Fed-PEMC: A Privacy-Enhanced Federated Deep Learning Algorithm for Consumer Electronics in Mobile Edge Computing

Qingxin Lin, Shui Jiang, Zihang Zhen, Tianchi Chen, Chenxiang Wei, and Hui Lin

***Abstract*—Consumer electronic devices often involve processing and analyzing a large amount of user personal data. Nevertheless, owing to apprehensions regarding privacy and security, users are hesitant to transmit this sensitive data to centralized cloud servers for training. The combination of mobile edge computing and federated learning (FL) enables local devices to access computational power and storage resources, allowing them to engage in distributed learning and model training while safeguarding user privacy. However, these resources are not unlimited. Furthermore, as artificial intelligence technology progresses, inference attacks have become a major threat to privacy in traditional federated learning. To address these challenges, we propose an innovative federated deep learning algorithm, called Fed-PEMC. This algorithm combines local differential privacy and model compression techniques. By leveraging deep reinforcement learning for model compression, Fed-PEMC reduces model size while maintaining model accuracy, improving communication efficiency. We also introduce customized label sampling to accelerate model training. Before uploading the model, we implement local differential privacy protection on the compressed model, reducing privacy budget and addressing privacy leakage caused by inference attacks. Theoretical analysis and experimental results validate that Fed-PEMC adheres to (ϵ, δ)-differential privacy and exhibits a communication cost linked to the model size. Experimental results show that compared to baseline algorithms, Fed-PEMC excels in ensuring privacy, maintaining model accuracy, and optimizing communication efficiency, and Fed-PEMC outperforms the baseline solution DP-Fed by 2.27 and 2.02 percentage points in testing accuracy on the Mnist and Cifar10 datasets, respectively.**

***Index Terms*—Federated learning, mobile edge computing, consumer electronics, deep reinforcement learning, local differential privacy.**

Manuscript received 30 August 2023; revised 30 October 2023 and 11 December 2023; accepted 1 January 2024. Date of publication 9 January 2024; date of current version 26 April 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 61702103 and Grant U1905211, and in part the Natural Science Foundation of Fujian Province under Grant 2020J01167 and Grant 2020J01169. *(Corresponding author: Hui Lin.)*

Qingxin Lin is with the Fuzhou University Zhicheng College, Department of Computer Engineering, Fuzhou 350002, Fujian, China (e-mail: lqx@fdzcxy.edu.cn).

Shui Jiang, Zihang Zhen, Tianchi Chen, Chenxiang Wei, and Hui Lin are with the College of Computer and Cyber Security, Fujian Normal University, Engineering Research Center of Cyber Security and Education Informatization, Fujian Province University, Fuzhou 350117, Fujian, China (e-mail: jiangshui87176@gmail.com; zzihang@foxmail.com; 17633806860@163.com; 18965004813@163.com; linhui@fjnu.edu.cn).

Digital Object Identifier 10.1109/TCE.2024.3351648

I. Introduction

AN EMERGING technology in the field is Mobile Edge Computing (MEC), it has experienced a surge in popularity in recent years. It has emerged as a solution to the challenges encountered in traditional cloud computing, including issues related to high latency, limited bandwidth, and security concerns. The core concept of mobile edge computing revolves around the deployment of computing resources, encompassing processing power, data storage, and networking equipment, in closer proximity to end-users, typically at the network's edge. The primary objective of MEC is to deliver swifter response times and reduced latency, ultimately ensuring a smooth and efficient user experience when accessing the Internet. Zhou et al. improved the privacy and resilience of mobile machine systems through distributed P2P federated learning [1].

The inception of mobile edge computing can be attributed to the widespread adoption of IoT devices, particularly within the consumer electronics industry. Zhou et al. achieved excellent results by applying federated reinforcement learning enhanced with digital twins to mobile networks [2]. Consumer electronics products, such as smartphones and smart home devices, demand rapid data processing and real-time responsiveness, prompting the need for data processing and analysis to occur in closer proximity to users. In the consumer electronics industry, federated learning is widely employed to enhance model performance, enhance data privacy, and enable personalized services. For instance, federated learning can be utilized to enhance the user experience on smartphones. By conducting local training on user devices, phones can learn personalized inputs, preferences, and behavioral patterns without the need to transmit sensitive data to a central server. This contributes to providing more accurate predictions and suggestions, such as autocomplete suggestions, voice recognition, and personalized recommendations. It's essential to emphasize that mobile edge computing doesn't replace cloud computing but rather complements it as a service. By harnessing the combined capabilities of mobile edge computing and cloud computing technologies, businesses can establish efficient and secure IT infrastructure to augment the performance and user experience of consumer electronics products. This paves the way for a wide array of opportunities for the application of mobile edge computing in the consumer electronics industry, enabling swifter and more intelligent functionalities and services for smartphones, smart home devices, smart wearables, and beyond.

Federated learning stands as a potential paradigm in the realm of artificial intelligence, offering efficient model training and built-in privacy protection [3], [4], [5], [6]. Zhou et al. applied technologies like federated learning to the Internet of Things, making IoT smarter and more efficient [7], [8]. Federated learning involves two primary entities: the client and the server. In federated learning, each client assumes the responsibility of training a local model based on an initial model and subsequently transmits the trained model to the server for aggregation. The combined model is then disseminated back to all participants as the new initial model. This iterative process continues until convergence is attained. It is widely acknowledged that federated learning is particularly effective in addressing the challenges of data silos. This suggests that the application of federated learning in the context of mobile edge computing can efficiently tackle the issue of distributed model computation. For instance, it has become increasingly common for banks to collect users' biometric data through mobile IoT devices for identity verification in banking services, such as facial recognition payments. However, banks encounter the challenges of operating in diverse industries, across multiple business scenarios, and collaborating with various terminal manufacturers (e.g., education, transportation, retail). Combining federated learning with mobile edge computing effectively resolves various business requirements. Specifically, the central computing node of the bank (head office) distributes basic models and SDK development toolkits to the branch banks (edge computing nodes). Various IoT application terminals, such as facial recognition payment POS machines, are developed using the SDK. Data collected by mobile IoT devices is uploaded to the edge computing nodes, where further training and optimization are performed based on the original basic model to adapt to specific usage scenarios. Simultaneously, the edge computing nodes upload the training results to the central computing node to optimize the basic model. Through this iterative process involving the edge computing infrastructure and the federated learning model, different business requirements are addressed, and the model of the central computing node gradually improves.

Federated learning is generally regarded as providing a specific degree of privacy safeguarding, primarily due to the fact that, in the federated learning process, individual client data is inaccessible to the server, and there is no sharing of data among the clients. However, with the continuous advancement of artificial intelligence, hackers are becoming increasingly sophisticated, employing intelligent and diverse attack strategies. Of particular concern are inference attacks, which pose a significant threat to data privacy by utilizing AI techniques to reverse-engineer the model and infer sensitive information [9]. This type of attack can result in data privacy disclosure during the process of uploading the model from the client to the server. Consequently, when attackers intercept the uploaded local model, they may potentially gain access to the client data embedded within that model. As a result, traditional federated learning approaches may fall short in effectively safeguarding the data privacy of clients.

Differential privacy, initially proposed by Dwork [10], is founded on rigorous mathematical principles. It is based on a method known as "randomized response," which introduces randomized noise to input or output data. This approach ensures that the impact of a single record on the dataset, when outputting information, always remains below a certain threshold. This prevents third parties from determining whether a single record has been modified, added, or deleted based on changes in the output. As a result, differential privacy is considered the highest level of security among current perturbation-based privacy protection methods. Differential privacy can be categorized into two variants: local differential privacy and global differential privacy, depending on where the noise is added. Given that both local differential privacy training and privacy safeguarding processes can be implemented on the client side, local differential privacy is better suited for federated learning. In fact, to address the issue of privacy leakage resulting from inference attacks, local differential privacy technology has been integrated into federated learning [11], [12], [13], [14]. While this effectively resolves the problem of data privacy disclosure in federated learning, it gives rise to another challenge, known as the "privacy budget explosion." This issue stems from the direct correlation between the privacy budget in the differential privacy mechanism and the growth of the model and the number of communication rounds. This leads to a substantial increase in the privacy budget, and it is important to note that there exists an inverse relationship between the privacy budget and model accuracy. Moreover, taking into account the extensive quantity of parameters in neural networks, adding noise to the model can significantly diminish model accuracy.

Numerous existing studies have focused on enhancing the privacy safeguarding of federated learning through the utilization of local differential privacy. These studies aim to mitigate the problem of a privacy budget surge by minimizing the size of local models. For instance, Hu et al. [15] consider model sparsity as a means of reducing the transmitted model's size. Ruixuan et al. [16] address the privacy budget concern by sampling model parameters to shrink the model. These studies address the trade-off between data privacy and accuracy in federated differential privacy by randomly selecting model parameters to drop. However, in federated learning, the impact of parameters on model training accuracy varies; in other words, some model parameters have a significant influence on training accuracy, and randomly discarding these crucial model parameters is not a viable solution. These algorithms treat all model parameters equally when compressing them, disregarding the differences in importance for each parameter. This approach can significantly hinder model training accuracy because it may lead to the removal of parameters that have a substantial impact on model accuracy, ultimately reducing the precision of the model.

Based on the above analysis, it is evident that addressing the following issues is crucial for ensuring the reliability of federated learning and safeguarding clients' privacy in the field of consumer electronics through mobile edge computing.

1) *Restricted Resources*: The computational and storage resources of devices in mobile edge computing are not unlimited.
2) *Privacy Disclosure*: Federated learning suffers from inference attacks that cause data privacy disclosure on clients.
3) *Privacy Budget Explosion*: Introducing a differential privacy mechanism to resist inference attacks can lead to a privacy budget surge, primarily due to the large number of model parameters involved.

To address the aforementioned challenges, this paper introduces a novel federated learning algorithm, named Fed-PEMC, for consumer electronics using mobile edge computing. Fed-PEMC incorporates a deep reinforcement learning algorithm(MCT) that takes into account the importance of model parameters, MCT(Model Compression during Training) achieves model compression in federated differential privacy by discarding parameters with minimal impact on model accuracy, addressing the trade-off between data privacy and usability. It utilizes deep reinforcement learning to identify and reduce model parameters that have minimal impact on model accuracy. By striking a balance between model accuracy and compression, Fed-PEMC effectively reduces privacy budgets.

Fed-PEMC has the potential for widespread applications in electronic consumer devices, enhancing user experience, strengthening data privacy protection, and reducing energy consumption. It has significant advantages in the following aspects. Electronic consumer devices typically contain users' personal data, such as smartphones and smart home devices. Federated learning allows model training to occur locally on the device, eliminating the need to send data to a central server, thus providing better user privacy protection. User data stays on their devices, reducing the risk of data leaks and privacy breaches(Privacy Protection). Federated learning enables on-device model training, reducing the latency associated with transmitting data to cloud servers and receiving results. This is particularly beneficial for applications requiring real-time feedback, such as smart homes, smartphones, and wearable devices(Low Latency). Local on-device model training reduces energy consumption associated with data transmission and cloud computing. This is crucial for electronic consumer devices that rely on battery power, as it can extend battery life and lower energy costs(Energy Efficiency). Federated learning allows electronic consumer devices to train models based on users' personalized needs, providing a better user experience. For example, smartphones can optimize recommendation algorithms based on users' habits and interests, and smart home devices can adapt to the needs of household members(Personalized Services). Since data doesn't need to be transmitted to cloud servers, federated learning can reduce network congestion, improving data transmission efficiency(Mitigation of Network Congestion). The principal contributions of this paper are delineated as follows.

- To protect the data privacy of federated learning clients, we incorporate a local differential privacy mechanism. To mitigate the issue of privacy budget explosion arising from an excessive number of model parameters, we only introduce noise to the compressed model. Through rigorous theoretical analysis, we have demonstrated that the proposed Fed-PEMC algorithm satisfies (ϵ, δ)-differential privacy, where ϵ represents the allocated privacy budget and δ denotes the confidence parameter.
  To ensure the accuracy of the compressed model, we have designed a deep reinforcement learning algorithm called MCT, we employ MCT during the training process to compress the model. Additionally, we utilize customized label sampling techniques to accelerate the training process. Uploading the compressed model to the server further enhances the communication efficiency of federated learning. Theoretical analysis reveals that the communication cost of Fed-PEMC is estimated to be $o(R/2|w|)$, where R represents the number of federated learning rounds, and $|w|$ denotes the model size.
- We conducted comprehensive experiments using the MNIST and CIFAR10 datasets to assess the efficiency of the proposed Fed-PEMC algorithm. The experimental results clearly demonstrate the superiority of Fed-PEMC over baseline algorithms, namely FedAvg [17] and DP-Fed [18], in terms of privacy protection, model accuracy, and communication efficiency.

The structure of this paper is detailed as follows: Section II covers the related work, Section III explains the system model, Section IV outlines the implementation specifics of the proposed Fed-PEMC, Section V presents the theoretical analyses, Section VI delves into the performance evaluation, and Section VII encapsulates the paper's conclusion.

II. Related Work

Privacy-enhanced federated learning has garnered substantial attention from both academia and industry, particularly within the realm of consumer electronics, resulting in a multitude of publications in this domain. Existing research predominantly centers on the implementation of privacy protection through either global differential privacy or local differential privacy mechanisms, which involves introducing noise to client data during the federated learning process.

Federated Learning based on Global Differential Privacy: In [15], Hu et al. presented the sparse amplification privacy, a novel federated learning framework that combines random sparsification with gradient perturbation to bolster privacy assurance. In [19], Chuanxin et al. introduced the Noisy-FL algorithm, a Gaussian differential privacy-based federated learning approach that offers user-level privacy protection. Ruixuan et al. [16] utilized the privacy amplification in the recently introduced shuffle model, incorporating differential privacy to optimize the utilization of the privacy budget. Sun et al. [20] proposed a newly designed local differential privacy mechanism for joint learning, addressing privacy explosion and weight range disparities in various deep learning model layers. This mechanism not only provides robust privacy guarantees but also enhances training efficiency.

TABLE I
THE SYMBOL LIST

Symbol	Representatives
D_i	the dataset of client i
B	the local mini-batch size
E	the number of local epochs
η	the learning rate
η_g	the global learning rate
θ	the model parameter
ϵ	the privacy budget
δ	the failure probability
G	the Bounded Gradient
g	the gradient
σ	the Gaussian distribution variance
$\phi()$	the L2 sensitivity
L	the L-smooth
μ	the μ-strongly convex
e^{th}	the error threshold
α	the privacy budget of RDP
β	the failure probability of RDP
d	the dimensions of the model parameters
ξ	the dataset of small batch
c	the compression rate

Hu et al. [21] introduced a novel differential private federated learning scheme, Fed-SMP, which offers client-level differential privacy safeguarding while preserving high model accuracy. Hu et al. [22] devised an effective personalized model learned from distributed user data, ensuring the differential privacy of client's data. Gong et al. [23] improve privacy and communication efficiency under a federated framework by using unlabeled cross-domain public data for one-time offline knowledge differentiation. Lu et al. [24] proposes a novel asynchronous federated learning scheme for vehicle network resource sharing, and further put forward a stochastic distributed update scheme to fortify federated learning security.

Federated Learning based on Local Differential Privacy: In [25], Wang et al. introduced a local differential privacy-based federated framework and provided theoretical assurance for data privacy and model accuracy. Sun et al. [26] addressed the trade-off between privacy assurance and model accuracy by proposing a novel framework utilizing a noise-free differential privacy mechanism in a federated model distillation setup, effectively securing local data privacy with minimal impact on model utility. In [27], Luping et al. proposed a new federated scheme that can greatly lower the communication cost while ensuring training convergence. In [28], Wu et al. adjusted the gradient descent process by an adaptive approach for adjusting learning rate to make the training process more efficient under resource constraints. For the problem of utility inequity in wireless IoT, Alvi et al. [29] made the training process more efficient by using differential privacy to limit the aggregation model leakage, so as to upgrade the quality of the aggregation model. In [30], Jiang et al. proposed a novel multidimensional selection method in federated learning to further ameliorate the convergence process and performance of the model. In [31], a new blended differential privacy is proposed for industrial data processing with a federated edge framework to address the issue of models subject to inference attacks. Yin et al. [32] introduced an edge federated architecture that employs a dynamic local differential privacy algorithm to resist internal attacks on edge devices. In federated learning setting, the subject granular privacy was introduced by Marathe et al. [33]. In such setting, a subject is referring to an individual whose private information is embodied by several data items either distributed across multiple federation users or confined within a single federation user.

The comparison of mainstream privacy-enhanced federated learning algorithms is given in Table II. In [11], the authors employ a three-plane framework and Local Differential Privacy (LDP) to safeguard privacy in federated learning. While LDP enhances privacy safeguarding in federated learning, it can have a negative impact on data availability. In [34], the authors expedite the convergence process through local custom training, significantly improving communication efficiency. However, they do not address privacy security issues in VEC. In [35], the authors devise a fairness-aware and time-sensitive task allocation mechanism to resolve coordination and scheduling problems in CEI systems, greatly enhancing the responsiveness of CEI systems. Nevertheless, privacy protection is not extensively considered in these works. Even though they substantially enhance communication efficiency in federated learning, and these algorithms can provide a certain level of privacy safeguarding for the federated learning, but these recent studies primarily have two main issues. On the one hand, They cannot simultaneously address privacy, communication efficiency, and data availability in all three aspects. On the other hand, in federated differential privacy, reducing model parameters randomly for model compression has a significant negative impact on the final training model accuracy. To solve these problems, the algorithm Fed-PEMC is proposed in this paper, which solves the problem of privacy budget explosion by implementing local differential privacy protection for the compressed model.Fed-PEMC uses the MCT algorithm to consider the impact of parameters on model accuracy when deciding whether to discard a parameter, thus achieving model compression while maintaining model training accuracy. This addresses the trade-off between resource constraints and data accuracy and privacy. As far as I'm aware, our approach is the first to study model compression in federated differential privacy while considering model training accuracy, and it resolves the issues mentioned above.

III. SYSTEM MODEL

In this section, we present a comprehensive system model overview for the proposed Fed-PEMC algorithm, and discuss potential threats that it may face. Additionally, we introduce the concept and application of local differential privacy as a means of privacy safeguarding in the scenario of our federated learning.

Federated Learning Architecture: The system model of the Fed-PEMC is shown in Fig. 1, which consists of two entities, namely the client and server.

1) *Client:* In the context of consumer electronics, each participating client, such as a smartphone or wearable equipment, trains its local model by locally collected data. To ensure efficient and privacy-preserving federated learning, each client compresses the uploaded model before send it to the server.

TABLE II
COMPARISON OF MAINSTREAM DIFFERENTIAL PRIVACY ENHANCED FEDERATED LEARNING ALGORITHMS

Ref	Scenes	Advantages	Limitations
[11]	Cross-silo FL	Ensure user-level privacy	No consideration of data availability
[12]	FL	Adaptively perturb the residual weights using LDP	Adaptive perturbation adds additional cost
[13]	LDP-FL	Reduce communication overhead while protecting privacy	Privacy budget explosion
[14]	FL	Achieves greater utility and smaller transfer rates	Limited user privacy protection
[10]	DP	Quantitative protection of user privacy	Limited user privacy protection
[19]	DP-FL	Protect users' privacy through DP	Privacy budget explosion
[15]	LDP-FL	Improve communication efficiency, reduce privacy budget	Compression model fails in model accuracy
[16]	DP-FL	Solve the privacy budget explosion by model shuffling	Shuffling model adds extra cost
[20]	LDP-FL	Proposed mechanism bypasses the curse of dimensionality	Model accuracy is not considered
[21]	FL	Improve model accuracy, protect client-level DP	Model sparsity cause model accuracy loss
[25]	LDA	Solve the privacy problem in LDA by LDP-FL	Privacy budget is not considered
[26]	Fed-Distillion	Noise-free DP	Supports no privacy preserving ML methods
[27]	FL	Mitigate communication overhead	Reducing model parameters affects accuracy
[28]	DP-FL	Adaptive gradient descent, reduce communications costs	Privacy budget explode
[29]	IoT	Address privacy concerns in wireless IoT	Fails in the tradeoff between privacy and accuracy
[30]	Signds-FL	Improve model convergence and accuracy	Limited user privacy protection

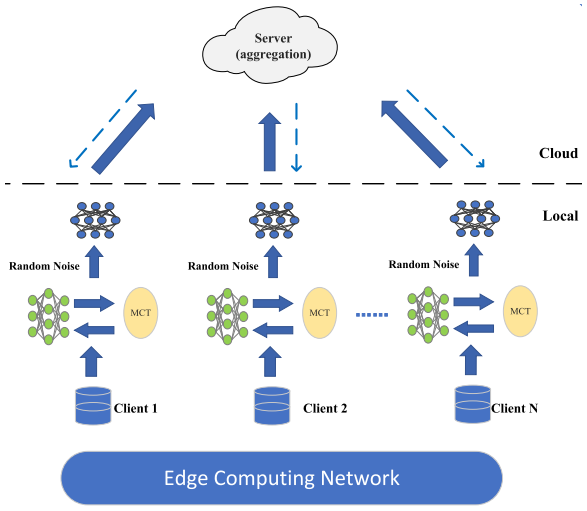

Fig. 1. System model of the proposed Fed-PEMC.

2) *Server:* The server acts as the central aggregator and receives the compressed models from all the clients. It then combines the models to create an aggregated model and dispatches it back to each client. This iterative process continues until the aggregated model achieves the desired accuracy or the predetermined maximum number of federated learning rounds is reached.

By incorporating compression techniques, federated learning in consumer electronics optimizes the transmission of models, decreasing the communication cost and improving the overall efficiency of the learning process. Additionally, privacy safeguarding mechanisms, for example differential privacy, are applied to guarantee the privacy of user data during the model aggregation.

Security Threats: In this paper, we specifically concentrate on privacy disclosure as the primary security threat. Protecting user privacy data is crucial in the context of consumer electronics. Therefore, we assume that all roles in the system model, including both clients and servers, are honest and curious. Under this assumption, clients are not expected to engage in malicious behavior, such as launching poisoning attacks to generate malicious models. Similarly, servers are not expected to conduct malicious operations like attacking, decrypting, or reverse engineering client-uploaded models.

IV. THE IMPLEMENTATION DETAILS OF THE PROPOSED FED-PEMC

A. Overview of Fed-PEMC

The framework of the proposed Fed-PEMC is designed to be applicable to consumer electronic products. It consists of two main components: cloud update and local update. (see Fig. 2).

Local Differential Privacy: Differential privacy (DP) was proposed by Dwork to enhance data privacy by adding random noise. It is extensively employed in various domains, such as federated learning, to thwart inference attacks. However, traditional DP requires a central trusted server, which may not be practical due to increased noise on the server side. To address this limitation, local differential privacy (LDP) was

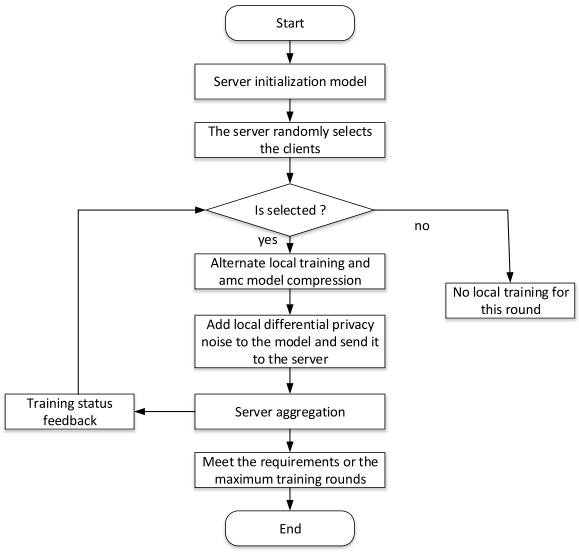

Fig. 2. The flow chart of the proposed Fed-PEMC.

introduced. LDP allows clients to add random perturbation to their local models, protecting the privacy of the models. LDP enables federated learning systems to maintain privacy without relying on a central trusted server. By incorporating LDP into federated learning, clients can contribute their local models while preserving data privacy. This technique enhances privacy protection and reduces the potential for privacy breaches during model aggregation.

Cloud Update: In the cloud update phase, the server randomly initializes the model weights. In the context of consumer electronics, the server can represent a cloud-based service or a centralized server. There exists n local clients, which could be consumer electronic devices such as smartphones, wearables, or IoT devices. In each round of communication, the server stochastically selects k clients (where $k \leq n$) for local training. This selection process can be performed based on factors such as device availability, battery level, or network conditions. After collecting the trained local models from the client-side, the server performs adaptive updates and sends the final result to all client agents.

Local Update: In the local update phase, each client, representing a consumer electronic device, utilizes a stochastic gradient descent algorithm for r rounds of local training, where $r \leq m$. The number of local training rounds m, can be determined based on factors for example equipment capability, battery life, or user preferences. During the local training, each client can leverage its own dataset, which may include sensor readings, user behavior logs, or other relevant data collected by the consumer electronic device. To accelerate the training process, a customized label sampling method can be employed to efficiently select training samples. The client feeds the semi-trained local model to a machine learning algorithm [36], [37], which helps to reduce the size of the model while maintaining its accuracy. The training and compression processes alternate until a balance between model accuracy and size is achieved. Additionally, to enhance privacy protection, random noise can be added to the trained model before uploading it to the server. Finally, the selected clients upload their trained local models to the server and receive a new model as the initial model for the next round of local training. This iterative process of cloud update and local update enables collaborative model training while considering privacy protection, computational efficiency, and resource-constrained nature of consumer electronic devices.

In order to strengthen the privacy protection of clients, local differential privacy is often employed. However, LDP introduces the privacy budget explosion problem, which necessitates the compression of models before adding noise. A model compression algorithm called AMC has been proposed in literature [38]. AMC utilizes deep reinforcement learning to provide compression policies for different models. However, AMC compresses the model after it has been trained, which can potentially impact the accuracy of the local model and subsequently the global model. To address this issue and achieve higher compression ratios while maintaining better model accuracy, we propose a novel compression method called MCT. MCT compresses the model during the training process, which allows us to reduce the privacy budget while ensuring model accuracy. This approach enables us to strike a tradeoff between privacy preservation and model accuracy. Furthermore, uploading compressed models to the server improves the efficiency of client-to-server communication. Compressed models require less bandwidth for transmission, reducing communication costs and improving overall system efficiency. The pseudo-code of Fed-PEMC, is presented in Algorithm 1.

B. MCT: Model Training During Compressing

1) Deep Reinforcement Learning Algorithm in MCT: MCT uses the deep deterministic policy gradient algorithm (DDPG) to train models while compressing them. The DDPG algorithm is built upon AC framework, which includes the Actor and Critic network, and each with its respective target network.

To illustrate the mode of training while compressing, we first define the three basic elements for the DDPG algorithm used in MCT, namely State, Action, and Reward.

- *State:* The local model considered in this paper is a convolutional neural network (CNN), and the MCT algorithm will compress the CNN layer by layer. This means that the state is defined based on the CNN layer. For each layer l, we use 11 feature parameters to represent the state as

$$S^l: \big(t, reduced, n, c, h, rest, w, stride, k, FLOPs[l], a(l-1)\big), \quad (1)$$

The kernel has a dimension of $(n \times c \times k \times k)$, while the input size is $(c \times h \times w)$. $FLOPs[t]$ denotes the quantity of floating-point operations in layer l. Additionally, *reduced* is indicative of the total count of reduced *FLOPs* across all prior layers, while *rest* signifies the remaining count of stream points in subsequent layers. Moreover, $a(l-1)$ refers to the action taken in the preceding layer, which will be scaled within the range of $[0, 1]$ before being relayed to the agent.

Algorithm 1 The Fed-PEMC Algorithm

Require: the dataset of client i D_i, the number of local client n, the local mini-batch size B, the number of local epochs E, the learning rate η, and the global learning rate η_g.

ClientUpdate:

Receive θ^{t+1} from server
for all each local client $i \in k$ in parallel **do**
  compute $P_i(y_j)$ according to Eq. (11)
  sample B according to $P_i(y_j)$ from D_i
  for all each local epoch $s = 0, 1, \ldots, E$ **do**
    for all each batch $b \in B$ **do**
      $\theta_i^{t,s+1} \leftarrow \theta_i^{t,s} - \eta g_i^{t,s}$
      If$(s\%r == 0)$
      $\theta'^{t,s+1}_i \leftarrow MCT(\theta_i^{t,s+1})$
    end for
  end for
  upload θ'^{t}_i after adding noise $b_i^{t,s} \sim N(0, \sigma^2 I_d)$
end for

ServerUpdate:

Initialize model θ^0.
Send θ^0 to all clients.
for all each round $t = 1, 2, \ldots, T$ **do**
  randomly select $k(k < n)$ local clients.
  collect all models θ^t from selected clients
  perform adaptive updates according to formula (13)
  send θ^{t+1} to all clients
end for

- *Action:* The model compression ratio of the model will be taken as an action, and the action will be taken within the continuous action space denoted as $a \in (0, 1]$, allowing for precise and fine-grained compression.
- *Reward:* Since in Fed-PEMC, we compress models as much as possible while maintaining model accuracy, we need to set reward functions that are sensitive to model accuracy. We know that the error exhibits an inverse relationship with either $log(FLOPs)$ or $log(\#Param)$, thereby we set the reward function as follows:

$$R_{FLOPs} = -log(FLOPs) \times Error, \tag{2}$$
$$R_{Param} = -log(\#Param) \times Error. \tag{3}$$

MCT obtains the rational model compression ratio by learning the optimal policy π from the actor network $\pi(.|\theta^{\pi})$, which take the state as the input and the action as the output, i.e., $\pi(s|\theta^{\pi}) \rightarrow a$. To better explore actions, we use noise generation and truncated normal distribution based on the Ornstein-Uhlenbeck process for action generation, as

$$\pi'(s_t) \sim TN\Big(\pi\big(s(t)|\theta_t{}^{\mu}, \sigma^2, 0, 1\big)\Big), \tag{4}$$

where the noise σ is initialized to 0.5 and decays exponentially in the course of each iteration thereafter. Fig. 3 presents the basic idea of the MCT algorithm.

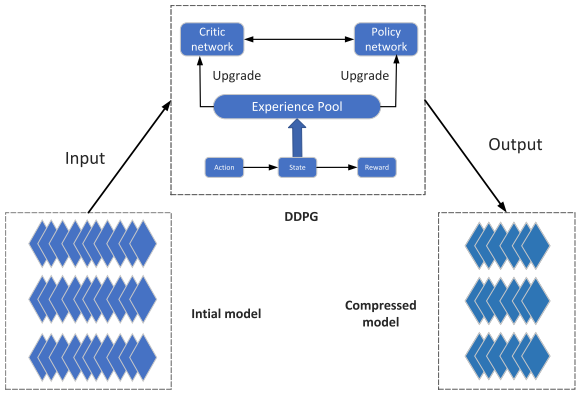

Fig. 3. The model compression by MCT.

2) The Train Process of MCT: The sampling process of the DDPG algorithm is to randomly select an action $s(t)$ based on noise and current strategy, calculate the reward value and obtain a new state $s(t+1)$ after executing the action, and finally store the experience in the experience pool. The objective in training an actor and a critic network within deep reinforcement learning is to maximize accumulated rewards and minimize the error between the evaluation value and the target value, respectively.

For the critic network, the loss function of which is defined as follows:

$$L(\theta_Q) = \frac{1}{N}\sum_t \Big(y(t) - Q\big(a_\lambda(t)|\theta^Q, s(t)\big)\Big)^2. \tag{5}$$

The Q value in Eq. (5) is calculated by

$$Q(s(t), a_\lambda(t)) = E\Big[r\big(s(t), a_\lambda(t)\big) + \gamma Q\Big(s(t+1), \pi\big(s(t+1)\big)\Big)\Big], \tag{6}$$

While the target value $y(t)$ is computed by

$$y(t) = \gamma Q'\Big(s(t+1), \pi'\big(s(t+1)|\theta^{\pi'}\big)|\theta^{Q'}\Big) + r(t), \tag{7}$$

where the reward value can be computed based on the operational state $s(t)$ and the action $a(t)$, and Q' is computed by the critic target network built on the next state $s(t+1)$. For the actor network, the loss function of which is defined as follows:

$$\nabla_{\theta^\pi} J \approx_\pi \Big[\nabla_a Q\Big(s, a|\theta^Q\Big)\big|_{s=s(t), a=\pi\left(s(t)\right)} \nabla_{\theta_\pi} \pi\big(s|\theta^{\pi}\big)\big|_{s=s(t)}\Big] \tag{8}$$

By softly updating the parameters in the target network, the output tends to be more stable. This makes using the target network to compute the target value naturally more steady, thereby ensuring a more stable learning process for the critic network. Thereby, the following equations are used for target network update:

$$\theta^{Q'} \leftarrow (1-\tau)\theta^{Q'} + \tau\theta^{Q}, \tag{9}$$
$$\theta^{\pi'} \leftarrow (1-\tau)\theta^{\pi'} + \tau\theta^{\pi}. \tag{10}$$

In general, the DDPG Agent generally takes the embedding state $s(t)$ of the layer $L(t)$ from the environment and generates

a sparse ratio as an action. It employs a designated compression algorithm (channel pruning) to compress the underlying layer at a rounded feasible fraction. The agent then progresses to the subsequent layer $L(t+1)$ and receives the state $s(t+1)$. Upon finishing the final layer $L(T)$, the accuracy of the reward is assessed on the validation set and conveyed back to the agent.

C. Customized Label Sampling For Training Accelerating

Using deep reinforcement learning to compress local models in federated learning takes a certain amount of time, which may lead to a long period of training time for Fed-PEMC. How to speed up the training process of the Fed-PEMC is an open problem. As shown in [39] the deep neural networks often prioritize learning the majority class while neglecting the minority class. For example, the majority class in the sample facilitates the overall training, the accuracy of the training model can be impacted by the data of the minority class in the sample. This suggests that we can solve the problem of long training time by adjusting the sampling probability. That is, for most types of data in the sample, we give a relatively large sampling probability, while for a few types of data in the sample, we give a relatively small sampling probability. Combing the customized label sampling proposed by Zhang et al. [40], we formulate the sampling probability of class label y_j in client k by

$$P_k(y_j) = \frac{n_k^{y_j}}{\sum_{i=1}^{N} n_k^{y_j}}, \tag{11}$$

where N is the total count of tag categories for client k.

In line 20 of Algorithm 1, add noise $b_i^{t,s} \sim N(0, \sigma^2 I_d)$ to the locally trained model before dispatching it to the server.

D. Server-Side Adaptive Updates

Considering that compressing the federation local model by deep reinforcement learning slows down the whole process of federation training, we utilize customized label sampling to accelerate the local training of clients in the edge computing network. The Adam algorithm is an extension of the stochastic gradient algorithm, where the learning rate is kept constant, and the Adam algorithm optimizes the training process by continuously and adaptively changing the learning rate during the training process. In edge computing networks, the resources of local training devices or clients are very limited, and the clients do not always participate in local training. Therefore, it is not a good choice for the local client to optimize the training process by Adam's algorithm. Inspired by [41] and [15], we consider the server side to optimize the training process by using Adam's algorithm. The server-side retains two momentum vectors $u, m \in R_d$, which are updated once per round. Specifically, in round t, subsequent to the local training of the client agents, agent $i \in W$ sends its model update $\Delta\theta_t$ to the server, which then updates the aggregation model by the following algorithm:

$$\begin{cases} m_t = \alpha_1 m_{t-1} + (1-\alpha_1)G, \\ u_t = \alpha_2 u_{t-1} + (1-\alpha_2)G^2, \\ \theta_{t+1} = \theta_t - \eta_g m_t / \left(\sqrt{u_t} + \lambda\right). \end{cases} \tag{12}$$

where α_1 and α_2 are the momentum parameters, $G = \sum_{i \in W} \Delta_i^t / |W|$, η_g is the global learning rate, and λ is the adaptive degree parameter.

V. Theoretical Analysis

A. Privacy Analysis

In this section, we'll discuss the privacy guarantees of Fed-PEMC. The assumptions and lemmas necessary for the privacy analysis are presented first.

Assumption 1 (Bounded Gradient [15]): The loss function $l_{(i)}(x, z)$ has $G/\sqrt{d}$-bounded gradients, i.e., for any data sample z from D_i, we get $|[\nabla l_i(x, z)]_j| \le G/\sqrt{d}$ for all $j \in [d]$, $i \in [n]$ and $x \in R^d$.

Lemma 1 (RDP Composition [42]): If M_1 satisfies $(\alpha, \rho_1) - RDP$ and M_2 satisfies $(\alpha, \rho_2) - RDP$, then their composition $M_1 \circ M_2$ satisfies $(\alpha, \rho_1 + \rho_2) - RDP$.

Lemma 2 (Gaussian Mechanism [42]): Let $h{:}D \to R^d$ be a vector-valued function over datasets. The Gaussian mechanism $M = h(D) + b$ with $b \sim N(0, \sigma^2 I_d)$ satisfies$(\alpha, \alpha\phi^2(h)/2\sigma^2) - RDP$, where $\phi(h)$ is the L_2 sensitivity of h defined by $\phi(h) = sup_{D,D'} ||h(D) - h(D')||_2$ with D, D' being two neighboring datasets in D.

Lemma 3 (RDP to DP conversion [43]): If M satisfies $(\alpha, \rho) - RDP$, then it also satisfies$(\rho + \frac{log(1/\delta)}{\alpha - 1}, \delta) - DP$.

Lemma 4 (RDP for Subsampling Mechanism [43], [44]): For a Gaussian mechanism M and any m-datapoints dataset D, we define $M \circ$ SUBSAMPLE as applying M on the subsampled dataset as input, where B datapoints are subsampled without replacement from the dataset with $q = B/m$ as the sampling ratio.

If M satisfies$(\alpha, \rho(\alpha)) - RDP$ concerning the subsampled dataset for all integers $\alpha \ge 2$, then the new randomized mechanism $M \circ$ SUBSAMPLE satisfies $(\alpha, \rho'(\alpha)) - RDP$ concerning D, where

$$\rho'(\alpha) \le \frac{1}{\alpha - 1} log(1 + q^2 \binom{\alpha}{2} min\{4(e^{\rho(2)} - 1)\} + \sum_{j=3}^{\alpha} q^2 \binom{\alpha}{j} 2e^{(j-1)\rho(j)}) \tag{13}$$

If $\sigma'^2 = \sigma^2/\phi^2(h)$ and $\alpha \le (2/3)\sigma^2 log(1/q\alpha(1 + \sigma'^2)) + 1$, then $M \circ$ SUBSAMPLE satisfies $(\alpha, 3.5q^2\phi^2(h)\alpha/\sigma^2) - RDP$.

Theorem 1 (Privacy Guarantee): Assuming small batches $\xi_i^{t,s}$ are sampled each iteration without replacement, we denote the data sampling rate by $q = B/m$, We let τ represent the number of local iterations, I_i represent the number of rounds the agent i has participated in, $r_{i,t}$ represent the compression rate of agent i at tth round iteration, and R denote the average compression rate over T federated learning rounds. Under **Assumption 1**, if $\sigma'^2 = \sigma^2 B^2/2RG^2 \ge 0.7$, then Fed-PEMC complishes $(\epsilon, \delta) - DP$ for client i, where

$$\epsilon = \frac{log(1/\delta)}{\alpha - 1} + \frac{7q^2 I_i \tau \alpha R G^2}{B^2 \sigma^2} \tag{14}$$

for any $\delta \in (0, 1)$ and $\alpha \le (2/3)\sigma^2 log(1/q\alpha(1 + \sigma'^2)) + 1$.

Proof: Our differential privacy mechanism for Fed-PEMC leverages the privacy guarantee of Gaussian noise, which is

amplified by the compression model. To assess end-to-end privacy, we begin by analyzing the sensitivity of Algorithm 1 at line 9. We denote the model gradient and added Gaussian noise of agent i after the tth compression round as $g_i^{t,s}$ and $b_i^{t,s}$, respectively. We first analyze the sensitivity of $g_i^{t,s}$ and then calculate the privacy guarantee after adding $b_i^{t,s}$ for agent i, assuming that any two neighboring data sets $\xi_i^{t,s}$ and $\xi_i'^{t,s}$ have the same size B but differ in one data sample (e.g., $z \in \xi_i^{t,s}$ and $z' \in \xi_i'^{t,s}$). Since the model parameters are compressed by deep reinforcement learning in Fed-PEMC, the compression rate of each agent varies at different iteration rounds. Therefore, the $L2$ sensitivity can be expressed as $\phi_{i,t}^2 = max||g_i^{t,s} - [\bigtriangledown f_i(\theta_i^{t,s}, \epsilon_i^{t,s})]||^2 = max||(1/B)[\bigtriangledown l(\theta_i^{t,s}, z) - \bigtriangledown l(\theta_i^{t,s}, z')]||^2$ and under Assumption 1 with $2r_{i,t}G^2/B^2$ as a bound. We find that the sensitivity of $g_i^{t,s}$ is proportional to the compression ratio, which reduces the privacy loss according to Lemma 2.

We then proceed to prove the end-to-end differential privacy guarantee for Fed-PEMC in Theorem 1. By leveraging Lemma 2, we can determine that a small batch $\xi_i^{t,s}$ of subsampling satisfies $(\alpha, \alpha = r_{i,t}G^2/B^2\sigma^2) - RDP$ in each local iteration of Algorithm 1. Furthermore, Lemma 4 guarantees that $M\circ$ SUBSAMPLE satisfies $(\alpha, 3.5q^2\phi^2(h)\alpha/\sigma^2) - RDP$ by subsampling. Finally, by applying Lemma 1 to Lemma 4, we derive Theorem 1.

Theorem 1 concludes that, for a constant value of δ, ϵ is determined through numerical computation by seeking the optimal α that minimizes ϵ.It's notable that the noise magnitude σ scales with R, indicating that reducing the compression ratio below 1 could decrease the Gaussian noise amplitude and enhance model accuracy.

B. Communication Cost

In this section, we examine the communication expense of the proposed Fed-PEMC.

Theorem 2 (Communication Cost): Assume that the number of federated learning global training rounds is T rounds. The compression rate of each round is different because the local client compresses the model by the AMCT algorithm individually. Let the compression rate of the local model in round i as $C_i = \frac{|w|}{|w_i|}$, where w is the model before compression and w_i is the model after compression. The communication cost of Fed-PEMC is $o(T/C|w|)$ for any agent i.

Proof: First we give the total amount of communication data for the federated average algorithm as

$$V_{(FedAvg)} = T|w|. \tag{15}$$

Thus, the total amount of communication data for the proposed Fed-PEMC can be calculated by

$$V_{(Fed-PEMC)} = \sum_{i=1}^{T} |w_i| = \sum_{i=1}^{T} \frac{1}{C_i}|w|. \tag{16}$$

Let C denote the average compression rate over the entire training process, then we have $\sum_{i=1}^{T} \frac{1}{C_i}|w| = \frac{T}{C}|w|$, then Theorem 3 is proved. Since the communication cost of Fed-PEMC is $o(T/C|w|)$ for any agent i. it can be inferred that its computational complexity is inversely proportional to C. The larger the value of C, the smaller its computational complexity. Without using the MCT algorithm, the communication overhead for the baseline solution is $o(T/|w|)$, which means that Fed-PEMC's communication overhead is $1/C$ of the baseline solution, our approach significantly reduces communication overhead.

C. Convergence Analysis

In this subsection, we're going to analyze the convergence of the Fed-PEMC scheme. Our primary analytical conclusion is that Fed-PEMC has a $O(\frac{1}{T})$ convergence rate, which ensures that it converges to the global optimum.

For the convenience of illustration, we assume that the local data size for all clients is equal, i.e., $D_i = D_j$ for any $i, j \in K$. We select N clients to train locally for E rounds per iteration. Below, we present three assumptions that support the proof of our convergence analysis.

1) *L-smooth* [45]: $\forall x, y, F(y) \leq F(x) + (y-x)^T \bigtriangledown F(x) + \frac{L}{2}||y-x||^2$;
2) *U-strongly convex* [45]: $\forall x, y, F(y) >= F(x) + (y-x)^T \bigtriangledown F(x) + \frac{\mu}{2}||y-x||^2$;
3) *Bounded gradient and bounded variance of gradient* [45]: $E[||\bigtriangledown F(x^k[t], \xi^k[t]) - \bigtriangledown F(x^k[t])||^2] \leq \sigma^2$ and $E[||\bigtriangledown F(x^k[t], \xi^k[t])||] \leq G^2$.

Theorem 3 (Convergence Analysis): If the three aforementioned assumptions hold, we define $K = L/\mu$, $\gamma = max\{8K, E\}$, and the learning rate $\eta = \frac{2}{\mu(\gamma+t)}$. If the error threshold in Fed-PEMC is as follows:

$$e^{th}[t] <= \eta_t^2 = \frac{4}{\mu^2(\gamma+t)^2} \sim O\left(\frac{1}{t^2}\right), E\left[e^i\right] = 0 \quad \forall i \in K. \tag{17}$$

Then all local clients participating in training in Fed-PEMC satisfy:

$$E\left[F(\bar{y}[T]) - F\left(y^*\right)\right] \leq \frac{K}{\gamma + T - 1}\left(\frac{2B}{\mu} + \frac{\mu\gamma}{2}\left[||y[1] - y^*||^2\right]\right) \tag{18}$$

where

$$B = \sum_{i=1}^{N} \frac{\sigma_i^2}{N^2} + 6L\Gamma + 8(E-1)G^2 + E\left[||e^{th}||^2\right]. \tag{19}$$

Proof: By the L-smoothness assumption, we can derive the following inequality:

$$E\left[F(\bar{y}[t])\right] - F\left(y^*\right) \leq \frac{L}{2}E\left[||\bar{y}[t] - y^*||^2\right] \tag{20}$$

In the following section, we will utilize the results presented in [46]. However, it's important to note that, in Fed-PEMC, the uplink errors from different users are not independent. To address this issue, we constrain the error term of the uplink as follows:

$$E\left[||y^*[t] - \bar{y}[t]||^2\right] = E\left[||e[t]||^2\right] = \frac{1}{N^2}E\left[||\sum_{i=1}^{N} e^i[t]||^2\right] \leq e^{th}[t]. \tag{21}$$

Using the result in [46] we can derive the following inequality:

$$E\left[||\bar{y}[t+1]-y^*||^2\right] \le (1-\eta_t\mu)E\left[||\bar{y}[t]-y^*||^2\right] + e^{th}[t] + \eta_t^2\left[\sum_{i=1}^{N}\frac{\sigma_i^2}{N^2} + 6L\Gamma + 8(E-1)G^2\right]. \tag{22}$$

We denote $\Delta_t = E[||\bar{y}[t+1]-y^*||^2]$, according to equation(18) and (19), the error threshold of Fed-PEMC $e^{th}[t] \le \eta_t^2$, we have $\Delta_{t+1} \le (1-\eta_t\mu)\Delta_t + \eta_t^2 B$, where $B = \sum_{i=1}^{N}\frac{\sigma_i^2}{N^2} + 6L\Gamma + 8(E-1)G^2 + E[||e^{th}||^2]$. We denote the learning rate $\eta_t = \frac{\beta}{t+\gamma}$, where $\beta \ge \frac{1}{\mu}$, $\gamma \ge 0$, $\eta_1 \le min\{1/\mu, 1/4L\} = 1/4L$, $\eta_t \le 2\eta_{t+E}$. Then there is $\Delta_t \le \frac{v}{\gamma+t}$, where $v = max\{\frac{\beta^2 B}{\beta\mu-1}, (\gamma+1)\Delta_0\}$ is as follows:

$$\begin{aligned}\Delta_{t+1} &\le (1-\eta_t\mu)\Delta_t + \eta_t^2 B \\ &= \left(1-\frac{\beta\mu}{t+\gamma}\right)\frac{v}{t+\gamma} + \frac{\beta^2 B}{(t+\gamma)^2} \\ &\le \frac{v}{t+\gamma+1}.\end{aligned} \tag{23}$$

We substitute Δ_t into the above formula and let $t = T$, then Theorem 3 is proved.

VI. Performance Evaluation

In this section, we evaluate the performance of the proposed algorithm Fed-PEMC. The experiment was conducted on a Windows 10 computer with a 12th generation Core i7 processor with a single-core frequency of up to 4.70 GHz and an RTX3060 GPU.

A. Experiment Setup

We present a comparative analysis between the proposed algorithm, Fed-PEMC, and the baseline algorithms DP-Fed, Fed-SPA, and Fed-Avg, and we will experimentally compare the latest federated learning algorithms QMAMCC-FL [47] and FLIDS-BSAFSC [48] with the algorithm we propose. DP-Fed is an algorithm that augments the Fed-Avg approach by introducing Gaussian noise to enhance privacy protection. To assess the advantages of Fed-PEMC, we conduct a performance evaluation of these algorithms, focusing on three key aspects: training loss, accuracy, and the balance between privacy budget and model accuracy. It's worth noting that to better illustrate the effectiveness of our solution in an e-consumer scenario, we conducted experimental simulations of Fed-PEMC on a PUSH dataset associated with an e-consumer device. The PUSH dataset is collected from the PUSH wearable device worn on the forearm to measure athlete movement. It comprises 49194 sets of 449260 repeated exercise repetitions from 1441 male athletes and 307 female athletes.

The performance evaluation is performed using Python based on the dataset MNIST, which consists of 10 sets of 28×28 handwritten digital image, and the dataset CIFAR10, which is composed of 10 sets of 32×32 images. For the MNIST dataset, we adopted a CNN model that incorporates two 5×5 convolutional layers—initially, with 10 filters in the first convolutional layer and 20 filters in the second. Each convolutional layer is succeeded by a 2×2 pooling layer and Relu activation. Additionally, a dropout layer was implemented to prevent overfitting. The model also includes two fully connected layers: the first has an input of 320 dimensions and an output of 50 dimensions, while the second layer has an input of 50 dimensions and an output of 10 dimensions. We employed a CNN model for the CIFAR10 dataset, consisting of two convolutional layers sized at 5×5 (the initial convolutional layer containing 6 filters and the subsequent layer with 16 filters), each paired with a 2×2 pooling layer and Relu activation. Additionally, three fully-connected layers (the first fully-connected layer with $16 \times 5 \times 5$ dimensional inputs and 120 dimensional outputs, the second one with 120 dimensional inputs and 84 dimensional outputs, and the last one with 84 dimensional inputs and 10-dimensional output) were utilized. For the MNIST dataset, each client is provided with a training set consisting of 600 images and labels, and a test set containing 100 images and labels. For the CIFAR10 dataset, every client is allocated a training set comprising 500 images and labels, along with a test set of 100 images and labels. In each training round, a random selection of 10 out of the total 100 clients is made. These selected clients proceed to undertake local training for 5 rounds, followed by a global training phase spanning 50 rounds. Both the Fed-PEMC and the baseline DP-Fed implementations incorporate local differential privacy measures by introducing Gaussian noise. Specifically, they initialize this noise with values of 0.6 for ϵ and 0.0001 for δ.

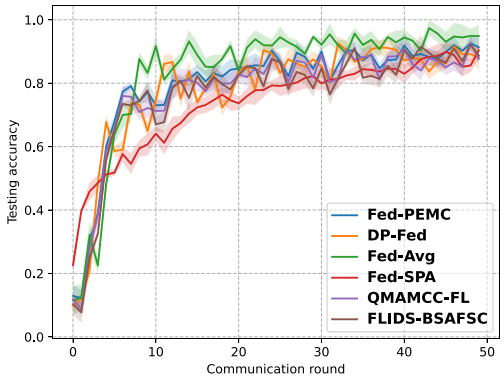

Fig. 4. The accuracy of Fed-PEMC, Fed-SPA, Fed-Avg, DP-Fed, QMAMCC-FL and FLIDS-BSAFSC on the MNIST dataset for 50 rounds.

B. Experiment Result

Fig. 4 and Fig. 5 illustrate the training accuracy and loss of the following algorithms: Fed-PEMC, Fed-SPA, QMAMCC-FL, FLIDS-BSAFSC, Fed-Avg, and DP-Fed. The overall testing accuracy and loss performance of our algorithm on the MNIST dataset are superior to the latest federated learning algorithms QMAMCC-FL and FLIDS-BSAFSC, these figures depict the performance of these algorithms over 50 rounds of training on the MNIST dataset. Regarding training accuracy, our proposed algorithm, Fed-PEMC, demonstrates higher

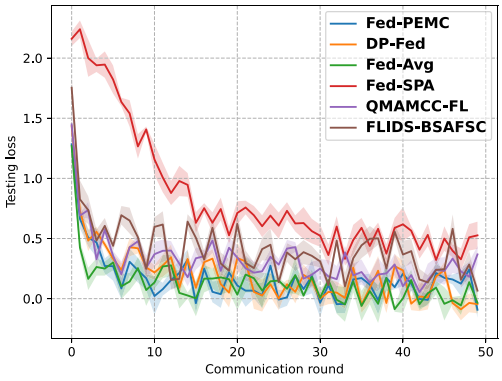

Fig. 5. The loss of Fed-PEMC, Fed-SPA, Fed-Avg, DP-Fed, QMAMCC-FL and FLIDS-BSAFSC on the MNIST dataset for 50 rounds.

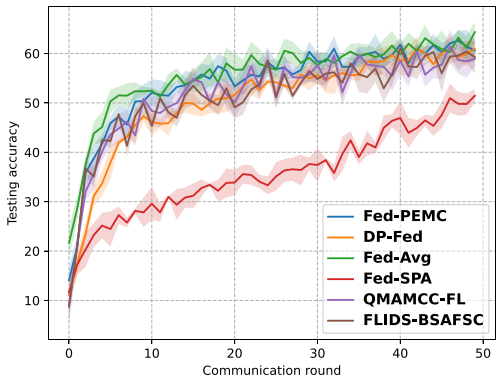

Fig. 6. The accuracy and of Fed-PEMC, Fed-SPA, Fed-Avg, DP-Fed, QMAMCC-FL and FLIDS-BSAFSC on the CIFAR10 dataset for 50 rounds.

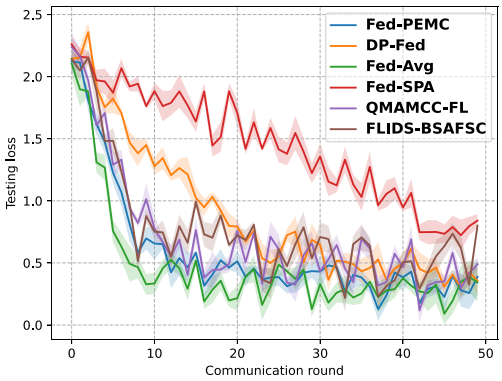

Fig. 7. The loss of Fed-PEMC, Fed-SPA, Fed-Avg, DP-Fed, QMAMCC-FL and FLIDS-BSAFSC on the CIFAR10 dataset for 50 rounds.

accuracy compared to the DP-Fed scheme. Furthermore, Fed-PEMC exhibits a more stable accuracy throughout the training process. However, the Fed-Avg scheme achieves higher accuracy than both Fed-PEMC and DP-Fed schemes since Fed-Avg does not incorporate privacy protection by adding noise. In terms of training losses, we observe that the losses of Fed-PEMC and Fed-Avg are comparable, with both algorithms achieving lower losses than DP-Fed throughout the entire training process.

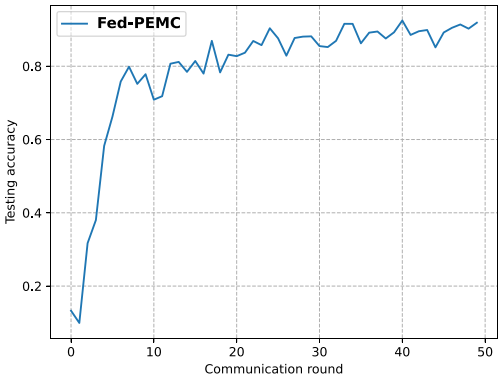

Fig. 8. The accuracy and of Fed-PEMC on the PUSH dataset for 50 rounds.

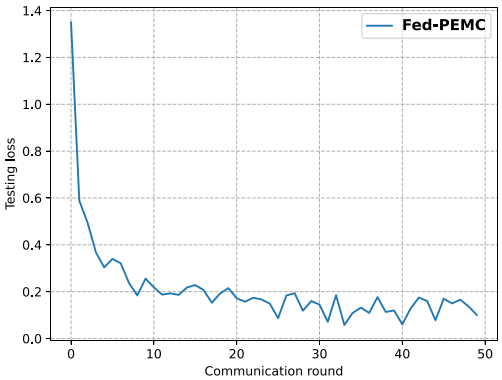

Fig. 9. The loss of Fed-PEMC on the PUSH dataset for 50 rounds.

Fig. 6 and Fig. 7 present the accuracy and loss of algorithms Fed-PEMC, Fed-SPA, QMAMCC-FL, FLIDS-BSAFSC, Fed-Avg, and DP-Fed after 50 rounds of training on the Cifar 10 dataset. The overall testing accuracy and loss performance of our algorithm on the MNIST dataset are superior to the latest federated learning algorithms QMAMCC-FL and FLIDS-BSAFSC. In the training accuracy, it can be seen that due to the absence of noise, the accuracy of Fed-Avg is higher than that of the algorithms Fed-PEMC and DP-Fed, while the accuracy of the proposed Fed-PEMFC is higher than that of DP-Fed. In the training loss, the loss of Fed-PEMC is slightly higher than that of Fed-Avg. However after 20 rounds of training, the loss values of both are very similar. We also can observe that the loss value of Fed-PEMC is far smaller than that of DP-Fed. Especially, at the 10th round of training, the difference is greatest. Fig. 8 and Fig. 9 demonstrates the testing accuracy and loss performance of the Fed-PEMC solution on the PUSH dataset, indicating its performance is comparable, and in some cases, even superior to that on the MNIST dataset. This highlights the effectiveness of our solution in the context of e-consumer devices.

Fig. 10 and Fig. 11 demonstrate the performance of Fed-PEMC, Fed-SPA and DP-Fed in terms of the tradeoff between privacy and accuracy on MNIST and Cifar 10 datasets, respectively. According to the tradeoff on the MNIST dataset, when the privacy budget ranges from 0.2 to 1, the training

TABLE III
SUMMARY OF RESULTS ON MNIST AND CIFAR10 DATASET

	Mnist(The number of model parameters)			Cifar(The number of model parameters)			Mnist	Cifar
Algorithm	10th round	30th round	50th round	10th round	30th round	50th round	Accuracy	Accuracy
Fed-SPA	8400	8400	8400	104197	104197	104197	87.28	51.39
DP-Fed	16800	16800	16800	208394	208394	208394	87.56	59.62
Fed-PEMC	7660	8366	8769	95861	117951	89609	90.83	61.64

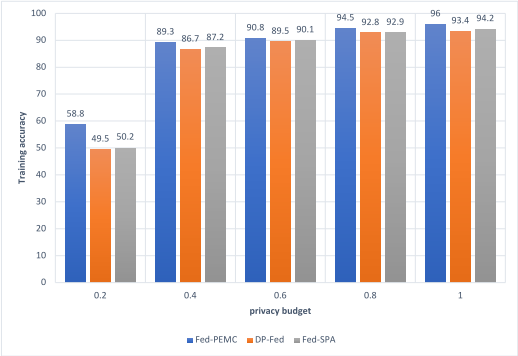

Fig. 10. The performance of privacy and accuracy tradeoff for Fed-PEMC, Fed-SPA and DP-Fed on MNIST(The horizontal coordinate is the privacy budget ϵ).

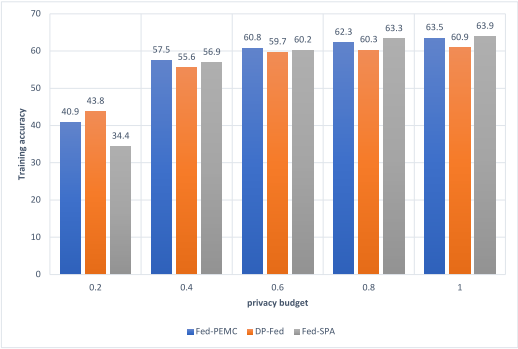

Fig. 11. The performance of privacy and accuracy tradeoff for Fed-PEMC, Fed-SPA and DP-Fed on CIFAR10(The horizontal coordinate is the privacy budget ϵ).

accuracy of the Fed-PEMC is higher than that of the DP-Fed. Especially, when the privacy budget values ranges from 0.2 to 0.3, the difference is greatest, with the accuracy of the Fed-PEMC being 8% higher than that of the DP-Fed. According to the tradeoff on the CIFAR10 dataset, DP-Fed has higher accuracy than Fed-PEMC with the privacy budget ranges from 0.2 to 0.35, while Fed-PEMC has higher accuracy than DP-Fed when the privacy budget varies from 0.35 to 1. In particular, when the privacy budget is set to 1, the difference between these two is greatest, and the accuracy of Fed-PEMC is about 3% higher than that of DP-Fed.

Table II shows the number of different round models and training accuracy of Fed-PEMC, DP-Fed and Fed-SPA in mnist and cifar datasets, our Fed-PEMC scheme has a compression rate similar to 50% sparsity rate of Fed-SPA, and our scheme outperforms Fed-SPA and DP-Fed in terms of accuracy on both MNIST and CIFAR datasets. Table III shows the training accuracy of Fed-PEMC, Fed-SPA, and DP-Fed for different privacy budgets on Mnist and Cifar, the

TABLE IV
RELATIONSHIP BETWEEN TESTING ACCURACY AND PRIVACY BUDGET ON MNIST AND CIFAR10 DATASET

Algorithm	DP-Fed(accuracy)		Fed-PEMC(accuracy)		Fed-SPA(accuracy)	
ϵ	mnist	cifar	mnist	cifar	mnist	cifar
0.2	49.5	43.8	58.8	40.9	50.2	34.4
0.3	77.3	50.6	84.9	49.8	75.6	51.1
0.4	86.7	55.6	89.3	57.5	87.2	56.9
0.5	88.2	58.9	89.9	59.9	89.2	58.9
0.6	89.5	59.7	90.8	60.8	90.1	60.2
0.7	90.2	59.9	92.7	61.7	91.7	62.3
0.8	92.8	60.3	94.5	62.3	92.9	63.3
0.9	93.2	61.2	95.2	62.9	93.2	63.1
1.0	93.4	60.9	96.0	63.5	94.2	63.9

TABLE V
TRAINING ACCURACY OF SCHEME FED-PEMC UNDER DIFFERENT PRIVACY BUDGET

Algorithm	Fed-PEMC(accuracy)		NoA-Fed-PEMC		NoB-Fed-PEMC	
ϵ	MNIST	CIFAR	MNIST	CIFAR	MNIST	CIFAR
0.2	58.8	40.9	57.2	39.6	57.4	40.2
0.4	89.3	57.5	86.1	57.8	87.6	55.7
0.6	90.8	60.8	88.6	59.5	90.2	60.3
0.8	94.5	62.3	91.9	60.6	91.2	61.2
1.0	96.0	63.5	94.3	61.2	93.9	62.3

training accuracy of DP-Fed, Fed-PEMC, and Fed-SPA on the MNIST and CIFAR datasets increases with an increase in privacy budget, and the training accuracy of Fed-PEMC is better than DP-Fed and Fed-SPA throughout the process of varying privacy budget.

C. Ablation experiment

In this section, we conduct ablation experiments on our Fed-PEMC scheme to demonstrate the effectiveness of the customized label sampling and server-side adaptive update, as well as to showcase how these two components further optimize our scheme.

The results of the ablation experiments are shown in Table IV, the entries NoA-Fed-PEMC and NoB-Fed-PEMC in Table IV respectively denote the Fed-PEMC schemes without customized label sampling and without server-side adaptive updates. Table IV shows the training accuracy of Fed-PEMC, NoA-Fed-PEMC, and NoB-Fed-PEMC on the MNIST and CIFAR datasets under different privacy budgets. It can be observed that Fed-PEMC has a higher training accuracy than the other two schemes throughout the varying privacy budget process. Scheme NoB-Fed-PEMC has a generally higher training accuracy than NoA-Fed-PEMC on both MNIST and CIFAR datasets across the entire experiment, indicating that custom class-level sampling has a greater impact on training optimization than server-side adaptive updates.

VII. Conclusion

The combination of federated learning and mobile edge computing provides personalized intelligent features and services for consumer electronics products. However, traditional federated learning is vulnerable to attacks and privacy breaches, and mobile edge computing faces challenges due to limited resources. To solve these issues, we introduce a federated learning algorithm called Fed-PEMC, which is based on privacy enhancement and model compression. This algorithm protects client privacy, ensures training accuracy, and improves communication efficiency. Specifically, in federated learning, we apply deep reinforcement learning algorithms to compress the model while maintaining accuracy, and accelerate local training through customized label sampling. By adding noise to the compressed model and dispatching it to the server, we solve the issue of privacy budget explosion and significantly improve communication efficiency between clients and servers. The theoretical analysis proves that Fed-PEMC satisfies (ϵ, δ)-differential privacy, and the communication overhead is only related to the model's size. Experimental results present that the proposed algorithm outperforms baseline algorithms in aspects of privacy protection, model accuracy, and communication efficiency. In future research, we will explore the enhancement of privacy protection and data availability in the context of federated differential privacy frameworks for consumer electronics, considering the scenario where client data is not independently and identically distributed (non-IID).

References

[1] X. Zhou et al., "Decentralized P2P federated learning for privacy-preserving and resilient mobile robotic systems," *IEEE Wireless Commun.*, vol. 30, no. 2, pp. 82–89, Apr. 2023.

[2] X. Zhou et al., "Digital twin enhanced federated reinforcement learning with lightweight knowledge distillation in mobile networks," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 10, pp. 3191–3211, Oct. 2023.

[3] A. N. Jahromi, H. Karimipour, and A. Dehghantanha, "An ensemble deep federated learning cyber-threat hunting model for Industrial Internet of Things," *Comput. Commun.*, vol. 198, pp. 108–116, Feb. 2023.

[4] J. Shu et al., "Clustered federated multitask learning on non-iid data with enhanced privacy," *IEEE Internet Things J.*, vol. 10, no. 4, pp. 3453–3467, Feb. 2023.

[5] X. You, X. Liu, N. Jiang, J. Cai, and Z. Ying, "Reschedule gradients: Temporal non-iid resilient federated learning," *IEEE Internet Things J.*, vol. 10, no. 1, pp. 747–762, Jan. 2023.

[6] F. Hu, W. Zhou, K. Liao, and H. Li, "Contribution- and participation-based federated learning on non-iid data," *IEEE Intelligent Systems*, vol. 37, no. 4, pp. 35–43, Jul./Aug. 2022.

[7] X. Zhou et al., "Hierarchical federated learning with social context clustering-based participant selection for Internet of Medical Things applications," *IEEE Trans. Comput. Social Syst.*, vol. 10, no. 4, pp. 1742–1751, Aug. 2023.

[8] X. Zhou, Y. Hu, J. Wu, W. Liang, J. Ma, and Q. Jin, "Distribution bias aware collaborative generative adversarial network for imbalanced deep learning in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 19, no. 1, pp. 570–580, Jan. 2023.

[9] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proc. IEEE Symp. Security Privacy (SP)*, 2017, pp. 3–18.

[10] C. Dwork, "Differential privacy, in: Automata, languages and programming," in *Proc. 33rd Int. Colloquium*, Venice, Italy, Jul. 2006, pp. 1–12.

[11] C. Wang, X. Wu, G. Liu, T. Deng, K. Peng, and S. Wan, "Safeguarding cross-silo federated learning with local differential privacy," *Digit. Commun. Netw.*, vol. 8, no. 4, pp. 446–454, 2022.

[12] Y. Miao, R. Xie, X. Li, X. Liu, Z. Ma, and R. H. Deng, "Compressed federated learning based on adaptive local differential privacy," in *Proc. 38th Annu. Comput. Security Appl. Conf.*, 2022, pp. 159–170.

[13] H. Zong, Q. Wang, X. Liu, Y. Li, and Y. Shao, "Communication reducing quantization for federated learning with local differential privacy mechanism," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, 2021, pp. 75–80.

[14] M. Kim, O. Günlü, and R. F. Schaefer, "Federated learning with local differential privacy: Trade-offs between privacy, utility, and communication," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, 2021, pp. 2650–2654.

[15] R. Hu, Y. Gong, and Y. Guo, "Federated learning with sparsification-amplified privacy and adaptive optimization," 2020, *arXiv:2008.01558*.

[16] L. Ruixuan, C. Yang, C. Hong, G. Ruoyang, and Y. Masatoshi, "FLAME: Differentially private federated learning in the shuffle model," 2020, *arXiv:2009.08063*.

[17] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artif. Intell. Stat.*, 2017, pp. 1273–1282.

[18] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," 2017, *arXiv:1712.07557*.

[19] Z. Chuanxin, S. Yi, and W. Degang, "Federated learning with Gaussian differential privacy," in *Proc. 2nd Int. Conf. Robot., Intell. Control Artif. Intell.*, 2020, pp. 296–301.

[20] L. Sun, J. Qian, and X. Chen, "LDP-FL: Practical private aggregation in federated learning with local differential privacy," 2020, *arXiv:2007.15789*.

[21] R. Hu, Y. Gong, and Y. Guo, "Federated learning with sparsified model perturbation: Improving accuracy under client-level differential privacy," 2022, *arXiv:2202.07178*.

[22] R. Hu, Y. Guo, H. Li, Q. Pei, and Y. Gong, "Personalized federated learning with differential privacy," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9530–9539, Oct. 2020.

[23] X. Gong et al., "Preserving privacy in federated learning with ensemble cross-domain knowledge distillation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 36, 2022, pp. 11891–11899.

[24] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Differentially private asynchronous federated learning for mobile edge computing in urban informatics," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 2134–2143, Mar. 2020.

[25] Y. Wang, Y. Tong, and D. Shi, "Federated latent Dirichlet allocation: A local differential privacy based framework," in *Proc. AAAI Conf. Artif. Intell.*, Vol. 34, 2020, pp. 6283–6290.

[26] L. Sun and L. Lyu, "Federated model distillation with noise-free differential privacy," 2020, *arXiv:2009.05537*.

[27] W. Luping, W. Wei, and L. Bo, "CMFL: Mitigating communication overhead for federated learning," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2019, pp. 954–964.

[28] X. Wu, Y. Zhang, M. Shi, P. Li, R. Li, and N. N. Xiong, "An adaptive federated learning scheme with differential privacy preserving," *Future Gener. Comput. Syst.*, vol. 127, pp. 362–372, Feb. 2022.

[29] S. A. Alvi, Y. Hong, and S. Durrani, "Utility fairness for the differentially private federated-learning-based wireless IoT networks," *IEEE Internet Things J.*, vol. 9, no. 19, pp. 19398–19413, Oct. 2022.

[30] X. Jiang, X. Zhou, and J. Grossklags, "SignDS-FL: Local differentially private federated learning with sign-based dimension selection," *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 5, pp. 1–22, 2022.

[31] B. Jiang, J. Li, H. Wang, and H. Song, "Privacy-preserving federated learning for industrial edge computing via hybrid differential privacy and adaptive compression," *IEEE Trans. Ind. Informat.*, vol. 19, no. 2, pp. 1136–1144, Feb. 2023.

[32] K. Yin et al., "DLDP-FL: Dynamic local differential privacy federated learning method based on mesh network edge devices," *J. Comput. Sci.*, vol. 63, Sep. 2022, Art. no. 101789.

[33] V. J. Marathe and P. Kanani, "Subject granular differential privacy in federated learning," 2022, *arXiv:2206.03617*.

[34] J. Lu, H. Liu, Z. Zhang, J. Wang, S. K. Goudos, and S. Wan, "Toward fairness-aware time-sensitive asynchronous federated learning for critical energy infrastructure," *IEEE Trans. Ind. Informat.*, vol. 18, no. 5, pp. 3462–3472, May 2022.

[35] S. Liu, J. Yu, X. Deng, and S. Wan, "FedCPF: An efficient-communication federated learning approach for vehicular edge computing in 6g communication networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 2, pp. 1616–1629, Feb. 2022.

[36] X. Zhou et al. "Information theoretic learning-enhanced dual-generative adversarial networks with causal representation for robust OOD generalization," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Nov. 17, 2023, doi: 10.1109/TNNLS.2023.3330864.
[37] X. Zhou, W. Liang, K. I.-K. Wang, and L. T. Yang, "Deep correlation mining based on hierarchical hybrid networks for heterogeneous big data recommendations," *IEEE Trans. Comput. Social Syst.*, vol. 8, no. 1, pp. 171–178, Feb. 2021, doi: 10.1109/TCSS.2020.2987846.
[38] Y. He, J. Lin, Z. Liu, H. Wang, L.-J. Li, and S. Han, "AMC: AutoML for model compression and acceleration on mobile devices," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 784–800.
[39] C. Fang, H. He, Q. Long, and W. J. Su, "Layer-peeled model: Toward understanding well-trained deep neural networks," 2021, *arXiv:2101.12699*.
[40] L. Zhang, L. Shen, L. Ding, D. Tao, and L.-Y. Duan, "Fine-tuning global model via data-free knowledge distillation for non-iid federated learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 10174–10183.
[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017, *arXiv:1412.6980*.
[42] I. Mironov, "Rényi differential privacy," in *Proc. IEEE 30th Comput. Security Found. Symp. (CSF)*, 2017, pp. 263–275.
[43] Y.-X. Wang, B. Balle, and S. P. Kasiviswanathan, "Subsampled Rényi differential privacy and analytical moments accountant," in *Proc. 22nd Int. Conf. Artif. Intell. Stat.*, 2019, pp. 1226–1235.
[44] L. Wang, B. Jayaraman, D. Evans, and Q. Gu, "Efficient privacy-preserving stochastic nonconvex optimization," 2019, *arXiv:1910.13659*.
[45] M. Beitollahi and N. Lu, "FLAC: Federated learning with autoencoder compression and convergence guarantee," in *Proc. IEEE Global Commun. Conf.*, 2022, pp. 4589–4594.
[46] X. Wei and C. Shen, "Federated learning over noisy channels: Convergence analysis and design examples," *IEEE Trans. Cogn. Commun. Netw.*, vol. 8, no. 2, pp. 1253–1268, Jun. 2022, doi: 10.1109/TCCN.2022.3140788.
[47] M. A. Elfaki et al., "Metaheuristics algorithm-based minimization of communication costs in federated learning," *IEEE Access*, vol. 11, pp. 81310–81317, 2023.
[48] T. Vaiyapuri et al., "Metaheuristics with federated learning enabled intrusion detection system in Internet of Things environment," *Expert Syst.*, vol. 40, no. 5, 2023, Art. no. e13138.