Federated Continual Learning based on Central Memory Rehearsal

Chen Zhang[1], Tielin Huang[1], Wenjie Mao[1], Hang Bai[1], and Bin Yu[1]
[1]School of Computer Science and Technology, Xidian University, Xi'an, Shaanxi, 710126, China

Federated learning, hailed as a transformative approach, fosters collaborative and secure acquisition of a unified model within the domain of Industrial Internet of Things (IIoT). This innovative paradigm enables multiple clients to collectively contribute to model training while preserving data privacy, leveraging the coordination of a central server. In the real world, most smart edge devices of the IIoT are always confronted with considerable data in the form of sequential data streams. However, current federated learning models suffer a sharp drop in performance when dealing with sequential data, which is called catastrophic forgetting. Consequently, A crucial obstacle encountered in practical implementations of federated learning revolves around the need to address the issue of catastrophic forgetting, thus enabling it to acquire and retain knowledge across multiple tasks, akin to human capabilities. In this paper, we propose a novel framework, called Federated Central Memory Rehearsal (FedCMR), which is inspired by the rehearsal method of continual learning. Specifically, the Generator model, trained by the central server, is tasked with the responsibility of creating the pseudo data (Central Memory) associated with previous tasks. The pseudo data refers to synthetic data generated by the model, which serves to mimic the data from older tasks. This synthetic data is crucial for rehearsal-based learning, allowing local models to retain knowledge from earlier tasks even when only the current task's data is available for training. Upon the arrival of a new task, the local client mixes a small amount of pseudo data with the local dataset for training, with the aim of maintaining the knowledge of old tasks (Rehearsal). Upon the completion of training for the current task by each local client, they proceed to upload their respective local models and the sampled data from the current task, fortified with differential privacy noise, to the central server. Subsequently, the server consolidates the collected local models, crafting a novel global model. Additionally, it generates a limited amount of synthetic data representing past tasks, which is then disseminated to each client for secure and collaborative training purposes. Experimental results demonstrate that FedCMR overcomes catastrophic forgetting while realizing privacy preserving and reducing communication costs.

***Index Terms*—Federated Learning, Privacy Preserving, Sensitive and Private Information.**

I. Introduction

The concept of Federated Learning (FL) emerges as an innovative machine learning paradigm that combines privacy preservation and collaborative distributed learning, effectively addressing the challenges of data isolation. In 2016, Google introduced Federated Learning as an approach initially applied to tackle the task of next-word prediction on Android mobile keyboards within a distributed environment[1]. This revolutionary framework characterizes a distinctive scenario of distributed machine learning, wherein multiple local clients actively participate in the training process of a shared model, with the guidance of a central server. The local client can be a device with limited local resources such as a mobile phone, a personal computer, or an Internet of Things (IoT) device [2]. However, local data available to individual clients is often insufficient for comprehensive training of deep learning models. To address this, FL aggregates locally trained models into a new global model on the server, which requires effective and efficient aggregation algorithms.

Several aggregation algorithms have been proposed to enable the central server to combine local models effectively, with the Federated Averaging (FedAVG) algorithm[3] being one of the most prominent. The FedAVG algorithm revolves around the following core principle: Initially, the server initializes the global model weight $\mathbf{w}_t^{(g)}$ and transmits it to a randomly selected set of clients, denoted as $\mathbf{S}_t$. Subsequently, these selected clients independently perform local updates on their initial local weights $\mathbf{w}_t$ in parallel. They then transmit the updated local weight $\mathbf{w}_{t+1}$ back to the server. Lastly, the server aggregates the weights uploaded by the selected clients using weighted averaging, resulting in the updated global weight $\mathbf{w}_{t+1}^{(g)}$. This updated global weight serves as the initial weight for the subsequent round of training, and the process continues iteratively until convergence is achieved.

Federated learning has found widespread use in various domains such as smart cities[4], healthcare[5], and open banking[6]. It allows for secure and privacy-preserving collaborative model training across multiple remote institutions while complying with privacy protection regulations like GDPR[7]. However, when dealing with sequential arrival data in practical applications of federated learning, the challenge of catastrophic forgetting always persists. For example, there are multiple hospitals (clients) distributed in different geographic regions currently participating in the federated training of the Cerebral Infarction diagnosis model as shown in Figure 1. When the epidemic COVID-19 [8] needs to be diagnosed in these hospitals, each hospital has to feed the local image data (chest X-ray, CT, MRI) of COVID-19 to the federated model. Unfortunately, traditional federated learning is going to drift to the diagnosis of COVID-19, which means that the diagnostic knowledge of Cerebral Infarction in each hospital is forgotten. Therefore, federated continual learning is needed to solve the above problems in similar scenarios.

Recently, continual learning (CL) has garnered significant attention in the realm of learning from infinite data streams. This approach deviates from the traditional assumption of having complete data readily available [9].Continual learning can continually accumulate the knowledge of sequential tasks, which can be applied to the future task to overcome

Manuscript received September 17, 2024; revised October 28, 2024. Corresponding author: Bin Yu (email: yubin@mail.xidian.edu.cn).

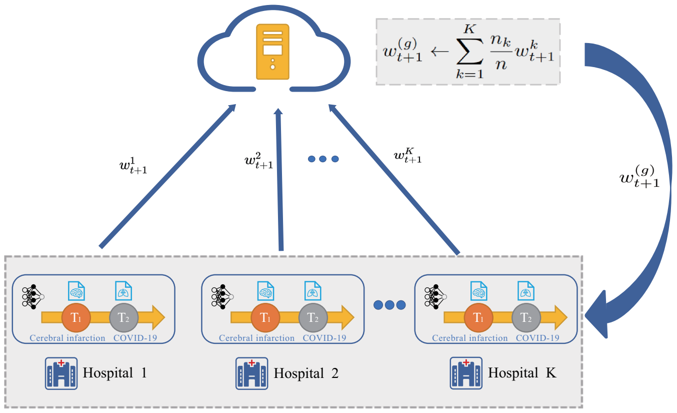


Fig. 1. Multiple hospitals participate in the federated training of the sequential disease diagnosis model. As a local client, each hospital uploads the local model weights w_{t+1} to the central server for global aggregation. Additionally, each hospital trains multiple tasks in a sequential manner. However, the knowledge of the newly arrived task (COVID-19) overwrites the knowledge of the old task (Cerebral infarction), which is called catastrophic forgetting.

catastrophic forgetting [10]. There are three main categories of methods used in continual learning: regularization-based methods, dynamic structure-based methods, and episodic memory-based methods.

However, the issue of catastrophic forgetting in the federated learning setting has rarely been discussed. The naive solution to this problem is to apply the above continual learning methods directly on the local clients to solve local catastrophic forgetting. Nevertheless, this naive method is bound to increase the computational burden on the local devices and reduce the training accuracy. Existing research [11] [12] on federated continual learning ignores the recovery of old task knowledge, which inevitably leads to the degradation of model performance on old tasks. Due to the uniqueness of its own architecture and training methods, federated learning has a large number of privacy leakage risks, such as Membership Inference Attack [13]. Consequently, we analyze several challenges for achieving anti-forgetting federated learning: 1) Restoring old task knowledge efficiently: When the local model of clients in federated learning learns a new task in a sequential dataset, its performance on old tasks often experiences a significant decline, a phenomenon commonly referred to as catastrophic forgetting. In order to restore the capability of a federated model to learn previous tasks, clients often need to retrain the model, which is the biggest challenge in achieving federated continual learning. 2) Lossless knowledge transfer is a crucial aspect that deserves attention in research. While some existing studies emphasize the isolation of model parameters, it is essential to recognize that placing excessive computation burdens on clients can have detrimental effects on model performance. Therefore, it is indispensable to ensure that the knowledge of old tasks of the model is preserved without affecting the training efficiency of the local clients. 3) Privacy concern: Federated learning strives to bridge data silos while ensuring privacy preservation. Safeguarding the data privacy of participants is a fundamental prerequisite for enticing clients to engage in the training process. Therefore, the federated continual learning approach that can be implemented in practice must possess the ability to guarantee the privacy of individual client data.

To provide promising solutions to learn networks without catastrophic forgetting through leveraging federated communication mechanisms, we propose a novel framework named Federated Central Memory Rehearsal (FedCMR) to address the above challenges, which utilizes Generative Adversarial Networks (GANs) [14] on the server side for episodic memory generation. Specifically, in this framework, the central server trains a Generator model to create pseudo data (Central Memory) for old tasks. When a new task arrives, local clients use some of this pseudo data in combination with their own data for training. This helps them retain knowledge from earlier tasks (Rehearsal). Subsequent to the training of the current task on local clients, each client uploads their local model and the sampled data from the current task, with the inclusion of differential privacy noise, to the central server. The server then aggregates the collected local models to form a fresh global model and generates a limited amount of synthetic data for previous tasks using the Generator. These aggregated global model and pseudo data for old tasks are then distributed synchronously to each client for secure and interactive training. Finally, the effectiveness of the proposed method is evaluated using sequential datasets. The main contributions of this paper are summarized as follows:

1) We presents a new FedCMR framework to effectively address the challenges faced by anti-forgetting federated learning: restoring old task knowledge, knowledge transferring and privacy concerns.
2) In the FedCMR, the generative model Memory Generator trained on the server side is for the purpose of generating episodic memory pseudo data, which restores the old task knowledge and reduces the computational burden of resource-constrained clients.
3) We assess the effectiveness of our approach using three distinct datasets: perm-mnist, five-mnist, and svhn-mnist. Extensive experiments reveal that compared with baselines, FedCMR overcomes catastrophic forgetting efficiently while providing privacy preserving.

The subsequent sections of this paper are structured as follows. Section II provides a brief review of the existing literature on federated learning and continual learning. Section III overviews the proposed FedCMR framework and provides the basic process in the framework. Section IV details the implementation of the FedCMR algorithm. Next, we conduct several experiments on sequential datasets to evaluate our framework in Section V. Finally, Section VI concludes the main efforts of this paper.

II. Related Works

In this section, we present the basic concepts of federated learning and continuous learning and discuss some of the related works.

A. Federated Learning

Recently, the researches on federated learning have greatly proliferated [15]. Federated learning, as introduced earlier, facilitates collaborative training of a shared model among distributed users while ensuring the privacy of local user data is preserved. Federated Averaging (FedAVG) is proposed in[3] which uses a synchronized global aggregation. Building upon the FedAVG framework, Li et al. [16] introduced a framework called FedProx. FedProx incorporates a correction term known as the proximal term into the objective function of local clients. Additionally, it dynamically adjusts the number of local iteration epochs, enhancing the overall performance of the federated learning process. Li et al. [17] discussed the existing challenges and solutions of federated learning, and summarizes possible directions for future work. In [18], a recommendation system for charging stations in the Electric Vehicles (EVs) industry is introduced, which utilizes the Vertical Federated Learning technique. The system is designed to operate on secure cloudlets. Cheng et al. [19] developed a privacy-preserving approach for detecting icing on wind turbine blades using heterogeneous federated learning, with a focus on data sharing among multiple wind farms. In [20], a new federated learning framework called PCFed is introduced, which ensures strong privacy guarantees and improves communication efficiency while achieving higher model accuracy.

With the rapid growth of big data, federated learning faces a new challenge in handling multiple tasks. However, it is often the case that the training data for all tasks is not readily accessible. Consequently, federated learning models must be trained on sequentially arriving data streams, posing a distinct challenge. Therefore, neural networks are expected to be able to continuously learn new knowledge with newly arrived data while not forgetting the previously learned knowledge. However, traditional deep neural networks suffer from design flaws that lead to the iterative overwriting of parameters in each layer during the training process [21]. As a consequence, the existing knowledge is overshadowed by the newly acquired task knowledge, resulting in what is commonly referred to as catastrophic forgetting. A naive solution is to retrain the model on the local old task data to regain the lost knowledge. But this method not only greatly reduces the training efficiency, but also wastes the limited resources of the client in the federated environment. Therefore, our objective is to develop an anti-forgetting framework tailored to the distinctive features of federated learning, with the aim of overcoming the aforementioned challenges.

B. Continual Learning

In the age of big data and deep learning, we face a situation where our model trains ever more tasks on their own datasets in a sequential manner, which differs from the traditional assumption that complete data are accessible [10]. Therefore, continual learning (CL) is designed to learn from an infinite data stream, which aims to gradually expand the knowledge base without forgetting previous knowledge [22]. Continual learning, also referred to as lifelong learning or incremental learning, involves the introduction of new tasks. However, a notable challenge in this process is the substantial performance decline of existing neural network models on previously learned tasks, which is commonly known as catastrophic forgetting. The occurrence of catastrophic forgetting in neural networks can be attributed to the stability-plasticity dilemma, a challenge that neural networks face. In this dilemma, stability refers to the ability of neurons to retain prior knowledge, while plasticity refers to the capacity to incorporate new knowledge [23]. A host of recent efforts have addressed the catastrophic forgetting of continual learning with task sequences. Li et al. [24] proposed a novel approach for continuous learning in classification. Their method focuses on enhancing the real-time classification performance by continuously incorporating newly labeled data into the learning process. Ren et al. [25] proposed a method that combines Extreme Learning Machine (ELM) with global Bayesian and continuous learning techniques. This approach enables the direct acquisition of new knowledge from additional data, allowing for the updating of model parameters without the need to retrain the entire model from scratch. Hanul et al. [26] proposed a novel Deep Generative Replay (DGR) with a dual-model architecture, which alleviates forgetting with generated data from previous tasks. Elastic Weight Consolidation (EWC) [27] is for the purpose of restricting changes of crucial model parameters through Fisher Information Matrix in the training process of later tasks. Li et al. [28] proposed the Learning without Forgetting (LwF) method that uses previous model outputs as soft labels to transfer knowledge and alleviate forgetting.

The continual learning in a distributed environment is crucial in the context in which distributed devices are increasingly interconnected. The paradigm of federated learning holds great promise in the field of distributed machine learning. However, there are merely a few discussions about continuous learning in a federated environment. Casado et al. [29] presented a novel framework called Light Federated and Continual Consensus (LFedCon2). This framework offers a federated and continual learning solution that specifically caters to devices with limited resources, enabling them to engage in continuous and localized learning processes. Huang et al. [30] introduced an approach called Federated Cross-Correlation and Continual Learning (FCCL), which addresses the issue of catastrophic forgetting. FCCL incorporates knowledge distillation during local updates, enabling the exchange of inter and intra-domain information while maintaining privacy. The primary objective of this approach is to mitigate the adverse effects of catastrophic forgetting throughout the learning process. In order to prevent the interference from other clients and reduce the communication cost, Yoon et al. [11] proposed Federated Weighted Inter-client Transfer (FedWeIT), which decomposes the model parameters of each client into sparse task-specific parameters and global federated parameters. However, none of the above methods considers leveraging the federated mechanism to transfer knowledge. Therefore, inspired by the rehearsal method, we propose federated central memory rehearsal, where the episodic memory is centrally generated in the server to perform pseudo-rehearsal.

III. Methodology

This section provides a detailed overview of our FedCMR framework. Section III-A describes the architecture of our

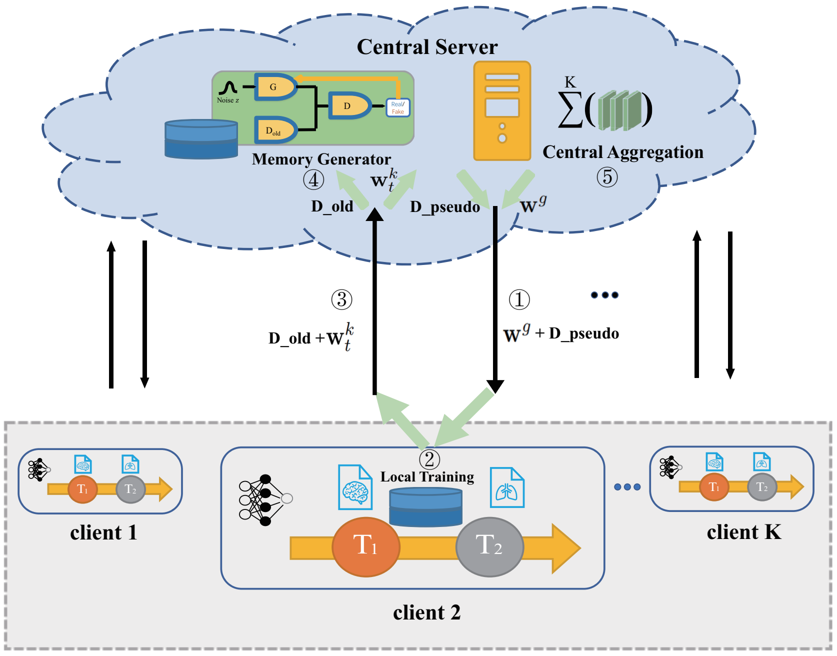


Fig. 2. The schematic architecture of the proposed Federated Central Memory Rehearsal (FedCMR). The Memory Generator model, which generates the pseudo data of old tasks, is depicted in the server cloud. The lower part of the diagram is the local clients participating in the federated learning, and each client learns on a sequential data stream.

framework. Section III-B outlines our approach, providing insights from both the server and client sides.

A. Overview

Fig. 2 illustrates the overall architecture of our proposed Federated Central Memory Rehearsal (FedCMR), which reflects that the main characteristics of the FedCMR are sequential, distributed and collaborative. The novelty of this framework resides in our pioneering proposition of maintaining a Memory Generator model on the server side. This model serves to address the challenge of catastrophic forgetting within the context of federated learning. Our inspiration for this approach stems from the generative replay method employed in continual learning. In this framework, we consider that there are K local clients involved in the federated training and they need to process the sequential arrival data streams collected in real time. The server situated in the cloud consolidates the locally uploaded models from the clients, resulting in the formation of a novel global model. Simultaneously, it maintains a Memory Generator model that generates old task pseudo data, which is fed with old task data sampled from local clients. When each local client trains the current task on the current dataset, the client samples a small part of data and uploads it to the server synchronously. At the time that a new task arrives, the old task pseudo data generated by the central server are mixed with the new task data according to the importance ratio for training. Accordingly, the knowledge of the old tasks is preserved when new tasks arrive, which means that the catastrophic forgetting challenge of federated learning is alleviated. We illustrate the proposed proposal in this paper formally in the algorithm diagram below. Table I lists several essential notations in this framework.

TABLE I
LIST OF NOTATIONS

Notations	Description
$\mathbf{D}_{old}$	the sampled old task data from local clients.
$\mathbf{D}_{pseudo}$	the generated pseudo data of old tasks.
$\mathbf{D}_i^k$	the local dataset for task i of client k.
$\mathbf{w}_t^k$	the local model weight maintained by the client k at global training epoch t
$\mathbf{w}^g$	the global model weight maintained by the central server

B. Our Framework

The proposed FedCMR framework comprises two primary components: the central server and the local client update mechanism. The central server plays a dual role by aggregating the locally uploaded weights from clients and housing a Memory Generator module responsible for generating pseudo data corresponding to previous tasks. On the other hand, the local clients conduct updates on their individual datasets, which are augmented with pseudo-data obtained from the central server, which helps mitigate the problem of catastrophic forgetting.

1) Central Server

As one of the significant components in federated learning, the central server is deployed in the cloud for interactive training with distributed local clients by communicating model weight parameters. Prior to the commencement of federated training, the global weight $\mathbf{w}^g$ is initialized by the central server and distributed to m local clients as their respective local initial models. During each global communication epoch, a subset of local clients is selected, and upon completing the model training on their respective datasets, they promptly transmit their updated model weights $\mathbf{w}_t^k$ to the central server. There are many federated aggregation algorithms, among which the FedAVG algorithm, which is often used as a baseline, is the most classic and effective. In this work, we choose FedAVG as the baseline method for aggregating the local models provided by the clients into a new global model. Consequently, this updated global model weight $\mathbf{w}^g$ can be used as the initial model weight for the next epoch training to continue integrating the efforts of individual clients. In the proposed FedCMR algorithm, to be different, we use the differential privacy mechanism to protect the privacy of communication data in interactive training. Common differential privacy noise mechanisms that perturb the original dataset or intermediate results are: Gaussian mechanism, Exponential mechanism, and Laplace mechanism. To ensure privacy preservation while minimizing the impact of differential privacy noise on the original data, we employ the Gaussian mechanism for gradient computation instead of the Laplacian mechanism. This choice is motivated by the faster decay of the tail in the Gaussian distribution compared to the Laplacian distribution. In the context of approximating a deterministic function f using a differential privacy mechanism, a common approach is to add noise calibrated to f based on its sensitivity S, which is defined as the maximum absolute difference $|f(d') - f(d)|$ between adjacent datasets d' and d. Overall, the additive noise mechanism of Gaussian is described as follows:

$$\mathcal{M}(d) \triangleq f(d) + \mathcal{N}\left(0, S^2 \cdot \sigma^2\right) \tag{1}$$

where $\mathcal{N}\left(0, S^2 \cdot \sigma^2\right)$ represents the Gaussian distribution with a mean of 0 and a standard deviation of S. It is worth noting that the method described above can provide a relaxed guarantee of (ε, δ)-differential privacy.

The central server in the cloud not only aggregates the local model weights, but also maintains a Memory Generator model which is fed with the sampled old task data synchronously uploaded from the client and generates the pseudo data of the old task. The Memory Generator model consists of a generator **G** and a discriminator **D**. The generator **G** is responsible for generating pseudo data of the old task, while the discriminator **D** helps to distinguish between real old task data and the generated pseudo data [14]. The **G** and **D** play against each other to minimize the difference between the pseudo data generated by **G** and the old task data, which is corresponding to a zero-sum game. The generated pseudo data are distributed to each local client to alleviate their forgetting of old task knowledge as new tasks arrive. Using the synthetic samples generated by the Generator model to replace the sensitive data can ensure that the privacy of the original sensitive data is not compromised during knowledge transfer.

2) Local Client

As the fundamental body of federated learning, the local clients can be composed of an ocean of communicable devices (Mobile Phones, IoT Devices, Automobiles, etc.) distributed in different geographical regions. However, with the development of communication networks (5G, WIFI, 6G, etc.), the data collected in real time by these devices are usually not permanently stored locally due to the memory or other resource limitations. Therefore, the data of local clients are always in the form of sequential data streams. In general, traditional federated model training is found that the knowledge of previous tasks is overwritten by the knowledge of the newly arrived task in the data streams, which is called catastrophic forgetting.

Algorithm 1 Federated Central Memory Rehearsal (FedCMR). There are K clients indexed by k. t is the number of sequential tasks, and E_l is the local epochs. B is the local mini-batch size, C is the participation ratio, and η is the learning rate.

```
Input: Global initial parameter $w_0$, local dataset of client $k$ $\mathcal{D}_k^{(1:t)}$, old task data $D_{old} \leftarrow \{\}$
Output: Anti-forgetting global model weights $w^g$
1: initialize $w^g$ and $D_{pseudo}$                      ▷ Server executes
2: for each task $t = 1, 2...t$ do
3:     $m \leftarrow \max(C \cdot K, 1)$
4:     $S_t \leftarrow$ (random set of $m$ clients)
5:     distribute $D_{pseudo}$ and $w^g$ to each client in $S_t$
6:     for each client $k \in S_t$ in parallel do
7:         $w_{t+1}^k, D_{old} \leftarrow$ ClientUpdate($w_t^k, D_{pseudo}$)
8:     end for
9:     $w^g \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$
10:    $D_{pseudo} \leftarrow$ GAN($D_{old}$)
11: end for
12: return $w^g$
13:
14: ClientUpdate($w_t^k, D_{pseudo}$):                   ▷ Client $k$ executes
15: upload $D_{old} \leftarrow sampled(D_k)$ to Server
16: $\mathcal{B} \leftarrow$ (split ($\mathcal{D}_k + D_{pseudo}$) into batches of size $B$ )
17: for local round $i = 1, 2...E_l$ do
18:     for batch $b \in \mathcal{B}$ do
19:         $w_{t+1}^k \leftarrow w_t^k - \eta \nabla \ell(w_t; b)$
20:     end for
21: end for
22: return $w_{t+1}^k$ to server
```

In the proposed FedCMR algorithm, each stochastically selected client initializes its local model weights $\mathbf{w}_t^k$ using global weights $\mathbf{w}^g$ downloaded from the server. When a new task arrives, the clients incorporate the generated pseudo data from previous tasks with the local data for training in order to mitigate the issue of catastrophic forgetting and preserve the knowledge of the old tasks. The mixing ratio is based on the importance metric r. While training the current task on the local dataset, each local participant uses the idle network bandwidth at this time to upload a slice of sampled current task data to the server. This method of asynchronous transfer neatly offsets the communication cost incurred by uploading the sampled old task data. Since uploading a small amount of old task data directly may cause privacy leakage, we must take measures to preserve the private data of these clients. Specifically, we apply a differential privacy mechanism to add Gaussian noise to these sampled old task data as above, which can provide a relaxed (ε, δ)-differential privacy guarantee. In FedCMR, we add noise to the original data to hide the privacy identifying information. Therefore the client's private data can be guaranteed not to be leaked. Upon completing the local training, the clients transmit their respective local model weights $\mathbf{w}_t^k$ to the server for aggregation.

Algorithm 1 details the overall framework of FedCMR. Line 2 initializes the global model w^g and the pseudo data D_{pseudo}. Lines 4-6 distribute the w^g and D_{pseudo} to the clients participating in this round of training. In line 8, the clients perform individual updates on their local models and send both the updated local model and the sampled current task data to the server. In line 10, the server consolidates the models uploaded by the clients to construct a new global model. In line 11, the server utilizes the Generator model to update the pseudo data sample D_{pseudo} with the old task data D_{old}. Lines 15-23 are the steps for the local client update. In line 16, the client samples the current task data and uploads it to the server. The pseudo data D_{pseudo} distributed by the server is mixed with the current task data and divided into training mini-batches in line 17. In lines 18-22, each client uses stochastic gradient descent for local training. Finally, in line 23, the updated model is transmitted to the server.

IV. Implementation

In this paper, inspired by the replay method in continual learning, we envision performing the replay method on edge client nodes for the purpose of mitigating the catastrophic forgetting challenge of federated learning. However, the participants in federated learning are often resource-constrained distributed devices (IoT, Mobile Phones, etc.) that may not have sufficient resources to store all of the old tasks data. Therefore, we design a novel approach FedCMR to address this issue. This section introduces the specific implementation of FedCMR framework.

A. Problem Statement

We first illustrate several basic terminologies in this paper.

1) **Statement 1:** In our federated continual learning framework FedCMR, we consider a sequence of local training tasks $\mathbf{T} = (\mathbf{T_1}, \mathbf{T_2}, ..., \mathbf{T_n})$ for all participants. Correspondingly, the clients are fed with a sequential data stream $\mathbf{D} = (\mathbf{D_1}, \mathbf{D_2}, ..., \mathbf{D_n})$, $\mathbf{D}_i = (\mathbf{X}_i, \mathbf{Y}_i)$, which consists of samples $\mathbf{X}_i$ with corresponding labels $\mathbf{Y}_i$. More importantly, the data of previous tasks $(\mathbf{D_0}...\mathbf{D_{i-1}})$ are not accessible when the participant trains current task $\mathbf{T}_i$ on dataset $\mathbf{D}_i$.
2) **Statement 2:** The Memory Generator model maintained in the central server is a tuple $< \mathbf{G}, \mathbf{D} >$, where $\mathbf{G}$ generates real-like samples, and the discriminator $\mathbf{D}$ evaluates the generated samples to identify the differences between them and the real samples.

B. Central Memory Generation of Central Server

In the proposed FedCMR, we focus on task-incremental sequential training cases in the continual learning. The central server utilizes the Federated Averaging algorithm (FedAVG) to perform weighted averaging of each client's model parameter $\mathbf{w}_t^k$. This process produces a new global model parameter $\mathbf{w}^g$, which serves as the initial model weight for each client in the next global training epoch. The weighted aggregation of the federated averaging is as follows:

$$w_{t+1}^{(g)} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k \tag{2}$$

where $w_{t+1}^{(g)}$ is the updated global weight, and w_{t+1}^k is the local weight trained on client k.

The server needs to train a Memory Generator model simultaneously while aggregating local model weights uploaded from randomly selected clients, which can generate pseudo data samples as close as possible to the real old task data. The Generator model integrates the data from the current task with the data from the previous task according to the importance ratio r, thereby it can accumulatively reconstruct a new input sample space and return the pseudo data with the same distribution to the clients.

The FedCMR framework proposed in this paper leverages the Memory Generator maintained in the central server to achieve central memory rehearsal. The Memory Generator is a generative model (GANs) that can generate observable samples from the real data distribution uploaded by local clients. The GANs consists of a generator $\mathbf{G}$ and a discriminator $\mathbf{D}$. The generator $\mathbf{G}$ is trained to generate pseudo data that closely approximates the distribution of real data. On the other hand, the discriminator $\mathbf{D}$ is tasked with differentiating between the distribution of real data and the pseudo data generated by $\mathbf{G}$. The generator network (Generator) and the discriminant network (Discriminator) are interactively trained via the zero-sum game. Through this training process, the Generator learns to approximate the data distribution of real old task data from clients and generate pseudo-samples (episodic memory) that exhibit similar characteristics. The objective functions of the two network tuple $< \mathbf{G}, \mathbf{D} >$ in the Memory Generator model can be defined as

$$\begin{aligned}\max_D V(D,G) =& \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] \\ &+ \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]\end{aligned} \tag{3}$$

$$\min_G V(D,G) = \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))] \tag{4}$$

where 3 is the objective function of discriminator $\mathbf{D}$, and 4 is that of Generator $\mathbf{G}$. In this paper, we use an optimized GAN model, WGAN-GP [31], to enhance the performance of the Generator for generating pseudo data.

C. Local Training of Clients

Each client in the federated environment has a local sequential task stream $\mathbf{T} = (\mathbf{T_1}, \mathbf{T_2}, ..., \mathbf{T_n})$. The client needs to preserve the knowledge acquired from the previous $(t-1)$ tasks while training the current task t. Correspondingly, in FedCMR, the participants need to upload a small amount of sampled data of the current task with DP noise for central memory rehearsal. For the purpose of optimizing communication efficiency, we adopt a technique of asynchronous communication to transmit data samples. Each local client can transmit data during the training process of the current task. After the current task training is completed, the clients upload the local model weights for aggregation. Therefore, this approach guarantees the efficient usage of the limited resources (CPU, bandwidth) of distributed edge client.

After downloading the global weight $\mathbf{w}^g$ and the pseudo data $\mathbf{D}_{pseudo}$ generated from the central server, the local client k initializes the local model weight $\mathbf{w}^k = \mathbf{w}^g$. Then the client k samples the current task i dataset $\mathbf{D}_i^k$ and uploads it to the central server, meanwhile, the pseudo data $\mathbf{D}_{pseudo}$ and the current task data $\mathbf{D}_i^k$ are mixed according to the importance ratio r for joint training. The overall loss function for local training is as 5:

$$\begin{aligned} L'\left(w_i^k\right) &= r\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim D_i^k}\left[L\left(S\left(\boldsymbol{x};w_i^k\right),\boldsymbol{y}\right)\right] \\ &+ (1-r)\mathbb{E}_{(\boldsymbol{x'},\boldsymbol{y'})\sim D_{pseudo}}\left[L\left(S\left(x';w_i^k\right),\boldsymbol{y'}\right)\right] \end{aligned} \tag{5}$$

where (x, y) denotes the samples and labels of the current task, (x', y') denotes the generated pseudo-samples and labels of the old tasks, $\mathbf{w}^k$ denotes the model weight parameters of client k on the task sequence, and L is the loss function.

V. Experiment

In this section, we assess the performance of our FedCMR on sequential arrival tasks on three different datasets: five-mnist, permu-mnist and svhn-mnist datasets. The central memory rehearsal based FedCMR outperforms other federated continuous learning approaches in alleviating forgetting. In Section V-A, we provide detailed information regarding the design of the dataset and the experimental setup. In Sections V-B - V-H we not only validate the effectiveness of the proposed method in anti-forgetting on sequential datasets, but also demonstrate its advantages by comparing the FedCMR framework with other baseline models and state-of-art models.

A. Experimental Settings

To evaluate the effectiveness of our framework and cover a broad range of learning problems, we separately train classical neural networks on sequential datasets with various federated learning algorithms. The configuration is described in detail below.

1) Learning environment

The basic configuration of the federated learning simulation system environment can be simply described by the following:

TABLE II
Environment Configuration

CPU:	Intel(R) Xeon(R) E5-2650 v4
GPU:	NVIDIA TITAN Xp.
Programming language:	Python 3.7.0
Machine learning library:	Pytorch 1.7.1

2) Dataset and models

The sequential arriving data used in this paper consists of the MNIST dataset [32] and the SVHN dataset [33]. The MNIST dataset, a classic handwritten digits dataset in machine learning, contains 60,000 image samples for training and 10,000 image samples for testing. SVHN is a real-world dataset of house number images in Google Street View imagery used to develop machine learning and object recognition algorithms. The SVHN dataset, which stands for Street View House Numbers, consists of 73,257 training image samples and 26,032 test image samples.

In this paper, we design three different sequential arrival datasets five-mnist, permu-mnist, svnh-mnist. The five-mnist dataset is formed by dividing the MNIST dataset into 5 groups according to the labels $\{0, 1, ..., 9\}$. The permu-mnist dataset uses 5 different random permutations to rearrange the pixels of the MNIST data, resulting in 5 different but related sets of data. The svnh-mnist dataset is sampled from two datasets MNIST and SVHN. We divide the datasets equally according to the number of federated clients. The number of sequential arriving tasks is the number of groups in the dataset. For the five-mnist dataset, our five groups of data are sequentially assigned to each client at five times, and the client will discard the old task dataset once the new task arrives. We demonstrate the essential information of these three sequential datasets in Table III.

TABLE III
Dataset Description

Dataset	Number of Tasks	Division Basis	Original Dataset
five-mnist	5	5 groups of labels	MNIST
permu-mnist	5	5 permutations	MNIST
svhn-mnist	2	direct sampling	SVHN, MNIST

A four-layer convolutional neural network is trained on the three different sequential arrival datasets in local clients. A generative adversarial network (GAN) is employed on the central server to accumulate client knowledge and generate old task data. The GAN architecture includes a generator network (Generator) and a discriminator network (Discriminator). The Generator is composed of an encoder, a convolutional long short-term memory network (LSTM), and a decoder. On the other hand, the Discriminator is a five-layer convolutional neural network. The specific configuration parameters of the learning model can be found in Table IV.

TABLE IV
Configuration Parameters of FedCMR Model

Parameters	Value
Total available clients(T):	100
Client participation ratio(C):	0.1
Local batch size(B):	32
Local epoch each task(E):	200
Mixing ratio(r):	0.5
Learning rate(l):	0.01

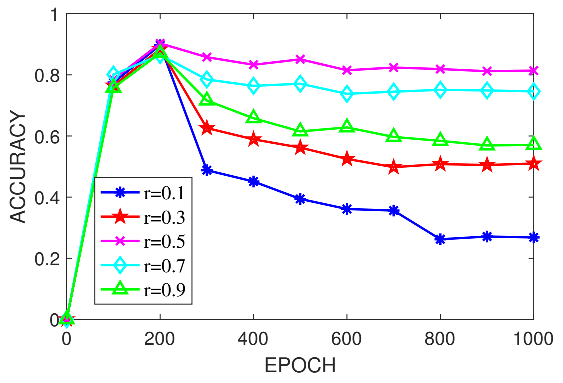


Fig. 3. The precision results of Task 1 in FedCMR framework with different mixing ratios.

B. Experimental Results with Different Mixing Ratios

In the first experiment, the performance of the FedCMR framework on the five-MNIST dataset is investigated by examining the influence of different mixing ratios r. We analyze the performance changes of FedCMR frameworks on Task 1 in the sequential tasks by setting the mixing ratio in the training as 0.1, 0.3, 0.5, 0.7, and 0.9. The experimental results are shown in Figure 3.

Based on the observations from Figure 3, it is evident that the FedCMR framework effectively mitigates the issue of catastrophic forgetting in federated models when new tasks are introduced, across different mixing ratios. Among the various mixing ratios, the model performs best when the mixing ratio r is set to 0.5, which may be because the knowledge of the old task can be preserved without compromising the prediction performance of the model for the current task.

C. Experimental Results with Different DP Noise Powers

In this subsection, we investigate the impact of different levels of differential private noise on the performance of the FedCMR model. To ensure data privacy during data transmission between clients and the server, we introduce noise perturbation on the client side. Specifically, Gaussian noise $N \sim N(0, \delta)$ is added to the transmitted data during communication. The resulting figure illustrates that the accuracy performance of the FedCMR model is significantly influenced by the magnitude of the added noise power. Notably, if a very large amount of noise is added, it can cause the SGD algorithm to converge to a local minimum solution, resulting in training failure of the model. Please refer to Figure 4 for the experimental results.

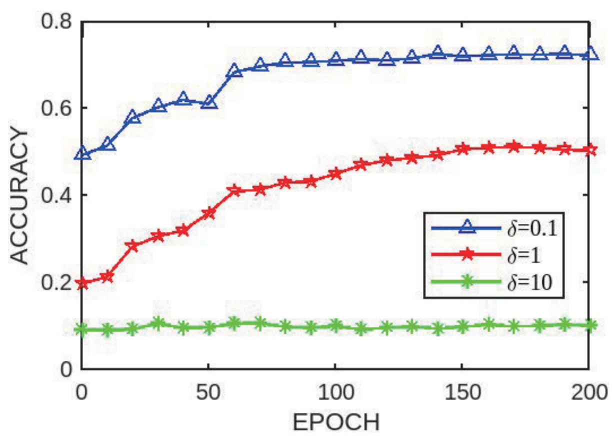


Fig. 4. The precision results of Task 1 in FedCMR framework with different Gaussian noise powers.

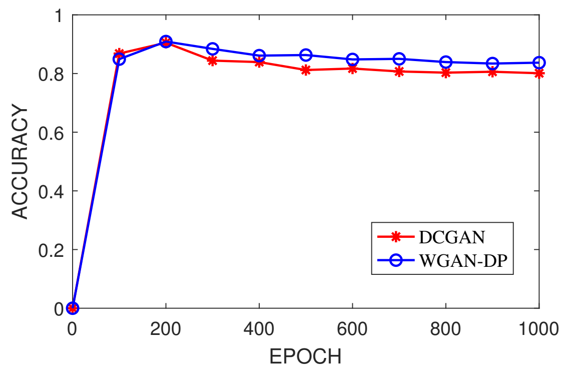


Fig. 5. The precision results of Task 1 in FedCMR framework with different GAN models.

D. Experimental Results with Different GAN Models

In this experiment, we explore the impact of different GAN models on the performance of the FedCMR framework on the five-mnist dataset. We analyze the performance changes of FedCMR frameworks on Task 1 in the sequential tasks by using different GAN models in the central server. The experimental results can be seen in Figure 5:

By employing the DCGAN [34] model or the WGAN-DP [31] model within the FedCMR framework, the issue of catastrophic forgetting when new tasks are introduced is significantly alleviated, as demonstrated in Figure 5. The performance of WGAN-DP model is about 3% higher than that of DCGAN, which may be because WGAN-DP adopts the gradient penalty instead of weight clipping to address the challenges posed by gradient instability and vanishing gradients. Therefore, the FedCMR framework adopts the WGAN-DP model to obtain better quality pseudo-data of old tasks to solve the catastrophic forgetting problem.

E. Performance Against Other Models

In this section, we perform a comparison experiment on the five-mnist dataset to verify the anti-forgetting capability of the client model in the FedCMR framework for Task 1 in the sequential task stream. The models involved in the comparison include the federated learning model FedAVG, the naive federated continual learning model Fed-EWC, and the state-of-art federated continual learning model FedWeIt and FCCL.

In the experimental results, we use the precision metric to accurately estimate the forgetting level on Task 1 of each model. The experimental results of the comparison of each framework are shown in Figure 6. Based on the figure, it is evident that FedAVG exhibits significant performance degradation in the sequential task stream, indicating a notable decline in its ability to retain previously learned knowledge. Obviously, our proposed FedCMR model demonstrates significant improvement in alleviating forgetting on Task 1 compared to baselines, and it achieves slightly better performance than the state-of-the-art method FedWeIt and FCCL, as shown in the experimental results. We compare these four methods as follows:

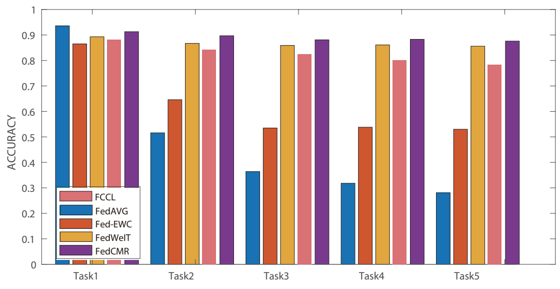
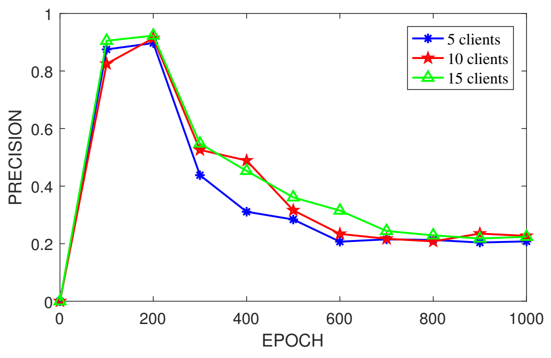


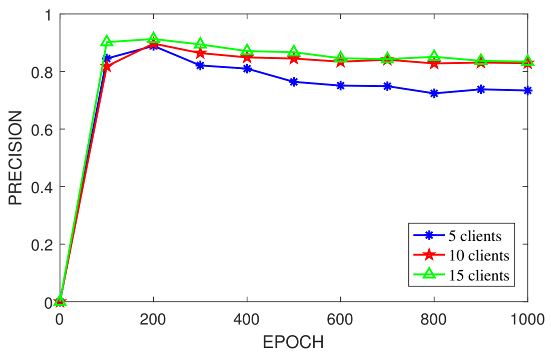
Fig. 6. The precision results of Task 1 in four different algorithm models when the sequential tasks arrive.

1) **The comparison of FedAVG and Fed-EWC:** Obviously, compared to FedAVG, Fed-EWC tempers the performance loss of Task 1 when subsequent tasks arrive. The reason behind this is that the EWC method restricts the variety of important weights by adding regularization to the weights. However, the local EWC model has to expand the network to obtain a higher precision due to the regularization items.
2) **The comparison of Fed-EWC learning and FedCMR:** Compared to Fed-EWC, the proposed method Fed-CMR significantly alleviates the performance drop of Task 1 when subsequent tasks arrive. Resource-constrained local clients are unaffordable for EWC network scaling. As a result, EWC can not work as efficiently as expected. On the contrary, the central memory rehearsal method in FedCMR is perfectly suited for federated learning mechanisms to restore old task knowledge.
3) **The comparison of state-of-the-art models and Fed-CMR:** FedWeIT decomposes each local model parameter into sparse task-specific parameters and global federated parameters based on the calculation of the importance and FCCL utilizes the available unlabeled public data to facilitate communication and employs a cross-correlation matrix to acquire a transferable representation that can generalize well in the presence of domain shift. These additional steps contribute significantly to the computational complexity, distinguishing it from the proposed FedCMR approach. Comparative experimental results show that FedCMR's method exploiting federated mechanisms to transfer knowledge is slightly superior at recovering old task knowledge than the state-of-the-art method FedWeIT. This is presumably because the limited computing resources of the clients cannot afford the calculation of the importance of the local model weights in the FedWeIT framework.

Furthermore, we found that traditional federated learning suffers from severe catastrophic forgetting on real-world streaming datasets from the above comparison. For this reason, federated learning may be difficult to gain practical application in the real world. But fortunately, we can clearly notice the fruitful results of the proposed FedCMR framework on streaming datasets that it not only alleviates the forgetting of old tasks knowledge in local clients when a new task arrives, but also leverages the communication mechanism of federated learning to not hinder the computational efficiency of local clients.



(a) FedAVG



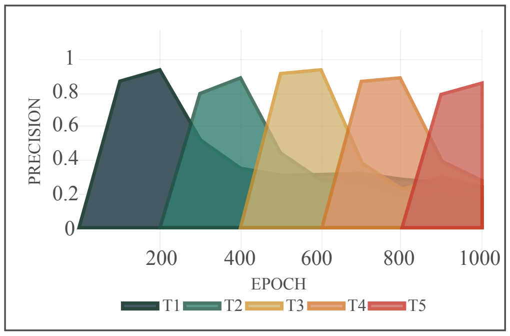
(b) FedCMR

Fig. 7. The experimental results of FedAVG and FedCMR frameworks on Task 1 in the sequential tasks with the number of clients participating in the training as 5, 10, and 15.

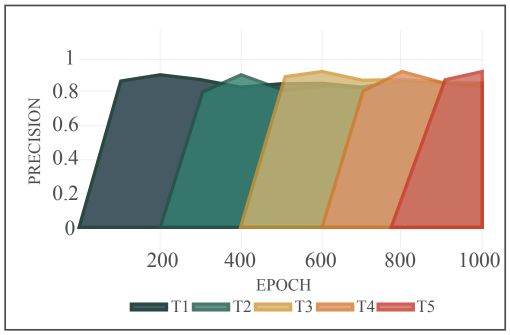
F. Experimental Results with Different Amounts of Participants

In this subsection, we explore the impact of different numbers of clients on the performance of the FedCMR framework on the five-mnist dataset. We analyze the performance changes of FedAVG and FedCMR frameworks on Task 1 in the sequential tasks by setting the number of clients involved in the training as 5, 10, and 15. Figure 7 presents the experimental results:

We can observe that the traditional FedAVG framework suffers from severe forgetting of Task 1 in three different numbers of participating clients. Furthermore, Figure 7 illustrates how different client numbers affect the anti-forgetting level of the proposed FedCMR framework. We can notice that when the number of clients is 5, the accuracy of Task 1 of FedCMR drops by about 13% when Task 5 arrives. With an increasing number of clients, the FedCMR framework demonstrates improved anti-forgetting performance. This enhancement can be attributed to the federated communication mechanism employed by FedCMR, which effectively mitigates catastrophic forgetting. The more client efforts aggregated, the more accurate the memory of old tasks generated by the central server. Hence, it can be concluded that the anti-forgetting capability of the FedCMR framework is positively correlated with the number of participants.



(a) five-mnist-none



(b) five-mnist-FedCMR

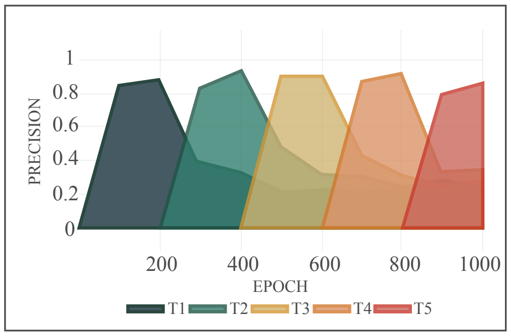
Fig. 8. Accuracy variation of individual tasks in the sequential task streams in traditional federated learning and FedCMR on five-mnist dataset.

G. *Effectiveness Experiments of FedCMR*

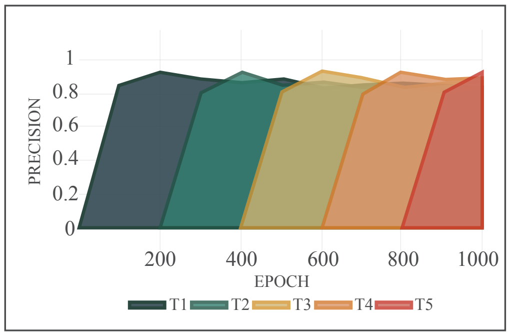
In this section, we estimate the effectiveness of the proposed FedCMR method on three sequential datasets, five-mnist, permu-mnist, and svnh-mnist, compared with the baseline. We validate and visualize the performance variation of each task in the sequential tasks in the following experiments.

1) Experiment on five-mnist dataset

In this experiment, we separate mnist into 5 groups of data by label, which arrives sequentially in each client. In Figure 8(a), we can clearly see that in traditional federated learning, whenever a new task arrives at the client, the previous task accuracy will drop sharply from about 90% to about 23%. This suggests that the client node faces the challenge of catastrophic forgetting, where the knowledge acquired from the previous task in the local model is overwritten by the knowledge gained from the new task. However, in Figure 8(b), whenever a new task arrives, the accuracy of the previous task barely drops. This shows that the local client model in the FedCMR framework fruitfully alleviates catastrophic forgetting.



(a) permu-mnist-none



(b) permu-mnist-FedCMR

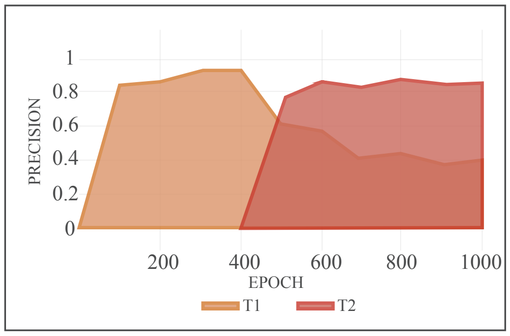
Fig. 9. Accuracy variation of individual tasks in the sequential task streams in traditional federated learning and FedCMR on permu-mnist dataset.

2) Experiment on permu-mnist dataset

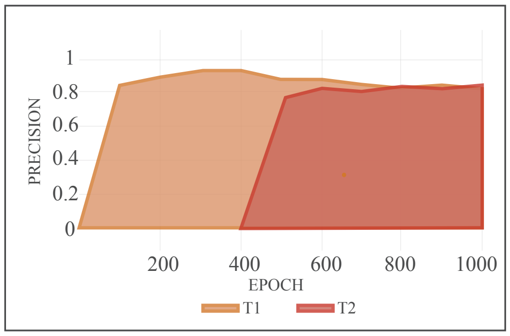
In this experiment, we first define 5 random permutations of MNIST image pixels. The pixels of the images in MNIST are then rearranged according to these 5 permutations. In this fashion, the clients can obtain 5 groups of sequential arriving datasets. In Figure 9(a), we can clearly see that in traditional federated learning, the previous task accuracy drops sharply from about 88% to about 25% whenever a new task arrives at the client. This suggests that the knowledge of the previous task is overwritten by the knowledge of the new task to cause the problem of catastrophic forgetting. However, in Figure 9(b), when each new task arrives, the accuracy of the previous task only approximately drops by 5%. This shows that the FedCMR framework has the ability to avoid the catastrophic forgetting problem in the task sequence.

3) Experiment on svnh-mnist dataset

In this experiment, we use two different but related datasets, MNIST and SVHN datasets, to test the effectiveness of FedCMR. As shown in Figure 10 below, when the model trained on the SVHN dataset by the client in the traditional federated learning framework encounters a new task using the MNIST

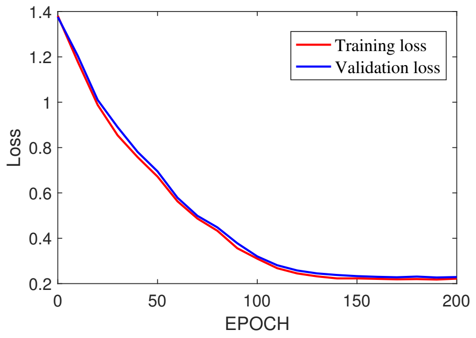


(a) svhn-mnist-none

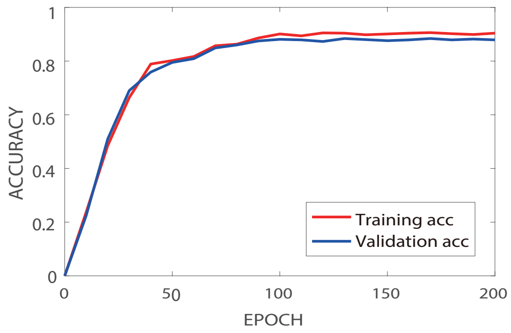


(b) svhn-mnist-FedCMR

Fig. 10. Accuracy variation of individual tasks in the sequential task streams in traditional federated learning and FedCMR on svnh-mnist dataset.



(a) training and validation loss



(b) training and validation accuracy

Fig. 11. Training and validation results of Task 1 in the sequential task streams for FedCMR training on five-mnist dataset.

dataset, the performance of the previous task drops drastically. Conversely, in the FedCMR framework, old tasks can maintain high accuracy when new tasks arrive, which demonstrates that FedCMR can still alleviate forgetting when faced with task sequences from different datasets.

Therefore, from the above three experiments on the 5 random permutations of pixels dataset, the 5 disjoint groups of dataset, and different but related dataset respectively, we can observe that in FedCMR the performance of old tasks decreases on average by about 5% when new task arrives, which demonstrates that FedCMR can effectively mitigate the catastrophic forgetting problem in most related sequential datasets.

H. Training and Validation Results for FedCMR

In this section, we examine the training progress of the federated model in the FedCMR framework to observe the fluctuations in the accuracy and loss function on the five-mnist dataset. The plots in Figure 11 display the curves representing the accuracy and loss of the model during the initial 200 epochs of training on Task 1.

I. FedCMR in Real-World Applications

FedCMR is particularly well-suited for deployment in real-world scenarios such as smart cities and healthcare systems. In smart cities, where edge devices continuously generate data streams for tasks like traffic management and environmental monitoring, catastrophic forgetting is a significant concern. FedCMR can help retain knowledge across tasks such as predicting traffic patterns or detecting anomalies, ensuring that historical data is not forgotten as new data comes in.

Similarly, in healthcare, where patient data streams continuously feed into models for diagnoses and treatments, FedCMR can preserve knowledge from older diagnoses while incorporating data for new medical conditions. For instance, models trained for the diagnosis of chronic conditions can retain their effectiveness even when new, emergent diseases, such as COVID-19, require additional training. The key challenges in these applications include ensuring data privacy under regulations like HIPAA and GDPR and managing the limited computational resources of edge devices in both smart cities and healthcare environments. FedCMR's design, which offloads intensive computations to the central server, mitigates some of these challenges by reducing the burden on local devices.

VI. Conclusion

In this paper, we propose a FedCMR framework to address one of the key challenges faced by federated learning on current real-world sequential data streams: catastrophic forgetting. Specifically, in this paper we explore leveraging the federated learning communication mechanism to achieve efficient recovery of old task knowledge. FedCMR ensures the preservation of knowledge from old tasks in the sequential task stream without compromising communication efficiency. Finally, we estimate the effectiveness of the FedCMR framework in mitigating forgetting on three sequential datasets. The experimental results confirm the effectiveness of the proposed method in alleviating catastrophic forgetting, surpassing baselines, and achieving slight improvements compared to the state-of-the-art method FedWeIt. Going forward, our future work will focus on optimizing the proposed framework through techniques such as model compression and multi-center FedCMR. These optimizations aim to enhance the learning performance and further improve the overall efficiency of the framework.

Acknowledgment

This work was supported by the Key Research and Development Program of Shaanxi (Program No. 2022ZDLGY01-13, 2019ZDLGY17-01), the National Natural Science Foundation of China (Grant No. 62106131), the National Key Research and Development Program of China (Grant No. 2020YFC1523304), and the Fund Program for the Scientific Activities of Selected Returned Overseas Professionals in Shanxi Province (Grant No. 20220002).

References

[1] Jakub Konečný, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[2] Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. A survey on federated learning. *Knowledge-Based Systems*, page 106775, 2021.

[3] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282, 2017.

[4] Weixi Wang, Fan He, Yulei Li, Shengjun Tang, Xiaoming Li, Jizhe Xia, and Zhihan Lv. Data information processing of traffic digital twins in smart cities using edge intelligent federation learning. *Information Processing & Management*, 60(2):103171, 2023.

[5] Wenshuo Wang, Xu Li, Xiuqin Qiu, Xiang Zhang, Vladimir Brusic, and Jindong Zhao. A privacy preserving framework for federated learning in smart healthcare systems. *Information Processing & Management*, 60(1):103167, 2023.

[6] Zhaohua Zheng, Yize Zhou, Yilong Sun, Zhang Wang, Boyi Liu, and Keqiu Li. Applications of federated learning in smart cities: recent advances, taxonomy, and open challenges. *Connection Science*, 34(1):1–28, 2022.

[7] Jan Philipp Albrecht. How the gdpr will change the world. *Eur. Data Prot. L. Rev.*, 2:287, 2016.

[8] Ittai Dayan, Holger R Roth, Aoxiao Zhong, Ahmed Harouni, Amilcare Gentili, Anas Z Abidin, Andrew Liu, Anthony Beardsworth Costa, Bradford J Wood, Chien-Sung Tsai, et al. Federated learning for predicting clinical outcomes in patients with covid-19. *Nature medicine*, 27(10):1735–1743, 2021.

[9] Viktor Losing, Barbara Hammer, and Heiko Wersing. Incremental on-line learning: A review and comparison of state of the art algorithms. *Neurocomputing*, 275:1261–1274, 2018.

[10] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[11] Jaehong Yoon, Wonyong Jeong, Giwoong Lee, Eunho Yang, and Sung Ju Hwang. Federated continual learning with weighted inter-client transfer. In *International Conference on Machine Learning*, pages 12073–12086. PMLR, 2021.

[12] Jiahua Dong, Lixu Wang, Zhen Fang, Gan Sun, Shichao Xu, Xiao Wang, and Qi Zhu. Federated class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10164–10173, 2022.

[13] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[15] Syreen Banabilah, Moayad Aloqaily, Eitaa Alsayed, Nida Malik, and Yaser Jararweh. Federated learning review: Fundamentals, enabling technologies, and future applications. *Information Processing & Management*, 59(6):103061, 2022.

[16] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020.

[17] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[18] Zeinab Teimoori, Abdulsalam Yassine, and M Shamim Hossain. A secure cloudlet-based charging station recommendation for electric vehicles empowered by federated learning. *IEEE Transactions on Industrial Informatics*, 2022.

[19] Xu Cheng, Fan Shi, Yongping Liu, Jiehan Zhou, Xiufeng Liu, and Lizhen Huang. A class-imbalanced heterogeneous federated learning model for detecting icing on wind turbine blades. *IEEE Transactions on Industrial Informatics*, 2022.

[20] Qing Han, Shusen Yang, Xuebin Ren, Peng Zhao, Cong Zhao, and Yimeng Wang. Pcfed: Privacy-enhanced and communication-efficient federated learning for industrial iots. *IEEE Transactions on Industrial Informatics*, 2022.

[21] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

[22] Zhiyuan Chen and Bing Liu. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12:1–207, 2018.

[23] Martial Mermillod, Aurélia Bugaiska, and Patrick Bonin. The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects. *Frontiers in psychology*, page 504, 2013.

[24] Dong Li, Shulin Liu, Furong Gao, and Xin Sun. Continual learning classification method with new labeled data based on the artificial immune system. *Applied Soft Computing*, 94:106423, 2020.

[25] Qiubing Ren, Heng Li, Mingchao Li, Ting Kong, and Runhao Guo. Bayesian incremental learning paradigm for online monitoring of dam behavior considering global uncertainty. *Applied Soft Computing*, 143:110411, 2023.

[26] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017.

[27] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[28] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[29] Fernando E Casado, Dylan Lema, Roberto Iglesias, Carlos V Regueiro, and Senén Barro. Collaborative and continual learning for classification tasks in a society of devices. *arXiv e-prints*, pages arXiv–2006, 2020.

[30] Wenke Huang, Mang Ye, and Bo Du. Learn from others and be yourself in heterogeneous federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10143–10153, 2022.

[31] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

[32] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[33] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[34] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Chen Zhang received the Ph.D. degree in computer application technology from Xidian University, Xi'an, China, in 2012. She is currently an Associate Professor with School of Computer Science and Technology. Her research interests include federated learning, machine learning, and software theory.

Tielin Huang is a master candidate at Xidian University. His research interests include federated learning and unsupervised learning.

Wenjie Mao is a PhD student at Xidian University. His main research interests are deep learning, federated learning, and representation learning.

Hang Bai , a master, focuses on federated learning and information security.

Bin Yu received the Ph.D. degree in computer software and theory from Northwest University, Xi'an, China, in 2003. He is currently a Full Professor with School of Computer Science and Technology. His research interests include artificial intelligence and information security.