Adaptive Federated Learning for Large Models with Scarce and Non-IID Data in Cloud-Edge Networks

Benteng Zhang[a], Yingchi Mao[a], Peng Zhang[b], Haotian Zheng[a], Xiaoming He[c], Jiawen Kang[d], and Jie Wu[e]
[a] College of Computer Science and Software Engineering, Hohai University, Nanjing, China
[b] Research and Development Center of Science and Technology, Huaneng Lancang River Hydropower Inc., China
[c] College of Internet of Things, Nanjing University of Posts and Telecommunications, Nanjing, China
[d] Guangdong University of Technology, Guangzhou, China
[e] Center for Networked Computing, Temple University, Philadelphia, USA
230407040003@hhu.edu.cn, yingchimao@hhu.edu.cn, zpenghn@163.com, 231607010125@hhu.edu.cn,
hexiaoming@njupt.edu.cn, kavinkang@gdut.edu.cn, jiewu@temple.edu

***Abstract*—Federated Learning (FL) can coordinate numerous IoT devices to train large models collaboratively and provide intelligent services and applications (FL-AIoT). However, large models' training requires significant memory and data, which creates challenges for resource-constrained IoT devices. State-of-the-art studies indicate that IoT devices have limited storage capacity, fresh data collected by devices can overwrite the outdated data and exhibit high data heterogeneity. This causes the global model on the cloud server to forget outdated data's characteristics and degrades model accuracy (i.e., catastrophic forgetting). Moreover, IoT devices with limited data collection capacity and energy cannot collect enough data for large models' training (i.e., data scarcity), which exacerbates catastrophic forgetting. Existing methods generate training data but overlook that catastrophic forgetting can significantly degrade the stability of the generative network's training. To address these issues, we propose a Federated Fine-tuning adaptive aggregation method via Knowledge Distillation (FedFKD). Specifically, FedFKD utilizes global model updates to aid the data generator's training. FedFKD dynamically assigns aggregation weights to each device by evaluating the model consistency, which can guide the global model to update in the optimal direction. Furthermore, FedFKD guides local models' training by combining fresh and outdated data characteristics through decoupled distillation with variable distillation temperatures. Experiments on CIFAR-10/100 and Tiny-ImageNet datasets show that compared to the best of 4 baselines, FedFKD can improve the average global model accuracy by 1.37%, and achieve the lowest global training loss.**

***Index Terms*—Federated learning, artificial intelligence of things, catastrophic forgetting, large model training**

I. Introduction

Large models need to be fed with a large amount of data, but in real-world IoT scenarios, IoT data (e.g., images, exhibit highly heterogeneous distributions) is not centralized but distributed across numerous devices [1, 2]. This creates communication burdens and privacy concerns for traditional Machine Learning (ML, which transfers row data to a cloud server for centralized training) in Artificial Intelligence of Things (AIoT). Therefore, we need to explore more efficient large model training methods to fully utilize the computing and storage resources on IoT devices. To our knowledge, Federated Learning (FL) can enable IoT devices to collaboratively train large ML models in a distributed manner without sharing raw data (FL-AIoT) [3]. This makes FL a promising cloud-edge collaborative large model training framework for AIoT [4].

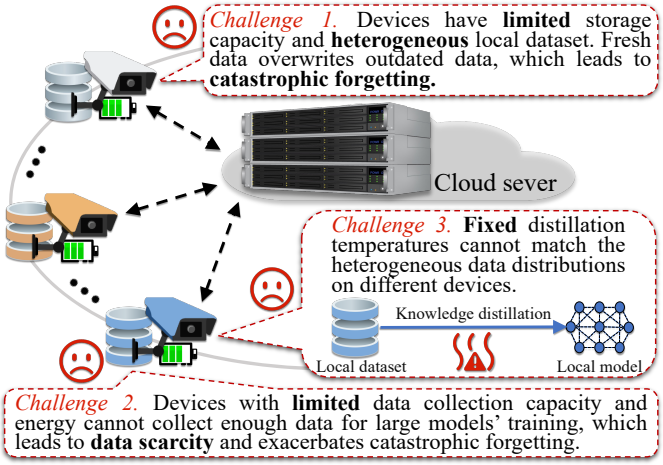


Fig. 1. Three challenges in large model training for FL-AIoT.

However, as illustrated in ***Challenge 1*** of Fig. 1, in real-world IoT scenarios, due to IoT devices have limited storage capacity, fresh data collected by devices can overwrite outdated data continuously and exhibit high data heterogeneity. State-of-the-art studies have indicated that FL tends to use fresh data from the last few rounds for training. Since large models require a lot of data for FL training (e.g., Large Language Models, LLMs), this can cause the aggregated global model to forget outdated data's characteristics (i.e., catastrophic forgetting) and degrade the model accuracy [5]. Moreover, as illustrated in ***Challenge 2*** of Fig. 1, IoT devices with limited data collection capacity and energy cannot continuously collect enough data for local training (i.e., data scarcity) [6], which exacerbates catastrophic forgetting and creates data scarcity for the training of large models. Therefore, to improve the global model accuracy with heterogeneous and scarce data, IoT devices need to overcome catastrophic forgetting while generating training data that follows the real-world distribution. Existing methods generate data for training.

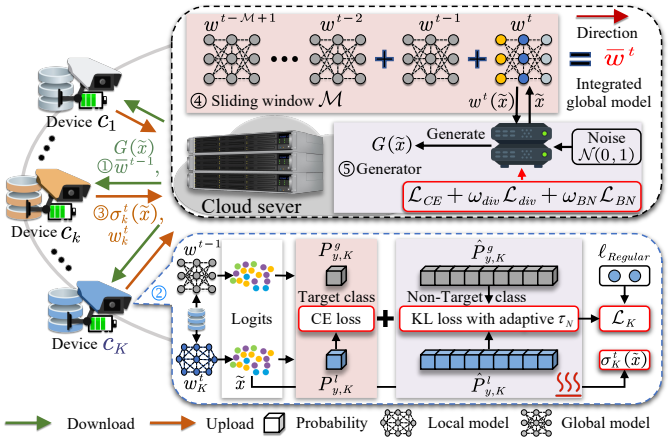


Fig. 2. The design details of FedFKD. ① Global model delivery. ② Decoupled distillation and local training. ③ Upload parameters. ④ Global aggregation and model integration. ⑤ Generate data.

For instance, FedGAN simultaneously uses a generator and a discriminator to generate training data and adds noise to train the generative network [7], but this requires significant computational overhead. Based on knowledge distillation, DENSE trains the data generator from the distilled knowledge without relying on additional shared information or datasets [8]. However, as training goes on, due to catastrophic forgetting, the discriminators in these methods may forget outdated data's characteristics, which undermines the stability of the generative network's training. This means that good training of the generator requires first overcoming catastrophic forgetting.

Introducing knowledge distillation into FL (i.e., Federated Distillation, FD) to extract and combine characteristics from both fresh and outdated data can effectively guide local training and mitigate catastrophic forgetting [12]. FedKD utilizes FD to integrate the dark knowledge mined from outdated data into the new training round [9], which can effectively save computational costs. To save storage space on devices, FedDistill extracts features from the dataset and shares them with other client devices through the cloud server [10], but this raises privacy concerns. Unlike the above methods, DKD extracts characteristics from the Logits vector and decomposes the classic distillation loss into the target class *Cross-Entropy* loss and non-target distillation loss [11], which is proven to improve the distillation effect. However, as illustrated in ***Challenge 3*** of Fig. 1, these methods use fixed distillation temperatures across different devices, which ignores that fixed distillation temperatures cannot match the heterogeneous data distributions on devices. This leads to poor distillation performance and fails to mitigate catastrophic forgetting. Therefore, dynamically adjusting distillation temperatures for devices based on the heterogeneous data distribution to overcome catastrophic forgetting and stabilize the generator's training remains a significant challenge in FL-AIoT.

Motivated by these issues, we propose FedFKD to generate data, while effectively overcoming catastrophic forgetting.

The **contributions** of our paper are depicted as follows.

- **IoT device side.** FedFKD builds an adaptive distillation temperature-aware mechanism to adjust distillation temperatures for each device dynamically. Furthermore, FedFKD dynamically assigns aggregation weights to each IoT device by evaluating model consistency.
- **Cloud server side.** A global model updates-assisted data generator is used to generate training data. FedFKD utilizes a sliding window on the cloud server to integrate $\mathcal{M}$ historical global models.
- **Effectiveness.** Experiments on three different datasets demonstrate that compared to the best of 4 baselines, FedFKD can improve the average global model accuracy by 1.37% and achieve the lowest global training loss.

The remainder of this paper is organized as follows. Our system model is shown in Section II. The design details of FedFKD are discussed in Section III. The experiments are reported in Section IV. Finally, we conclude with Section V.

II. System Model

Our system model is shown in Fig. 2, which consists of N IoT devices and K devices are randomly selected in each training round. Each IoT device c_k has a private local dataset $\mathcal{D}_k$, which contains C types of samples, and each sample is represented as $x \in \mathbb{R}^d$ with a corresponding label y. The global dataset of the entire system is $\mathcal{D} = \sum_{k\in N}^{N} \mathcal{D}_k$. All IoT devices are initialized with a global model w^0. In the t-th training round, the local model of IoT device c_k is denoted as w_k^{t-1}, and the aggregated global model parameters are denoted as w^t. Each IoT device optimizes the local model and the optimization objective is given by

$$w_k^t = \arg\min \mathbb{E}_{(x,y)\sim\mathcal{D}_k}[\mathcal{L}(w; w^{t-1}, x, y)], \tag{1}$$

where $\mathcal{L}$ is the composite loss function. Device c_k uses its local dataset $\mathcal{D}_k$ as input to both the local model w_k^t and the global model w^{t-1} from the cloud server for characteristic extraction. The Logits vector output (i.e., local data characteristic) of client c_k for sample x is denoted as

$$s(w_k^t, x) = \underbrace{[z_{1,k}, z_{2,k}, ..., z_{i,k}, ..., z_{C,k}]}_{z_k^l} \in \mathbb{R}^{1\times C}. \tag{2}$$

In the $(t-1)$-th training round, the characteristic of the global model for the device's local data is denoted as

$$s(w^{t-1}, x) = \underbrace{[z_1, z_2, ..., z_i, ..., z_C]}_{z_k^g} \in \mathbb{R}^{1\times C}. \tag{3}$$

The target class prediction probability $p_{y,k}^l$ and the non-target class probability $\hat{p}_{y,k}^l$ for sample x of device c_k are given by

$$p_{y,k}^l = \frac{\exp(z_{i,k}^l)}{\sum_{j=1}^{C}\exp(z_{j,k}^l)}, \hat{p}_{y,k}^l = \frac{\exp(z_{\tilde{i},k}^l)}{\sum_{\tilde{i}}^{C}\exp(z_{\tilde{i},k}^l)}, \tag{4}$$

where $\tilde{i} \in [1, C] \wedge \tilde{i} \neq y$ and $z_{i,k}^l$ is the Logits output of device c_k for the local sample x. The global representation prediction probability for non-target classes is given by

$$\hat{p}_{y,k}^g = \frac{\exp(z_{\tilde{i},k}^g)}{\sum_{\tilde{i}}^{C}\exp(z_{\tilde{i},k}^g)}. \tag{5}$$

TABLE I
LIST OF MAIN SYMBOLIC PARAMETERS

Symbol	Symbol Meaning
d	Dimensions of the model
x	Samples in the dataset $\mathcal{D}$
y	Sample x corresponds to the label
r	Local iteration epoch
η	Learning rate
K	Total number of randomly selected IoT devices
N	Total number of IoT devices
C	Total number of categories of images
T	Training rounds
B	Mini-batch size
c_k	The k-th IoT device
w^t	Global model of t-th training round
w_k^t	Local model of c_k in t-th training round
z_i	The Logits output of sample x on a Class i sample
z_y	Sample x Logits output on the target class
$\bar{w}^t$	Integrated model in t-th training round
σ_k^t	Model consistency of c_k in t-th training round
α_k^t	Aggregate weight of c_k in t-th training round
$s(\cdot)$	The Logits output of the model
$\mathcal{M}$	Sliding window size
$\mathcal{D}_k$	Local dataset for c_k

III. Large model training in FedFKD

A. Adaptive Distillation Temperature-Aware Mechanism

We observed that the Logits values of most large models are distributed near zero (with fewer values far from zero, resembling a normal distribution). Therefore, we approximate the Logits distribution as a normal distribution with a mean μ and a variance ϵ. Then, we can rescale the sample's probability distribution p_i as

$$\tilde{p}_i = \frac{\exp((z_i-\mu)/\epsilon)}{\sum_{j=1}^{C}\exp((z_j-\mu)/\epsilon)} = \frac{\exp(z_i/\epsilon)}{\sum_{j=1}^{C}\exp(z_j/\epsilon)}. \tag{6}$$

From (6), it can be observed that the effect of the distillation temperature τ is similar to the role of ϵ in a normal distribution. Thus, τ can alter the scale of the probability distribution without affecting the classification results. The distillation temperature τ_k of device c_k dynamically changes with the Logits output vector of the model, which is given by

$$\tau_k = \tau[s(w_k^t, [\mathcal{D}_k]_b)], \tag{7}$$

where $\tau[\cdot]$ is the standard deviation function and $[\mathcal{D}_k]_b$ is the b-th mini-batch of data randomly sampled from $\mathcal{D}_k$. After softening by the distillation temperature τ_k, the local probability prediction and global probability prediction for non-target classes on device c_k are given by

$$\begin{aligned}\hat{p}_{y,k}^l(\tau_k) &= \frac{\exp(z_{\tilde{i},k}^l/\tau_k)}{\sum_{\tilde{i}}^{C}\exp(z_{\tilde{i},k}^l/\tau_k)},\\ \hat{p}_{y,k}^g(\tau_k) &= \frac{\exp(z_{\tilde{i},k}^g/\tau_k)}{\sum_{\tilde{i}}^{C}\exp(z_{\tilde{i},k}^g/\tau_k)}.\end{aligned} \tag{8}$$

We redefine the decoupled distillation loss $\mathcal{L}_{DeD}$ as

$$\begin{aligned}\mathcal{L}_{DeD}(w_k^t, w^{t-1}, x) = \mathcal{L}_{CE}(p_{y,k}^l, 1_y)\\ + \beta{\tau_k}^2\mathcal{L}_{KL}(\hat{p}_{y,k}^g(\tau_k)\|\hat{p}_{y,k}^l(\tau_k)),\end{aligned} \tag{9}$$

where 1_y is the label of x, β is the extraction intensity of the characteristic in the global distribution, $\mathcal{L}_{CE}$ is the *Cross-Entropy* loss, and $\mathcal{L}_{KL}$ is the *Kullback-Leibler* divergence loss. The covariance matrix of the characteristic representation z_i of x is equal to its correlation matrix $\mathbb{K}$:

$$\begin{aligned}\mathbb{K} &= \frac{1}{C-1}\sum_{i=1}^{C}(z_i-\overline{z})(z_i-\overline{z})^\top\\ &= (z_i-\overline{z})/\sqrt{Var(z)}.\end{aligned} \tag{10}$$

where z_i is the model output vector of the sample for the i-th class, $\bar{z}$ is the average vector representation of the sample across all classes, and $Var(\cdot)$ is the variance function. To eliminate the inter-class characteristic correlation of the sample and alleviate the imbalance in accuracy between classes, we introduce a regularization term $\mathcal{L}_{Regular}$ by minimizing the *Frobenius* norm of the matrix $\mathbb{K}$, which penalizes the variance between the singular values. $\mathcal{L}_{Regular}$ is given by

$$\mathcal{L}_{Regular}(w_k^t, x) = \frac{1}{d^2}\|\mathbb{K}\|_{\mathrm{F}}^2. \tag{11}$$

The local overall optimization objective of device c_k in the t-th training round is given by

$$\mathcal{L}_k = \mathcal{L}_{DeD}(w_k^t, w^{t-1}, x) + \theta\mathcal{L}_{Regular}(w_k^t, x), \tag{12}$$

where θ is the weight of $\mathcal{L}_{Regular}$.

B. Global Model Updates-Assisted Data Generator

We use the global model's network structure and model parameters to construct the sample generator G. We first input a noise $\mathcal{N}(0,1)$ into G and assign random labels $\tilde{y}$ to the synthetic data $\tilde{x}$. By minimizing the loss $\mathcal{L}_{CE}$ between the generator's output and the global model's output, we have

$$\mathcal{L}_{CE} = CE(\overline{w}^t(\tilde{x}), \tilde{y}). \tag{13}$$

To increase the diversity of the synthetic data, we introduce $\mathcal{L}_{KL}$ to simulate the classification probabilities of the synthetic data on other categories, which is given by

$$\mathcal{L}_{div} = -\mathcal{L}_{KL}(G(\tilde{x}), w^t(\tilde{x})). \tag{14}$$

Since the sample generated by model inversion may deviate from the true data distribution, we introduce batch normalization loss (i.e., performing statistical information similar to the global model's normalization layer, *BN* layer, on the synthetic data), which is given by

$$\mathcal{L}_{BN} = \sum_{l}(\left\|\mu_l(\tilde{x}) - \mu_l\right\| + \left\|\epsilon_l^2(\tilde{x}) - \epsilon_l^2\right\|), \tag{15}$$

where $\mu_l(\tilde{x})$ and $\epsilon_l^2(\tilde{x})$ are the mean and variance of the l-th *BN* layer of G, respectively. μ_l and ϵ_l^2 are the mean and variance of the l-th *BN* layer of the global model, respectively.

Therefore, the goal of G is to generate data that is similar to the real data distribution, which is denoted as

$$\min_G \mathcal{L}_{CE} + \omega_{div}\mathcal{L}_{div} + \omega_{BN}\mathcal{L}_{BN}, \tag{16}$$

where ω_{div} is the weight of $\mathcal{L}_{div}$ and ω_{BN} is the weight of $\mathcal{L}_{BN}$.

Algorithm 1: FedFKD

Input: N, T, E, $\mathcal{M}$, r, β, θ, ω_{div}, ω_{BN}
Output: Global model w

Initial w^0 and w_G
begin
  for *each training round* $t = 1, 2, 3, ..., T$ **do**
    for *device* c_k *in parallel* **do**
      Save global model w^{t-1} and $w_k^t \leftarrow w^{t-1}$
      $w_k^t \leftarrow ClientUpdate(w_k^t, \mathcal{D}_k)$
      $\tilde{x} = Generate(w^{t-1})$
      $\sigma_k^t(\tilde{x}) = Var(s(w_k^t, \tilde{x}))$
    end
    $\alpha_k^t(\tilde{x}) = \sigma_k^t(\tilde{x}) / \sum_k \sigma_k^t(\tilde{x})$
    $w^t = \frac{1}{K} \sum_{k=1}^{K} \alpha_k^t(x) w_k^t$
    if $t < \mathcal{M}$ **then**
      Send w^t to devices
    else
      Send $w^t = \overline{w}^t = \frac{1}{\mathcal{M}} \sum_{m=1}^{\mathcal{M}} w^{t-m+1}$ to devices
    end
  end
  return w
end
function $ClientUpdate(w_k^t, \mathcal{D}_k)$
begin
  for *local epoch* $e = 1, ..., r$ **do**
    for *batch* $b \in B$ **do**
      $\tau_k = \tau[s(w_k^t, [D_k]_b)]$
      $\mathcal{L}_k = \mathcal{L}_{DeD}(w_k^t, w^{t-1}, [D_k]_b) + \theta \mathcal{L}_{Regular}(w_k^t, [D_k]_b)$
      $w_k^t \leftarrow w_k^{t,0} - \frac{\eta}{|[D_k]_b|} \sum_{e=0}^{r-1} \sum_{b=1}^{B} \nabla \mathcal{L}_k$
    end
  end
  return w_k^t to sever
end
function $Generate(w^{t-1})$
begin
  Input noise $z \sim \mathcal{N}(0, 1)$ and $w_G \leftarrow w^{t-1}$
  Generate $\tilde{x}$ with label $\tilde{y}$
  Update w_G by $\min_G \mathcal{L}_{CE} + \omega_{div} \mathcal{L}_{div} + \omega_{BN} \mathcal{L}_{BN}$
  return $\tilde{x}$
end

C. Model Consistency-Based Adaptive Global Aggregation

In the t-th training round, after the local training, the cloud server evaluates the model consistency coefficient $\sigma_k^t(w_k^t, \tilde{x})$ by calculating the variance of the Logits vector $s(w_k^t, \tilde{x})$, which is given by

$$\sigma_k^t(\tilde{x}) = Var(s(w_k^t, \tilde{x})), \tag{17}$$

where, $\tilde{x}$ is the generated data sample. The higher the variance $\sigma_k^t(\tilde{x})$, the greater the confidence of device c_k in its prediction for sample $\tilde{x}$, and thus, device c_k should be assigned a higher aggregation weight. The global aggregation weight $\alpha_k^t(\tilde{x})$ of device c_k is given by

$$\alpha_k^t(\tilde{x}) = \sigma_k^t(\tilde{x}) / \sum^{K} \sigma_k^t(\tilde{x}). \tag{18}$$

The cloud server aggregates the global model based on the aggregation weight of each device, which is denoted as

$$w^t = \sum^{K} \alpha_k^t(\tilde{x}) w_k^t. \tag{19}$$

After aggregating the global model, we set up a sliding window on the cloud server to integrate $\mathcal{M}$ historical global models, obtaining the integrated model $\bar{w}^t$, which is given by

$$\overline{w}^t = \frac{1}{\mathcal{M}} \sum_{m=1}^{\mathcal{M}} w^{t-m+1}. \tag{20}$$

In the $(t+1)$-th training round, the cloud server sends $\bar{w}^t$ as the latest global model w^t to randomly selected devices. Then devices perform mini-batch SGD on their local datasets. The local model update for device c_k is given by

$$w_k^{t,r} = w_k^{t,0} - \frac{\eta}{|[\mathcal{D}_k]_b|} \sum_{e=0}^{r-1} \sum_{b=1}^{B} \nabla \mathcal{L}_k, \tag{21}$$

where r is the local update epoch and η is the learning rate. The design details of FedFKD are presented in ***Algorithm 1***.

IV. Performance Evaluation

A. Experimental Settings

Experimental Environment. All experiments are conducted on a server with 256GB of RAM (equipped with 2 NVIDIA $A100$ GPUs, each with 80GB of memory). The server runs the Ubuntu 20.04 operating system and is powered by a 64-core Intel(R) Xeon(R) Gold 6326 CPU @2.90GHz and CUDA 11.8 with PyTorch 1.8 framework.

Non-IID Datasets, Target Models, and Hyperparameter Settings. To simulate a realistic heterogeneous data environment in FL-AIoT, we use the *Dirichlet* function $Dir(\alpha)$ to partition the training dataset to each device to generate non-IID datasets [13]. Note that a smaller parameter α indicates higher data heterogeneity among different devices. To simulate a small training data sample, we distribute the datasets to $N = 100$ devices and randomly select $K = 10$ devices to participate in FL training in each training round. For the CIFAR-10/100 datasets [14], we train a CNN model (two convolutional layers and two fully connected layers). For the Tiny-ImageNet dataset [15], we train a ResNet-18 model. To ensure fairness, we use the same settings as in FedProxGAN [19], we set the generator's learning rate to 0.01, $r = 5$, $batchsize = 50$, $\eta = 0.1$, and $T = 200$. To achieve better performance of FedFKD, we set $\beta = 1.0$, $\theta = 0.1$, $\omega_{div} = 1$, $\omega_{BN} = 0.1$, and $\mathcal{M} = 5$.

Baselines. Four comparative methods are as follows.

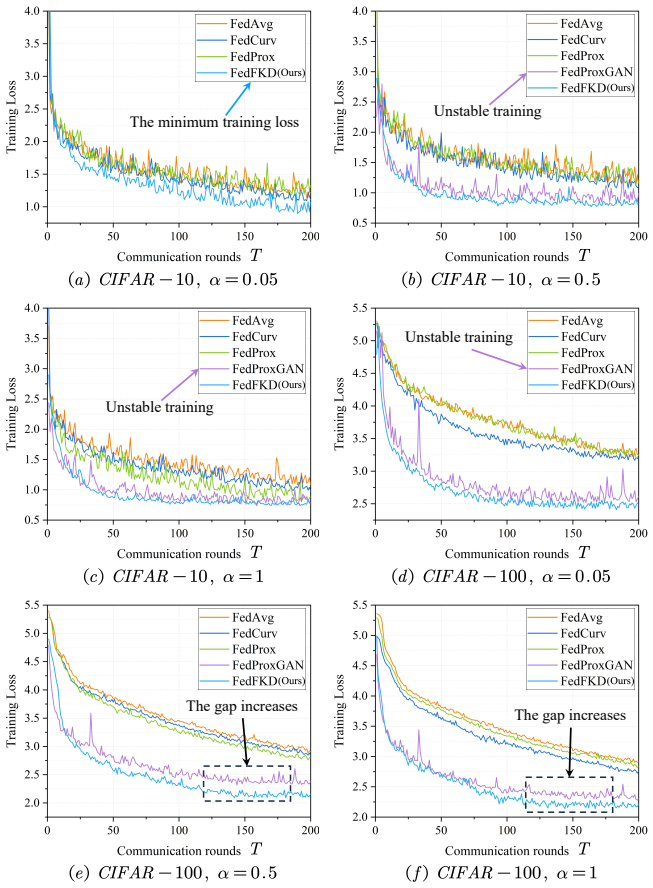
- FedAvg is the most classic baseline method in FL [16], which averages the local model parameters.
- FedCurv utilizes a penalty term to mitigate catastrophic forgetting based on the EWC algorithm [17], with the penalty term's weight determined by the *Fisher* matrix.
- FedProx introduces a constraint term to the local loss [18], which can mitigate catastrophic forgetting.
- FedProxGAN is the state-of-the-art FL method based on data generation that adds a data generator on top of FedProx and fine-tunes the global model [19].

Metrics. Two evaluation criteria are depicted as follows.

- *Global model accuracy* Acc_g. A higher Acc_g indicates better model training performance of the method.

TABLE II
GLOBAL MODEL ACCURACY Acc_g OF DIFFERENT TRAINING METHODS (%)

Method	CIFAR-10			CIFAR-100			Tiny-ImageNet			Average
	$Dir(0.05)$	$Dir(0.5)$	$Dir(1)$	$Dir(0.05)$	$Dir(0.5)$	$Dir(1)$	$Dir(0.05)$	$Dir(0.5)$	$Dir(1)$	
FedAvg	35.13	66.97	73.25	31.31	37.32	38.62	13.62	16.93	17.70	36.76
FedCurv	34.51	67.34	72.82	30.85	35.99	39.16	14.26	17.97	19.62	36.95
FedProx	38.16	63.98	62.65	32.13	32.10	35.70	16.73	18.17	21.57	35.85
FedProxGAN	-	69.73	74.38	35.33	40.23	**43.31**	20.11	23.15	26.43	41.58
FedFKD (Ours)	**43.96**	**71.70**	**74.65**	**36.90**	**42.50**	42.56	**21.24**	**24.72**	**28.36**	**42.95**



(a) $CIFAR-10$, $\alpha=0.05$ (b) $CIFAR-10$, $\alpha=0.5$ (c) $CIFAR-10$, $\alpha=1$ (d) $CIFAR-100$, $\alpha=0.05$ (e) $CIFAR-100$, $\alpha=0.5$ (f) $CIFAR-100$, $\alpha=1$

Fig. 3. Global model training loss $Loss_g$ of different training methods on the CIFAR-10/100 datasets.

- Global model training loss $Loss_g$. A lower $Loss_g$ indicates better training performance of the model.

B. Analysis of Global Model Accuracy Acc_g

Table II reports the global model accuracy Acc_g of five methods on three datasets with $\alpha = \{0.05, 0.5, 1\}$.

1) CIFAR-10 dataset. Under all levels of data heterogeneity, FedFKD achieves the highest global accuracy. When the level of data heterogeneity is low ($\alpha = 1$), the Acc_g of FedFKD outperforms the optimal baseline method (FedProxGAN) by approximately 0.27%. As data heterogeneity increases ($\alpha = \{0.05, 0.5\}$), the Acc_g of FedFKD is significantly higher than the optimal baseline methods (FedProxGAN and FedProx) by approximately 1.97% and 5.80%, respectively. Notably, when $\alpha = 0.05$, since the generator in FedProxGAN samples based on local model averaging, catastrophic forgetting on devices leads to severe oscillations in the discriminator's training (i.e., it fails to converge). Consequently, the specific accuracy for this case is not reported in Table II.

2) CIFAR-100 dataset. When $\alpha = 1$, FedProxGAN achieves the highest $Acc_g = 43.31\%$. However, when data heterogeneity $\alpha = \{0.05, 0.5\}$, FedFKD achieves the highest $Acc_g = 36.90\%$ and 42.50%, which is higher than FedProxGAN by 1.57% and 2.27%, respectively. This indicates that FedFKD can maintain good performance under high data heterogeneity.

3) Tiny-ImageNet. We find that FedFKD achieves the highest $Acc_g = 21.24\%, 24.72\%$, and 28.36% when $\alpha = \{0.05, 0.5, 1\}$, respectively, significantly outperforming other baseline methods. This indicates that FedFKD can maintain good performance on more complex datasets.

C. Analysis of Global Model Training Loss $Loss_g$

Fig. 3 shows the global model training loss $Loss_g$ of five methods on the CIFAR-$10/100$ datasets with $\alpha = \{0.05, 0.5, 1\}$. The training loss clearly illustrates the convergence speed and training stability of a model training method.

1) CIFAR-10 dataset. From (a), (b), and (c) in Fig. 3, we find that FedFKD achieves the lowest training loss across all levels of data heterogeneity. In particular, when $\alpha = \{0.5, 1\}$, the $Loss_g$ of FedProxGAN is similar to that of FedFKD, but the training of FedFKD is noticeably more stable. When facing high data heterogeneity ($\alpha = 0.05$), the $Loss_g$ of FedFKD is significantly lower than that of FedProxGAN. This indicates that FedFKD's training is more stable, and FedFKD can more effectively mitigate catastrophic forgetting under highly heterogeneous data.

2) CIFAR-100 dataset. From (d), (e), and (f) in Fig. 3, we find that the growing complexity of the CIFAR-100 dataset has caused a noticeable rise in the $Loss_g$ for all methods compared to CIFAR-10. FedFKD achieves the lowest $Loss_g$ and the fastest convergence speed across all levels of data heterogeneity. When $\alpha = \{0.5, 1\}$, the $Loss_g$ of FedFKD is significantly lower than that of the other four methods. When facing a high level of data heterogeneity ($\alpha = 0.05$), the $Loss_g$ of FedProxGAN is close to that of FedFKD, but still higher than that of FedFKD, and the training of FedProxGAN is extremely unstable. The above experimental results indicate that FedFKD can maintain lower training loss and faster convergence speed on more complex datasets.

V. Conclusion

The proposed FedFKD can tackle the following challenges. *1)* Diverse IoT devices have limited storage capacity, fresh data collected by devices can continuously overwrite the outdated data and exhibit high data heterogeneity, which leads to catastrophic forgetting in the global model. *2)* IoT devices with limited data collection capacity and energy cannot continuously collect enough data for local training, which exacerbates catastrophic forgetting. To this end, we propose a Federated Fine-tuning adaptive aggregation method via Knowledge Distillation (FedFKD). Specifically, FedFKD utilizes global model updates to aid the data generator's training. To guide the global model to update in the optimal direction, FedFKD dynamically assigns aggregation weights to each device and utilizes a sliding window on the cloud server. FedFKD builds an adaptive distillation temperature-aware mechanism to adjust distillation temperatures for each device dynamically. Experiments on three datasets demonstrate that FedFKD can achieve higher global model accuracy and lower global training loss with different data heterogeneity. These improvements indicate that FedFKD is a promising method to overcome catastrophic forgetting for IoT devices with scarce and Non-IID data.

Acknowledgment

This work is supported in part by the National Key R&D Program of China under Grant 2022YFC3005401; in part by the Key Research and Development Program of China, Yunnan Province under Grant 202203AA080009; in part by the Technology Talent and Platform Program of Yunnan Province under Grant 202405AK340002; in part by the Science and Technology Projects of China Huaneng Group under Grant HNKJ20-H46; and in part by the National Natural Science Foundation of China under Grant 62402246. *(Corresponding author: Yingchi Mao.)*

References

[1] A. Imteaj, U. Thakker, S. Wang, J. Li and M. H. Amini, "A Survey on Federated Learning for Resource-Constrained IoT Devices," in *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 1-24, 2022.

[2] X. Wang, C. Wang, X. Li, V. C. M. Leung and T. Taleb, "Federated Deep Reinforcement Learning for Internet of Things With Decentralized Cooperative Edge Caching," in *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9441-9455, 2020.

[3] A. Abu-Khadrah, A. M. Ali, and M. Jarrah, "An Amendable Multi-Function Control Method using Federated Learning for Smart Sensors in Agricultural Production Improvements," *ACM Trans. Sen. Netw.*, 2023.

[4] X. Wang, Y. Zhao, C. Qiu, Z. Liu, J. Nie and V. C. M. Leung, "InFEDge: A Blockchain-Based Incentive Mechanism in Hierarchical Federated Learning for End-Edge-Cloud Communications," in *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 12, pp. 3325-3342, 2022.

[5] X. Wang, Y. Han, V. C. M. Leung, D. Niyato, X. Yan and X. Chen, "Convergence of Edge Computing and Deep Learning: A Comprehensive Survey," in *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 869-904, 2020.

[6] N. H. Chu, D. T. Hoang, D. N. Nguyen, N. Van Huynh and E. Dutkiewicz, "Joint Speed Control and Energy Replenishment Optimization for UAV-Assisted IoT Data Collection With Deep Reinforcement Transfer Learning," in *IEEE Internet of Things Journal*, vol. 10, no. 7, pp. 5778-5793, 2023.

[7] Y. Zhang, Q. Zhang, M. Yang, T. Xiao and Z. Wang, "FedGAN: Federated GAN for Few-shot Image Generation," *Advances in 2023 3rd International Conference on Electronic Information Engineering and Computer Science (EIECS)*, 2023, pp. 1020-1024.

[8] J. Zhang *et al.*, "DENSE: Data-Free One-Shot Federated Learning," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, pp. 21414–21428, 2022.

[9] C. Wu, F. Wu, L. Lyu, Y. Huang, and X. Xie, "Communication-Efficient Federated Learning via Knowledge Distillation," *Nat Commun*, vol. 13, no. 1, p. 2032, 2022.

[10] H. Seo *et al.*, "16 Federated Knowledge Distillation," in *Machine Learning and Wireless Communications, Cambridge University Press,* 2022, pp. 457–485,

[11] B. Zhao, Q. Cui, R. Song, Y. Qiu, and J. Liang, "Decoupled Knowledge Distillation," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, 2022, pp. 11953–11962.

[12] Z. Wu *et al.*, "Exploring the Distributed Knowledge Congruence in Proxy-data-free Federated Distillation," *ACM Trans. Intell. Syst. Technol.*, p. 3639369, 2023.

[13] S. Reddi *et al.*, "Adaptive Federated Optimization," In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[14] A. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images". 2009.

[15] L. Yao and J. Miller, "Tiny ImageNet Classification with Convolutional Neural Networks". 2015.

[16] B. McMahan *et al.*, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proc. Int. Conf. Artif. Intell. Stat., AISTATS,* PMLR, 2017, pp. 1273–1282.

[17] N. Shoham *et al.*, "Overcoming Forgetting in Federated Learning on Non-IID Data," 2019, *arXiv:* arXiv:1910.07796.

[18] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated Optimization in Heterogeneous Networks," in *Proc. Mach. Learn. Syst.*, vol. 2, pp. 429–450, 2020.

[19] L. Zhang *et al.*, "Fine-Tuning Global Model via Data-Free Knowledge Distillation for Non-IID Federated Learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 10174–10183.