Energy-Efficient Personalized Federated Continual Learning on Edge

Zhao Yang, Haoyang Wang, *Graduate Student Member, IEEE*, and Qingshuang Sun

***Abstract*—Federated learning (FL) on the edge devices must support continual learning (CL) to handle continuously evolving the data and perform the model training in an energy-efficient manner to accommodate the devices with limited computational and energy resources. This letter proposes an energy-efficient personalized federated CL (FCL) framework for the edge devices. The network structure on each device is divided into parts for retaining old knowledge and learning new knowledge, training only part of the model to reduce overhead. A data-free parameter selection approach selects important parameters from the trained model to retain old knowledge. During new task learning, a federated search method determines a resource-adaptive personalized model structure for each device. Experimental results demonstrate that our method can effectively support FCL in an energy-efficient manner on the edge devices.**

***Index Terms*—Edge devices, energy-efficient, federated continual learning (FCL), personalized federated search.**

I. Introduction

FEDERATED learning (FL) [1] effectively leverages the massive amounts of data on the edge devices to train the models while maintaining the data privacy. However, when implementing FL on the edge devices, the data on these devices evolves over time. This dynamic nature of data necessitates the capability for continuous learning in FL on the edge devices.

Deploying FL on the edge devices while learning from continually evolving data faces the problem of catastrophic forgetting (CF), where the new knowledge acquisition overwrites the existing model knowledge, affecting generalization. FL's inability to perceive the local data distribution along with the heterogeneous data (e.g., non-IID data) across the devices, impacts the model training performance and convergence. Edge devices' limited computational resources and battery power exacerbate this, as reduced training convergence leads to higher costs and faster energy depletion, potentially causing devices to go offline. This disrupts the FL process and hinders continual learning (CL). Therefore, energy-efficient federated CL (FCL) is essential for edge devices.

Existing methods for FCL [2], [3], [4] typically rely on the CL techniques [5], [6], [7]. These approaches often require storing/generating replay training samples [5], expanding the network structure [6], or setting up additional distillation networks [7], imposing extra computational and storage burdens on the edge devices. In contrast, parameter isolation [8] for CL, where the network pruning creates space for new tasks, avoids increasing network capacity and reduces computational overhead, making it more suitable for the edge devices. However, applying this to FL introduces challenges, such as data privacy and non-IID data. Determining the appropriate network structure on each device to retain old knowledge, learn new knowledge, and aggregate knowledge is crucial for the FCL performance, impacting energy consumption and device online time, ultimately determining the feasibility of a more continual FL process.

Therefore, this letter proposes an energy-efficient FCL method for the edge devices. Neural networks on different devices are divided into two modules: one part retains original knowledge and the other part learns new knowledge, with only the latter being trained in FL. This approach ensures generalization and reduces training costs. A data-free parameter selection method is used for the old knowledge retention, selecting important parameters without retraining. To further reduce training costs, the training model's scale is determined based on the available resources of the deployed device, and automated method is used to search for the optimal personalized network structure on each device. We design search strategy and controller for the knowledge learning and aggregation to handle the non-IID data, ensuring performance without incurring extra training costs. Additionally, a lightweight search controller is established to reduce the search overhead. These approaches reduce energy consumption and extend the online time of edge devices, supporting sustainable FL for new samples.

II. Related Work

FL, a privacy-preserving distributed learning framework, excels at processing the data locally on the edge devices. However, integrating FL with diverse and mobile edge devices presents challenges, particularly due to the evolving nature of data. Therefore, FL on the edge devices needs CL capabilities to retain old knowledge while acquiring new information, thus avoiding CF and maintaining the model generalization.

Several methods have been proposed to address CF in FCL. For instance, [2] explores continual edge learning through knowledge transfer and an ADMM-based federated meta-learning algorithm. Yoon et al. [3] proposed FedWeIT, involving selective knowledge transfer but requiring high local storage. Usmanova et al. [4] presented a distillation-based approach using LwF [9]. Mori et al. [10] used progressive neural networks [6] to integrate new neural networks. These methods impose additional computational or storage demands,

Manuscript received 26 July 2024; accepted 30 July 2024. This work was supported in part by the Fundamental Research Funds for the Central Universities, CHD under Grant 300102374102. This manuscript was recommended for publication by A. Shrivastava. *(Corresponding author: Qingshuang Sun.)*

Zhao Yang is with the School of Future Transportation, Chang'an University, Xi'an 710064, Shaanxi, China.

Haoyang Wang and Qingshuang Sun are with the School of Computer Science, Northwestern Polytechnical University, Xi'an 710129, Shaanxi, China (e-mail: sunqsh@mail.nwpu.edu.cn).

Digital Object Identifier 10.1109/LES.2024.3439552

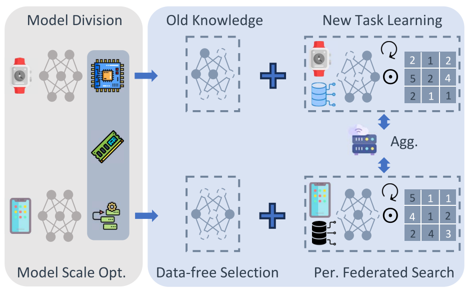

Fig. 1. Overview of our proposed method.

challenging FL on the edge devices due to limited resources. This increased computational burden also heightens energy consumption, reducing the edge devices' online time and complicating support for persistent FL.

III. Energy-Efficient Personalized FCL

A. System Overview

This letter aims to enable each device in FL to perform FCL efficiently, ensuring the network performs well on both the old and new data. This requires reducing training overhead in each round and ensuring training convergence to minimize the overall energy consumption. Therefore, we propose energy-efficient personalized FCL.

The proposed method, shown in Fig. 1, divides the network into parts for retaining old knowledge and learning new knowledge. During new task training, only part of the network is trained to reduce overhead on the edge devices. With limited storage, new training samples overwrite the old ones. We use data-free parameter selection to retain important parameters and preserve old knowledge. To ensure efficiency and performance despite old knowledge and heterogeneous data, a resource-adaptive personalized federated search method determines each device's network structure. Detailed methods for the parameter selection and federated search are provided below.

B. Retention of Old Knowledge With Parameter Selection

In FCL, once a new task arrives, learning for the previous task is complete. Each device must then select and retain important parameters related to the old samples to ensure performance on the previous tasks. This process, akin to model pruning, must be executed on the edge devices with minimal computational cost, avoiding the retraining and fine tuning required by the traditional methods. Since, the edge devices have limited storage and overwrite the previous task samples with new ones, a data-free approach for the parameter selection is necessary. Therefore, we designed a data-free method for retaining old knowledge based on the pruning method proposed by [11].

When selecting parameters, we first set the model size M_{Retain} for the part that retains old knowledge. The size of M_{Retain} is the size of the model structure outside the part determined for learning new knowledge. The expectation of the sum of the importance scores of all the weights is used as a proxy for the model accuracy, and the maximization of this proxy yields the optimal layer-wise density

$$\max_{p_l} \mathop{\mathbb{E}}_{M,W}\left[\sum_{l=1}^{N} S^l \cdot M^l\right] \text{ subject to } \sum_{l=1}^{N} \alpha_l \cdot p_l \leq M_{\text{Retain}} \quad (1)$$

where N is the number of layers in the network, p_l is the layer-wise density of the layer l, and W_l and M_l represent the weight matrix and the pruning mask matrix of the lth layer, respectively. l is the number of parameters in the layer l. S_l represents any important score matrix in the lth layer, which can be calculated as

$$S^l\left(w_{ij}^l\right) = \left[\mathbb{1}^\top \prod_{k=l+1}^{N} \left|W^k \odot M^k\right|\right]_i \left|W_{ij}^l M_{ij}^l\right| \left[\prod_{k=1}^{l-1} \left|W^k \odot M^k\right| \mathbb{1}\right]_j \quad (2)$$

where $S^l(w_{ij}^l)$ is the important score for a single weight w_{ij}^l, $\odot$ denotes the Hadamard product, $|\cdot|$ is element-wise absolute operation, and $\mathbb{1}$ is an all-one vector.

As a result, the layer-wise density p_l is the only variable in (1), and can be obtained by

$$\max_{p_l} \sum_{l=1}^{N} \log p_l \text{ subject to } \sum_{l=1}^{N} \alpha_l \cdot p_l \leq M_{\text{Retain}}. \quad (3)$$

From the above equations, it can be seen that p_l is obtained through the matrix computation, which involves relatively low computational overhead. After obtaining the density p_l for each layer, only the first $\sqrt{p_l} C_l$ network structures (e.g., the channels C_l) are retained as the part for preserving old knowledge.

C. Personalized Search for New Task Learning

When searching for the model structure for the new task on each device, we first determine a resource-adaptive model structure to reduce the training overhead for each round. The performance bottleneck in neural network processing is primarily caused by two constraints: 1) computation capacity and 2) memory occupancy. The computation of a neural network is driven by multiply-accumulate operations (MACs). The total number of MACs in a neural network, denoted by C, is closely tied to the scale of the model

$$C = \sum_{i=1}^{L} \sum_{j=1}^{n_i} {k_{i,j}}^2 n_{i-1} h_{i,j} w_{i,j} \quad (4)$$

where $k_{i,j}$ is the kernel size of the jth filter in the ith layer. $h_{i,j}$ and $w_{i,j}$ are the height and width of the corresponding output feature map, respectively. L is the total number of layers. n_i is the number of filters in the ith layer.

In addition to computation, the memory occupancy of a neural network is critical to its overall performance. Storing and loading data incurs additional time and energy consumption, making it essential to carefully manage the memory usage. Memory occupancy refers to the storage of a neural network's training weights and feature maps, which are used during the inference process. The total memory occupancy of a neural network, denoted by M, can be calculated as follows:

$$M = B \sum_{i=1}^{L} \sum_{j=1}^{n_i} {k_{i,j}}^2 n_{i-1} + B \sum_{i=1}^{L} \sum_{j=1}^{n_i} h_{i,j} w_{i,j} \quad (5)$$

TABLE I
PERFORMANCE COMPARISON WITH SIX BASELINES. A_5 IS AVERAGE ACCURACY(%) ON FIVE TASKS, F IS AVERAGE FORGETTING(%) ON FIVE TASKS

Method	Split-Caltech		Split-MNIST		Split-CIFAR-100		Split-Traffic Sign		IID-CIFAR-100	
	A_5	F	A_5	F	A_5	F	A_5	F	A_5	F
STL	49.89	/	63.01	/	42.54	/	51.84	/	48.19	/
FedAvg	27.23	19.56	40.12	14.34	20.34	36.45	34.67	19.88	29.47	27.47
FedEWC	18.21	22.25	39.24	18.92	17.51	42.58	35.02	23.56	26.46	28.67
FedLwF	52.49	3.76	62.49	2.78	40.12	5.88	51.87	3.98	44.21	5.12
CHFL	58.33	4.19	66.83	3.72	41.07	6.01	57.25	4.49	46.23	4.84
FedWeIT	65.02	4.07	66.23	2.37	44.83	7.25	55.29	3.77	53.16	4.58
Ours	67.56	5.02	71.29	3.86	46.12	8.37	62.37	5.72	53.49	5.97

where B is the data bitwidth, which in most hardware platforms is 32 bits. In this letter, our goal is to construct a high-efficiency partial model A that satisfies the following constraint: $C_m^{con}(A) \leq b_m$, where C_m^{con} represents the mth type of computation resource constraint mentioned earlier, and b_m is the corresponding budget. To achieve this, we select a subset of filters from each layer. The determined model scale is also the search space that will be used in the subsequent personalized federated search process.

After determining the search space for each device, we construct a search controller and strategy to enable the devices to find an optimized network structure for new tasks. Since, the correlation between the current and original training samples is difficult to determine due to overwriting, we randomly select a predetermined number of parameters from the initial network structure (trained on the old tasks) for each device. This model structure is then trained on the new samples locally.

When the local training on different devices is completed, the trained parameters are uploaded to the server for aggregation. Since, the number and type of samples collected by the edge devices are limited, parameter aggregation allows the models on different devices to utilize the knowledge contained in the training samples from the other devices. Finally, the aggregated parameters are returned to the different devices and validated on the current task's data to determine whether the parameter selection for that round of the model search was appropriate. If validation accuracy improves, the current parameter selection is deemed appropriate and used in subsequent training. If it decreases, the network structure is considered inappropriate, and new parameters are selected for the next round.

The search process shows the need for a parameter sampling controller to record the parameter search status during training. Thus, we establish a weight matrix (shown in Fig. 1) as a search controller on each device. This matrix activates parameters for training and records their selection frequency. If selected parameters improve the training accuracy, their count increases by one. Ultimately, the network structure on each device, composed of the most frequently selected parameters, forms the final personalized network structure.

Since, we use filters as the basic unit for the parameter selection, the search controller's dimensions depend only on the number of filters per layer and the number of layers. This limits its storage space. Additionally, updating the search controller involves only the matrix addition, making its computational burden negligible. Thus, it is a lightweight search controller suitable for the edge devices.

After learning new knowledge, the network structure from the personalized search is combined with the retained old knowledge to form a new model. When a new task arrives, the process of parameter selection for the knowledge retention and personalized federated search is repeated. This allows the selected parameters to preserve the knowledge from both the previous task and earlier tasks.

IV. EXPERIMENT

A. Experimental Setup

Datasets and Baselines (Split-Caltech-256 [12]): We selected 250 classes from the Caltech-256 dataset and organized them into 50 non-IID subtasks. *Split-MNIST Series:* Including MNIST [13], FashionMNIST [14], and Not-MNIST [15]. We divide a total of 30 classes from the three datasets into six non-IID tasks. *Split-CIFAR-100:* We partition the dataset into 20 non-IID tasks, with each task consisting of five distinct classes. *Split-Traffic Sign [16]:* We divide this dataset into eight non-IID tasks, with each task consisting of five classes. *IID-CIFAR-100:* The total of 100 image classes is divided into disjoint ten-class datasets. Each dataset is further divided into six IID subdatasets, each consisting of ten classes. In the experiments, tasks belonging to the same dataset are randomly allocated to various local devices. Each device receives a new task corresponding to the dataset when it enters a new time step. We use AlexNet as the backbone structure for each dataset. And six baselines are selected for comparison, including single-task learning (STL), FedAvg [17], and four FCL methods: 1) FedEWC [18]; 2) FedLwF [4]; 3) CHFL [10]; and 4) FedWeIT [3].

Evaluation Metrics (Average Accuracy): This metric quantifies the model's performance after training on a sequence of consecutive tasks

$$A_t = \frac{1}{t}\sum_{i=1}^{t} a_{t,i} \tag{6}$$

where $a_{t,i}$ refers to the model performance on the task i after being trained on the task t. *Average Forgetting:* This metric measures the accuracy decline for each task by comparing the highest accuracy achieved during training to the final accuracy obtained when the model training is completed

$$F = \frac{1}{T-1}\sum_{i=1}^{T-1} \max_{1,\ldots,T-1}\left(a_{t,i} - a_{T,i}\right). \tag{7}$$

B. Results

In the experiment, we randomly selected ten Raspberry Pi 3B, 3B+, and 4B devices as deployment devices to participate in FL. These ten devices learned over five consecutive tasks. The experimental results are shown in Table I. From the results, it can be seen that the traditional FL method FedAvg struggles to handle the heterogeneous data and CL tasks. Our method outperforms FedEWC and FedLwF. Although CHFL

TABLE II
ACCURACY(%) OF LOCAL MODELS ON SPLIT-CIFAR100 DATASET

Device	**FedAvg**	**FedLwF**	**CHFL**	**FedWeIT**	**Ours**
0	57.98	67.43	72.65	77.47	79.81
1	48.12	72.46	73.45	66.81	78.23
2	51.53	60.12	60.98	63.68	76.57
3	53.78	61.39	61.56	69.45	76.41
4	46.83	62.76	68.71	70.56	72.88
5	49.54	70.27	72.67	71.34	76.19
6	50.83	61.59	69.69	65.67	74.44
7	56.67	67.22	64.86	70.37	78.21
8	54.83	68.09	68.58	71.45	79.58
9	49.59	71.62	66.77	72.67	77.62
Avg.	51.97	66.29	67.99	69.94	76.99
std	3.52	4.29	4.23	3.69	2.06

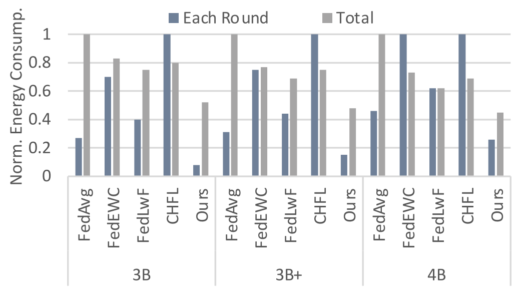

Fig. 2. Comparison of different methods in terms of energy consumption in one round and the whole training process.

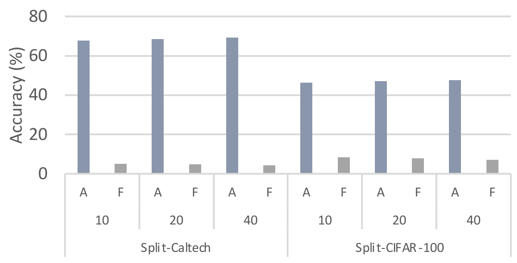

Fig. 3. Scalability evaluation of our method with more participants.

and FedWeIT retain more old knowledge, they perform better in terms of average forgetting. However, the retention of old knowledge has a greater impact on the acquisition of new knowledge. Therefore, our proposed method achieves higher average accuracy.

Next, we verified the performance of different methods in handling the heterogeneous data. This part of the experiment is conducted on the Split-CIFAR-100 dataset. In the experiment, each device was randomly assigned only five different classes of data. The experimental results are shown in Table II. It can be seen that the personalized search method adopted in this letter can still achieve better accuracy for different devices under highly non-IID settings with the smallest accuracy fluctuation between the devices.

Subsequently, we validated an important aspect of this letter, namely the energy consumption advantage of the proposed method. The experimental results are shown in Fig. 2. Using our method in each training round, due to training only part of the network structure has significant advantages in terms of energy consumption. Additionally, we compared the energy consumption of different methods when achieving the same accuracy. The proposed method achieves the lowest overall energy consumption.

Finally, we verified the scalability of the proposed method. In this part of the experiment, we expanded the number of participating devices to 20 and 40. The experimental results are shown in Fig. 3. As the number of participating devices increases, our method can achieve better performance. However, more devices lead to increased communication overhead and bandwidth contention, reducing communication efficiency and providing only limited performance gains.

V. CONCLUSION

Our paper proposes an energy-efficient personalized FCL framework. Each device's model is divided into parts for retaining old knowledge and learning new knowledge. Training only part of the model reduces overhead. Important parameters are selected to retain old knowledge and a personalized search method improves convergence of new knowledge learning, lowering the overall training energy consumption. Experimental results show the proposed method's advantages in learning performance and energy consumption.

REFERENCES

[1] P. Kairouz et al., "Advances and open problems in federated learning," *Found. Trends® Mach. Learn.*, vol. 14, nos. 1–2, pp. 1–210, 2021.

[2] S. Yue, J. Ren, J. Xin, S. Lin, and J. Zhang, "Inexact-ADMM based federated meta-learning for fast and continual edge learning," in *Proc. Mobihoc*, 2021, pp. 1–20.

[3] J. Yoon, W. Jeong, G. Lee, E. Yang, and S. J. Hwang, "Federated continual learning with weighted inter-client transfer," in *Proc. ICML*, 2021, pp. 12073–12086.

[4] A. Usmanova, F. Portet, P. Lalanda, and G. Vega, "A distillation based approach integrating continual learning and federated learning for pervasive services," in *Proc. CML-IOT IJCAI Workshop*, 2021, pp. 1–7.

[5] A. Chaudhry, M. Ranzato, M. Rohrbach, and M. Elhoseiny, "Efficient lifelong learning with a-GEM," in *Proc. ICLR*, 2019, pp. 1–20.

[6] A. A. Rusu et al., "Progressive neural networks," 2016, *arXiv:1606.04671*.

[7] J. Zhang et al., "Class-incremental learning via deep model consolidation," in *Proc. WACV*, 2020, pp. 1131–1140.

[8] A. Mallya and S. Lazebnik, "PackNet: Adding multiple tasks to a single network by iterative pruning," in *Proc. CVPR*, 2018, pp. 1–9.

[9] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 2935–2947, Dec. 2018.

[10] J. Mori, I. Teranishi, and R. Furukawa, "Continual horizontal federated learning for heterogeneous data," in *Proc. IJCNN*, 2022, pp. 1–8.

[11] Y. Cai, W. Hua, H. Chen, G. Suh, S. C. De, and Z. Zhang, "Structured pruning is all you need for pruning CNNs at initialization," 2022, *arXiv:2203.02549*.

[12] G. Griffin, A. Holub, and P. Perona, 2022, "Caltech-256 object category dataset," Dataset, Caltech. [Online]. Available: https://data.caltech.edu/records/nyy15-4j048

[13] Y. LeCun, "The MNIST database of handwritten digits," Dataset. Accessed: Aug. 16, 2024. [Online]. Available: https://goo.gl/t6gTEy

[14] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*.

[15] Y. Bulatov, "not-MNIST dataset," Dataset. Accessed: Aug. 16, 2024. [Online]. Available: https://goo.gl/t6gTEy

[16] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Netw.*, vol. 32, pp. 323–332, Aug. 2012.

[17] H. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Artif. Intell. Statist.*, 2017, pp. 1–10.

[18] J. Kirkpatrick et al., "Overcoming catastrophic forgetting in neural networks," *Proc. Nat. Acad. Sci.*, vol. 114, no. 13, 2017, Art. no. 35213526.