbb{E}_t[\mathcal{L}(\tilde{f}_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t})] \leq \sum_{j=1}^{D} p_{ij,t} \sum_{k=1}^{|\mathbb{B}_j|} \left(\prod_{m=1}^{D} p_{im,t}^{b_{m,k}}\right) \frac{w_{ij,t}}{W_{i,t}^{(k)}} \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\rho}_j), y_{i,t}) \tag{E.41}$$

where $\mathbb{B}_j$ is the set of all possible solutions for the linear equation in (E.38) condition on $a_{ij,t} \geq 1$. Since for any k, we have $w_{ij,t} \leq W_{i,t}^{(k)}$, (E.41) can be relaxed to

$$\mathbb{E}_t[\mathcal{L}(\tilde{f}_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t})] \leq \sum_{j=1}^{D} p_{ij,t} \sum_{k=1}^{|\mathbb{B}_j|} \left(\prod_{m=1}^{D} p_{im,t}^{b_{m,k}}\right) \mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\rho}_j), y_{i,t}). \tag{E.42}$$

Since $\sum_{k=1}^{|\mathbb{B}_j|}\left(\prod_{m=1}^{D} p_{im,t}^{b_{m,k}}\right)$ includes all possibilities in $\mathbb{B}_j$, we can conclude that $\sum_{k=1}^{|\mathbb{B}_j|}\left(\prod_{m=1}^{D} p_{im,t}^{b_{m,k}}\right) = 1$. Combining this with (E.42), we obtain

$$\mathbb{E}_t[\mathcal{L}(\tilde{f}_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t})] \leq \sum_{j=1}^{D} p_{ij,t}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\rho}_j), y_{i,t}). \tag{E.43}$$

Combining (E.43) with (E.37), we arrive at

$$\sum_{t=U}^{T}\mathbb{E}_t[\mathcal{L}(\tilde{f}_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t})] - \sum_{t=U}^{T}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\rho}_j), y_{i,t}) \leq \frac{\ln D}{\eta_c} + \frac{\eta_c}{2}D(T-U). \tag{E.44}$$

Since $0 \leq \mathcal{L}(f(\boldsymbol{x};\boldsymbol{\theta}), y) \leq 1$, it can be written that

$$\sum_{t=1}^{U}\mathbb{E}_t[\mathcal{L}(\tilde{f}_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t})] - \sum_{t=1}^{U}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\rho}_j), y_{i,t}) \leq U. \tag{E.45}$$

Combining (E.45) with (E.44), we get

$$\sum_{t=1}^{T}\mathbb{E}_t[\mathcal{L}(\tilde{f}_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t})] - \sum_{t=1}^{T}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\rho}_j), y_{i,t}) \leq \frac{\ln D}{\eta_c} + \frac{\eta_c}{2}D(T-U) + U. \tag{E.46}$$

Since $\bar{f}_{i,t}(\cdot)$ similar to $f_{i,t}(\cdot)$ is the ensemble of two models, following the same derivation steps from (E.9) to (E.21) by substituting $\bar{f}_{i,t}(\boldsymbol{x}_{i,t})$, $f_{i,t}(\boldsymbol{x}_{i,t})$ and $\tilde{f}_{i,t}(\boldsymbol{x}_{i,t})$ with $f_{i,t}(\boldsymbol{x}_{i,t})$, $f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t)$ and $f(\boldsymbol{x}_{i,t};\boldsymbol{\phi}_{i,t})$, respectively, we can conclude that

$$\sum_{t=1}^{T}\mathcal{L}(\bar{f}_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t}) - \sum_{t=1}^{T}\mathcal{L}(f_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t}) \leq \frac{\ln(2)}{\eta_c} + \frac{\eta_c T}{2}, \tag{E.47a}$$

$$\sum_{t=1}^{T}\mathcal{L}(\bar{f}_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t}) - \sum_{t=1}^{T}\mathcal{L}(\tilde{f}_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t}) \leq \frac{\ln(2)}{\eta_c} + \frac{\eta_c T}{2}. \tag{E.47b}$$

Taking the expectation from both sides of (E.47b) with respect to randomization in model selection along with combining (E.47b) with (E.46) yields

$$\begin{aligned}&\sum_{t=1}^{T}\mathbb{E}_t[\mathcal{L}(\bar{f}_{i,t}(\boldsymbol{x}_{i,t}),y_{i,t})]-\sum_{t=1}^{T}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\rho}_j),y_{i,t})\\ \leq&\frac{\ln 2D}{\eta_c}+\frac{\eta_c}{2}(D+1)T+(1-\frac{\eta_c}{2}D)U.\end{aligned}\tag{E.48}$$

Furthermore, combining (E.47a) with (E.21) and (E.24) and taking the expectation with respect to model selection randomization, we obtain

$$\sum_{t=1}^{T}\mathbb{E}_t[\mathcal{L}(\bar{f}_{i,t}(\boldsymbol{x}_{i,t}),y_{i,t})]-\sum_{t=1}^{T}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\phi}_{i,t}),y_{i,t})\leq\frac{\ln(4)}{\eta_c}+\eta_cT,\tag{E.49a}$$

$$\sum_{t=1}^{T}\mathbb{E}_t[\mathcal{L}(\bar{f}_{i,t}(\boldsymbol{x}_{i,t}),y_{i,t})]-\sum_{t=1}^{T}\mathcal{L}(f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t),y_{i,t})\leq\frac{\ln(4)}{\eta_c}+\eta_cT.\tag{E.49b}$$

Recall that $h_j(\cdot)$ associated with the j-th model parameter in $\mathbb{D}_t$ be defined as $h_j(\boldsymbol{x}_{i,t})=f(\boldsymbol{x}_{i,t};\boldsymbol{\rho}_j)$ while $h_{\text{loc}}(\cdot)$ and $h_{\text{fed}}(\cdot)$ correspond to the local and federated models, respectively, defined as $h_{\text{loc}}(\boldsymbol{x}_{i,t})=f(\boldsymbol{x}_{i,t};\boldsymbol{\phi}_{i,t})$ and $h_{\text{fed}}(\boldsymbol{x}_{i,t})=f(\boldsymbol{x}_{i,t};\boldsymbol{\theta}_t)$. Also recall that $\mathbb{H}:=\{h_j\mid\forall j:1\leq j\leq|\mathbb{D}_T|\}\cup\{h_{\text{loc}},h_{\text{fed}}\}$. Comparing the right hand sides of (E.49a) and (E.49b) with that of (E.48) and considering the fact that $D\geq 2$, for any $h\in\mathbb{H}$ we can write

$$\begin{aligned}&\sum_{t=1}^{T}\mathbb{E}_t[\mathcal{L}(\bar{f}_{i,t}(\boldsymbol{x}_{i,t}),y_{i,t})]-\sum_{t=1}^{T}\mathcal{L}(h(\boldsymbol{x}_{i,t}),y_{i,t})\\ \leq&\frac{\ln 2D}{\eta_c}+\frac{\eta_c}{2}(D+1)T+(1-\frac{\eta_c}{2}D)U.\end{aligned}\tag{E.50}$$

By substituting $h(\cdot)$ with $h_i^*(\cdot)$ as defined in (5.18b) and considering the fact that $D\leq U/n$, we obtain the personalized regret upper bound for client i as shown in (5.20). Moreover, taking the average of (E.50) across clients and substituting $h(\cdot)$ with $h^*(\cdot)$, we arrive at

$$\begin{aligned}&\frac{1}{N}\sum_{t=1}^{T}\sum_{i=1}^{N}\mathbb{E}_t[\mathcal{L}(\bar{f}_{i,t}(\boldsymbol{x}_{i,t}), y_{i,t})] - \frac{1}{N}\sum_{t=1}^{T}\sum_{i=1}^{N}\mathcal{L}(h^*(\boldsymbol{x}_{i,t}), y_{i,t})\\ \leq&\frac{\ln 2D}{\eta_c} + \frac{\eta_c}{2}(D+1)T + (1-\frac{\eta_c}{2}D)U \end{aligned} \tag{E.51}$$

which proves the theorem.

E.4 Supplementary Experimental Details

This section presents supplementary experimental results and details about experimental setup. All experiments were carried out using Intel(R) Core(TM) i7-10510U CPU @ 1.80 GHz 2.30 GHz processor with a 64-bit Windows operating system.

E.4.1 Regression Data Distribution

As it is pointed out in section 5.5, the present chapter tests the performance of algorithms for online regression task on Air and WEC datasets. These datasets are downloaded from UCI Machine Learning Repository [84]. Each data sample in Air dataset includes air quality information such as concentration of some chemicals in the air. Data samples in Air dataset collected from four different geographical locations. Moreover, data samples in WEC, collected from wave energy converters in four different geographical locations. In order to distribute data, clients are partitioned into 4 groups. For each group, 70% of data samples observed by each client in the group belongs to a specific geographical location while 10% of observed data samples belong to each of the rest 3 locations.

E.4.2 Random Feature Kernel-based Models

As it is pointed in section 5.5, the proposed Fed-POE and all baselines utilize a random feature kernel-based model to perform online regression task. In what follows we explain random feature-based kernel models. Let $\kappa(\cdot, \cdot)$ be a positive-definite function called kernel such that $\kappa(\boldsymbol{x}, \boldsymbol{x}')$ measures the similarity between $\boldsymbol{x}$ and $\boldsymbol{x}'$. In online kernel learning context, at time step $t+1$, the following prediction is made for $\boldsymbol{x}$ (see e.g. [148, 77, 131]):

$$f_\kappa(\boldsymbol{x}; \boldsymbol{\alpha}_t) = \sum_{\tau=1}^{t} \sum_{i=1}^{N} \alpha_{i,\tau} \kappa(\boldsymbol{x}, \boldsymbol{x}_{i,\tau}) \tag{E.52}$$

where $\boldsymbol{\alpha}_t = [\alpha_{1,1}, \dots, \alpha_{N,1}, \dots, \alpha_{1,t}, \dots, \alpha_{N,t}]$ denotes the learnable parameters. Therefore, the number of parameters that should be learned grows with time. In order to alleviate the computational complexity of online kernel learning, random feature approximation [123] can be employed. In fact, using random feature approximation, the number of parameters that needs be learned is time-invariant and is selected by the algorithm. Assume that $\kappa(\cdot)$ is a shift-invariant kernel function such that $\kappa(\boldsymbol{x}, \boldsymbol{x}') = \kappa(\boldsymbol{x} - \boldsymbol{x}')$. Also, suppose that $\kappa(\cdot)$ is scaled such that $\kappa(\mathbf{0}) = 1$. Let $\xi(\cdot)$ denotes the Fourier transform of $\kappa(\cdot)$. According to definition of inverse Fourier transform $\kappa(\mathbf{0}) = \int_{-\infty}^{\infty} \xi(\boldsymbol{\omega}) d\boldsymbol{\omega} = 1$. Therefore, it can be concluded that $\xi(\cdot)$ is a probability density function (PDF). Let $\boldsymbol{\omega}_1, \dots, \boldsymbol{\omega}_D$ be drawn randomly from $\xi(\cdot)$ and called random features. Using the random features $\boldsymbol{\omega}_1, \dots, \boldsymbol{\omega}_D$, the representation $\boldsymbol{z}(\boldsymbol{x})$ is defined as

$$\boldsymbol{z}(\boldsymbol{x}) = \frac{1}{\sqrt{D}} [\sin(\boldsymbol{\omega}_1^\top \boldsymbol{x}), \dots, \sin(\boldsymbol{\omega}_D^\top \boldsymbol{x}), \cos(\boldsymbol{\omega}_1^\top \boldsymbol{x}), \dots, \cos(\boldsymbol{\omega}_D^\top \boldsymbol{x})]. \tag{E.53}$$

Given random features $\boldsymbol{\omega}_1, \dots, \boldsymbol{\omega}_D$ and using the proposed Fed-POE, at time step t, client i makes prediction $f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t) = \boldsymbol{\theta}_t^\top \boldsymbol{z}(\boldsymbol{x}_{i,t})$. Clients and the server employ the proposed Fed-POE to learn the parameter $\boldsymbol{\theta}_t$. As it can be inferred since the model $f(\cdot; \cdot)$ is linear with respect to model parameter $\boldsymbol{\theta}_t$, using convex loss functions, the loss $\mathcal{L}(f(\boldsymbol{x}_{i,t}; \boldsymbol{\theta}_t), y_{i,t})$ is convex as well.

Furthermore, in section 5.5, the proposed Fed-POE utilizes three Gaussian kernels for online regression on Air and WEC datasets. In order to implement multi-kernel learning for Fed-POE, the prediction of kernels are linearly combined and the weights for linear combination is learned locally by each client. Let $f_{0.1}(\boldsymbol{x}_{i,t})$, $f_1(\boldsymbol{x}_{i,t})$ and $f_{10}(\boldsymbol{x}_{i,t})$ represent predictions of Gaussian kernels with variances of 0.1, 1 and 10, respectively. Then at time step t, client i makes prediction $w_{0.1,it}f_{0.1}(\boldsymbol{x}_{i,t}) + w_{1,it}f_1(\boldsymbol{x}_{i,t}) + w_{10,it}f_{10}(\boldsymbol{x}_{i,t})$. In order to update weights $w_{0.1,it}$, $w_{1,it}$ and $w_{10,it}$, client i employs multiplicative update rule. As an example after observing the loss $\mathcal{L}(f_1(\boldsymbol{x}_{i,t}), y_{i,t})$, client i updates $w_{1,it}$ as $w_{1,i(t+1)} = w_{1,it}\exp(-\gamma_i\mathcal{L}(f_1(\boldsymbol{x}_{i,t}), y_{i,t}))$ where γ_i is a learning rate specified by client i.

E.4.3 Image Classification Experimental Setup

The pre-trained CNN used by Fed-POE and other baselines is biased toward class label 0. For CIFAR-10, the pre-trained CNN is trained on a subset of the CIFAR-10 training data, consisting of $5,000$ samples with label 0 and 500 samples from each of the other 9 class labels. For FMNIST, the model is trained on a subset of the FMNIST training data, consisting of $6,000$ samples with label 0 and 500 samples from each of the other 9 class labels. The CNN models are trained using Tensorflow 2.16.1. We used the SGD optimizer with a learning rate of 10^{-3} and momentum of 0.9. The models for CIFAR-10 and FMNIST were trained for 100 epochs and 10 epochs, respectively.

Clients receive test data samples sequentially and make prediction for the newly received sample. To distribute test data samples of CIFAR-10 among clients, we split clients into 10 groups. For CIFAR-10, 55% of samples observed by a client belongs to a specific class label while only 5% of received samples belong to each of other 9 class labels. For FMNIST, client data distribution is time-variant. Since the number of test samples is $10,000$ and the number of clients is 20, it can be concluded that time horizon T is 500. In the first half of time steps (i.e. $t \leq 250$), each client observes 200 samples from the first 5 class labels and 50 samples from other 5 class labels. In the second half, this is reversed: clients observe 200 samples from the last 5 class labels and 50 samples from the other class labels. In each half, each client is biased toward one of the five majority classes. For example, if a client is biased toward class 0 in the first half, it observes 100 samples from class label 0, 25 samples from each of class labels 1 to 4, and 10 samples from each of class labels 5 to 9. Therefore, in each half, each client observes 100 samples from one class, 25 samples from each of four other classes, and 10 samples from each of the remaining five classes.

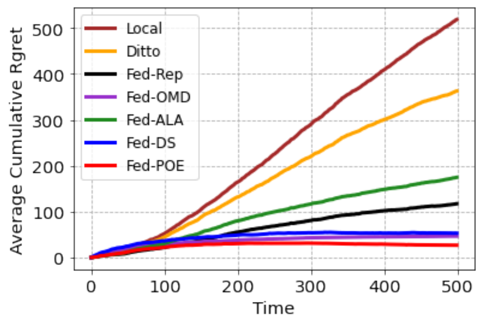
To implement Fed-OMD for both regression and image classification, we used the ℓ_2-norm as a regularizer function for mirror descent. For implementing Ditto for both regression and image classification, we set the regularization factor λ to 1. In the case of image classification using Fed-Rep, clients locally fine-tune the last two layers of the CNN model, while the rest of the network is used as the global backbone to generate representations. Furthermore, Fed-POE is compatible with any federated learning method and can utilize any federated algorithm. For CIFAR-10, Fed-POE uses Fed-OMD, while for FMNIST, Fed-POE uses Fed-Rep. To fine-tune the CNN model, Fed-POE and all baselines use the SGD optimizer and the cross-entropy loss function.

E.4.4 Supplementary Results

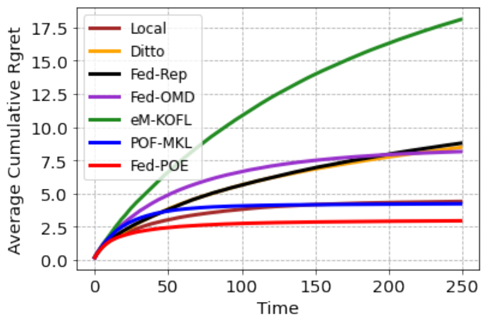
Table E.1 presents the accuracy of clients for image classification using Fed-POE with varying values of M. As can be seen for both CIFAR-10 and FMNIST, when $M > 0$, the accuracy is higher than in the case where $M = 0$. The case where $M = 0$ corresponds to clients not using models stored by the server in their ensemble. Therefore, these results show that constructing the ensemble using previous federated model parameters stored by the server improves the

Table E.1: Average accuracy and standard deviation across clients employing Fed-POE for image classification with varying values of M

Datasets	$M = 0$	$M = 4$	$M = 8$	$M = 16$
CIFAR-10	$65.55\% \pm 8.77\%$	$66.50\% \pm 8.00\%$	$\mathbf{66.54\% \pm 8.08\%}$	$66.46\% \pm 7.98\%$
FMNIST	$79.03\% \pm 1.76\%$	$79.12\% \pm 1.87\%$	$\mathbf{79.23\% \pm 1.88\%}$	$79.18\% \pm 1.85\%$



(a) CIFAR-10



(b) WEC

Figure E.1: Cumulative regret over time on CIFAR-10 and WEC datasets.

accuracy of Fed-POE. This indicates the effectiveness of the model selection procedure of Fed-POE presented in Algorithm 13. Additionally, these results show that increasing M does not necessarily lead to further accuracy improvement. This implies the effectiveness of Fed-POE's model selection in pruning model parameters from the ensemble that have relatively lower accuracy. Figure E.1 illustrates the average cumulative global regret of clients over time using Fed-POE and all other baselines. As depicted, Fed-POE achieves sublinear regret for both CIFAR-10 and WEC datasets. This corroborates the theoretical results in Theorems 5.2 and 5.3.