Fisher Information-based Efficient Curriculum Federated Learning with Large Language Models

Ji Liu, Jiaxiang Ren, Ruoming Jin, Zijie Zhang, Yang Zhou, Patrick Valduriez, Dejing Dou

► **To cite this version:**

HAL Id: lirmm-04734309

https://hal-lirmm.ccsd.cnrs.fr/lirmm-04734309v1

Submitted on 13 Oct 2024

Fisher Information-based Efficient Curriculum Federated Learning with Large Language Models

Ji Liu[1†*], Jiaxiang Ren[2†], Ruoming Jin[3], Zijie Zhang[4],
Yang Zhou[2*], Patrick Valduriez[5], Dejing Dou[6]

[1]HiThink Research, Hangzhou, Zhejiang, China,

[2]Auburn University, Auburn, United States, [3]Kent State University, Kent, United States,

[4]University of Texas at San Antonio, San Antonio, United States,

[5]Inria, University of Montpellier, CNRS, LIRMM, France, and LNCC, Petropolis, Brazil,

[6]Fudan University, Shanghai, China, and BEDI Cloud, Beijing, China

Abstract

As a promising paradigm to collaboratively train models with decentralized data, Federated Learning (FL) can be exploited to fine-tune Large Language Models (LLMs). While LLMs correspond to huge size, the scale of the training data significantly increases, which leads to tremendous amounts of computation and communication costs. The training data is generally non-Independent and Identically Distributed (non-IID), which requires adaptive data processing within each device. Although Low-Rank Adaptation (LoRA) can significantly reduce the scale of parameters to update in the fine-tuning process, it still takes unaffordable time to transfer the low-rank parameters of all the layers in LLMs. In this paper, we propose a Fisher Information-based Efficient Curriculum Federated Learning framework (FibecFed) with two novel methods, i.e., adaptive federated curriculum learning and efficient sparse parameter update. First, we propose a fisher information-based method to adaptively sample data within each device to improve the effectiveness of the FL fine-tuning process. Second, we dynamically select the proper layers for global aggregation and sparse parameters for local update with LoRA so as to improve the efficiency of the FL fine-tuning process. Extensive experimental results based on 10 datasets demonstrate that FibecFed yields excellent performance (up to 45.35% in terms of accuracy) and superb fine-tuning speed (up to 98.61% faster) compared with 17 baseline approaches). Our code will be publicly available.

1 Introduction

As a promising paradigm to collaboratively train models with decentralized data, Federated Learning (FL) can be exploited to fine-tune Large Language Models (LLMs) without aggregating the raw

data from a large number of devices (Fan et al., 2023; Kuang et al., 2023; Che et al., 2023a; Liu et al., 2024b; Che et al., 2023b; Liu et al., 2022b,a; Zhou et al., 2022). A number of stringent legal regulations (Official Journal of the European Union, 2016; Californians for Consumer Privacy, 2020) have been set up in order to protect the security and the privacy of personal data, which hinders the aggregation of the decentralized raw data. FL typically utilizes a parameter server (Li et al., 2014; Liu et al., 2024c, 2023a,b) to aggregate the distributed model updates in devices, which only transfers the parameters or the gradients of the updated models in replace of the raw personal data. By leveraging the distributed raw data of end users, Large Language Models (LLMs) can be trained on devices with excellent performance (Zhao et al., 2024).

While ChatGPT (OpenAI, 2022) has achieved remarkable progress, LLMs (Touvron et al., 2023; Jiang et al., 2023; Du et al., 2022; Zeng et al., 2023; Zhang et al., 2024) have attracted extensive attention. The size of LLMs ranges from several million parameters, e.g., RoBERTa$_{\text{LARGE}}$ (Liu et al., 2020), to several hundreds billion parameters (Wang et al., 2019). While the large scale brings strong capability in various Natural Language Processing (NLP) tasks (Zhao et al., 2023b), the pre-training and the fine-tuning process of LLMs significant communication and computation costs (et al., 2023).

Two types of parameter-efficient approaches exist for reducing the number of parameters within the fine-tuning process of LLMs. Prompt tuning (Liu et al., 2021; Lester et al., 2021; Liu et al., 2022d) can dynamically adjust the prompts to fine-tune LLMs with only a few trainable parameters, which may introduce performance degradation. Although Low-Rank Adaptation (LoRA) (Hu et al., 2021) can significantly reduce the scale of parameters to update in the fine-tuning process of LLMs so as to enable the training of LLMs on edge devices (Xu et al., 2024), it still takes unaffordable time to

---

† Equal contribution.
* Corresponding author: jiliuwork@gmail.com, yangzhou@auburn.edu

update the low-rank parameters of all the layers in LLMs when dealing with decentralized data.

While being an effective method to improve the efficiency and effectiveness of training process, curriculum learning (Bengio et al., 2009) is exploited to train large-scale models (Li et al., 2022a). Inspired by the learning strategy of starting small (Elman, 1993), the curriculum training process starts with easier data and then gradually increase the difficulty. Instead of randomly sampling the batch from training dataset, curriculum learning allows the model to gradually learn from easy samples to hard samples during the training or the fine-tuning process. Existing approaches generally measure the complexity of samples based on heuristic methods (Li et al., 2024) or a simple mode-based method (Xu et al., 2022), both of which cannot provide an accurate estimation of difficulty of data samples and cannot be directly applied in FL. In the context of FL, the data is generally non-Independent and Identically Distributed (non-IID), which requires an adaptive difficulty evaluation approach for diverse devices (Vahidian et al., 2023).

Model compression methods, e.g., pruning (Wu et al., 2021; Liu et al., 2024d; Zhang et al., 2022) or sparse training (Bibikar et al., 2022), are exploited in FL to reduce computation and communication costs. Sparse training can achieve personalization so as to further improve the performance of FL (Liu et al., 2023c; Dai et al., 2022). However, the pruning or sparse training incurs severe accuracy degradation due to lossy strategies or simple component (neuron) selection mechanisms. In addition, the model can be split into two parts, i.e., the server part and the device part, in order to achieve both the generalization and personalization capability (Han et al., 2023).

Fisher information can be exploited to accelerate the training process of LLMs (Ollivier, 2015; Martens and Grosse, 2015; Osawa et al., 2023). Fisher information is defined as the amount of information carried by a random variable corresponding to some unknown parameters (Duy et al., 2022). As a measure of the local curvature (Martens, 2020), Fisher Information Matrix (FIM) defines the Riemannian metric of the parameter space (Karakida et al., 2019), which can indicate the difficulty of data samples and the importance of each component of the network along with the generalization performance (Jastrzebski et al., 2021).

In this paper, we propose FibecFed, i.e., a Fisher Information-based Efficient Curriculum Federated Learning framework. FibecFed is composed of two novel methods, i.e., adaptive federated curriculum learning and efficient sparse parameter update. To the best of our knowledge, we are among the first to exploit the fisher information to perform curriculum learning and sparse training at the same time within FL settings. We summarize out major contributions as follows:

- We propose an adaptive federated curriculum learning method to sample easy data samples first and to gradually improve the difficulty of samples so as to improve the effectiveness of the FL fine-tuning process. We exploit a fisher information-based method to measure the difficulty of training data within each device.

- We propose an efficient sparse parameter update method to select proper layers for global aggregation and to adaptively update sparse parameters to achieve excellent efficiency and effectiveness. We utilize fisher information to evaluate the importance of diverse components of LLMs and propose a lossless method for global aggregation and local update.

- We conduct extensive experimentation to validate our approach using 10 datasets. The experimental results reveal that FibecFed significantly outperforms 17 baseline approaches in terms of accuracy (up to 45.35% higher) and fine-tuning speed (up to 98.61% faster).

The rest of the paper is organized as follows. The related work is presented in Section 2. We formulate the problem to address in Section 3. We present the architecture of FibecFed and propose the adaptive federated curriculum learning and the efficient sparse parameter update method in Section 4. We demonstrate the experimental results in Section 5. Finally, Section 6 concludes.

2 Related Word & Preliminaries

Inspired by the learning strategy of starting small (Elman, 1993), curriculum learning (Bengio et al., 2009) is exploited in large-scale model training (Li et al., 2022a). Existing works measure the complexity of samples based on static characteristics of data samples, e.g., sequence length (Li et al., 2024; Platanios et al., 2019). Although a simple global mode-based method is proposed to predict the performance improvement based on several training

states (Xu et al., 2022), it still cannot provide an accurate estimation of difficulty of data samples due to non-IID data in FL settings. Direct evaluation based on the inference loss of models (Vahidian et al., 2023) cannot well explore the impact on the generalization of the training process. The attention scores can analyze the dependency among diverse layers, but varies significantly between heads (Vig and Belinkov, 2019), which cannot be directly utilized in FL settings. While a sharpness-aware minimization method (Foret et al., 2021) can help minimize loss value and loss sharpness to improve model generalization, it does not consider federated fine-tuning settings of LLMs.

FIM can be exploited to enable the second-order optimization so as to improve the training process (Osawa et al., 2023; Jin et al., 2022) and to compute a global posterior for federated learning (Jhunjhunwala et al., 2024). In addition, continual learning can be used to improve the performance of trained models while addressing the forgetting problem (Wu et al., 2022a). Different from (Osawa et al., 2023; Wu et al., 2022a; Jhunjhunwala et al., 2024), we exploit the sum of diagonal of FIM to evaluate the difficulty of samples within the efficient curriculum learning method and to calculate the importance score of each layer and neuron within the LLM.

Model compression methods (Wu et al., 2021; Bibikar et al., 2022) are exploited in FL to reduce both computation and communication costs. Although pruning methods can reduce the size of large models (Wang et al., 2020; Ma et al., 2023; Xia et al., 2023), it is complicated to choose a proper pruning rate and may incur inferior performance in terms of accuracy (Wu et al., 2021). Sparse training can achieve personalization (Bibikar et al., 2022; Liu et al., 2023c; Setayesh et al., 2022; Dai et al., 2022) while addressing the client shift problem brought by the non-IID data (Setayesh et al., 2022; Karimireddy et al., 2020). However, the existing sparse training methods may incur severe accuracy degradation with poor generalization capacity due to simple component selection mechanisms. The model can be split into a server part and a device part to achieve both generalization and personalization capability (Han et al., 2023), which still incurs severe computation and communication costs in the FL settings of LLMs. Please note our approach is orthogonal with model compression methods.

For NLP tasks, prompt tuning (Liu et al., 2021; Lester et al., 2021; Liu et al., 2022d) can fine-tune LLMs with only a few parameters. With prompt tuning, an extra network is exploited to generate proper prompts or prefix, which is concatenated with the input to guide LLMs to generate proper answers. Furthermore, LoRA updates trainable rank decomposition matrices while freezing the parameters of the original network, which can significantly reduce the scale of parameters to update (Hu et al., 2021). However, both the prompt tuning and LoRA still incur significant communication costs due to the update for all the layers.

3 Problem Formulation

In this paper, we delve into the problem of how to efficiently fine-tune a large language model within a FL setting. The FL setting is composed of a parameter server and K devices. We assume that the data samples are distributed among the devices, each of which contains a dataset $D_k = \{s_i, m_i\}^{n_k}$ with s_i, m_i, and n_k referring to a data sample, the corresponding label, and the cardinality of D_k. We denote the cardinality of the whole dataset $D = \{D_1, D_2, ..., D_k\}$ by N.

We consider a LLM $\mathcal{M}$ of L layers, each layer contains a full parameter matrix $\mathcal{W}_o^l$. We exploit the LoRA method to reduce the parameters to update in this paper (Hu et al., 2021), and denote the LoRA parameters of the the LLM $\mathcal{M}$ by P with P_l representing the set of LoRA parameters in Layer l of $\mathcal{M}$. We denote the updated LoRA parameters on Device k by $\mathcal{P}_k$. Then, we formulate the problem to address in this paper as how to efficiently update $\mathbf{P}$ so as to minimize the global loss:

$$\min_{\mathbf{P}} \left[\mathcal{F}(\mathcal{M}, \mathbf{P}) \triangleq \frac{1}{K} \sum_{k=1,\ \mathcal{P}_k \in \mathbf{P}}^{K} n_k F_k(\mathcal{M}, \mathcal{P}_k) \right],$$
(1)

where $\mathcal{F}(\mathcal{M}, \mathbf{P})$ is the global loss, $F_i(\mathcal{M}, \mathcal{P}_k) \triangleq \frac{1}{n_l} \sum_{\{s_i, m_i\} \in \mathcal{D}_k} f(\mathcal{M}, \mathcal{P}_k, s_i, m_i)$ represents the local loss function on Device k with $f(\mathcal{M}, \mathcal{P}_k, s_i, m_i)$ calculating the local loss on Device k.

4 Efficient Curriculum Federated Learning

In this section, we first explain the system model. Then, we propose the adaptive federated curriculum learning method. Afterward, we further detail the efficient sparse tuning method.
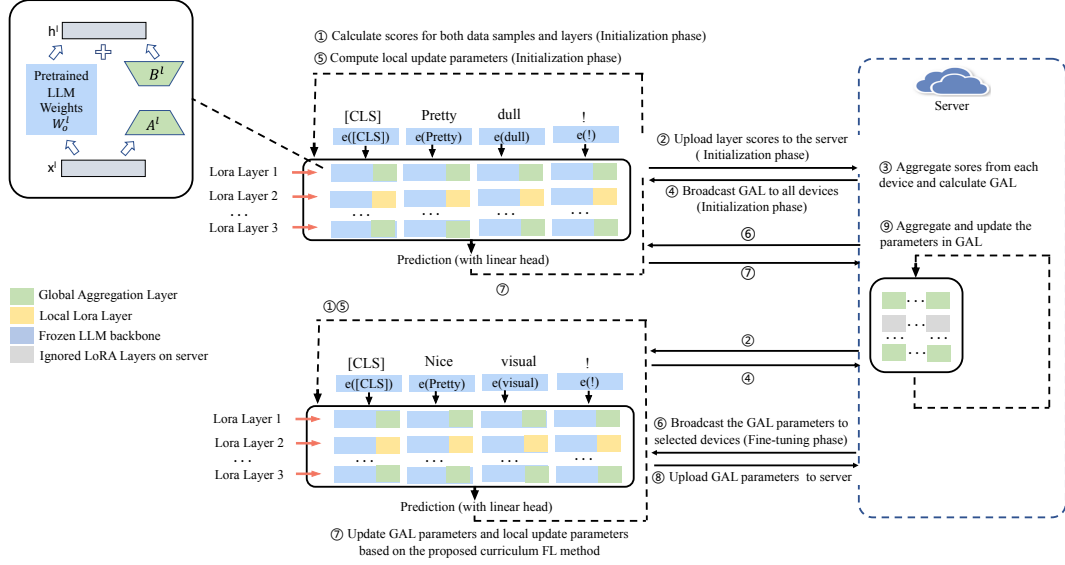
Figure 1: The system model of FibecFed.

4.1 System Model

The system model of FibecFed is shown in Figure 1. We assume that the LLM ($\mathcal{M}$) is deployed on each device. The parameters of $\mathcal{M}$ stays frozen while we update the LoRA parameters (Hu et al., 2021). As shown on the top left of Figure 1, on Device k, the parameters (W_o^l) at each layer (l) is decomposed into two matrices (LoRA), i.e., A_k^l and B_k^l, which can be updated during the fine-tuning process. Then, the hidden values (h) generated at Layer l with the input x is calculated based on Formula 2.

$$h = W_o^l x + B_k^l A_k^l x. \tag{2}$$

The fine-tuning process is composed of two phases, i.e., initialization and tuning. Within the initialization phase, we evaluate the difficulty score for each batch of data samples (see Formulas 3-5 in Section 4.2 and Lines 1-4 in Algorithm 1) and the importance score for each layer (see Formulas 6-11 and details in Section 4.3.1) based on fisher information on each device (Step ①). Then, the importance score of each layer is transferred to the server (Step ②), which aggregates the scores and selects proper layers as the Global Aggregation Layers (GAL) (see details in Section 4.3.1, Line 7 in Algorithm 1, Step ③). Afterward, the GAL are boradcasted to all the devices (Step ④). Finally, the parameters, which are not in the GAL, are locally evaluated on each device so as to generate local update part of parameters and the local static

parameters to be frozen (see details in Section 4.3.2, Lines 8-10 in Algorithm 1, Step ⑤). During the tuning phase, only the parameters in GAL and the local update part of parameters are updated while the local static parameters are kept frozen. The tuning phase consists of multiple rounds, each of which consists of five steps. First, the server randomly selects $\mathcal{K}$ devices and broadcasts the global parameters in GAL on the server to the selected devices (Step ⑥, Line 12 in Algorithm 1). Then, the parameters in GAL and the local update part of parameters are updated based on our proposed curriculum FL (see details in Appendix) on each selected device (Step ⑦, Lines 13-17 in Algorithm 1). Afterward, the updated parameters in GAL are uploaded to the server (Step ⑧). Finally, the server aggregate and update the global parameters in GAL (Step ⑨, Line 19 in Algorithm 1).

The tuning phase is composed of multiple rounds, each of which consists of five steps. First, the server randomly selects $\mathcal{K}$ devices and broadcasts the global parameters in GAL on the server to the selected devices (Step ⑥). Then, the parameters in GAL and the local update part of parameters are updated based on our proposed curriculum FL (see details in Section 4.2) on each selected device (Step ⑦). Afterward, the updated parameters in GAL are uploaded to the server (Step ⑧). Finally, the server aggregate and update the global parameters in GAL (Step ⑨).

4.2 Fisher information-based Curriculum Federated Learning

Inspired by the starting small strategy (Elman, 1993), we propose a fisher information-based curriculum FL method to enable efficient federated fine-tuning. As the FIM can help indicate the amount of information carried by each data sample to generate the response (Ly et al., 2017), we propose utilizing the FIM to measure the difficulty of data samples. The FIM is defined in Formula 3:

$$\mathbf{F}_i \triangleq \mathbb{E}_{s_i} \left[(\nabla \log p_k(s_i)) (\nabla \log p_k(s_i))^T \right], \quad (3)$$

where $\mathbf{F}_i$ represents the FIM corresponding to data sample s_i, $p_k(s_i)$ represents the probability density function of the inference with the LLM $\mathcal{M}$, the LoRA parameters $\mathcal{P}_k$, and data sample s_i, $\nabla \log p_k(s_i)$ denotes the first-order derivative of the LoRA parameters, calculated by the gradient of the loss respect to $\mathcal{P}_k$, T refers to the transpose of a matrix. Practically, the expected FIM can be approximated by empirical FIM (Kunstner et al., 2019) as defined in Formula 4:

$$\mathbf{F}_i \approx \frac{1}{N} \sum_{i=1}^{N} \left[(\nabla \log p_k(s_i)) (\nabla \log p_k(s_i))^T \right], \quad (4)$$

However, calculating the FIM is computational expensive as the multiplication of $\nabla \log p_k(s_i)$ is both time and memory consumption when the size of the derivative matrix, i.e., $|\nabla \log p_k(s_i)|$, is substantial. Inspired by (Pascanu and Bengio, 2013), we calculate the diagonal of FIM to approximate the FIM as shown in Formula 5.

$$\tilde{\mathbf{F}}_i = I_{|\nabla \log p_k(s_i)|} \odot \mathbf{F}_i \quad (5)$$

where $I_{|\nabla \log p_k(s_i)|}$ is the identity matrix with the same size of the derivative matrix. Then, we calculate the sum of the trace of $\tilde{\mathbf{F}}_i$ (Jastrzebski et al., 2021) as the score of the data sample. Finally, we can calculate the difficulty score of a batch of data samples (see details in Appendix).

In order to improve the training efficiency, we propose a curriculum data selection strategy. We take the simplest $\mathcal{B}_k^t$ data samples for the local update on Device k in Round t. $\mathcal{B}_k^t$ becomes bigger along with the epoch number within the training process (see details in Apendix).

4.3 Efficient Sparse Parameter Update

In this section, we propose an efficient sparse parameter method composed of a global aggregation layer selection method and a local update parameter selection method.

4.3.1 Global Aggregation Layer Selection

In order to reduce communication costs, we only transfer the LoRA parameters in important layers (GAL) between the server and devices for global aggregation. In this section, we propose a global aggregation layer selection method with a novel layer importance score calculation technique and a global aggregation layer selection technique based on the importance score.

While important layers generally capture distinguishable features of data (Mellor et al., 2021), we select the layers that are sensitive to the input data samples. When a layer exhibit less resilience against the noise on the data, it corresponds to higher sensitivity, and thus is more important. We calculate the output difference of a certain layer with two similar input data samples to indicate its resilience, which represents the importance score.

In order to get two similar input data samples, we add noise to an original sample. Within a predefined noise budget, we calculate the noise that maximizes the loss, so as to well evaluate the sensitivity of the layer. Then, the noise (ϵ_i) corresponding to s_i is calculate based on Formula 6:

$$\epsilon_i = \underset{||\epsilon_i||_p < \gamma}{\mathrm{argmax}} \underbrace{f(\mathcal{M}, \mathcal{P}_k, s_i + \epsilon_i, m_i)}_{L_k(s_i + \epsilon_i)} - \underbrace{f(\mathcal{M}, \mathcal{P}_k, s_i, m_i)}_{L_k(s_i)}, \quad (6)$$

where L_k is the local loss, $|| \cdot ||_p$ represents the ℓ_p-norm of the noise, and γ refers to the noise budget. We decompose $L_k(s_i + \epsilon_i) - L_k(s_i)$ via the first-order Talyor extension as defined in Formula 7:

$$\begin{aligned} &L_k(s_i + \epsilon_i) - L_k(s_i) \\ &\approx L_k(s_i) + \epsilon_i^T \nabla_{\mathcal{P}_k} L_k(s_i) - L_k(s_i), \quad (7) \\ &= \epsilon_i^T \nabla_{\mathcal{P}_k} L_k(s_i). \end{aligned}$$

Then, we can solve the approximation by the solution to a classic dual problem (Foret et al., 2021) as defined in Formula 8.

$$\epsilon_i^* = \gamma \frac{sign(\nabla_{\mathcal{P}_k} L_k(\mathcal{P}_k)) |\nabla_{\mathcal{P}_k} L_k(\mathcal{P}_k)|^{q-1}}{(||\nabla_{\mathcal{P}_k} L_k(\mathcal{P}_k)||_q^q)^{1/1-p}} \quad (8)$$

where $| \cdot |^{q-1}$ denotes the absolute value and power in terms of each element, q is a factor that satisfies $\frac{1}{p} + \frac{1}{q} = 1$. Afterward, we take ϵ_i^* as the noise ϵ_i to calculate the sensitivity.

As Frobenius norm can characterize features in the latent space (Chen et al., 2021), we exploit a relative difference of the Frobenius norm to measure the sensitivity of each layer. The relative difference can avoid the bias brought by the absolute values. The relative difference of of Frobenius norm is defined in Formula 9.

$$\mathcal{F}^l(s_i) = \frac{||h^l(s_i + \epsilon_i^*)||_F - ||h^l(s_i)||_F}{||h^l(s_i)||_F}, \quad (9)$$

where $\mathcal{F}^l(s_i)$ is the relative difference of the Frobenius norm, $h^l(s_i)$ represents the output embeddings at Layer l of the LLM $\mathcal{M}$ for data sample s_i, $|| \cdot ||_F$ refers to the Frobenius norm. Then, the importance score of Layer l on Device k is calculated based on its local data D_k as defined in Formula 10.

$$\mathcal{I}_k^l = \frac{1}{n_k} \sum_{s_i \in D_k} \mathcal{F}^l(s_i), \quad (10)$$

where $\mathcal{I}_k^l$ represents the importance score of Layer l on Device k. Afterward, the global importance score $\mathcal{I}^l$ is calculated based on Formula 11.

$$\mathcal{I}^l = \frac{1}{N} \sum_{k=1}^{K} n_k \mathcal{I}_k^l, \quad (11)$$

We propose a lossless method to select proper important layers as GAL. On each device k, the LoRA parameters are initialized as $\mathcal{P}_k^0$. After $\mathcal{T}$ rounds of fine-tuning, the LoRA parameters are denoted by $\mathcal{P}_k^{\mathcal{T}}$. We construct a base function as $\Delta_k = \mathcal{P}_k^0 - \mathcal{P}_k^{\mathcal{T}}$. We calculate the Hessian matrix of the local loss function with its eigenvalues sorted in ascending order ($\{\lambda_k^1, \lambda_k^2, ..., \lambda_k^r ..., \lambda_k^{R_k}\}$ with r representing the index of an eigenvalue and R indicating the rank of the Hessian matrix). We calculate the Lipschitz constant ($\mathscr{L}_k$) of a base function $H_k(\mathcal{P}_k^{\mathcal{T}})\Delta_k - \nabla L_k'(\Delta_k + \mathcal{P}_k^{\mathcal{T}})$ with $\nabla L_k'(\cdot)$ being the gradient of the local loss function and H_k referring to the Hessian matrix of the local loss function. Inspired by (Zhang et al., 2021), we find the first r_k that satisfies $\lambda_{r_{k+1}} - \lambda_{r_k} > 4\mathscr{L}_k$ to achieve lossless performance. Then, we calculate the expected number of layers in GAL on Device k as $\mathcal{N}_k^* = (1 - \frac{r_k}{R_k})L$ with L being the number of layers in $\mathcal{M}$. Then, we calculate the number of layers in GAL as $\mathcal{N}^* = \frac{\mu}{N} \sum n_k \mathcal{N}_k^*$, where μ is a hyper-parameter to adjust the ratio between global and local number. Finally, we select $\mathcal{N}^*$ layers with the highest importance scores.

4.3.2 Local Update Parameter Selection

In order to reduce computation costs, we only update important LoRA parameters within the local update while freeze the remaining parameters. Apart from the parameters in GAL, we dynamically select an important part of parameters in other layers to update. In this section, we propose a novel fisher information-based local update parameter selection method with momentum.

While the LoRA parameters may significantly vary during the fine-tuning process, we calculate the FIM with momentum within first $\mathcal{T}'$ epochs by $\mathbf{F}_k^t = \gamma * \mathbf{F}_k^{t-1} + (1-\gamma)\tilde{\mathbf{F}}_k$, where γ represents the coefficient that controls the step size of the moving average, $\mathbf{F}_k^t$ refers to the FIM on Device k at Round t, $\tilde{\mathbf{F}}_k$ is the empirical average diagonal approximation of the FIM, i.e., $\tilde{\mathbf{F}}_k = \frac{1}{n_k} \sum_{s_i \in D_k} \tilde{\mathbf{F}}_i$ with $\tilde{\mathbf{F}}_i$ calculated based on Formula 5, while $\mathbf{F}_k^0$ is directly calculated without moment. Finally, we get a FIM $\mathbf{F}_{k,l}^{\mathcal{T}'}$ for each layer l outside of GAL. Inspired by (Diao et al., 2023), we exploit a neuron-wise aggregation of the FIM to indicate the importance score of Neuron μ in Layer l as defined in Formula 12.

$$\int_{k,l}^{\mu} = \sum_{\upsilon=0}^{|\mathcal{W}_{\mu:}|-1} \mathbf{F}_k^{\mathcal{T}'}[\mu * |\mathcal{W}_{\mu:}| + \upsilon] \quad (12)$$

Where $|\mathcal{W}\mu :|$ denotes the number of elements in the μth row of the full weight matrix $\mathcal{W}_o^l$ in $\mathcal{M}$, $\mathbf{F}_{k,l}^{\mathcal{T}'}[\nu]$ represents the νth diagonal element in $\mathbf{F}_{k,l}^{\mathcal{T}'}$. Afterward, we exploit the lossless method to calculate the proper local update parameter ratio as $\rho_{k,l} = 1 - \frac{r_{k,l}}{R_{k,l}}$ (see details in Section 4.3.1, with $r_{k,l}$ and $R_{k,l}$ representing the corresponding r_k and R_k in Layer l. Finally, we take the most important $\rho_{k,l}$ neurons in terms of the importance score $\int_{k,l}^{\mu}$ as the local update parameters to be updated with the parameters in GAL and freeze the other parameters within the local update.

4.4 FibecFed Algorithm

The FibecFed algorithm is shown in Algorithm 1. Within the initialization phase (Lines 1 - 10), the difficulty scores of each batch are calculated based on Formula 7 (Lines 2 - 4), the batches of data samples are sorted in ascending order in terms of the difficulty scores for the curriculum data selection strategy (Line 5), the GAL are calculated in Line 7 (see details in Section 4.3.1), and the local update parameters are computed in Line 9 (see details in Section 4.3.2). Then, within the fine-tuning phase

Algorithm 1 Fisher Information-base Efficient Curriculum Federated Learning (FibecFed)

Input:
T: The maximum number of rounds
K: The number of devices
$D = \{D_1, D_2, ..., D_K\}$: The set of datasets on each device
$\eta = \{\eta_1, \eta_2, ..., \eta_T\}$: The learning rates

Output:
$\mathbf{P}^t$: The set of LoRA parameters at Round t

1: **for** k in $\{1, 2, ..., K\}$ (in parallel) **do**
2:      **for** $B_j \in D_k$ **do**
3:          $\smallint_j \leftarrow$ Calculation based on Formula 7
4:      **end for**
5:      Sort $B_j \in D_k$ in ascending order of $\smallint_j$
6: **end for**
7: $GAL \leftarrow$ Compute the GAL
8: **for** k in $\{1, 2, ..., K\}$ (in parallel) **do**
9:      $\mathcal{P}_k^u \leftarrow$ Compute local update parameters
10: **end for**
11: **for** t in $\{1, 2, ..., T\}$ **do**
12:      Sample $\mathcal{K} \subseteq \{1, 2, ..., K\}$ devices
13:      **for** k in $\mathcal{K}$ **do**
14:          Select $\mathcal{B}_k^t$ data samples based on Formula 8
15:          $\mathcal{P}_k^{t-\frac{1}{2}} \leftarrow$ Update $\mathcal{P}_k^{t-1}$ with $\mathcal{P}_{GAL}^{t-1}$
16:          $\mathcal{P}_k^t \leftarrow$ Update $\mathcal{P}_k^u \subset \mathcal{P}_k^{t-\frac{1}{2}}$ using $\mathcal{B}_k^t$ data samples
17:      **end for**
18:      $\mathcal{P}_{GAL}^t \leftarrow$ Aggregate $\mathcal{P}_{GAL,k}^t$ with $k \in \mathcal{K}$
19: **end for**

is performed in Lines 11 - 19. A set of devices $\mathcal{K}$ is randomly selected (Line 12). Then, on each device, local update is carried out in Lines 13 - 17. First, the data samples are selected based on the curriculum data selection strategy in Line 14. Second, the LoRA parameters are updated with the global parameters in PSL transferred from the server in Line 15 (see details in Section 4.1). Third, the LoRA parameters are updated based on the local training with the selected $\mathcal{B}_k^t$ data samples in Line 16 (see details in Section 4.3.2). Finally, the parameters in the global aggregation layers are aggregated using the FedAvg algorithm (McMahan et al., 2017) on the server (Line 18).

5 Experiments

In this section, we demonstrate extensive experimentation with 17 baseline approaches and 10 NLP tasks to reveal the advantages of FibecFed.

5.1 Experimental Setup

We take an FL environment composed of 100 devices and a parameter server in the experimentation. We randomly sample 10 devices in each epoch. We utilize 10 commonly-used NLP tasks in the experimentation, i.e., QNLI (Rajpurkar et al., 2016), SST-2 (Socher et al., 2013), CoLA (Warstadt et al., 2019), MRPC (Dolan and Brockett, 2005), RTE (Giampiccolo et al., 2007), BoolQ (Clark et al., 2019), MPQA (Wiebe et al., 2005), Subj (Pang and Lee, 2004), TREC (Voorhees and Tice, 2000), and MR (Pang and Lee, 2005). The input data of the tasks are non-IID among the 100 devices. We compare FibecFed with 17 baseline approaches, i.e., a parameter efficient fine-tuning-based approaches (Adapter (Houlsby et al., 2019)), 6 prompt-based tuning methods (FedPrompt (Zhao et al., 2023a), P-tuning v2 (Liu et al., 2022d), IDPG (Wu et al., 2022b), ATTEMPT (Asai et al., 2022), LPT (Liu et al., 2022c), LoRA (Hu et al., 2021)), 4 curriculum learning-based approaches (Shortformer (Press et al., 2021), VOC (Platanios et al., 2019), SLW (Li et al., 2024), SE (Peng et al., 2023)), 3 personalized FL methods (PFedGate (Chen et al., 2023), FedDST (Bibikar et al., 2022), FedALT (Pillutla et al., 2022)), and 2 Lora based methods (SLoRA (Babakniya et al., 2023), AdaLoRA (Zhang et al., 2023)). We carry out the experimentation based on two language models, i.e., RoBERTa$_{\text{LARGE}}$ (Liu et al., 2020) and LLaMA (Touvron et al., 2023).

5.2 Evaluation of FibecFed

In this section, we present the evaluation of FibecFed based on RoBERTa$_{\text{LARGE}}$ and LLaMA.

5.3 Evaluation based on RoBERTa$_{\text{LARGE}}$

Table 1 present the convergence accuracy of diverse approaches based on RoBERTa$_{\text{LARGE}}$. FibecFed significantly outperforms baseline methods in terms of the convergence accuracy (up to 45.35%, 38.37%, 12.69%, 37.60%, 42.38%, 18.72%, 7.01%, 6.36%, 5.70%, 5.90%, 8.79%, 28.45%, 16.70%, 4.96%, 5.44%, 14.11% and 5.49% compared with Adapter, FedPrompt, P-tuning v2, IDPG, ATTEMPT, LTP, LoRA, Shortformer, VOC, SLW, PFedGate, FedDST, SE, FedALT, sLoRA, AdaLoRA, and Delta-LoRA respectively). In addition, we analyze the time to achieve target accuracy (see details in Appendix), which demonstrates that FibecFed outperforms baseline methods in terms of efficiency

Method	QNLI	SST-2	CoLA	MRPC	RTE	BoolQ	MPQA	Subj	Trec	MR	Avg
Adapter	49.46	90.83	54.17	84.77	47.29	62.17	90.95	51.65	96.2	91.30	58.32
FedPrompt	87.73	94.38	19.79	76.31	64.98	74.58	90.10	94.25	92.6	91	78.57
P-tuning v2	88.74	94.04	50.23	78.16	76.17	74.89	88.75	95.5	90.8	90.65	82.79
IDPG	66.7	89.11	4.59	72.22	52.35	68.93	71.8	59.4	73.4	86.35	64.48
ATTEMPT	50.74	50.92	4.63	76.01	54.15	62.17	90.35	88.85	77.2	91.15	64.16
LPT	89.38	94.27	50.78	82.38	80.86	62.2	90.15	95.75	92.4	90.6	82.87
LORA	89.86	94.72	54.78	83.15	78.7	75.96	89.2	95.8	94.4	91.35	84.72
Shortformer	90.17	94.04	54.16	84.49	79.78	78.62	90.6	96.85	95.2	91.1	85.5
VOC	91.89	95.18	53.64	85.15	81.31	78.32	90.85	96.85	96.6	91.25	86.1
SLW	91.91	94.1	54.75	85.01	79.06	78.41	91.25	96.6	95.8	91.15	85.80
PFedGate	90.72	93.81	50.73	83.33	76.17	76.88	89.8	94.5	94	87.65	83.75
FedDST	90.15	94.61	30.12	81.41	71.84	77.13	90.05	95.85	86.2	91.2	80.85
SE	76.42	93.69	55.47	81.72	71.84	76.42	89.8	96.3	87.2	91.85	82.07
FedALT	91.76	94.15	<u>55.5</u>	<u>85.89</u>	80.95	80.46	90	96.15	96.6	91.25	86.23
sLORA	92.40	<u>94.38</u>	55.51	85.41	82.31	80.24	<u>91.30</u>	96.8	<u>97.2</u>	91.35	<u>86.69</u>
adaLORA	91.14	92.32	44.46	81.83	75.81	77.16	89.65	96.25	91	<u>91.8</u>	83.14
Delta-LoRA	<u>92.41</u>	94.27	54.95	85.36	<u>83.39</u>	<u>80.18</u>	90.80	96.75	96.8	91.25	86.61
Ours	**93.12**	**95.76**	**58.57**	**90.85**	**84.69**	**80.92**	**91.35**	**97.0**	**97.8**	**92.95**	**88.31**

Table 1: The convergence accuracy with FibecFed and diverse baseline approaches. The evaluation with GLUE benchmark is based on development sets while others are based on test sets. The best results are highlighted in **bold** and the second bests are marked with <u>underline</u>. The results are obtained using RoBERTa$_{\text{LARGE}}$.

(up to 94.7%, 97.43%, 98.61%, 61.64%, 96.12%, 91.6%, 96.69%, 89.12%, 80.15%, 84.26%, 80.82%, 97.01%, 96.52%, 82.16%, 85.18%, 95.66% and 69.65% compared with Adapter, FedPrompt, P-tuning v2, IDPG, ATTEMPT, LTP, LoRA, Shortformer, VOC, SLW, PFedGate, FedDST, SE, FedALT, sLoRA, AdaLoRA, and Delta-LoRA respectively). The advantages are expected as our proposed curriculum data selection strategy on each device can well improve both the efficiency and the effectiveness. In addition, the proposed important layer selection method can reduce the scale of parameters to transfer between the server and devices. Furthermore, the local update parameter selection method can well reduce useless computation on each devices while freezing unimportant parameters may mitigate the effect of overfitting brought by the non-IID data.

5.4 Evaluation based on LLaMA 7B

We carried out the experimentation with a LLM, i.e., LLaMA 7B on MRPC, MR, and SST-2 dataset. As shown in Table 2, FibecFed significantly outperforms baseline approaches in terms of both performance (up to 29.60%, 33.93%, 11.91%, 12.27%, 5.41%, 11%, 12.27%, 26.35%, 4.69% higher accuracy compared with FedPrompt, P-tuning v2, ATTEMPT, LPT, LORA, VOC, SE, SLoRA and DeltaLoRA) and efficiency (up to 45.75%, 39.87%, 28.8%, 32.7%, 26.58%, 5.63%, 16.2%, 25.6%, 47.1% faster compared with FedPrompt, P-tuning v2, ATTEMPT, LPT, LORA, VOC, SE, SLoRA

Method	COLA		MRPC		RTE	
	Acc	Time	Acc	Time	Acc	Time
FedPrompt	59.30	2527	80.54	1365	56.68	1296
P-tuning v2	3.89	2159	80.18	935	52.35	842
ATTEMPT	54.77	1825	81.38	1069	74.37	934
LPT	51.8	1645	80.13	958	74.01	1045
LoRA	59.55	1768	79.56	941	80.87	984
Voc	60.64	1364	38.64	859	75.28	745
SE	58.69	1628	79.35	846	74.01	839
SLoRA	60.56	1602	80.39	947	59.93	945
DeltaLoRA	58.91	1674	81.67	1034	81.95	1329
FibecFed	**61.48**	**1298**	**81.93**	**832**	**86.28**	**703**

Table 2: Convergence accuracy and fine-tuning time on COLA, MRPC and RTE with LLaMA.

and DeltaLoRA). The advantages reveal the our proposed approach improves both the efficiency and effectiveness with LLM.

5.5 Robustness & Scalability

In this section, we demonstrate the robustness and the scalability of FibecFed with divers degrees of non-IID data and different device numbers. FibecFed achieves comparable accuracy across divers degrees of data heterogeneity, the difference of which is smaller than 1.83%. In addition, we conduct the experimentation with the device number ranging from 20 to 100 based on MRPC dataset and find that the disparity is smaller than 0.79% in terms of accuracy.

5.6 Communication overhead

The absolute communication overhead is shown in the Table 13 of Appendix (with RoBERTa$_{\text{LARGE}}$).

The communication overhead of FibecFed is higher than that of FedPrompt (up to 3.51 times), IDPG (up to 3.3 times), and ATTEMPT (up to 1.85 times). This is expected as the these three methods are prompt tuning-based methods, which corresponds to much fewer parameters to update during the training phase compared with FibecFed. However, these three methods correspond to significantly lower performance (compared with FibecFed), i.e., low convergence accuracy (from 1.25% to 38.68% for FedPrompt, from 6.6% to 53.98% for IDPG, and from 1% to 53.94% for ATTEMPT), as shown in Table 1. The communication overhead of FibecFed is significantly lower than the other methods (6.25 times for Adapter, 9.67 times for P-tuning V2, 1.9 times for LPT, and 25% for LORA, SHORTFORMER, Voc, SLW, PFedGate, FedDST, SE, FedAlt, sLora, AdaLora, Delta-LoRA). This is expected as well as FibecFed only transfers the global aggregation layers instead of the parameters of all the layers in LLM.

In addition, the relative communication overhead (i.e., the ratio between the absolute communication overhead and the total training time) is shown in Table 14 of Appendix. Similar to the absolute communication overhead, the relative communication overhead of FibecFed is higher than that of Fed-Prompt (from 2.3% to 37.4%), IDPG (from 2.4% to 34.4%), and ATTEMPT (from 1.9% to 31.7%) as well. This is expected as explained before. As the total training time of FibecFed becomes shorter, the relative communication overhead of FibecFed becomes slightly more significant than FedAlt (from 1.4% to 20.1%) and AdaLora (from 0.8% to 5.7%). In addition, the relative communication overhead of FibecFed becomes similar to that of sLora (from 7.8% smaller to 24.8% bigger) and Delta-LoRA (from 7.5% smaller to 5.5% bigger). The relative communication overhead of FibecFed is still smaller than the rest approaches, i.e., Adapter (up to 43.9%), P-tuning V2 (up to 55.6%), LPT (up to 25.1%), LORA (up to 14.5%), SHORTFORMER (up to 13.8%), Voc (up to 8.0%), SLW (up to 8.8%), PFedGate (up to 7.0%), FedDST (up to 7.1%), SE (8.1%). Please note that FibecFed corresponds to higher convergence accuracy (as shown in Table 1) and shorter training time (as shown in Table 2).

5.7 Ablation Study

To analyze the impact of each module in FibecFed, we demonstrate the ablation study in terms of the curriculum data selection method, the important layer selection method, and the local update parameter selection method. First, we conduct a comparative analysis among four curriculum strategies, i.e., SLW, VOC, Shortformer, SE, and that without curriculum (NULL). FibecFed corresponds to superior performance in terms of accuracy (up to 5.73%, 9.12% 5.84%, 6.41%, 7.7% compared with Voc, SE, SLW, Shortformer, and NULL, respectively) and efficiency (up to 26.53%, 34.26% 68.92%, 68.36%, 58.57% compared with Voc, SE, SLW, Shortformer, and NULL, respectively). Afterward, we compare the importance layer selection method with Ascending Order (AO), Descending Order, Random Order (RO) and that with full layer synchronization (FULL), which reveals the advantages of our layer selection method in terms of accuracy (up to 3.42%, 2.76%, 2.65%, 1.02%) and efficiency (up to 29.3%, 18.46%, 23.1%, 15.3%) compared with AO, DO, RO, and FULL, respectively. Furthermore, we compare our local update parameter selection method to that without selection, i.e., all the parameters are updated. The advantage of our local update parameter selection method can achieve 2.48% higher accuracy and 11.8% faster.

6 Conclusion

In this paper, we propose an original fisher information-based efficient curriculum federated learning, i.e., FibecFed. Within FibecFed, we propose an adaptive federated curriculum learning method and an efficient sparse parameter update method. We exploit fisher information to calculate the difficulty scores of data samples and propose the an original curriculum data selection strategy. In the sparse parameter update method, we propose a new sensitivity-based important layer selection technique and a novel fisher information-based important parameter technique method while freezing the remaining parameters to achieve both efficiency and effectiveness. We demonstrate the results of extensive experimentation to compare FibecFed with 17 baseline approaches based on 10 NLP tasks, which reveal significant advantages of FibecFed in terms of accuracy (up to 45.35%) and fine-tuning speed (up to 98.61% faster).

Acknowledgements

Limitations

While our approach can be to exploited to significantly improve the performance and efficiency of LLM federated learning, we assume that a central parameter server is exploited to coordinate the training process. When there is not a central parameter or the heterogeneous devices (Che et al., 2022; Li et al., 2022b) are connected based on diverse topologies (decentralized setting (Liu et al., 2024a)), e.g., ring, it would be complicated to directly exploit our proposed approach. In addition, our approach can be combined with model compression methods, e.g., model pruning and quantization (Jia et al., 2024), to achieve better performance. In the future, we anticipate exploiting model compression methods in LLM federated learning with a decentralized setting.

References

Akari Asai, Mohammadreza Salehi, Matthew E. Peters, and Hannaneh Hajishirzi. 2022. ATTEMPT: parameter-efficient multi-task tuning via attentional mixtures of soft prompts. In *Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6655–6672.

Sara Babakniya, Ahmed Roushdy Elkordy, Yahya H. Ezzeldin, Qingfeng Liu, Kee-Bong Song, Mostafa El-Khamy, and Salman Avestimehr. 2023. Slora: Federated parameter efficient fine-tuning of language models. In *Int. Workshop on Federated Learning in the Age of Foundation Models in Conjunction with NeurIPS*, pages 1–13.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Int. Conf. on Machine Learning (ICML)*, page 41–48. Association for Computing Machinery.

Sameer Bibikar, Haris Vikalo, Zhangyang Wang, and Xiaohan Chen. 2022. Federated dynamic sparse training: Computing less, communicating less, yet learning better. In *AAAI Conf. on Artificial Intelligence (AAAI)*, pages 6080–6088. AAAI Press.

Californians for Consumer Privacy. 2020. California consumer privacy act home page. https://www.caprivacy.org/. Online; accessed 09/05/2022.

Tianshi Che, Ji Liu, Yang Zhou, Jiaxiang Ren, Jiwen Zhou, Victor Sheng, Huaiyu Dai, and Dejing Dou. 2023a. Federated learning of large language models with parameter-efficient prompt tuning and adaptive optimization. In *Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7871–7888.

Tianshi Che, Zijie Zhang, Yang Zhou, Xin Zhao, Ji Liu, Zhe Jiang, Da Yan, Ruoming Jin, and Dejing Dou. 2022. Federated fingerprint learning with heterogeneous architectures. In *IEEE Int. Conf. on Data Mining (ICDM)*, pages 31–40.

Tianshi Che, Yang Zhou, Zijie Zhang, Lingjuan Lyu, Ji Liu, Da Yan, Dejing Dou, and Jun Huan. 2023b. Fast federated machine unlearning with nonlinear functional theory. In *Int. Conf. on Machine Learning (ICML)*, pages 4241–4268.

Daoyuan Chen, Liuyi Yao, Dawei Gao, Bolin Ding, and Yaliang Li. 2023. Efficient personalized federated learning via sparse model-adaptation. In *Int. Conf. on Machine Learning (ICML)*, volume 202, pages 5234–5256.

Xinyang Chen, Sinan Wang, Jianmin Wang, and Mingsheng Long. 2021. Representation subspace distance for domain adaptation regression. In *Int. Conf. on Machine Learning (ICML)*, pages 1749–1759.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, (NAACL-HLT)*, pages 2924–2936.

Rong Dai, Li Shen, Fengxiang He, Xinmei Tian, and Dacheng Tao. 2022. Dispfl: Towards communication-efficient personalized federated learning via decentralized sparse training. In *Int. Conf. on Machine Learning (ICML)*.

Enmao Diao, Ganghua Wang, Jiawei Zhang, Yuhong Yang, Jie Ding, and Vahid Tarokh. 2023. Pruning deep neural networks from a sparsity perspective. In *Int. Conf. on Learning Representations (ICLR)*, pages 1–12.

Dimitar I. Dimitrov, Mislav Balunović, Nikola Konstantinov, and Martin Vechev. 2022. Data leakage in federated averaging. *arXiv preprint arXiv:2206.12395*.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing, IWP@IJCNLP 2005, Jeju Island, Korea, October 2005, 2005*. Asian Federation of Natural Language Processing.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. Glm: General language model pretraining with autoregressive blank infilling. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 320–335.

Tran Trong Duy, Ly V Nguyen, Viet-Dung Nguyen, Nguyen Linh Trung, and Karim Abed-Meraim. 2022. Fisher information neural estimation. In *European Signal Processing Conference (EUSIPCO)*, pages 2111–2115. IEEE.

Jeffrey L. Elman. 1993. Learning and development in neural networks: the importance of starting small. *Cognition*, 48(1):71–99.

Gemini Team et al. 2023. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Tao Fan, Yan Kang, Guoqiang Ma, Weijing Chen, Wenbin Wei, Lixin Fan, and Qiang Yang. 2023. Fatellm: A industrial grade federated learning framework for large language models. *arXiv preprint arXiv:2310.10049*.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. 2021. Sharpness-aware minimization for efficiently improving generalization. In *Int. Conf. on Learning Representations (ICLR)*, pages 1–19.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third PASCAL recognizing textual entailment challenge. In *Workshop on Textual Entailment and Paraphrasing (ACL-PASCAL@ACL)*, pages 1–9.

Dong-Jun Han, Do-Yeon Kim, Minseok Choi, Christopher G Brinton, and Jaekyun Moon. 2023. Splitgp: Achieving both generalization and personalization in federated learning. In *IEEE Int. Conf, on Computer Communications (INFOCOM)*, pages 1–10. IEEE.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Int. Conf. on Machine Learning (ICML)*, volume 97, pages 2790–2799.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.0968*.

Stanislaw Jastrzebski, Devansh Arpit, Oliver Astrand, Giancarlo B Kerg, Huan Wang, Caiming Xiong, Richard Socher, Kyunghyun Cho, and Krzysztof J Geras. 2021. Catastrophic fisher explosion: Early phase fisher matrix impacts generalization. In *Int. Conf. on Machine Learning (ICML)*, pages 4772–4784. PMLR.

Divyansh Jhunjhunwala, Shiqiang Wang, and Gauri Joshi. 2024. Fedfisher: Leveraging fisher information for one-shot federated learning. In *Int. Conf. on Artificial Intelligence and Statistics*, pages 1612–1620.

Juncheng Jia, Ji Liu, Chendi Zhou, Hao Tian, Mianxiong Dong, and Dejing Dou. 2024. Efficient asynchronous federated learning with sparsification and quantization. *Concurrency and Computation: Practice and Experience*, 36(9):e8002.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Jiayin Jin, Jiaxiang Ren, Yang Zhou, Lingjuan Lyu, Ji Liu, and Dejing Dou. 2022. Accelerated federated learning with decoupled adaptive optimization. In *Int. Conf. on Machine Learning (ICML)*, pages 10298–10322.

Ryo Karakida, Shotaro Akaho, and Shun-ichi Amari. 2019. Universal statistics of fisher information in deep neural networks: Mean field approach. In *Int. Conf. on Artificial Intelligence and Statistics (AISTATS)*, pages 1032–1041. PMLR.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. 2020. SCAFFOLD: Stochastic controlled averaging for federated learning. In *Int. Conf. on Machine Learning (ICML)*, volume 119, pages 5132–5143.

Weirui Kuang, Bingchen Qian, Zitao Li, Daoyuan Chen, Dawei Gao, Xuchen Pan, Yuexiang Xie, Yaliang Li, Bolin Ding, and Jingren Zhou. 2023. Federatedscope-llm: A comprehensive package for fine-tuning large language models in federated learning. *arXiv preprint arXiv:2309.00363*.

Frederik Kunstner, Philipp Hennig, and Lukas Balles. 2019. Limitations of the empirical fisher approximation for natural gradient descent. *Advances in neural information processing systems (NeurIPS)*, 32.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3045–3059.

Conglong Li, Zhewei Yao, Xiaoxia Wu, Minjia Zhang, Connor Holmes, Cheng Li, and Yuxiong He. 2024. Deepspeed data efficiency: Improving deep learning model quality and training efficiency via efficient data sampling and routing. *arXiv preprint arXiv:2212.03597*, pages 1–19.

Conglong Li, Minjia Zhang, and Yuxiong He. 2022a. The stability-efficiency dilemma: Investigating sequence length warmup for training GPT models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Guanghao Li, Yue Hu, Miao Zhang, Ji Liu, Quanjun Yin, Yong Peng, and Dejing Dou. 2022b. Fedhisyn: A hierarchical synchronous federated learning framework for resource and data heterogeneity. In *Int. Conf. on Parallel Processing (ICPP)*, pages 1–11.

Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long,

Eugene J Shekita, and Bor-Yiing Su. 2014. Scaling distributed machine learning with the parameter server. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pages 583–598.

Ji Liu, Tianshi Che, Yang Zhou, Ruoming Jin, Huaiyu Dai, Dejing Dou, and Patrick Valduriez. 2024a. Aedfl: efficient asynchronous decentralized federated learning with heterogeneous devices. In *SIAM Int. Conf. on Data Mining (SDM)*, pages 833–841.

Ji Liu, Chunlu Chen, Yu Li, Lin Sun, Yulun Song, Jingbo Zhou, Bo Jing, and Dejing Dou. 2024b. Enhancing trust and privacy in distributed networks: a comprehensive survey on blockchain-based federated learning. *Knowledge and Information Systems*, pages 1–27.

Ji Liu, Jizhou Huang, Yang Zhou, Xuhong Li, Shilei Ji, Haoyi Xiong, and Dejing Dou. 2022a. From distributed machine learning to federated learning: A survey. *Knowledge and Information Systems*, 64(4):885–917.

Ji Liu, Juncheng Jia, Tianshi Che, Chao Huo, Jiaxiang Ren, Yang Zhou, Huaiyu Dai, and Dejing Dou. 2024c. Fedasmu: Efficient asynchronous federated learning with dynamic staleness-aware model update. In *AAAI Conf. on Artificial Intelligence*, volume 38, pages 13900–13908.

Ji Liu, Juncheng Jia, Beichen Ma, Chendi Zhou, Jingbo Zhou, Yang Zhou, Huaiyu Dai, and Dejing Dou. 2022b. Multi-job intelligent scheduling with cross-device federated learning. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 34(2):535–551.

Ji Liu, Juncheng Jia, Hong Zhang, Yuhui Yun, Leye Wang, Yang Zhou, Huaiyu Dai, and Dejing Dou. 2024d. Efficient federated learning using dynamic update and adaptive pruning with momentum on shared server data. *ACM Transactions on Intelligent Systems and Technology*.

Ji Liu, Zhihua Wu, Danlei Feng, Minxu Zhang, Xinxuan Wu, Xuefeng Yao, Dianhai Yu, Yanjun Ma, Feng Zhao, and Dejing Dou. 2023a. Heterps: Distributed deep learning with reinforcement learning based scheduling in heterogeneous environments. *Future Generation Computer Systems*, 148:106–117.

Ji Liu, Xuehai Zhou, Lei Mo, Shilei Ji, Yuan Liao, Zheng Li, Qin Gu, and Dejing Dou. 2023b. Distributed and deep vertical federated learning with big data. *Concurrency and Computation: Practice and Experience*, 35(21):e7697.

Xiangyang Liu, Tianxiang Sun, Xuanjing Huang, and Xipeng Qiu. 2022c. Late prompt tuning: A late prompt could be better than many prompts. In *Findings of the Association for Computational Linguistics: EMNLP*, pages 1325–1338.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022d. P-tuning:

Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 61–68. Volume 2: Short Papers.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

Xiaofeng Liu, Yinchuan Li, Qing Wang, Xu Zhang, Yunfeng Shao, and Yanhui Geng. 2023c. Sparse personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–15.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Roberta: A robustly optimized BERT pretraining approach. In *Int. Conf. on Learning Representations (ICLR)*, pages 1–15.

Alexander Ly, Maarten Marsman, Josine Verhagen, Raoul PPP Grasman, and Eric-Jan Wagenmakers. 2017. A tutorial on fisher information. *Journal of Mathematical Psychology*, 80:40–55.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. LLM-pruner: On the structural pruning of large language models. In *Conf. on Neural Information Processing Systems (NeurIPS)*, pages 1–19.

Tanguy Marchand, Regis Loeb, Ulysse Marteau-Ferey, Jean Ogier du Terrail, and Arthur Pignet. 2023. SRATTA: sample re-attribution attack of secure aggregation in federated learning. In *Int. Conf. on Machine Learning (ICML)*, volume 202, pages 23886–23914.

James Martens. 2020. New insights and perspectives on the natural gradient method. *The Journal of Machine Learning Research*, 21(1):5776–5851.

James Martens and Roger Grosse. 2015. Optimizing neural networks with kronecker-factored approximate curvature. In *Int. Conf. on Machine Learning (ICML)*, pages 2408–2417.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics (AISTATS)*, pages 1273–1282.

Joe Mellor, Jack Turner, Amos Storkey, and Elliot J Crowley. 2021. Neural architecture search without training. In *Int. Conf. on Machine Learning (ICML)*, pages 7588–7598.

Official Journal of the European Union. 2016. General data protection regulation. https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679. Online; accessed 09/05/2022.

Yann Ollivier. 2015. Riemannian metrics for neural networks i: feedforward networks. *Information and Inference: A Journal of the IMA*, 4(2):108–153.

OpenAI. 2022. Chatgpt. https://openai.com/blog/chatgpt. Online; accessed 05/02/2024.

Kazuki Osawa, Shigang Li, and Torsten Hoefler. 2023. Pipefisher: Efficient training of large language models using pipelining and fisher information matrices. *Machine Learning and Systems*, 5.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 271–278.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 115–124.

Razvan Pascanu and Yoshua Bengio. 2013. Revisiting natural gradient for deep networks. In *Int. Conf. on Learning Representations (ICLR)*, pages 1–18.

Keqin Peng, Liang Ding, Qihuang Zhong, Yuanxin Ouyang, Wenge Rong, Zhang Xiong, and Dacheng Tao. 2023. Token-level self-evolution training for sequence-to-sequence learning. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 841–850.

Krishna Pillutla, Kshitiz Malik, Abdelrahman Mohamed, Michael G. Rabbat, Maziar Sanjabi, and Lin Xiao. 2022. Federated learning with partial model personalization. In *Int. Conf. on Machine Learning (ICML)*, volume 162, pages 17716–17758.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. In *North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 1162–1172.

Ofir Press, Noah A. Smith, and Mike Lewis. 2021. Shortformer: Better language modeling using shorter inputs. In *Annual Meeting of the Association for Computational Linguistics and Int. Joint Conference on Natural Language Processing (ACL/IJCNLP)*, pages 5493–5505.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2383–2392.

Mehdi Setayesh, Xiaoxiao Li, and Vincent WS Wong. 2022. Perfedmask: Personalized federated learning with optimized masking vectors. In *Int. Conf. on Learning Representations (ICLR)*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1631–1642.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. LLama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Saeed Vahidian, Sreevatsank Kadaveru, Woonjoon Baek, Weijia Wang, Vyacheslav Kungurtsev, Chen Chen, Mubarak Shah, and Bill Lin. 2023. When do curricula work in federated learning? In *IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, pages 5061–5071.

Jesse Vig and Yonatan Belinkov. 2019. Analyzing the structure of attention in a transformer language model. *arXiv preprint arXiv:1906.04284*.

Ellen M. Voorhees and Dawn M. Tice. 2000. Building a question answering test collection. In *Annual International ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR)*, pages 200–207.

Chenguang Wang, Mu Li, and Alexander J. Smola. 2019. Language models with transformers. *arXiv preprint arXiv:1904.09408*.

Ziheng Wang, Jeremy Wohlwend, and Tao Lei. 2020. Structured pruning of large language models. In *Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6151–6162.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Trans. Assoc. Comput. Linguistics*, 7:625–641.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Lang. Resour. Evaluation*, 39(2-3):165–210.

Tongtong Wu, Massimo Caccia, Zhuang Li, Yuan Fang Li, Guilin Qi, and Gholamreza Haffari. 2022a. Pretrained language model in continual learning: A comparative study. In *Int. Conf. on Learning Representations (ICLR)*.

Xueyu Wu, Xin Yao, and Cho-Li Wang. 2021. Fedscr: Structure-based communication reduction for federated learning. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 32(7):1565–1577.

Zhuofeng Wu, Sinong Wang, Jiatao Gu, Rui Hou, Yuxiao Dong, V. G. Vinod Vydiswaran, and Hao Ma. 2022b. IDPG: an instance-dependent prompt generation method. In *Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, (NAACL)*, pages 5507–5521.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2023. Sheared LLaMA: Accelerating language model pre-training via structured pruning. In *Workshop on Advancing Neural Network Training: Computational Efficiency, Scalability, and Resource Optimization (WANT@NeurIPS)*, pages 1–22.

M Xu, D Cai, Y Wu, X Li, and S Wang. 2024. Fwdllm: Efficient fedllm using forward gradient. *arXiv preprint arXiv:2308.13894*.

Zifan Xu, Yulin Zhang, Shahaf S. Shperberg, Reuth Mirsky, Yuqian Jiang, Bo Liu, and Peter Stone. 2022. Model-based meta automatic curriculum learning. In *Decision Awareness in Reinforcement Learning Workshop at ICML*.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, Zhiyuan Liu, Peng Zhang, Yuxiao Dong, and Jie Tang. 2023. GLM-130b: An open bilingual pre-trained model. In *Int. Conf. on Learning Representations (ICLR)*.

Hong Zhang, Ji Liu, Juncheng Jia, Yang Zhou, Huaiyu Dai, and Dejing Dou. 2022. Fedduap: Federated learning with dynamic update and adaptive pruning using shared data on the server. In *Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 2776–2782. Int. Joint Conf. on Artificial Intelligence Organization.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024. Tinyllama: An open-source small language model. *arXiv preprint arXiv:2401.02385*.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023. Adaptive budget allocation for parameter-efficient fine-tuning. In *Int. Conf. on Learning Representations (ICLR)*.

Zeru Zhang, Jiayin Jin, Zijie Zhang, Yang Zhou, Xin Zhao, Jiaxiang Ren, Ji Liu, Lingfei Wu, Ruoming Jin, and Dejing Dou. 2021. Validating the lottery ticket hypothesis with inertial manifold theory. *Advances in Neural Information Processing Systems (NeurIPS)*, 34.

Haodong Zhao, Wei Du, Fangqi Li, Peixuan Li, and Gongshen Liu. 2023a. Fedprompt: Communication-efficient and privacy-preserving prompt tuning in federated learning. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Jujia Zhao, Wenjie Wang, Chen Xu, Zhaochun Ren, See-Kiong Ng, and Tat-Seng Chua. 2024. Llm-based federated recommendation. *arXiv preprint arXiv:2402.09959*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023b. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Chendi Zhou, Ji Liu, Juncheng Jia, Jingbo Zhou, Yang Zhou, Huaiyu Dai, and Dejing Dou. 2022. Efficient device scheduling with multi-job federated learning. In *AAAI Conf. on Artificial Intelligence (AAAI)*, volume 36, pages 9971–9979.

Algorithm 2 FedAvg

Input:
  t: The number of current round
  $\mathcal{P}^{t-1}$: The global LoRA parameters at Round $t-1$
  T: The maximum number of rounds
  K: The number of devices
  $D = \{D_1, D_2, ..., D_K\}$: The set of datasets on each device
  $\lambda = \{\lambda_1, \lambda_2, ..., \lambda_T\}$: The learning rates
Output:
  $\mathcal{P}^t$: The global LoRA parameters at Round t
  1: **for** k in $\{1, 2, ..., K\}$ (in parallel) **do**
  2: $\mathcal{P}_k \leftarrow \mathcal{P}^{t-1}$
  3: **for** $B_j \in D_k$ **do**
  4:    $\mathcal{P}_k \quad\quad \leftarrow \quad\quad \mathcal{P}_k -$
       $\lambda^t \sum_{s_i \in B_j} \nabla_{\mathcal{P}_k} f(\mathcal{M}, \mathcal{P}_k, s_i, m_i)$
  5: **end for**
  6: **end for**
  7: $m^t \leftarrow \sum_{k=1}^K |D_k|$
  8: $\mathcal{P}^t = \sum_{k=1}^K \frac{|D_k|}{m^t} \mathcal{P}_k$

A FedAvg Update

The original FedAvg update is shown in Algorithm 2.

B Gradient Calculation in LoRA

During the local training, we update LoRA parameters at Epoch t for data samples s_i as follows:

$$
\begin{aligned}
\mathcal{P}_A^t &= \mathcal{P}_A^{t-1} - \lambda^t \nabla_{\mathcal{P}_A^{t-1}} f(\mathcal{M}, \mathcal{P}_k, s_i, m_i) \\
\mathcal{P}_B^t &= \mathcal{P}_B^{t-1} - \lambda^t \nabla_{\mathcal{P}_B^{t-1}} f(\mathcal{M}, \mathcal{P}_k, s_i, m_i),
\end{aligned} \tag{13}
$$

and for a batch of data samples B_j as follows:

$$
\begin{aligned}
\mathcal{P}_A^t &= \mathcal{P}_A^{t-1} - \lambda^t \sum_{s_i \in B_j} \nabla_{\mathcal{P}_A^{t-1}} f(\mathcal{M}, \mathcal{P}_k, s_i, m_i) \\
\mathcal{P}_B^t &= \mathcal{P}_B^{t-1} - \lambda^t \sum_{s_i \in B_j} \nabla_{\mathcal{P}_B^{t-1}} f(\mathcal{M}, \mathcal{P}_k, s_i, m_i),
\end{aligned} \tag{14}
$$

where λ^t is the learning rate, $\mathcal{P}_k$ is the combination of $\mathcal{P}_A$ and $\mathcal{P}_B$. To get the gradient of reconstructed full-rank matrix at iteration t, we can calculate as follows:

$$
\nabla_{\mathcal{P}_k^t} f(\mathcal{M}, \mathcal{P}_k, s_i, m_i) = \mathcal{P}_A^t \mathcal{P}_B^t - \mathcal{P}_A^{t-1} \mathcal{P}_B^{t-1} \tag{15}
$$

C Formulas for Curriculum Federated Learning

We can calculate the difficulty score of a data sample as defined in Formula 16.

$$
\int_i = Tr(\tilde{\mathbf{F}}_i), \tag{16}
$$

where $\int_i$ is the difficulty score of data sample s_i. Then, the difficulty score of a batch of data samples can be calculated based on Formula 17:

$$
\int_j = \sum_{s_i \in B_j} Tr(\tilde{\mathbf{F}}_i), \tag{17}
$$

where $\int_j$ is the difficulty score of the batch B_j. The more significant the score is, the more difficulty the batch of data samples is. We calculate the difficulty score based on the initial model as the difficulty of data samples corresponds to negligible change during the fine-tuning process (Platanios et al., 2019; Li et al., 2022a). $\int_i$ can be computed using the square of the elements in the diagonal of the first-order derivative matrix, which can avoid the computation of the full FIM with the heavy multiplication of two matrices. This mechanism makes the calculation feasible in terms of computation time and memory consumption. Please note that FIM is calculated and stored locally, which does not need to be transferred to the server.

We calculate $\mathcal{B}_k^t$ based on Formula 18.

$$
\mathcal{B}_k^t = (\beta + (1 - \beta)\frac{t}{\alpha T})\frac{n_k}{\mathrm{B}}, \tag{18}
$$

where β represents the initial training sample ratio, α denotes the ratio of training epoch until all data is used, B refers to the batch size. Both β and α are hyper-parameters within the range of $[0, 1]$. Then, the batch of data samples is selected based on Formula 19:

$$
Select^t(B_j) = \begin{cases} True & \text{if } j < \mathcal{B}_k^t \\ False & \text{otherwise}, \end{cases} \tag{19}
$$

where $Select^t(B_j)$ represents the selection decision. When $Select^t(B_j) = True$, the batch of data samples is selected for local update. With this curriculum training strategy, the local training can learn from easy samples to challenging samples, which can achieve excellent performance with fewer data samples in early iterations.

D Novelty of FibecFed

Different from the existing approaches, we exploit the fisher information to measure both the complexity of training data and the importance of the components, e.g., neurons, in LLMs. We enable curriculum learning based on the complexity of the training data within each device to achieve superb accuracy. We exploit LoRA to achieve efficient fine-tuning with only a few parameters. In addition, we split the trainable parameters into three parts, each of which is for global aggregation, local update, and as frozen neurons on each device, respectively. Only the parameters of the global aggregation part are updated and synchronized among multiple devices during the fine-tuning process. We consider the relative change of each layer and get an excellent estimation of important layers to generate the part for global aggregation. We exploit the momentum of parameter updating and get a robust estimation of important neurons in order to select important neurons for local update.

E Prompt-tuning in LLM

When exploiting the prompt tuning, the parameters of the networks to generate prompts or prefix are adjusted instead of the parameters of LLMs. The prompt or the prefix is the added instruction concatenated to the input, which can guide LLMs to generate proper answers. For instance, a prompt ("This is [MASK]") can be concatenated to the input ("Wonderful movie!") to be sent to a LLM, which generates the label ("positive" or "negative") for a sentiment analysis task.

F Notations

In this paper, we use the notations summarized in Table 3.

G Experiment results

In this section, we present the details of extensive experiments. In Section G.1, we explain the details of experimental setup. In Section G.3, we compare the efficiency in terms of time to achieve target accuracy. In Section G.4, we show the convergence accuracy over 10 different datasets. In SectionG.5, we present the robustness of FibecFed respect to scalability and data heterogeneity. In Section G.6, we demonstrate the performance with learning rates. In Section G.7, we discuss the impact of different curriculum strategies.

G.1 Experimental Setup

We present the hyper-parameters used in the fine-tuning process in Table 8. We set the global training epochs to 100, except for QNLI, SST-2. We follow the Dirichlet distribution (with 1 as concentration α) to partition the whole data into splits and assign a certain number of samples based on Dirichlet distribution (with $\alpha = 5$), development sets are served as test data to evaluate performance in the GLUE benchmark. For 4 other datasets, we select a certain number of samples from the training set as the development set, and the number of samples for each label is determined according to its original label distribution of training set. For datasets in GLUE benchmark, we use their original data splits. For 4 other datasets with no default splits, we randomly split the dataset into train, development, and test sets.

With 100 devices, the number of samples ranges from 346 to 2607 for QNLI, 222 to 1676 for SST-2, 28 to 213 for COLA, 12 to 91 for MRPC, 8 to 62 for RTE, 31 to 235 for BoolQ, 25 to 189 for MPQA, 23 to 174 for Subj, 16 to 123 for Trec and 25 to 191 for MR. Additionally, a detailed number of samples distributed among 10 devices with the MRPC dataset is shown in Table 4.

RoBERTa$_{\text{LARGE}}$ consists of 24 layers of transformers followed by a classification head, which contains 355M parameters. LLaMA is composed of 32 transformer layers with 7B parameters.

The centralized methods (Adapter, P-tuning v2, IDPG, ATTEMPT, LPT, LoRA, Shortformer, VOC, SLW, AdaLoRA, Delta-LoRA) are adapted to the FL setting with FedAvg (McMahan et al., 2017) for a fair comparison. In addition, we exploit the LoRA with FL opmization-based methods, i.e., PFedGate, FedDst and FedALT.

G.2 Comparison with Random Data Selection

The heuristic based methods or mode-oriented methods cannot address the heterogeneity issue across local data samples with variance in such metric. To show the effectiveness of our proposed Fisher information-based metric, we implement another baseline method using random selecting in the curriculum data selection strategy. Table 5 shows the performance under different selecting strategies. Compared with other selecting method, FibecFed outperforms other baselines up to 8.51% in terms of accuracy, which demonstrates the effectiveness of proposed method. In addition, Ran-

dom corresponds to the lowest accuracy compared other methods, i.e., ShortFormer, SLW, Voc, and FibecFed. Table 6 also indicates that FibecFed achieves the target accuracy of 85% on the MRPC dataset in less time(up to 92.49% faster) compared with other selection strategies. It is noteworthy that the random method reaches the accuracy 85% at epoch 95, significantly increasing the required time. This will be clarified in the revised version.

G.3 Fine-tuning Efficiency

In order to show the efficiency of FibecFed, we present the time to achieve target accuracy in Table 7.

G.4 Accuracy & Time Figures in RoBERTa$_{LARGE}$

Figures 2 - 5 present the convergence process of FibecFed and other baselines on COLA, QNLI, SST-2, MRPC, RTE, BOOLQ, MPQA and Subj datasets. Figure 2 shows the convergence results over FibecFed and benchmarks on COLA, QNLI and SST-2 datasets, Figure 3 presents the results on MRPC, RTE and BOOLQ datasets, Figure 4 demonstrates the evaluation results on MPQA, Subj datasets and Figure 5 shows the performance on Trec and MR datasets.

G.5 Accuracy & Time for Robustness & Scalability

Figure 6 presents results of robustness of FibecFed. Figure 6(a) shows accuracy of FibecFed with different learning rate. Figure 6(b) shows the performance under varying client number. In addition, Figure 6(c) indicates FibecFed is robust to different degree of heterogeneity of data.

G.6 Impact of learning rate

As shown in Figure 6(a). the performance of FibecFed on MRPC dataset differs (up to 2.22%) among 4 varying learning rates from $1e^{-4}$ to $8e^{-4}$. Thus, we take the learning rate of $\lambda = 8e^{-4}$, which corresponds to the best accuracy 90.59% in practice.

G.7 Impact of Curriculum Strategies

The curriculum strategy controls the speed of exploiting difficult data samples during the fine-tuning process. We consider three strategies, i.e., linear, square (sqrt), and exponential (exp), which

are defined in Formulas 20, 21, and 22, respectively.

$$\mathcal{B}_k^t = (\beta + (1 - \beta)\frac{t}{\alpha T})\frac{n_k}{B}, \quad (20)$$

$$\mathcal{B}_k^t = (\beta + (1 - \beta)\frac{t^2}{\alpha T})\frac{n_k}{B}, \quad (21)$$

$$\mathcal{B}_k^t = (\beta + (1 - \beta)\frac{e^t}{\alpha T})\frac{n_k}{B}, \quad (22)$$

where e represents the Euler's number. As shown in Figure 7(c) the performance of linear (91%) is similar to that of sqrt (91%), which are much higher than exp (83.14%). However, as sqrt incurs more complicated calculation compared with linear, we exploit the linear strategy in the paper.

G.8 Generalization of FibecFed

To further evaluate the generalization of FibecFed on modest models, we conducted image classification task on the CIFAR-10 dataset using a seven-layer Multilayer Perceptron (MLP). As demonstrated in Table 9, our proposed method consistently achieves higher accuracy (from 2.09% to 3.53%) than FedAvg across all epochs. Furthermore, Table 10 illustrates that FibecFed corresponds to less time (from 0.78 to 2.12 times faster) than FedAvg. The result is aligned with the findings reported in the paper, which suggests that FibecFed can be effectively generalized across various machine learning models.

G.9 Low Bound of *K*

In fact, there is no strict lower bound for K (while K should be bigger than 1 for federated learning). As curriculum learning is effective in centralized learning scenarios (Bengio et al., 2009), i.e., $K = 1$, our method retains its efficacy even with the minimal device setting (1). When K is increased, the performance remains high (or even higher because of more data is exploited for the training) as shown in Figure 6(b). In addition, we conducted additional experiments with varying numbers of devices, i.e., 2, 5, and 10, to verify the effect of small device number K. Table 11 shows the performance in terms of accuracy across all tested scenarios on MRPC dataset, where the proposed method achieves an average accuracy of 90.56% and a variance of 0.326. These results demonstrate the robustness of our approach across different K values.
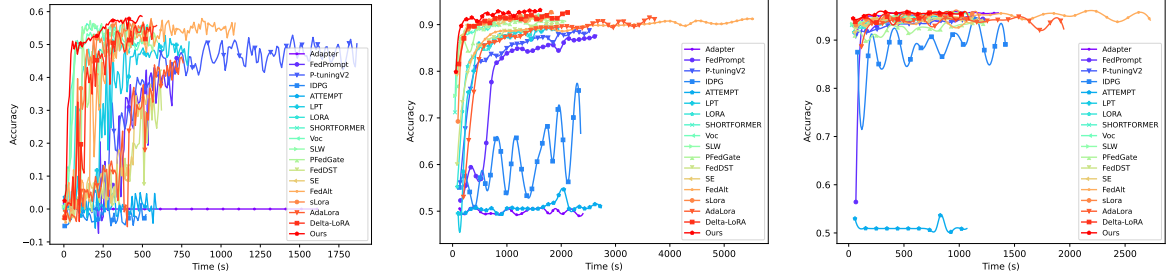
(a) Acc & COLA   (b) Acc & QNLI   (c) Acc & SST-2

Figure 2: The accuracy and training time with FibecFed and diverse baseline approaches.
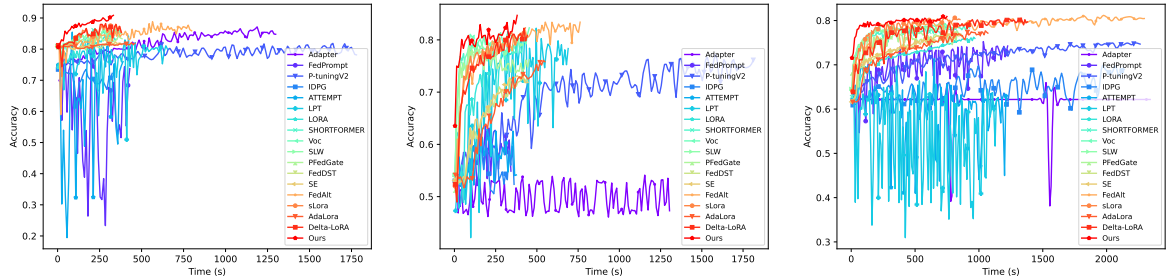
(a) Acc & MRPC   (b) Acc & RTE   (c) Acc & BOOLQ

Figure 3: The accuracy and training time with FibecFed and diverse baseline approaches.
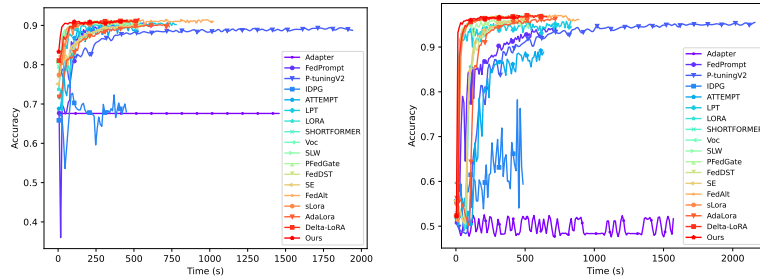
(a) Acc & MPQA   (b) Acc & Subj

Figure 4: The accuracy and training time with FibecFed and diverse baseline approaches.
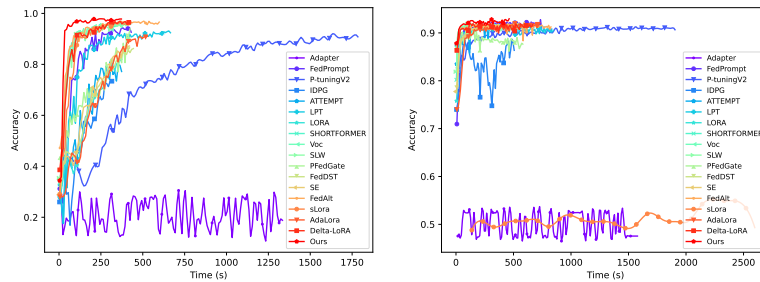
(a) Acc & Trec   (b) Acc & MR

Figure 5: The accuracy and training time with FibecFed and diverse baseline approaches.

G.10 Effect of Initial Sampling Rate

As shown in Table 12, a proper initial sample ratio can achieve optimal performance. when B_k^t is small, the model can learn nothing from beginning. However, when B_k^t is large, too many hard examples are exposed to model and might force model

generated some bad gradients. Both cases will lead to a poor quality of aggregation on server side, hence degrading the final convergent performance.

| (a) Acc & Lr | (b) Acc & # of devices | (c) Acc & Data heterogeneity |

Figure 6: The accuracy and training time with FibecFed under different settings

| (a) Acc & Score function | (b) Acc & # of transfer layers | (c) Acc & Curriculum strategy |
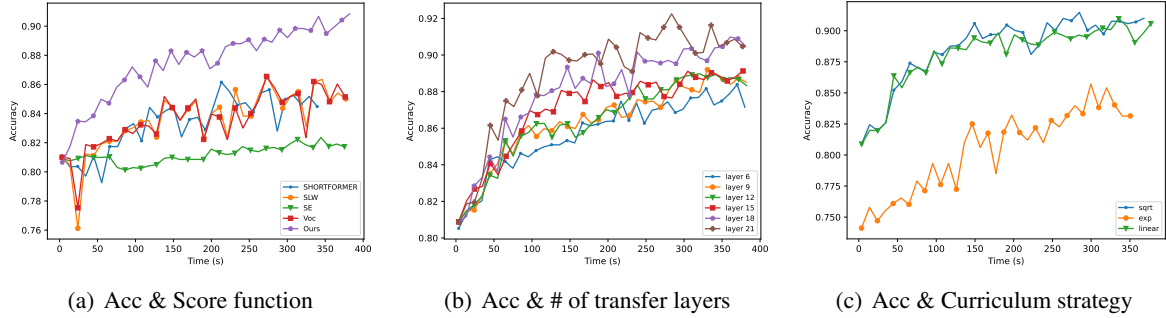
Figure 7: The accuracy and training time with FibecFed under different settings
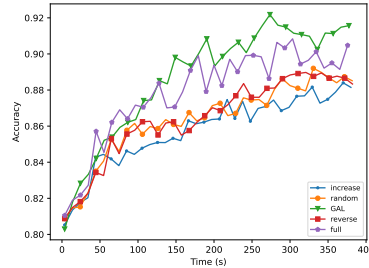
Figure 8: The accuracy and training time with FibecFed with diverse data selection strategies.

H Analysis of FibecFed

In this section, we present the analysis of FibecFed, including the motivation, the training efficiency, fake difficulty scores, task fairness, and the significance of each component.

H.1 Motivation

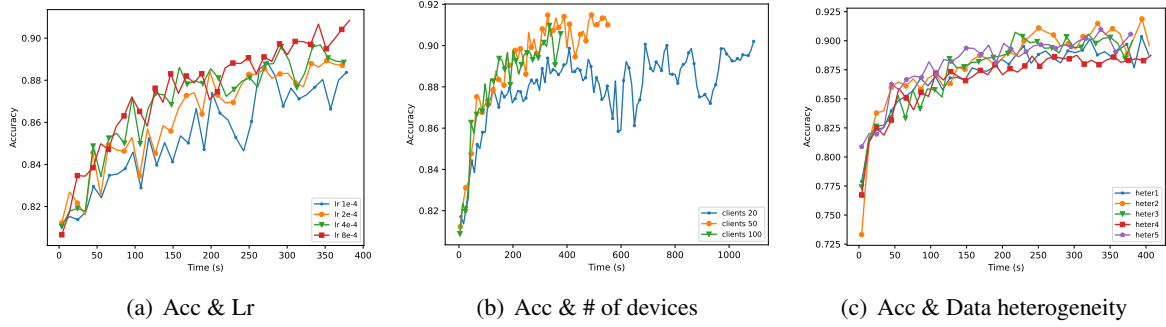While stringent legal regulations are carried out to protect the security and the privacy of decentralized raw data, federated learning becomes promising to enable the collaborative training process without aggregating the raw data into a centralized data center. While LLMs are too large to be directly trained with federated learning, we propose adaptive federated curriculum learning and efficient sparse parameter update with LoRA to enable the federated learning of LLMs while reducing the communication costs so as to improve the training speed (up to 98.61% faster) and improving the accuracy (up

to 45.35%). The use case is to enable the inference process of LLMs on edge devices, while the LoRA parameters are updated. The LoRA parameters are much smaller than the full model, the training of which is feasible on edge devices. However, the full update and aggregation of all the layers within the LoRA parameters of LLMs in traditional federated learning still takes much time and the performance is inferior. Our approach can well improve the training (LoRA fine-tuning) process and the performance in the setting of federated learning of LLMs.

H.2 Curriculum learning & Efficiency of the training

Our proposed method, i.e., efficient curriculum federated learning, can significantly improve the efficiency of the training instead of reducing the efficiency. As explained in Section 5.6, we conduct a comparative analysis among four curriculum

strategies: SLW, VOC, Shortformer, SE, and that without curriculum (NULL). FibecFed corresponds to superior performance in terms of accuracy (up to 5.73%, 9.12% 5.84%, 6.41%, 7.7% compared with Voc, SE, SLW, Shortformer, and NULL, respectively) and efficiency (up to 26.53%, 34.26% 68.92%, 68.36%, 58.57% compared with Voc, SE, SLW, Shortformer, and NULL, respectively). The efficient curriculum federated learning can reduce the total number of batches participating in the training process so as to reduce local training time. In addition, the curriculum federated learning still carries out the training process in parallel, i.e., the training process in each client is performed in parallel. While it may take some time to calculate the score of each batch, i.e., $\int_i$, the calculation is carried out in parallel on each device, and takes negligible time (less than 2.98%) compared with the training time. Once the difficulty score is determined, each training batch is sorted in ascending order and indexed starting from 0. Once the difficulty score is calculated, the training data is selected on each device based on Formula 9 for the training process of each epoch, and the selected batches of data samples are much smaller compared with those without curriculum learning at the beginning (when $j < \mathcal{B}_k^t$). The process of batch selection takes negligible time (a few microseconds) compared with the reduced training time brought by the reduced data samples. In addition, the curriculum federated learning strategy can well improve the accuracy of the trained model.

H.3 Curriculum Learning with Fake Difficulty Score

The Fisher Information Matrix (FIM) is calculated and stored locally, which does not need to be transferred to the server. FIM is utilized to calculate the difficult score so as to locally select the data samples based on our curriculum learning strategy on each device. When a device tricks the training by providing updates with fake lower FIM pretending to use a simpler dataset, the curriculum learning strategy can select the simple samples among the local dataset, which can reduce the training time and improve the training efficiency as well. In this work, our approaches focuses on improving the training efficiency and reducing the communication costs in federated learning of large language models based on our efficient curriculum learning method and efficient sparse parameter update, while the security issues may be addressed in future

work.

H.4 Task Fairness

Our efficient curriculum federated learning method does not incur unfair process of the tasks in devices. The curriculum strategy selects the data within each selected device. It does not change the device sampling mechanism, which determines which device participates in which epoch. Our efficient curriculum federated learning method guides the model to learn from simpler to more complex samples, facilitating stable and efficient convergence. In Federated Learning (FL), the inherent data heterogeneity often results in significant gradient divergence, especially when the data heterogeneity is severe. Our curriculum federated learning method can mitigate this effect, leading to more effective model aggregation and improve overall performance.

H.5 Communication overhead

The absolute communication overhead is shown in the Table 13 (with RoBERTa$_{\text{LARGE}}$). The communication overhead of FibecFed is higher than that of FedPrompt (up to 3.51 times), IDPG (up to 3.3 times), and ATTEMPT (up to 1.85 times). This is expected as the these three methods are prompt tuning-based methods, which corresponds to much fewer parameters to update during the training phase compared with FibecFed. However, these three methods correspond to significantly lower performance (compared with FibecFed), i.e., low convergence accuracy (from 1.25% to 38.68% for FedPrompt, from 6.6% to 53.98% for IDPG, and from 1% to 53.94% for ATTEMPT), as shown in Table 1. The communication overhead of FibecFed is significantly lower than the other methods (6.25 times for Adapter, 9.67 times for P-tuning V2, 1.9 times for LPT, and 25% for LORA, SHORTFORMER, Voc, SLW, PFedGate, FedDST, SE, FedAlt, sLora, AdaLora, Delta-LoRA). This is expected as well as FibecFed only transfers the global aggregation layers instead of the parameters of all the layers in LLM.

In addition, the relative communication overhead (i.e., the ratio between the absolute communication overhead and the total training time) is shown in Table 14. Similar to the absolute communication overhead, the relative communication overhead of FibecFed is higher than that of FedPrompt (from 2.3% to 37.4%), IDPG (from 2.4% to 34.4%), and ATTEMPT (from 1.9% to 31.7%) as well. This is expected as explained before. As the

total training time of FibecFed becomes shorter, the relative communication overhead of FibecFed becomes slightly more significant than FedAlt (from 1.4% to 20.1%) and AdaLora (from 0.8% to 5.7%). In addition, the relative communication overhead of FibecFed becomes similar to that of sLora (from 7.8% smaller to 24.8% bigger) and Delta-LoRA (from 7.5% smaller to 5.5% bigger). The relative communication overhead of FibecFed is still smaller than the rest approaches, i.e., Adapter (up to 43.9%), P-tuning V2 (up to 55.6%), LPT (up to 25.1%), LORA (up to 14.5%), SHORTFORMER (up to 13.8%), Voc (up to 8.0%), SLW (up to 8.8%), PFedGate (up to 7.0%), FedDST (up to 7.1%), SE (8.1%). Please note that FibecFed corresponds to higher convergence accuracy (as shown in Table 1 in the manuscript) and shorter training time (as shown in Table 2 in the manuscript).

H.6 Significance of Each Component

Our approach aims to improve the efficiency of LLM federated learning in two perspective: communication cost and local training efficiency, each of which plays a significant role in accelerating the training process of FL. In contrast to heuristic metrics, we utilize the FIM to assess sample difficulty, achieving an accurate estimation of difficulty score. This newly proposed metric enhances Curriculum Learning by facilitating a more stable and faster convergence training, reducing the number of epochs required for local training. Consequently, this approach reduces local training time and improves overall efficiency, offering a significant enhancement over traditional difficulty assessment methods. In addition, A novel noise-sensitive Layer Selection is proposed to identify the critical layers of the model, thereby reducing communication overhead without compromising performance. Furthermore, A novel parameter selection strategy focuses on identifying key neurons within the model for local update. By determining which parameters are most influential, this approach enhances local training efficiency, optimizes computational resources and speeds up the learning process on individual devices.

In Section 5.6 (Ablation Study), we demonstrate the ablation study in terms of the curriculum data selection method, the important layer selection method, and the local update parameter selection method. The experimentation reveals that the efficient curriculum federated learning corresponds to superior performance in terms of both accuracy (up

to 7.7%) and efficiency (up to 68.92%). In addition, the important layer selection method can significantly improve the efficiency (up to 23.1%) with slight improvement of accuracy (up to 3.42%) and the local update parameter selection method can further improve the efficiency up to 11.8% with slightly higher efficiency (up to 2.48%).

In our approach, efficient curriculum federated learning enables the training process begins with simple data samples and then gradually increase the difficulty, which can improve both the efficiency and the performance. The efficient sparse parameter update method is composed of global layer selection and local update parameter selection. The global layer selection method reduces communication costs by only transferring the layers of importance scores between devices and the server, without performance degradation. The local update parameter selection can improve the local training efficiency by only updating the important parameters. Our approach yields excellent performance (up to 45.35% in terms of accuracy) and superb fine-tuning speed (up to 98.61% faster).

H.7 Combination with Model Compression

Our approach is orthogonal with model compression methods. Various model compression methods exist, e.g., pruning and quantization. Our approach can be combined with these methods to achieve higher training speed or higher accuracy. However, the pruning and quantization methods may degrade the performance (accuracy) of LLMs. In addition, the training process based on the model compression methods may require additional low-level training operators in the forward propagation and back propagation, which incurs extra complexity. Thus, the combination of our approach with the model compression methods can be addressed in our future work.

H.8 LLM Training on Devices

In real-world scenarios, numerous governments and organizations have established regulations and laws to protect data privacy, making access to data on edge devices increasingly challenging. Moreover, pre-trained models are often not well-suited for domain-specific tasks due to lack of fine-tuning. By implementing our proposed approach, the LLMs can be fined based on the distributed raw data stored on diverse edge devices without aggregating the raw data into a central server or a central data center. This approach not only ad-

heres to privacy and regulatory standards but also ensures that LLMs can be fine-tuned to specific tasks with distributed end user data distributed in edge devices. A classical example is highlighted in (Zhao et al., 2024), where, by leveraging the behavioral data of end users, the server obtains a collaboratively fine-tuned LLM-based recommendation system. Furthermore, with the advancement of LLM technology, additional applications could emerge in highly confidential environments such as hospitals and banks. Our proposed approach provides a solution to develop good task-specific model while ensuring both the integrity and confidentiality of the sensitive raw data of end users on diverse edge devices. All the explanation will be added in our final version.

H.9 Data Heterogeneity & Layer Importance

The inherent heterogeneity of samples on each device contributes to the difference of the importance scores of each layer. In the settings of federated learning, the samples on each device are generally heterogeneous, i.e., non non-Independent and Identically Distributed (non-IID). In order to select the important layers as the global aggregation layers, we aggregate the importance scores on each device based on Formula 11 so as to calculate the global importance scores for each layer. This global aggregation can balance the diverse importance of each layer on each device. Afterward, we select the proper global aggregation layers with a lossless method on each device. This selection can reduce the data to communicate in each epoch so as to improve the efficiency without degrading performance (accuracy).

H.10 Perturbed Parameters & Increased Sensitivity

While the layer selection is conducted on each device, the importance scores of each layer are aggregated, which can balance the importance of layers among different devices. The data heterogeneity may have impact on the global aggregation layer selection within the lossless method, i.e., the number of selected aggregation layers are different across devices.

When more parameters are perturbed locally, the lossless method may result in more global aggregation layers so as to achieve excellent performance (accuracy). When the sensitivity to noise corresponding to specific layers is increased, the impact can be taken into consideration while calculating the importance scores. Afterward, the impact is reflect within the calculation of Formula 15 to generate the global importance score of each layers. Thus, the increase sensitivity to noise may impact the importance of layers. However, the fact of more parameters are perturbed can be captured by our lossless method, which determines a proper number of layers to be selected as the global aggregation layers so as to reduce the number of layers to transfer while ensuring the performance (accuracy) of LLM in federated learning. More parameters are perturbed locally across different clients when their increased sensitivity to noise has impacts on more layers of LLMs. Our proposed global aggregation layer selection with the lossless method can well reduce the communication costs so as to improve the efficiency (training speed) while ensuring the performance (accuracy).

H.11 Implementation of FibecFed

FibecFed is composed of two methods, i.e., adaptive federated curriculum learning and efficient sparse parameter update. While these two methods correspond to in-depth novel technical contributions based on Fisher Information, the implementation of FibecFed is straightforward. While Fisher Information is exploited, we calculate the square of the elements in the diagonal of the first-order derivative matrix to reduce the complexity of the Fisher Information computation as explained in Section 4.2.

In practice, the Fisher Information is implemented to calculate the importance scores of the data and layers in the initialization phase. In addition, the local update parameter selection is carried out in the initialization phase as well. Within the training phase, the curriculum data selection strategy is carried out. The execution can be easily carried out with the parameters, e.g., α, β, as shown in Table 8.

H.12 intuition for Fisher Information

Within the training phase of federated learning, it is pivotal to exploit the informative data and to update the critical parts of the LLM so as to achieve efficient training process. However, the existing methods to evaluate the training data or the LLM is static, which corresponds to inaccurate estimation and low performance. As a measure of the local curvature, Fisher Information Matrix (FIM) defines the Riemannian metric of the parameter space, which can indicate the difficulty of data samples and the

importance of each component of the LLM within the training phase. Fisher information is defined as how sensitive the model is to changes in θ at a particular θ (Ly et al., 2017), where θ represents the value to infer and corresponds to the ground truth output of the LLM in FibecFed. When the Fisher Information is not significant in respect of a certain training sample within the training process of an LLM, i.e., small score defined in Formula 17, the corresponding training data contains relatively few information for the training process and the data is considered easy for the LLM. Then, the model can quickly learn the knowledge within the easy data according to the starting small strategy of the curriculum learning (Bengio et al., 2009). In FibecFed, we exploit Formula 18 to carry out the Fisher Information-based curriculum learning, which can choose easy samples at the beginning to improve the training efficiency while achieving high accuracy. Besides the training data, Fisher Information can indicate the importance of the neuron in LLM within the training phase. When the Fisher Information corresponding to a neuron is significant, we consider it as an important part of the LLM that needs further adjustment (training). Otherwise, the corresponding neurons do not need update as they are not sensitive within the training phase (these neurons may stay unchanged even when they are considered as local update parameters). Thus, we take the neurons of high importance scores based on the Fisher Information (based on Formula 12) as the local update parameters. In this way, Fisher Information guides FibecFed to choose the simple data to begin with and to choose the sensitive neurons in the LLM to update within the training phase so as to achieve efficient and effective federated learning.

H.13 Privacy in FibecFed

Indeed, federated learning is designed to preserve privacy. While FibecFed focuses on improving the efficiency of federated learning, it can further enhance the data privacy without bringing extra privacy issues. Within the training phase of FibecFed, we only transfer the parameters in global aggregation layers between the server and devices, which can avoid transfer the whole model so as to protect the data privacy. In addition, we update the local update parameters instead of the full parameters, which can further avoid privacy and security issues due to potential gradient leakage. Traditional attack methods, e.g., gradient attack, assume that full gradients or models are transferred between the server and devices (Dimitrov et al., 2022; Marchand et al., 2023). Thus, compared with the traditional federated learning approaches that transfer the whole model between the server and devices, FibecFed can further enhance the privacy and security of federated learning.

Within the initialization of FibecFed, only the global aggregation layer selection incurs exchanging the importance scores of each layer in the LLM between the server and devices, while the other two modules, i.e., Fisher information-based curriculum learning and local update parameter selection, does not bring any extra information transfer between the server and devices. The importance scores of layers are insensitive data with few information of the raw training data on each device. The insensitive data resembles the number of samples to be transferred in traditional federated learning approaches, e.g., FedAvg, which still complies with privacy regulations. To the best of our knowledge, there is no attack methods based on the importance scores of each layer in the LLM.

FibecFed can be combined with other security or privacy methods, e.g., encryption, differential privacy, to further protect the data security and privacy in federated learning.

I Potential Risks

We propose an efficient LLM federated learning approach to enable collaborative LLM training with distributed raw data without aggregation, which can protect the privacy of the raw data. Our approach can be combined with other defense approaches or privacy protection methods to further enhance the security or the privacy of federated learning when there are curious or malicious attacks.

Table 3: Summary of main notations.

Notation	Definition
$\mathcal{M}; \mathbf{P}$	The set of LLM parameters; the set of trainable LoRA parameters
$\mathcal{K}; K$	The set of edge devices; the size of $\mathcal{M}$
$\mathcal{D}; N$	The global dataset; the size of $\mathcal{D}$
$\mathcal{D}_k; N_k$	The dataset on Device k; the size of $\mathcal{D}_k$
$\mathcal{F}(\cdot); F_k(\cdot)$	The global loss function; the local loss function on Device k
$s_i; m_i$	The single data sample; it's corresponding label of index i
L	The number of Layers
T	The maximum number of global rounds
η_k	The learning rate on device k
$\mathcal{K}$	The number of participated devices per round
$\mathcal{P}_k$	The set of trainable LoRA parameters for device k
W_0^l	The parameters at Layer l
A_k^l, B_k^l	The LoRA matrices at Layer l on device k
h^l	The hidden values generated at Layer l
$\tilde{\mathbf{F}}_i; \mathbf{F}_i$	The empirical and approximated FIM
$\odot$	The hadamard product operation
I	The identity matrices
Tr	The trace of the matrices
$\int_i; \int_j$	The difficulty score of Sample i and Batch j
ϵ_i	The generated noise for sample i
$p; q$	The dual norm factors
γ	The noise budget
$\mathcal{F}^l(s_i)$	The relative difference of Frobenius norm
$\mathcal{I}^l$	The importance score of Layer l
λ_k	The ith eigenvalues
$\mathscr{L}_k$	The Lipschitz constant
$H_k(\mathcal{P}_k^{\mathcal{T}})$	The Hessian Matrix with respect to $\mathcal{P}_k$ at round $\mathcal{T}$ on device k
$\mathbf{F}_k^t$	The FIM on Device k at Round t
$\tilde{\mathbf{F}}_k^t$	The empirical average diagonal approximation of $\mathbf{F}_k^t$
$\mathbf{F}_{k,l}^{\mathcal{T}'}$	The FIM on Device k at Round t for layer l
$\mathbf{F}_{k,l}^{\mathcal{T}'}[\nu]$	The vth diagonal element in $\mathbf{F}_{k,l}^{\mathcal{T}'}$
$R; r$	The Rank of the Hessian Matrix; the index of eigenvalue.
$\mathcal{P}_A^t; \mathcal{P}_B^t$	The fist and second part of LoRA parameters for device k

Device index	1	2	3	4	5	6	7	8	9	10
Number of samples	422	317	303	303	651	474	270	431	378	119

Table 4: The number of samples on each client for MRPC dataset.

Method	Accuracy
ShortFormer	86.05%
SLW	86.58%
Voc	86.49%
Random	85.56%
FibecFed	90.84%

Table 5: The accuracy under different sample selection strategies with RoBERTA-Large model on mrpc dataset.

Method	Time
ShortFormer	173
SLW	177
Voc	118
Random	786.6
FibecFed	59

Table 6: The time(seconds) under different sample selection strategies to achieve target accuracy 85% with RoBERTA-Large model on mrpc dataset.

Method	QNLI	SST-2	CoLA	MPRC	RTE	BoolQ	MPQA	Subj	Trec	MR
Adapter	/	70	/	529	/	/	/	/	/	/
FedPrompt	2381	221	/	/	317	1121	130	/	317	95
P-tuning v2	1365	731	894	/	415	1586	429	1384	/	992
IDPG	/	/	/	/	/	/	/	/	73	/
ATTEMPT	/	/	/	/	/	/	361	/	/	125
LPT	697	279	600	/	82	/	92	246	/	177
LORA	663	276	230	193	110	665	210	278	96	130
SHORTFORMER	193	359	65	76	17	66	31	94	204	38
VOC	111	99	71	89	14	45	32	71	142	31
SLW	113	194	69	60	17	45	37	71	178	37
PFedGate	85	/	110	146	28	71	33	70	421	82
FedDST	686	105	/	/	268	427	228	315	/	189
SE	100	153	115	/	230	378	194	283	/	198
FedALT	314	131	288	127	48	108	46	69	157	39
sLORA	170	64	145	90	45	72	69	84	189	/
adaLORA	875	271	738	/	296	507	265	344	/	227
Delta-LoRA	201	58	180	59	50	123	46	75	188	66
Ours	**61**	**39**	**50**	**28**	**8**	**22**	**14**	**25**	**46**	**29**

Table 7: The fine-tuning time (s) to achieve a target accuracy (87% for QNLI, 93% for SST-2, 50% for CoLA, 83% for MRPC, 70% for RTE, 74% for BoolQ, 88% for MPQA, 95% for Subj, 94% for Trec, and 91.1% for MR) with FibecFed and diverse baseline approaches. "/" represents that fine-tuning does not achieve the target accuracy. The best results are highlighted in **bold** and the second best ones are marked with underline. The results are obtained using RoBERTa$_{\text{LARGE}}$.

Method	QNLI	SST-2	CoLA	MPRC	RTE	BoolQ	MPQA	Subj	Trec	MR
batch size	8	8	8	8	8	8	8	8	8	8
non-IID degree	5	5	5	5	5	5 5	5	5	5	
epochs	20	20	100	100	100	100	100	100	100	100
local iteration	2	2	2	2	2	2	2	2	2	2
learning rate	4e-4	4e-4	3e-4	8e-4	4e-4	8e-4	2e-4	2e-4	8e-4	1e-4
β	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6
pace function	linear	linear	linear	linear	linear	linear	linear	linear	linear	linear
number of layers in GAL	18	18	18	18	18	18	18	18	18	18
clients num	100	100	100	100	100	100	100	100	100	100
clients num per round	10	10	10	10	10	10	10	10	10	10
α	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8

Table 8: The hyperparameters settings for each dataset

Method/Epoch number	5	20	35	50
FedAvg	26.07%	36.59%	39.85%	41.13%
FibecFed	29.60%	38.85%	41.94%	43.74%

Table 9: The accuracy of FibecFed and FedAvg with MLP across different epochs on cifar-10 dataset

Method/Epoch number	20	30	40
FedAvg	13.06	65.3	241.61
FibecFed	4.18	25.05	135.43

Table 10: The time (seconds) of FibecFed and FedAvg with MLP to achieve different target accuracy 40% on cifar-10 dataset

Number of devices K	Accuracy
2	91.64%
5	89.48%
10	91.19%
20	90.20%
50	91.0%
100	90.56%
110	90.75%

Table 11: The accuracy under varying number of devices on the RoBERTA-Large model.

initial sample ratio	Accuracy	Time
0.05	88.55%	139
0.1	89.40%	141
0.2	89.18%	147
0.4	89.93%	162
0.6	90.57%	175
1.0	89.80%	205

Table 12: The accuracy and training time(seconds) under varying initial sample ratios with RoBERTA-Large model on MRPC dataset.

Method	qnli	sst-2	cola	mrpc	rte	boolq	mpqa	subj	trec	mr
Adapter	217.4	217.4	1086.9	1086.9	1086.9	1086.9	1086.9	1086.9	1086.9	1086.9
FedPrompt	6.7	6.7	33.3	33.3	33.3	33.3	33.3	33.3	33.3	33.3
P-tuning v2	320.0	320.0	1600.0	1600.0	1600.0	1600.0	1600.0	1600.0	1600.0	1600.0
IDPG	7.0	7.0	34.9	34.9	34.9	34.9	34.9	34.9	34.9	34.9
ATTEMPT	10.5	10.5	52.6	5.26	52.6	52.6	52.6	52.6	52.6	52.6
LPT 87.0	87.0	434.8	434.8	434.8	434.8	434.8	434.8	434.8	434.8	434.8
LORA 40.0	40.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0
Shortformer	40.0	40.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0
VOC	40.0	40.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0
SLW	40.0	40.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0
PFedGate	40.0	40.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0
FedDST	40.0	40.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0
SE	40.0	40.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0
FedALT	40.0	40.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0
sLORA	40.0	40.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0
adaLORA	40.0	40.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0
Delta-LoRA	40.0	40.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0
FibecFed (ours)	30.0	30.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0

Table 13: Absolute communication overhead of FibecFed and diverse baseline approaches. The time unit is second.

Method	qnli	sst-2	cola	mrpc	rte	boolq	mpqa	subj	trec	mr
Adapter	0.091	0.161	0.674	0.837	0.833	0.464	0.746	0.692	0.815	0.689
FedPrompt	0.003	0.005	0.046	0.078	0.097	0.027	0.054	0.046	0.077	0.045
P-tuning v2	0.127	0.260	0.858	0.899	0.883	0.707	0.824	0.741	0.897	0.840
IDPG	0.003	0.005	0.067	0.107	0.096	0.016	0.078	0.072	0.111	0.069
ATTEMPT	0.004	0.010	0.088	0.135	0.140	0.043	0.100	0.083	0.141	0.078
LPT	0.045	0.098	0.546	0.664	0.630	0.379	0.558	0.525	0.653	0.513
LORA	0.021	0.042	0.416	0.570	0.539	0.207	0.456	0.454	0.535	0.441
Shortformer	0.048	0.072	0.370	0.589	0.478	0.218	0.429	0.428	0.531	0.409
VOC	0.023	0.037	0.351	0.531	0.441	0.254	0.383	0.398	0.471	0.373
SLW	0.023	0.036	0.366	0.530	0.482	0.252	0.382	0.397	0.475	0.377
PFedGate	0.019	0.032	0.336	0.521	0.441	0.232	0.351	0.378	0.471	0.344
FedDST	0.020	0.034	0.322	0.523	0.448	0.238	0.353	0.384	0.451	0.350
SE	0.021	0.032	0.338	0.533	0.454	0.231	0.375	0.385	0.476	0.370
FedALT	0.007	0.015	0.184	0.251	0.262	0.087	0.196	0.225	0.334	0.242
sLORA	0.022	0.034	0.354	0.530	0.465	0.235	0.375	0.372	0.462	0.077
adaLORA	0.011	0.021	0.263	0.435	0.364	0.187	0.273	0.278	0.379	0.280
Delta-LoRA	0.019	0.037	0.339	0.527	0.453	0.145	0.375	0.306	0.469	0.298
FibecFed (ours)	0.019	0.029	0.302	0.452	0.394	0.200	0.330	0.328	0.403	0.325

Table 14: Relative communication overhead of FibecFed and diverse baseline approaches. The time unit is second.