International Journal of Networking and Computing – www.ijnc.org, ISSN 2185-2847
Volume 14, Number 2, pages 123–144, July 2024

A Multi-Head Federated Continual Learning Approach for Improved Flexibility and Robustness in Edge Environments

Chunlu CHEN
Graduate School of Information Science and Electrical Engineering, Kyushu University
Fukuoka, Japan

Kevin I-Kai WANG
Dept. of Electrical, Computer and Software Engineering, The University of Auckland
Auckland, New Zealand

Peng LI
School of Computer Science and Engineering, The University of Aizu
Aizuwakamatsu, Japan

Kouichi SAKURAI
Faculty of Information Science and Electrical Engineering, Kyushu University
Fukuoka, Japan

Received: February 15, 2024
Revised: May 4, 2024
Accepted: May 25, 2024
Communicated by Hiroyuki Sato

Abstract

In the rapidly evolving field of machine learning, the adoption of traditional approaches often encounters limitations, such as increased computational costs and the challenge of catastrophic forgetting, particularly when models undergo retraining with new datasets. This issue is especially pronounced in environments that require the ability to swiftly adapt to changing data landscapes. Continual learning emerges as a pivotal solution to these challenges, empowering models to assimilate new information while preserving the knowledge acquired from previous learning phases. Despite its benefits, the continual learning process's inherent need to retain prior knowledge introduces a potential risk for information leakage.

Addressing these challenges, we propose a Federated Continual Learning (FCL) framework with a multi-head neural network model. This approach blends the privacy-preserving capabilities of Federated Learning (FL) with the adaptability of continual learning, ensuring both data privacy and continuous learning in edge computing environments. Moreover, this framework enhances our approach to adversarial training, as the constant influx of diverse and complex training data allows the model to improve its understanding and adaptability, thereby strengthening its defenses against adversarial threats. Our system features a architecture with dedicated fully-connected layers for each task, ensuring that unique features pertinent to each task are accurately captured and preserved over the model's lifetime. Data undergoes processing through these task-specific layers before a final label is determined, based on the highest prediction value.

This method exploits the model's full range of knowledge, significantly boosting prediction accuracy. We have conducted thorough evaluations of our FCL framework on two benchmark datasets, MNIST and CIFAR-10, with the results clearly validating the effectiveness of our approach.

Keywords: Federated Learning, Continual Learning, Adversarial Learning, Catastrophic Forgetting, Security Systems

1 Introduction

As Internet of Things (IoT) devices multiply and mobile computing power grows, edge computing has become a crucial approach for addressing the data processing and analytical challenges at the edge of network. It involves data processing near the source, either at the edge of network or directly within the devices [1], offering a potential solution to the limitations of traditional cloud computing, such as high latency and bandwidth constraints [2]. However, implementing edge computing carries its own set of risks, especially concerning the security and privacy of data within edge devices [3].

The nature of edge computing means that training data is often dispersed and subject to change over time, posing substantial obstacles to the training process [4]. For instance, autonomous vehicles must continuously adjust to new road conditions, while the finance sector needs to constantly learn from new patterns of fraudulent activities. Using conventional machine learning methods in these scenarios requires frequent retraining of models with new data sets, leading to increased computational demands and reduced efficiency due to the need for data integration and model retraining [5].In addition, a critical challenge in this process is catastrophic forgetting, where learning new information causes the model to forget previously acquired knowledge, diminishing its performance on tasks it had learned before [6, 7].

Continual learning, also known as lifelong learning, emerges as a robust countermeasure against the issue of catastrophic forgetting, enabling machine learning models to dynamically integrate new information and master new tasks while retaining insights from earlier tasks [8, 9]. This approach is distinguished by its iterative learning process, where the model sequentially encounters different tasks, each comprising distinct class sets that do not overlap with those of preceding or succeeding tasks [10, 11].

Implementing continual learning on edge devices enhances their ability to adapt to changing conditions by allowing them to learn incrementally from data acquired in real time. This capability is especially crucial for handling Non-Independent and Identically Distributed (Non-IID) data, which often exhibits significant variability in edge computing scenarios due to the diverse origins of data across devices [12]. Despite the advantages, security concerns remain, including issues related to authentication, network security, and secure computation [13]. Edge computing environments may expose vulnerabilities that could be exploited to degrade model performance or expose sensitive data, while the preservation of knowledge in continual learning poses risks of data leakage. Moreover, continual learning's requirement for models to assimilate data from varying distributions can lead to instability in the model's decision-making boundaries. When models are updated to learn new tasks, there's a risk of forgetting previously acquired knowledge, potentially increasing the model's vulnerability to adversarial attacks targeting earlier tasks.

To tackle the challenges of data privacy and the phenomenon of catastrophic forgetting in edge computing, we introduce a novel framework that combines the privacy-enhancing capabilities of Federated Learning (FL) with the adaptability of continual learning systems. This approach allows for local training on devices, obviating the need for central data storage and processing, thus preserving data privacy and security [14, 15]. By implementing continual learning within an FL framework, devices can more effectively adjust to changes in their operating environments and reduce the risk of catastrophic forgetting, enhancing model performance and data integrity [4, 16].

In our proposed framework, each device independently develops a local model using its data, which are subsequently aggregated to form a comprehensive global model. This strategy helps prevent the erasure of previously acquired knowledge, as the learning from individual devices enriches

[0] This is an abstract footnote

the global model without overriding each other's contributions, thus addressing catastrophic forgetting [17, 18]. Moreover, we incorporate adversarial training to bolster the model's defense against cyber threats [19]. The decentralized architecture of FL provides a diversified dataset across devices, facilitating the creation of adversarial examples that span a broad spectrum of potential security vulnerabilities. This diversity ensures that our model remains resilient against various types of attacks. In a continual learning setting, the generation of adversarial examples is continually updated to reflect the model's evolving understanding and potential security gaps, ensuring ongoing protection of the system [20]. By generating and utilizing adversarial examples locally, we further safeguard sensitive data from exposure.

As an extended version of our previous work [21], in this work, we leverage diverse data sources in the dynamic environment of continual learning to conduct adversarial training, extending the dynamic multi-head federated continual learning (FCL) framework. This approach ensures that our system can efficiently process and manage data in the challenging context of edge computing, providing a secure, adaptable, and efficient data handling solution. The primary contributions of our work include:

- Task-Designated Layers: We introduce an innovative design strategy that assigns an individual fully connected layer to each task. This approach facilitates task-specific optimization, specifically catering to the unique characteristics inherent in each task.

- Adaptive Classification: In our model, given a new input sample, the prediction for the label of the sample is determined by propagating the sample through each task-specific layer and selecting the label associated with the highest prediction value across all task-specific models. This process is called adaptive classification because it takes into account all the task-specific layers that have been learned, rather than just relying on a single model. As a result, it enhances prediction accuracy, reduces catastrophic forgetting, and provides flexibility for dynamic environments and tasks.

- Generalization Capability: The inherent flexibility of our model permits effective acclimation to a broad spectrum of tasks, irrespective of potential discrepancies in their label ranges. In our model, each input is processed by all task heads, with the highest prediction chosen as the final classification. This suggests that the output space is collectively constructed by all task heads, rather than independently determined by a single one. This integrated approach seeks to fully harness the model's cumulative knowledge base. As for the matter of catastrophic forgetting, by engaging multiple task heads and their associated knowledge, there's a continual reinforcement of previously learned representations. Each task head, through its shared and differential knowledge, provides a form of regularized learning environment, mitigating the abrupt shifts in the weight space that lead to forgetting.

- Data Privacy Protection: By harnessing the privacy-preserving advantages of FL and integrating them into a continual learning framework, we offer a solution to address the challenges related to privacy in edge computing data processing and analysis. In addition, using the diverse data samples obtained in continuous learning to generate adversarial samples and incorporating these samples into the training process can not only improve the generalization performance of the model, but also help improve the overall system's resistance to these potential attacks.

2 Related Work

Continual Learning: is a subset of machine learning that concentrates on the adaptation of models to evolving data and tasks post-deployment. This strategy promotes the continuous improvement of model performance without the extensive need for retraining, thereby streamlining the process of model refinement post-deployment [11, 22]. It is particularly adept at facilitating the transfer of knowledge between diverse tasks, thereby significantly improving the versatility of models, especially in the realm of reinforcement learning [23].

The domain of continual learning is categorized into three distinct scenarios: Task Incremental Learning (Task-IL), Domain Incremental Learning (Domain-IL), and Class Incremental Learning (Class-IL), each defined by unique challenges, data distribution variations, and specific constraints [24]. This research focuses primarily on Class-IL, which deals with the sequential integration of new classes into an existing model framework within a consistent application environment.

Addressing the stability-plasticity challenge inherent in Class-IL, strategies are generally grouped into three types: regularization techniques, replay mechanisms, and parameter isolation strategies [11]. Among these, replay and regularization strategies are favored for their effective compromise between retaining prior knowledge and accommodating new insights. Notable examples include regularization techniques such as Learning without Forgetting (LwF) [25] and Elastic Weight Consolidation (EWC) [26], and replay-based learning exemplified by iCaRL [27]. Despite facing obstacles such as increased computational demands, potential for overfitting, and confusion between tasks during significant variations, these approaches are pivotal in advancing continual learning.

To overcome the hurdles associated with hyperparameter adjustments and the computational intensity of learning across multiple tasks, recent advancements have introduced meta-learning methodologies [28]. These aim to equip models with the capability to derive overarching learning principles through exposure to a variety of tasks, thereby enhancing their ability to generalize to new challenges [29]. Continual learning and meta-learning, while differing in focus and technique, both contribute significantly to improving model adaptability and generalization across multiple tasks [30–32]. In addition, an asynchronous FCL approach [33] is introduced to address the challenge of asynchronous learning, where clients independently experience different tasks at varying time frames.

Federated Learning: has emerged as a cutting-edge strategy to address the security and privacy concerns associated with Distributed Machine Learning (DML) [34]. As a specialized iteration of DML, FL enables the development of comprehensive models while bypassing the requirement for direct data exchange among contributing entities. This approach allows for the sharing of model parameters or intermediary outcomes instead, thereby enhancing the protection against data breaches during the transfer process [35] and reducing the challenges posed by data compartmentalization within industries [36].

While traditional FL efforts have focused on creating an effective global model through decentralized training, our research seeks to enhance both server and device performances by endowing them with the ability to continuously learn and adjust to constantly changing environments. Previous research has explored the use of knowledge distillation to facilitate the transfer of insights across tasks, thereby mitigating the risk of catastrophic forgetting [37]. This technique allows for the device model to draw on the experiences of other devices through the server model, thereby improving overall model versatility and preventing overfitting on new tasks [5, 37]. Knowledge distillation also promotes the exchange of insights between local models and their consolidation from a local to a global scale [38]. Moreover, by differentiating between global federated weights and task-specific sparse parameters, and enabling devices to selectively incorporate knowledge from their counterparts through a calculated blend of their task-specific parameters, this approach effectively minimizes conflicts between tasks and facilitates a dynamic exchange of knowledge among participating devices [39]. However, these methods require the transfer of information or insights between devices for training purposes, which could potentially compromise data privacy. To address this concern, Variational Autoencoder (VAE) based techniques have been suggested to safeguard the privacy of data on devices [4]. In addition, Differential Privacy (DP) is often discussed in Continual Learning (CL). An approach is proposed that preserves DP in CL, quantifying privacy risk and ensuring privacy guarantees for data records across tasks [40].

Adversarial Training: Adversarial training is a pivotal technique to enhance the robustness of models against potential adversarial attacks [19, 41]. Given that continual learning models train on constantly evolving datasets, they are particularly susceptible to adversarial manipulations, especially during transitions between tasks [42, 43]. Adversarial training addresses these vulnerabilities by incorporating adversarial perturbation samples during the training process. These samples are designed to mimic potential manipulations by adversaries, enabling the model to learn robust features invariant to such manipulations. This process bolsters the model's ability to recognize and

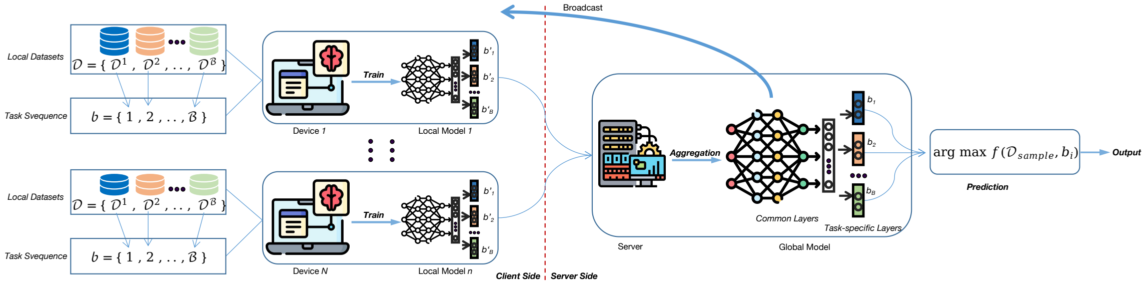

Local Datasets
$\mathcal{D} = \{ \mathcal{D}^1 , \mathcal{D}^2 , \ldots , \mathcal{D}^{\mathcal{B}} \}$
Task Svequence
$b = \{ 1 , 2 , \ldots , \mathcal{B} \}$
Train
b'_1
b'_2
b'_B
Device 1
Local Model 1
Local Datasets
$\mathcal{D} = \{ \mathcal{D}^1 , \mathcal{D}^2 , \ldots , \mathcal{D}^{\mathcal{B}} \}$
Task Svequence
$b = \{ 1 , 2 , \ldots , \mathcal{B} \}$
Train
b'_1
b'_2
b'_B
Device N
Local Model n
Client Side
Server Side
Broadcast
Aggregation
b_1
b_2
b_B
Common Layers
Task-specific Layers
Server
Global Model
$\arg\max f(\mathcal{D}_{sample}, b_i)$
Output
Prediction

Fig. 1. The training process of Continual Learning-based Federated Learning

withstand adversarial inputs, thus maintaining the integrity and reliability of cross-task predictions [44, 45]. Additionally, the decentralized nature of FL allows for training models on multiple devices without the need for centralized data aggregation. However, this distributed training environment also means that models must be individually fortified against adversarial threats, as each device may encounter unique adversarial patterns based on its local data. By incorporating adversarial training, the robustness of models can be extended to ensure their security and effectiveness. Generating adversarial examples locally on each device ensures the robustness of local models, and by aggregating model updates from devices that have undergone adversarial training, the global model can inherit the robustness features learned locally. CL further enables models to maintain their ability to counter new and complex adversarial strategies over time.

Existing adversarial training methods can be categorized into several types. Fast Gradient Sign Method (FGSM) [46] and Fast Gradient Method (FGM) [47] are simple yet effective methods for generating adversarial examples. By leveraging the gradient information of input data, these methods introduce a small perturbation in the direction that increases model loss. The difference between FGSM and FGM lies in their normalization techniques, and they are widely used due to their computational efficiency. Projected Gradient Descent (PGD) [48] is considered an iterative version of FGSM, applying multiple gradient updates to fine-tune perturbations and find more effective adversarial examples. PGD projects the perturbation back to a predetermined range after each update, ensuring the perturbation remains moderate, making PGD-generated adversarial examples more precise and effective. To further optimize the computational process of PGD, methods like Free Adversarial Training (FreeAT) [49] have been proposed. You Only Propagate Once (YOPO) [50] reduces the computational burden of gradient calculation by leveraging the structure of neural networks. The Adversarial Logit Pairing (ALP) [51] method increases model robustness by encouraging similar output distributions for both original and adversarial inputs. Ensemble adversarial training [19] enhances the robustness of a single model by mixing adversarial examples from multiple models, based on the assumption that different models may be sensitive to different adversarial examples, allowing for broader coverage of attack types.

3 Methodology

In this section, we elucidate the theoretical principles of Class-IL and FL, which lay the foundation for our proposed methodology. The details of the algorithm are given in the next subsection. Our approach is designed to overcome several challenges associated with class-incremental learning in a FL setting.

Firstly, our approach targets the challenge of catastrophic forgetting, a pervasive issue in continual learning where learning new tasks causes a model to forget the previously learned tasks. To address this, we designate independent fully connected layers to each task in the learning process. This ensures that the knowledge gained from each task is captured and retained separately, promoting stability and preventing interference from the learning of new tasks.

Secondly, our methodology improves prediction accuracy through the application of multi-head predictions. By propagating a batch of data through each task-specific layer and taking the maximum prediction value across all tasks, we effectively maximize the utilization of all the knowledge contained within the model. This ensures that each prediction leverages insights from all learned tasks, thereby enhancing the prediction accuracy.

In terms of privacy preserving, we also integrate adversarial training techniques to fortify model robustness against potential adversarial attacks, further ensuring the security of the FL system. In the context of FL, adversarial training plays a crucial role not only in protecting models from malicious inputs but also in maintaining data privacy.

3.1 Adversarial Sample Generation

Incorporating adversarial training into our framework, we adopt a strategy where generated adversarial examples and local samples are mixed at a 50% ratio for training purposes, utilizing the FGSM [46] as our method of choice, which is a widely recognized approach for generating adversarial examples, involves applying a small, carefully calculated perturbation to the input data. This perturbation is designed to maximize the model's prediction error, thus creating an adversarial example that is visually similar to the original but is classified incorrectly by the model. By training the model on a dataset composed of an equal mix of adversarial and original examples, we aim to equip the model with the ability to accurately classify both types of inputs, thereby improving its general resilience against attacks.

We consider the scenario of training on a single device, where device k has a local dataset $\mathcal{D}_{ori_k} = \{(\mathbf{x}_{k,i}, y_{k,i})\}_{i=1}^{m_k}$ with m_k samples. To enhance the robustness of the model M_k against potential adversarial attacks, we employ the FGSM to generate adversarial examples from $\mathcal{D}_{ori_k}$. These examples are visually similar to the original samples but are crafted to deceive the model. The creation of adversarial examples is approached as an optimization problem where the goal is to maximize the loss function $\mathcal{L}_k$ by adding a perturbation η to the original input $\mathbf{x}_{k,i}$. The perturbation is calculated using the gradient of the loss function with respect to the input, $\nabla_{\mathbf{x}_{k,i}} \mathcal{L}_k(\theta, \mathbf{x}_{k,i}, y_{k,i})$, ensuring that the magnitude of η is less than ϵ, thereby keeping the adversarial examples within the vicinity of the original data distribution.

The perturbation δ is defined as:

$$\delta = \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}_{k,i}} \mathcal{L}_k(\theta, \mathbf{x}_{k,i}, y_{k,i})) \tag{1}$$

Subsequently, the adversarial examples $\mathcal{D}_{adv_k}$ are created by adding this perturbation to the original inputs $\mathcal{D}_{ori_k}$:

$$\mathcal{D}_{adv_k} = \mathcal{D}_{ori_k} + \delta \tag{2}$$

By introducing FGSM into our Class-IL framework, we aim to continuously train a robust classifier that can defend against adversarial attacks, while each device k optimizes its local model M_k using both original and adversarial examples.

3.2 Class-Incremental Learning

Given a data stream with emerging new classes, Class-IL aims to continually incorporate the knowledge and build a unified classifier [52]. Refer to the definition of Class-IL [53], denote the sequence of $b \in \{1, 2, .., B\}$ training sets without overlapping classes as $\left\{\mathcal{D}^1, \mathcal{D}^2, \cdots, \mathcal{D}^B\right\}$, where $\mathcal{D}^b = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n^b}$ is the b-th training set with n^b samples. A training instance $\mathbf{x}_i \in \mathbb{R}^D$ belongs to class $y_i \in Y_b$. Y_b is the label space of task b, and $Y_b \cap Y_{b'} = \varnothing$ for $b \neq b'$. Following the typical Class-IL setting [27, 54], during the b-th incremental stage, we can only access data from $\mathcal{D}^b$ for model training. The target is to build a unified classifier for all seen classes $\mathcal{Y}_b = Y_1 \cup \cdots Y_b$ continually. In other words, we hope to find a model $f(\mathbf{x}) : X \to \mathcal{Y}_b$ that minimizes the expected loss:

$$f^* = \underset{f \in \mathcal{H}}{\operatorname{argmin}} \mathbb{E}_{(\mathbf{x}_j, y_j) \sim \mathcal{D}_t^1 \cup \cdots \mathcal{D}_t^b} \left[\ell\left(f\left(\mathbf{x}_j\right), y_j\right)\right], \tag{3}$$

where $\mathcal{H}$ denotes the hypothesis space and $\ell(\cdot,\cdot)$ is the CrossEntropy loss function. $\mathcal{D}_t^b$ denotes the data distribution of task b. We assume the global model based on FL is used as the initialization for $f(\mathbf{x})$, which will be introduced in the next Section.

3.3 Multi-Head Federated Continual Learning

We consider a FL system composed of a central server and a set of edge devices indexed by $k \in \{1, 2, \ldots, K\}$. Each edge device, also referred to as a client, can access a sequence of tasks denoted by B and is responsible for sequentially executing each task. For a given task b, device k uses its local dataset $\mathcal{D}_k^b$ to train the model. Additionally, each device generates adversarial samples $\mathcal{D}_{adv_k}^b$ based on its local dataset $\mathcal{D}_k^b$. The combined dataset $\mathcal{D}_k$ for device k is the union of its local datasets for all tasks, expressed as $\mathcal{D}_k = \bigcup_{b\in B} \mathcal{D}_k^b$.

The local loss function for device k when training on task b, which now includes both original and adversarial samples, is defined as:

$$\mathcal{L}_k^b(w) = \frac{1}{2n_k^b}\left(\sum_{i=1}^{n_k^b} \ell\left(f\left(x_{k,i}^b\right), y_{k,i}^b\right) + \sum_{i=1}^{n_k^b} \ell\left(f\left(x_{adv_{k,i}}^b\right), y_{k,i}^b\right)\right), \tag{4}$$

where $(x_{k,i}^b, y_{k,i}^b)$ are the individual samples and labels from the original dataset $\mathcal{D}_k^b$, and $(x_{adv_{k,i}}^b, y_{k,i}^b)$ are the adversarial samples and their corresponding labels generated from $\mathcal{D}_k^b$. The term n_k^b denotes the number of samples for task b on device k, $\ell(\cdot,\cdot)$ signifies the loss function, and $f(\cdot)$ represents the model's output.

Consequently, for each device k, the local loss function for task b can be expressed as $\mathcal{L}_k^b(w)$. Our goal is to minimize the global loss function in the server-side as follows:

$$\min_w F(w), \text{ where } F(w) \triangleq \frac{1}{N}\sum_{k=1}^{K}\sum_{b=1}^{B} n_k^b \mathcal{L}_k^b(w), \tag{5}$$

where $N = \sum_{k=1}^{K}\sum_{b=1}^{B} n_k^b$ represents the total number of samples across all devices and all tasks. The global loss function, $F(w)$, which represents the weighted average of the local loss functions, $\mathcal{L}_k^b(w)$, computed on each edge device k for each task b.

3.4 Federated Continual Learning Architecture

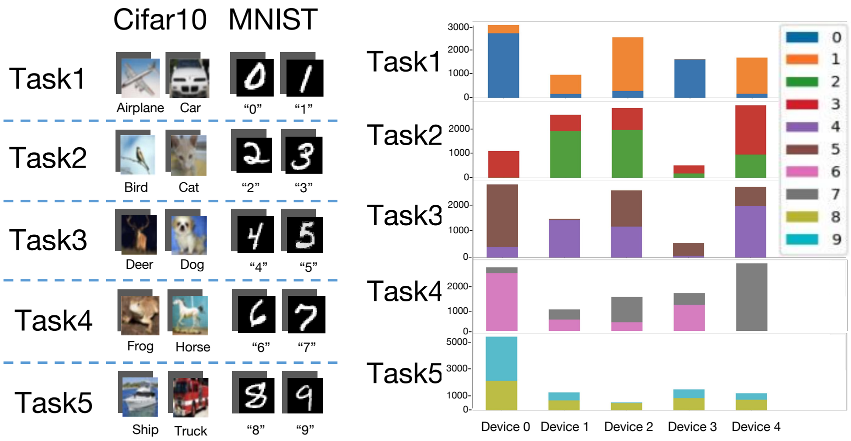
As illustrated in Figure 1, we present the architecture of our system. In this section, we will detail our framework from three aspects: data partitioning, model training, and prediction.

- **Data Partitioning:**

  We divided the dataset into different tasks based on the classes for our experiment. Specifically, we separated the data into five tasks, with each task containing two classes, as depicted in Figure 2(a). Moreover, within each task, we simulated a real data scenario by employing a Dirichlet distribution [55] for the division of the dataset across each device, as illustrated in Figure 2(b). The choice of the Dirichlet distribution stems from its ability to model variations in data distribution across devices, reflecting the inherent data heterogeneity often observed in practical settings. This ensures a more realistic and robust evaluation of the proposed methods in conditions akin to real-world deployments.

- **Training:**

  We consider a FL system comprising a server and a set of edge devices $k \in \{1, 2, .., K\}$, and a sequential training task set $b \in \{1, 2, .., B\}$. We assume the server is considerably more powerful than the devices. The data is distributed across multiple devices, which can be locally exploited for the FL training process without being transferred. We assume that, in a continuous time horizon requiring sequential training of multiple tasks, each device k can only



(a) Segmentation of Data for Distinct Tasks

(b) Data Distribution on the Client Side

Fig. 2. Task Division Based on Classes and Data Distribution.

access their private dataset for each task. For instance, in autonomous driving, a car typically needs to learn about its surrounding environment after passing through a road section. At this point, only current environmental data can be obtained, and a new environment needs to be learned upon entering the next road section.

The training process is as follows:

- Generate Adversarial Samples: Each device generates adversarial samples $\mathcal{D}^b_{adv_k}$ using its dataset $\mathcal{D}^b_k$. This is done by applying small perturbations to existing samples, designed to mislead the model while remaining imperceptible.
- Local Model Training: Each edge device (client) k trains its local model on its respective dataset $\mathcal{D}^b_k$ for each task b. This involves computing the local loss $\mathcal{L}^b_k(w)$ and applying a local update to the model parameters w to minimize this loss. The local loss is computed as the average loss over all samples in the device's local dataset for the given task.
- Model Parameter Uploading: Once local training is complete, each edge device uploads its locally updated model parameters to the server.
- Global Model Aggregation: The server aggregates the uploaded model parameters from all edge devices. This is typically done by taking a weighted average of the model parameters, where the weights correspond to the number of samples on each device for each task, i.e., n^b_k.
- Global Model Broadcasting: After aggregating the model parameters into a global model, the server broadcasts the global model back to the edge devices. Each device then replaces its local model with this updated global model, and the process repeats.
- Iterative Process: Repeat the above steps until the global loss $F(w)$ converges.

This iterative process allows the system to learn a global model for task b that generalizes well across all edge devices and all tasks, while preserving data privacy by avoiding the need to share raw data.

- **Prediction:**

In the prediction phase, we acquire the model w trained for task b and we have a batch of data for prediction, denoted as $\mathcal{D}_{\text{unknown}} = \{x_1, x_2, ..., x_i\}$. We propagate this batch of data through each task-specific layer, for each sample x_i obtaining a set of the prediction values $y^b_{\text{pred},i} = \{x^1_i, x^2_i, ..., x^b_i\}$ from each task-specific layer, where b represents the total number of tasks executed. We choose the highest prediction value as the label of the data. The detailed definition is given as follows:

We have a function $f_b(x; w_b)$ representing the model learned for task b with parameters w_b. Then, given a new input sample $x \in \mathcal{D}_{\text{unknown}}$, the prediction of the label is obtained by

$$y_{\text{pred}} = \arg\max_{b \in B} f_b\left(x; w_b\right)$$

This function chooses the label associated with the highest prediction value across all the task-specific models. The label y_{pred} is the prediction of the model for the input sample x.

4 Experiments

In this work, we focus on continual learning scenarios involving sequential tasks. Our primary aim is to assess the system's performance and its capacity to generalize to new tasks, while also considering the computational cost of our method. Performance is typically measured by the model's accuracy in predicting or classifying data. A model's generalization capability is its ability to apply learned knowledge to new tasks without forgetting information from previous tasks. These aspects are critical in continual learning, where models need to learn from new data without forgetting prior tasks.

In this section, we present our experimental setup, discuss the results, and consider the computational cost of our method compared to traditional approaches.

4.1 Evaluation Protocols

In a setting of continual learning, the network has a separate fully connected layer for each task to differentiate the classes it learns in specific tasks. However, it relies on a so-called oracle that decides the task at test time, which can lead to misleadingly high test accuracy [56]. In contrast, our model adopts a more practical and challenging setting. In this setup, each input passes through all task heads, and the highest predicted category is chosen as the final classification. This means that the output space is jointly formed by all task heads, rather than being determined independently by a single task head. This requires the model to learn how to resolve class confusion across different tasks at test time, even in the absence of explicit task identity. Not only does this strategy maximize the use of all the knowledge in the model, but it also effectively reduces the rate of forgetting, despite its practical challenges.

4.2 Experimental Setup

Datasets: We conducted evaluations of our training framework on two image classification tasks: MNIST [57] and CIFAR-10 [58]. Both are widely recognized as benchmark datasets for image classification tasks. They are extensively used within the computer vision community for the development and evaluation of image classification algorithms and models. As such, they provide a standardized platform for benchmarking the performance of various deep learning architectures and techniques. We have divided the dataset into an 8:1:1 ratio, with 80% allocated for training, 10% for validation, and the remaining 10% for testing. This ratio ensures a robust training set while providing enough data for validation and testing to gauge model performance during development and after training. It also aligns with commonly used practices in deep learning to ensure generalization while preventing overfitting [14, 59].

FL setup: We have constructed a central server accompanied by 5 devices (clients), where the distribution of data across these devices adheres to a Non-IID setting based on the Dirichlet distribution. The assumption of 5 devices is based on common practices in federated learning experiments, which aim to simulate a realistic environment where data is distributed across multiple clients [60, 61]. Additionally, this setup is aligned with our experimental environment to ensure efficiency. This setup provides a balanced scenario for training while allowing for effective aggregation of model

Table 1: Model Structure

Models	*Dataset*	*Model Architecture*	*Model Size (MB)*	*Parameter Size (mill.)*	*Training Time (h)*
EWC	MNIST	1-10C5 2M 10-20C5 2M D 320D 50D	0.083	0.022	1.6
	CIFAR-10	3-6C5 2M 6-16C5 2M 400D 120D 84D	0.24	0.062	1.6
LwF	MNIST	1-6C5 2M 6-16C5 2M 256D 120D 84D	0.169	0.044	1.95
	CIFAR-10	3-6C5 2M 6-16C5 2M 400D 120D 84D	0.24	0.062	2
Our Method	MNIST	1-6C5 2M 6-16C5 2M 256D 120D 84D 2D (multiplied by 5 for five tasks)	0.79	0.21	**0.63**
	CIFAR-10	3-6C5 2M 6-16C5 2M 400D 120D 84D 2D (multiplied by 5 for five tasks)	1.12	0.29	**0.58**

parameters at the central server. For each task, it is collectively trained and accomplished by these 5 devices.

Model: In light of existing research methodologies, we selected various models to evaluate the MNIST and CIFAR-10 datasets. For the method based on EWC [26], which functions by imposing penalties on crucial parameters to preserve knowledge from previously learned tasks during continuous learning, we used a Convolutional Neural Networks (CNN) classification model. The LwF [25] guides the training of new tasks using knowledge from previous tasks. This approach is also present in current FCL studies [5, 37]. For this method, we employed a CNN model with an incremental class capability, allowing for dynamic addition of output categories during training. In our proposal, the model is divided into two parts: the common layers and the task-specific layers. The common layers, based on CNN, serve as the backbone of the model and are shared across all tasks. The task-specific layers consist of independent fully connected layers for each task.

Table 1 details the model architecture and its size after completing five sequential tasks. In our method for CIFAR-10, the architecture begins with a 5x5 convolutional layer that transitions from 3 to 6 channels, followed by a 2x2 max pooling layer. It then has another 5x5 convolutional layer transitioning from 6 to 16 channels, complemented by another 2x2 max pooling layer. Subsequently, the model incorporates dense layers of 400, 120, and 84 units, and culminates with a final 2-unit layer. This structure is replicated for each of the five tasks.

Implementation: The experiments are implemented in PyTorch. We simulate a set of devices and a centralized server on one deep learning workstation (i.e., NVIDIA GeForce RTX 4090 GPU).

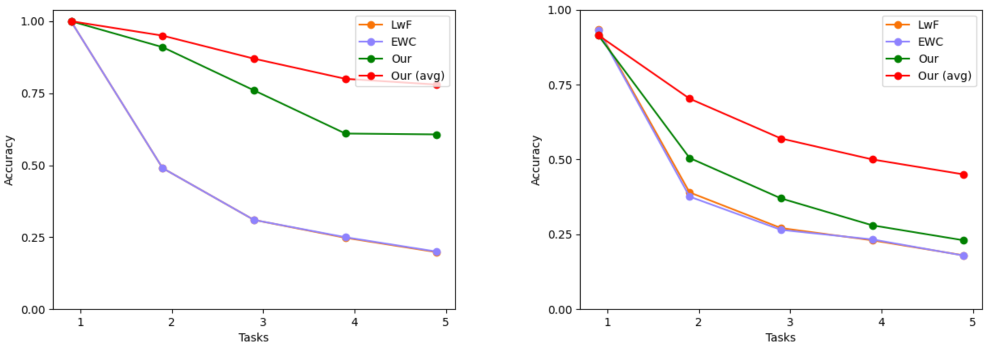
Hyperparameter: For different datasets, we have chosen varying numbers of communication rounds. We run 10 communication rounds and undertake local training on the device side for 20 epochs. We assumed that we have 5 devices (clients). We used a learning rate $\eta = 0.01$, batch size 128 and Stochastic Gradient Descent (SGD) optimizer.

4.3 Experimental Results and Discussions

To comprehensively evaluate model performance in a continual learning setting, we evaluate both the accuracy and the forgetting of the global model on each task upon the completion of the training. In our FCL setting, the data held by each device can vary significantly, and these data distributions exhibit Non-IID characteristics. As a result, the accuracy of the models on different devices could be quite disparate. In addition, after the completion of training for the current task, the global model is broadcasted to all device for updating in preparation for the next task's training. Consequently, the testing results on the current task should be similar across all devices. Hence, in this paper, we mainly analyze the performance of the global model.

The definition of accuracy is the model's average performance across all tasks, evaluated on a per-task basis [10, 11]. Specifically, it denotes the model's average predictive accuracy across all tasks undertaken up to the completion of the N-th task. This measurement helps us gauge how well the model has retained knowledge from previously learned tasks and its capacity to generalize to new tasks. In our experiments, we will assess both the accuracy after completing all tasks and the average accuracy across tasks.

On the other hand, the degree of forgetting serves to gauge the potential loss of information the model may experience as it learns new tasks [62]. It is defined as the difference between the expected task knowledge (i.e., the initial accuracy of the task) and the accuracy after training additional tasks. In simple terms, if a model's accuracy on old tasks drops more profoundly during the process of



(a) MNIST Dataset

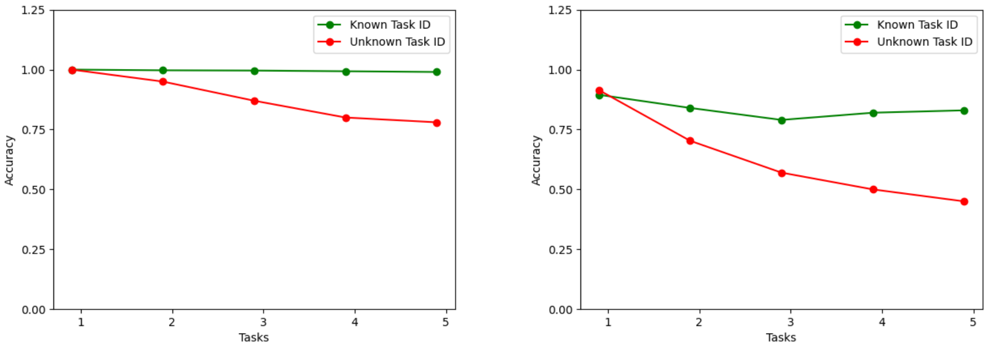
(b) CIFAR-10 Dataset

Fig. 3. The Accuracy Performance

learning new tasks, then its degree of forgetting is higher. This measurement standard aids us in understanding and quantifying the model's stability in retaining old knowledge while acquiring new knowledge in a continual learning environment.

In addition, each edge device generating adversarial samples in FCL, leveraging the unique aspects of its local dataset for every task. This approach exploits the specific data characteristics and inherent vulnerabilities of each device. These samples, highly pertinent to the model's defensive training, ensure the training process is both relevant and effective in bolstering the model's resilience against attacks, with minimal disruption to the training workflow. During local training, the generated adversarial samples are seamlessly integrated into the local dataset, thereby enriching the original data. This augmented dataset include authentic samples and adversarial samples, becomes the new training dataset for the local model. Engaging with this diverse set of data, the model can enhance its ability to discern and withstand malevolent inputs. Subsequent to the adversarial training at the local level, devices proceed to update their model parameters, which are then securely relayed to the central server for aggregation. This ensures the collective benefits of adversarial training permeate the entire network. The global model, thus refined, inherits robustness traits from the individual training episodes conducted locally on each device. To evaluate the model's accuracy, we introduce adversarial samples as a test mechanism. This not only challenges the model's capacity to make accurate predictions but also serves as a rigorous stress test, exposing potential weaknesses that could be exploited by real-world attacks. By rigorously testing with adversarial examples, we ensure that our global model is not only robust but also maintains high accuracy, thus providing a reliable defense mechanism in the operational environment.

In our experiments, we selected the MNIST and CIFAR-10 datasets and tested the performance in terms of accuracy and forgetting of three models—EWC, LwF, and our method. EWC mitigates forgetting by applying regularization to the model weights, assisting the model in retaining memories of old tasks while learning new ones. Although EWC avoids the need to store old data and is compatible with other techniques, it is computationally intensive and tuning its hyperparameters to achieve optimal performance is challenging. In contrast, the LwF method guides the model using the knowledge from old tasks when learning new ones, minimizing catastrophic forgetting. This method neither requires access to the original old task data nor has any compatibility issues with other deep learning methods. Furthermore, to assess the robustness of our model, we conduct a comparative analysis between scenarios where the model has undergone adversarial training and those where it has not. This allows us to observe the enhancements in model resilience as a direct result of our proposed method. The test results and discussion are as follows.



(a) MNIST Dataset

(b) CIFAR-10 Dataset

Fig. 4. Accuracy: Known vs. Unknown Task ID

4.3.1 Computational Cost

The computational cost of our method can be analyzed from various angles, including training time, memory usage, and resource allocation. The proposed method, with its task-specific layers, inherently requires additional storage and computational resources compared to conventional approaches. However, this cost is balanced by the flexibility and adaptability it offers in a continual learning context.

- **Training Time:** Compared to EWC and LwF methods, our method exhibited shorter training times for both the MNIST and CIFAR-10 datasets, table 1 provides the details. This is primarily due to the parallelization of tasks across multiple devices and the efficient aggregation of model parameters at the central server. The flexibility in training with our approach allows for effective distribution across devices, reducing bottlenecks and accelerating the overall training process.

- **Memory Usage:** Our method's structure, with shared common layers and task-specific fully connected layers, does require additional memory. However, the memory overhead is manageable due to the shared architecture, mitigating the need for significant duplication. Compared to the EWC and LwF methods, which rely on regularization or guidance from prior tasks, our method's memory requirements are competitive while offering better accuracy and adaptability.

- **Resource Allocation:** The centralized server model allows for efficient aggregation and distribution of model parameters, ensuring that computational resources are utilized optimally. This architecture, combined with the distribution of training tasks across multiple devices, helps reduce the computational burden on individual devices.

4.3.2 Accuracy

In our evaluation, we focused on two specific accuracies: one is the model's accuracy after the completion of the current task, and the other is the average accuracy of the model for all tasks completed so far. Our test set consists of the training categories of the current task and all the categories of tasks trained before. Figure 3(a) reveals the accuracy of the three models on the MNIST dataset, and Figure 3(b) showcases their performance on the CIFAR-10 dataset. Tables 2 and 3 provide more details on the accuracy of LwF, EWC, and our method on the MNIST and CIFAR datasets.

For the EWC and LwF strategies, in the MNIST dataset test shown in Figure 3(a), we mainly focus on the first type of accuracy; for our approach, we examine both accuracies. During the training phase, we observed that EWC and LwF have challenges in maintaining performance on old

Table 2: Accuracy of MNIST Dataset

Method	***Task1***	***Task2***	***Task3***	***Task4***	***Task5***
LwF	99%	49%	31%	24%	19%
EWC	99%	49%	31%	25%	20%
Our Method (Avg-Acc)	100%	95%	87%	80%	78%
Our Method (Avg-Acc, Known-Task)	100%	99.7%	99.6%	99.3%	99%

Table 3: Accuracy of CIFAR Dataset

Method	***Task1***	***Task2***	***Task3***	***Task4***	***Task5***
LwF	93%	39%	27%	23%	17.9%
EWC	93%	37%	26%	23.3%	17%
Our Method (Avg-Acc)	91.5%	70.3%	57%	50%	45%
Our Method (Avg-Acc, Known-Task)	89.5%	84%	79%	82%	83%

tasks—once a new task is completed, their performance on old tasks declines significantly. Surprisingly, EWC seems to have entirely forgotten the knowledge of old tasks. In contrast, our method outperforms these two strategies in terms of performance. After completing 5 tasks, the accuracy of our model remains above 60%, with an average accuracy reaching 78%. This stands in stark contrast to the performance of EWC and LwF, which have accuracies of around 20%. In the test on the CIFAR-10 dataset (Figure 3(b)), although our model's performance is not as good as on the MNIST dataset, it still has a clear advantage compared to EWC and LwF, maintaining an average accuracy of over 45%, while the latter two are only around 18%.

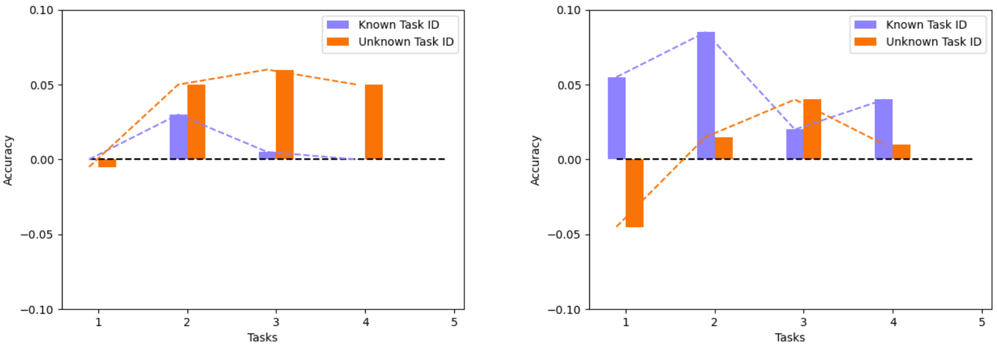
Furthermore, by utilizing the multi-head setup in our model architecture, if we know in advance which task the test data belongs to, the accuracy can be further improved. As shown in Figure 4(a), the average accuracy on the MNIST dataset can rise to over 99%, and as depicted in Figure 4(b), the average accuracy on the CIFAR-10 dataset can increase to over 83%. When the task is known, the accuracy improves significantly, especially in the later tasks, indicating that task recognition has a positive impact on model performance. Overall, these data suggest that our method has higher accuracy on both MNIST and CIFAR datasets, with a better ability to retain knowledge between tasks, while also showing excellent performance when the task is known.

4.3.3 Forgetting

In our evaluations, both EWC and LwF strategies exhibited noticeable performance degradation on old tasks when confronted with new ones, with the phenomenon of forgetting becoming increasingly evident. Alarmingly, under certain circumstances, EWC almost entirely lost its memory of the old tasks. LwF demonstrated a similar trend. Given these observations, we subjected our model to two distinct evaluation methodologies: one where the task ID of the test data was known in advance, and another where it remained unknown. In the latter scenario, the data sequentially passed through task heads, ultimately selecting the category with the highest prediction value as the final classification.

Figure 5(a) illustrates the results derived from the MNIST dataset, while Figure 5(b) presents the performance on the CIFAR-10 dataset. Comparatively, our approach demonstrated significantly reduced forgetting relative to the EWC and LwF methods, signifying its superior efficacy. Table 4 shows the degree of forgetting in models on CIFAR and MNIST datasets after each task. For CIFAR with an unknown task ID, the first two tasks have negative forgetting, indicating that the model's performance on old tasks improved after training new tasks. After that, the degree of forgetting turns positive but remains relatively small. With a known task ID, CIFAR shows positive forgetting across all tasks, particularly in Task 1 and Task 2, with a decrease in later tasks. The forgetting rate on the MNIST dataset oscillated within a range of ±6%, and on the CIFAR-10 dataset, it was around ±4.5%.

For MNIST with an unknown task ID, there's almost no forgetting in the first task, with a slight increase in later tasks, indicating good stability across tasks. With a known task ID, the degree of forgetting in MNIST significantly drops across all tasks, with little to no forgetting in later tasks.



(a) MNIST Dataset

(b) CIFAR-10 Dataset

Fig. 5. Forgetting Evaluation

Table 4: Forgetting

Dataset	***Task2***	***Task3***	***Task4***	***Task5***
CIFAR (Unknown Task ID)	-4.5%	1.5%	4%	1%
CIFAR (Known Task ID)	5.5%	8.5%	2%	4%
MNIST (Unknown Task ID)	-0.5%	5%	6%	5%
MNIST (Known Task ID)	0%	3%	0.5%	0%

This highlights the positive impact of task recognition on reducing the degree of forgetting. Overall, these data suggest that knowing the task ID can effectively reduce the degree of forgetting, especially in the MNIST dataset, where this effect is more pronounced.

In our study, we have elucidated various conjectures and assumptions informed by the data-driven behaviors we observed. It's important to highlight that these ideas, related to both accuracy and forgetting, require a deeper look into how the model processes information, how its parameters change, and how different tasks affect each other. Understanding these key aspects can help researchers and engineers create better strategies for continual learning. This also will guide our next steps and focus in the area of continual learning.

4.3.4 Adversarial Learning

We integrate adversarial learning into our framework. Adversarial learning is achieved by introducing perturbations into inputs, which are designed to challenge the model's predictive accuracy [63]. By training the model to recognize and correctly classify these adversarial samples alongside regular data, we enhance its robustness against potential attacks and unexpected variations in data. The incorporation of adversarial learning not only bolsters the model's defensive capabilities but also contributes to its generalization performance across tasks. It prepares the model to handle ambiguous or slightly altered data inputs that it might encounter in real-world scenarios, thus improving its overall adaptability and reliability. We integrate this approach into our training regimen and evaluate its impact on both the retention of previous knowledge and the acquisition of new information, as discussed in the following experimental evaluations.

In order to verify the effectiveness of adversarial training, we compared it with a model without adversarial training and observed its performance. We compared the model's accuracy on a test set with only normal data and a test set containing adversarial examples. We selected an equal number of normal samples and adversarial samples during training, i.e., a ratio of 50%, to ensure that the model pays equal attention to normal and adversarial samples throughout the training process. Additionally, we employed the FGSM [46, 47], which generates adversarial samples by utilizing the gradient of the input data plus a small perturbation. This perturbation is based on the gradient multiplied by a coefficient (ϵ), and we set it to 0.1 and 0.3.

Table 5: Average Accuracy of the Model Following Adversarial Training

Dataset	***With Normal Samples***	***With Adversarial Samples***	***Adversarial Training (epsilon=0.1)***	***Adversarial Training (epsilon=0.3)***
MNIST	78%	70%	66%	58%
CIFAR-10	45%	28%	**36%**	**30%**

The table 5 below displays the results of our experiments. The first column shows the average accuracy when the model is not subjected to adversarial training and the test set consists of normal samples only. The second column demonstrates the average accuracy when the model is not subjected to adversarial training and the test set consists of adversarial samples. The third and fourth columns display the average accuracy of the model when subjected to adversarial training and the test set consists of adversarial samples. After completing 5 sequential tasks, the average accuracy of both datasets decreased when faced with adversarial samples. Compared with the MNIST dataset, the CIFAR-10 dataset performed better after adversarial training. The feature space of the MNIST dataset is relatively simple, making it easier for the generated adversarial samples to deceive the model, potentially leading to a loss of accuracy on normal data. The improved performance of CIFAR-10 may be attributed to its more complex image features, making the model more susceptible to adversarial attacks when processing this data. Therefore, adversarial training can help the model better learn and adapt to these complex image features, improving the model's generalization ability and robustness. Additionally, the CIFAR-10 dataset is closer to real-world image data than the MNIST dataset, allowing adversarial training to be more effective in realistic scenarios.

5 Challenges and Future Directions

In this section, we delve into the challenges faced by continual learning paradigms and outline potential avenues for future research. As the digital landscape becomes increasingly dominated by the IoT, continual learning emerges as an indispensable trend, finding applicability in diverse sectors. These include image classification [64, 65], object detection [66], semantic segmentation [67, 68], healthcare [69], natural language processing [70, 71], robotics [72–75], Vision-Language Models [53], and federated semi-supervised learning [76].

Despite its potential, the implementation of FCL is fraught with difficulties, primarily due to the heterogeneity and Non-IID nature of data across devices. This diversity in data distribution poses significant challenges in achieving consistent model performance, necessitating innovative approaches to manage data imbalance, mitigate data drift effects, and ensure secure and private training on edge devices. Future research could explore the development of sophisticated FL strategies, such as more nuanced federated aggregation algorithms that account for data heterogeneity and adaptive learning rate mechanisms that optimize training across varied data landscapes.

Moreover, the integration of adversarial learning within FCL frameworks introduces a critical dimension of robustness against adversarial threats, emphasizing the need for balance between robustness and accuracy. To enhance security further, incorporating discussions on DP and encryption algorithm could be pivotal [77]. DP ensures data privacy by adding noise, making it difficult for attackers to extract specific information, while encryption algorithm enables computations on encrypted data, maintaining privacy in untrusted environments. Future research could explore innovative adversarial training techniques that incorporate these encryption algorithms to minimize accuracy loss.

Furthermore, the quest for enhanced model accuracy and efficiency remains central to the continual learning paradigm. Current methods often grapple with constraints engendered by communication overhead, computational intricacy, and the increased complexities intrinsic to FL. Future studies may want to focus on the deployment of efficient communication and computation strategies, like model compression techniques and optimization algorithms, to amplify model accuracy and efficiency while preserving data privacy.

In addition, our framework, which combines the advantages of FL and continual learning, has several potential specific applications in the context of edge computing and data privacy preservation:

- **Edge Computing in Healthcare:** The integration of FL can be particularly beneficial in healthcare settings where patient data privacy is critical [78, 79]. In healthcare, our framework enables medical devices and sensors at the edge to learn and adapt to changing patient conditions while safeguarding data privacy.

- **Financial Services:** In the financial sector, characterized by the utmost importance of data privacy and security [80], our framework can be applied to distributed financial transaction data. This approach enhances fraud detection and financial analysis without compromising customer privacy [81].
- **Personalized Learning and Content Delivery:** In education and content delivery platforms, FCL can enable personalized learning experiences and content recommendations by learning from user interactions in a privacy-preserving manner [82, 83]. This approach can significantly enhance user engagement and satisfaction while safeguarding user data [84].
- **Smart Cities:** Beyond healthcare and financial services, FCL can revolutionize smart city applications, where privacy-sensitive data from numerous IoT devices can be utilized to optimize traffic flow, energy consumption, and public safety measures without compromising individual privacy [85, 86].
- **Autonomous Vehicles:** Within the autonomous driving ecosystem, FCL empowers vehicles to learn from vast arrays of sensor data, while ensuring that sensitive information remains securely within the vehicle's own computing system. This enables a fleet of autonomous vehicles to share valuable insights and improvements in driving algorithms, traffic navigation, and obstacle detection strategies without directly exchanging raw data [87, 88]. Consequently, each vehicle not only becomes better equipped to handle diverse driving conditions and environments but also contributes to a collective intelligence that enhances the safety, efficiency, and reliability of autonomous transportation systems [89].

In summary, the integration of FL and continual learning in edge computing scenarios has the potential to benefit a wide range of applications, particularly in domains where data privacy is a concern. As part of our future work, we will also explore the development of systems in related fields to facilitate cross-domain integration.

6 Conclusion

FL introduces a paradigm that allows edge devices to train models locally, eliminating the need to send sensitive data to central servers and thus enhancing user privacy. Furthermore, continual learning supports the continuous update of models in reaction to changes in local data and immediate needs, ensuring that models remain relevant and optimized in fluctuating conditions. In our study, we introduce a FCL framework that not only champions the cause of ongoing learning but also prioritizes the safeguarding of privacy. Our approach involves assigning a distinct, fully connected layer to each specific task, guaranteeing that the data for each task is processed in a manner that is optimally tailored to its unique features. During the prediction process, this data traverses through every layer dedicated to the tasks, with the label of the highest prediction value identified as the final label. This method leverages the model's complete knowledge base to significantly improve prediction accuracy. Moreover, we enhance the model's robustness and security against potentially malicious inputs by incorporating adversarial training techniques. This multifaceted approach highlights the model's versatility and its suitability for a wide array of applications.

Acknowledgement

The research of the first author is partially supported by the Japan Science and Technology Agency, Support for Pioneering Research Initiated by the Next Generation (JST SPRING) under Grant JPMJSP2136. The research of the second author is partially supported by the International Exchange, Foreign Researcher Invitation Program of National Institute of Information and Communications

Technology (NICT), Japan. The first and fourth authors are funded by JSPS international scientific exchanges between Japan and India, Bilateral Program DTS-JSP, grant number JPJSBP120227718. The third and fourth authors are supported by KAKENHI-PROJECT- JP 24K02932.

References

[1] Dinh C Nguyen, Ming Ding, Quoc-Viet Pham, Pubudu N Pathirana, Long Bao Le, Aruna Seneviratne, Jun Li, Dusit Niyato, and H Vincent Poor. Federated learning meets blockchain in edge computing: Opportunities and challenges. *IEEE Internet of Things Journal*, 8(16):12806–12825, 2021.

[2] Lei Liu, Chen Chen, Qingqi Pei, Sabita Maharjan, and Yan Zhang. Vehicular edge computing and networking: A survey. *Mob. Netw. Appl.*, 26(3):1145–1168, jun 2021.

[3] Yuyi Mao, Changsheng You, Jun Zhang, Kaibin Huang, and Khaled B Letaief. A survey on mobile edge computing: The communication perspective. *IEEE communications surveys & tutorials*, 19(4):2322–2358, 2017.

[4] Tae Jin Park, Kenichi Kumatani, and Dimitrios Dimitriadis. Tackling dynamics in federated incremental learning with variational embedding rehearsal. *arXiv preprint arXiv:2110.09695*, 2021.

[5] Anastasiia Usmanova, François Portet, Philippe Lalanda, and German Vega. A distillation-based approach integrating continual learning and federated learning for pervasive services. *arXiv preprint arXiv:2109.04197*, 2021.

[6] Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi, Matt Gardner, Bryan Kisiel, et al. Never-ending learning. *Communications of the ACM*, 61(5):103–115, 2018.

[7] Gido M Van de Ven, Hava T Siegelmann, and Andreas S Tolias. Brain-inspired replay for continual learning with artificial neural networks. *Nature communications*, 11(1):4069, 2020.

[8] Raia Hadsell, Dushyant Rao, Andrei A Rusu, and Razvan Pascanu. Embracing change: Continual learning in deep neural networks. *Trends in cognitive sciences*, 24(12):1028–1040, 2020.

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

[10] Marc Masana, Xialei Liu, Bartłomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation on image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[11] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

[12] Marcos F Criado, Fernando E Casado, Roberto Iglesias, Carlos V Regueiro, and Senén Barro. Non-iid data and continual learning processes in federated learning: A long road ahead. *Information Fusion*, 88:263–280, 2022.

[13] Pasika Ranaweera, Anca Delia Jurcut, and Madhusanka Liyanage. Survey on multi-access edge computing security and privacy. *IEEE Communications Surveys & Tutorials*, 23(2):1078–1124, 2021.

[14] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[15] Viraaji Mothukuri, Reza M. Parizi, Seyedamin Pouriyeh, Yan Huang, Ali Dehghantanha, and Gautam Srivastava. A survey on security and privacy of federated learning. *Future Generation Computer Systems*, 115:619–640, 2021.

[16] Marcos F Criado, Fernando E Casado, Roberto Iglesias, Carlos V Regueiro, and Senén Barro. Non-iid data and continual learning processes in federated learning: A long road ahead. *Information Fusion*, 88:263–280, 2022.

[17] Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In *International conference on machine learning*, pages 4528–4537. PMLR, 2018.

[18] Kai Zhu, Wei Zhai, Yang Cao, Jiebo Luo, and Zheng-Jun Zha. Self-sustaining representation expansion for non-exemplar class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9296–9305, 2022.

[19] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses, 2020.

[20] Iqbal H Sarker. Multi-aspects ai-based modeling and adversarial learning for cybersecurity intelligence and robustness: A comprehensive overview. *Security and Privacy*, page e295, 2023.

[21] Chunlu Chen, I Kevin, Kai Wang, Peng Li, and Kouichi Sakurai. Flexibility and privacy: A multi-head federated continual learning framework for dynamic edge environments. In *2023 Eleventh International Symposium on Computing and Networking (CANDAR)*, pages 1–10. IEEE, 2023.

[22] Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application, 2023.

[23] Hadi Nekoei, Akilesh Badrinaaraayanan, Aaron C. Courville, and Sarath Chandar. Continuous coordination as a realistic scenario for lifelong learning. *ArXiv*, abs/2103.03216, 2021.

[24] Gido van de Ven, Tinne Tuytelaars, and Andreas Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 4:1–13, 12 2022.

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

[26] 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.

[27] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[28] Sungyong Baik, Myungsub Choi, Janghoon Choi, Heewon Kim, and Kyoung Mu Lee. Meta-learning with adaptive hyperparameters. *Advances in neural information processing systems*, 33:20755–20765, 2020.

[29] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):5149–5169, 2021.

[30] Jaewoong Shin, Hae Beom Lee, Boqing Gong, and Sung Ju Hwang. Large-scale meta-learning with continual trajectory shifting. *arXiv preprint arXiv:2102.07215*, 2021.

[31] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[32] Renkun Ni, Micah Goldblum, Amr Sharaf, Kezhi Kong, and Tom Goldstein. Data augmentation for meta-learning. In *International Conference on Machine Learning*, pages 8152–8161. PMLR, 2021.

[33] Donald Shenaj, Marco Toldo, Alberto Rigon, and Pietro Zanuttigh. Asynchronous federated continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5054–5062, 2023.

[34] Qinbin Li, Zeyi Wen, Zhaomin Wu, Sixu Hu, Naibo Wang, Yuan Li, Xu Liu, and Bingsheng He. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[35] Te-Chuan Chiu, Yuan-Yao Shih, Ai-Chun Pang, Chieh-Sheng Wang, Wei Weng, and Chun-Ting Chou. Semisupervised distributed learning with non-iid data for aiot service platform. *IEEE Internet of Things Journal*, 7(10):9266–9277, 2020.

[36] Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. A survey on federated learning. *Knowledge-Based Systems*, 216:106775, 2021.

[37] Anastasiia Usmanova, François Portet, Philippe Lalanda, and German Vega. Federated continual learning through distillation in pervasive computing. In *2022 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 86–91. IEEE, 2022.

[38] Xiaohan Zhang, Songlin Dong, Jinjie Chen, Qi Tian, Yihong Gong, and Xiaopeng Hong. Deep class-incremental learning from decentralized data. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–14, 2022.

[39] 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.

[40] Pradnya Desai, Phung Lai, NhatHai Phan, and My T Thai. Continual learning with differential privacy. In *Neural Information Processing: 28th International Conference, ICONIP 2021, Sanur, Bali, Indonesia, December 8–12, 2021, Proceedings, Part VI 28*, pages 334–343. Springer, 2021.

[41] Nanyang Ye, Qianxiao Li, Xiao-Yun Zhou, and Zhanxing Zhu. Amata: An annealing mechanism for adversarial training acceleration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10691–10699, 2021.

[42] Matteo Terzi, Alessandro Achille, Marco Maggipinto, and Gian Antonio Susto. Adversarial training reduces information and improves transferability. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 2674–2682, 2021.

[43] Hoki Kim, Woojin Lee, and Jaewook Lee. Understanding catastrophic overfitting in single-step adversarial training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8119–8127, 2021.

[44] Haotao Wang, Tianlong Chen, Shupeng Gui, TingKuei Hu, Ji Liu, and Zhangyang Wang. Once-for-all adversarial training: In-situ tradeoff between robustness and accuracy for free. *Advances in Neural Information Processing Systems*, 33:7449–7461, 2020.

[45] Yanxu Hu and Andy J Ma. Adversarial feature augmentation for cross-domain few-shot classification. In *European Conference on Computer Vision*, pages 20–37. Springer, 2022.

[46] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015.

[47] Takeru Miyato, Andrew M. Dai, and Ian Goodfellow. Adversarial training methods for semi-supervised text classification, 2021.

[48] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks, 2019.

[49] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *Advances in Neural Information Processing Systems*, 32, 2019.

[50] Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. You only propagate once: Accelerating adversarial training via maximal principle. *Advances in Neural Information Processing Systems*, 32, 2019.

[51] Harini Kannan, Alexey Kurakin, and Ian Goodfellow. Adversarial logit pairing. *arXiv preprint arXiv:1803.06373*, 2018.

[52] Da-Wei Zhou, Qi-Wei Wang, Zhi-Hong Qi, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Deep class-incremental learning: A survey. *arXiv preprint arXiv:2302.03648*, 2023.

[53] Da-Wei Zhou, Yuanhan Zhang, Jingyi Ning, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Learning without forgetting for vision-language models. *arXiv preprint arXiv:2305.19270*, 2023.

[54] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 831–839, 2019.

[55] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

[56] Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C-C Jay Kuo. Class-incremental learning via deep model consolidation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1131–1140, 2020.

[57] 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.

[58] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Handbook of Systemic Autoimmune Diseases*, 1(4), 2009.

[59] Yutao Huang, Lingyang Chu, Zirui Zhou, Lanjun Wang, Jiangchuan Liu, Jian Pei, and Yong Zhang. Personalized cross-silo federated learning on non-iid data. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 7865–7873, 2021.

[60] 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.

[61] Mirian Hipolito Garcia, Andre Manoel, Daniel Madrigal Diaz, Fatemehsadat Mireshghallah, Robert Sim, and Dimitrios Dimitriadis. Flute: A scalable, extensible framework for high-performance federated learning simulations. *arXiv preprint arXiv:2203.13789*, 2022.

[62] Ronald Kemker, Marc McClure, Angelina Abitino, Tyler L. Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press, 2018.

[63] Jindong Gu, Hengshuang Zhao, Volker Tresp, and Philip HS Torr. Segpgd: An effective and efficient adversarial attack for evaluating and boosting segmentation robustness. In *European Conference on Computer Vision*, pages 308–325. Springer, 2022.

[64] Eden Belouadah, Adrian Popescu, and Ioannis Kanellos. A comprehensive study of class incremental learning algorithms for visual tasks, 2020.

[65] Benedikt Pfülb and Alexander Gepperth. A comprehensive, application-oriented study of catastrophic forgetting in dnns. *arXiv preprint arXiv:1905.08101*, 2019.

[66] Juan-Manuel Perez-Rua, Xiatian Zhu, Timothy M Hospedales, and Tao Xiang. Incremental few-shot object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13846–13855, 2020.

[67] Fabio Cermelli, Massimiliano Mancini, Samuel Rota Bulo, Elisa Ricci, and Barbara Caputo. Modeling the background for incremental learning in semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9233–9242, 2020.

[68] Onur Tasar, Yuliya Tarabalka, and Pierre Alliez. Incremental learning for semantic segmentation of large-scale remote sensing data. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(9):3524–3537, 2019.

[69] Tanvi Verma, Liyuan Jin, Jun Zhou, Jia Huang, Mingrui Tan, Benjamin Chen Ming Choong, Ting Fang Tan, Fei Gao, Xinxing Xu, Daniel S Ting, et al. Privacy-preserving continual learning methods for medical image classification: a comparative analysis. *Frontiers in Medicine*, 10, 2023.

[70] Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. Lamol: Language modeling for lifelong language learning. *arXiv preprint arXiv:1909.03329*, 2019.

[71] Pankaj Gupta, Yatin Chaudhary, Thomas Runkler, and Hinrich Schütze. Neural topic modeling with continual lifelong learning. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

[72] Timothée Lesort, Vincenzo Lomonaco, Andrei Stoian, Davide Maltoni, David Filliat, and Natalia Díaz-Rodríguez. Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges. *Information Fusion*, 58:52–68, 2020.

[73] Ryan Julian, Benjamin Swanson, Gaurav S Sukhatme, Sergey Levine, Chelsea Finn, and Karol Hausman. Efficient adaptation for end-to-end vision-based robotic manipulation. In *4th Lifelong Machine Learning Workshop at ICML 2020*, 2020.

[74] Bo Liu, Xuesu Xiao, and Peter Stone. A lifelong learning approach to mobile robot navigation. *IEEE Robotics and Automation Letters*, 6(2):1090–1096, 2021.

[75] Sungho Suh, Vitor Fortes Rey, and Paul Lukowicz. Tasked: Transformer-based adversarial learning for human activity recognition using wearable sensors via self-knowledge distillation. *Knowledge-Based Systems*, 260:110143, 2023.

[76] Nan Yang, Dong Yuan, Charles Z Liu, Yongkun Deng, and Wei Bao. Fedil: Federated incremental learning from decentralized unlabeled data with convergence analysis. *arXiv preprint arXiv:2302.11823*, 2023.

[77] Zhiqi Bu, Ping Li, and Weijie Zhao. Practical adversarial training with differential privacy for deep learning, 2022.

[78] Jie Xu, Benjamin S Glicksberg, Chang Su, Peter Walker, Jiang Bian, and Fei Wang. Federated learning for healthcare informatics. *Journal of Healthcare Informatics Research*, 5(1):1–19, 2021.

[79] Jiachun Li, Yan Meng, Lichuan Ma, Suguo Du, Haojin Zhu, Qingqi Pei, and Xuemin Shen. A federated learning based privacy-preserving smart healthcare system. *IEEE Transactions on Industrial Informatics*, 18(3):2021–2031, 2022.

[80] Marco Schreyer, Timur Sattarov, and Damian Borth. Federated and privacy-preserving learning of accounting data in financial statement audits. *arXiv preprint arXiv:2208.12708*, 2022.

[81] Ahmed Imteaj and M Hadi Amini. Leveraging asynchronous federated learning to predict customers financial distress. *Intelligent Systems with Applications*, 14:200064, 2022.

[82] Guanyu Lin, Feng Liang, Weike Pan, and Zhong Ming. Fedrec: Federated recommendation with explicit feedback. *IEEE Intelligent Systems*, 36(5):21–30, 2020.

[83] Feng Liang, Weike Pan, and Zhong Ming. Fedrec++: Lossless federated recommendation with explicit feedback. In *AAAI conf. on artificial intelligence*, pages 4224–4231, 2021.

[84] Jinze Wu, Qi Liu, Zhenya Huang, Yuting Ning, Hao Wang, Enhong Chen, Jinfeng Yi, and Bowen Zhou. Hierarchical personalized federated learning for user modeling. In *The Web Conf.*, pages 957–968, 2021.

[85] Ji Chu Jiang, Burak Kantarci, Sema Oktug, and Tolga Soyata. Federated learning in smart city sensing: Challenges and opportunities. *Sensors*, 20(21):6230, 2020.

[86] Mohammed El Hanjri, Hibatallah Kabbaj, Abdellatif Kobbane, and Amine Abouaomar. Federated learning for water consumption forecasting in smart cities, 2023.

[87] Jin-Hua Chen, Min-Rong Chen, Guo-Qiang Zeng, and Jia-Si Weng. Bdfl: A byzantine-fault-tolerance decentralized federated learning method for autonomous vehicle. *IEEE Transactions on Vehicular Technology*, 70(9):8639–8652, 2021.

[88] Shiva Raj Pokhrel and Jinho Choi. Federated learning with blockchain for autonomous vehicles: Analysis and design challenges. *IEEE Transactions on Communications*, 68(8):4734–4746, 2020.

[89] Sumudu Samarakoon, Mehdi Bennis, Walid Saad, and Mérouane Debbah. Distributed federated learning for ultra-reliable low-latency vehicular communications. *IEEE Transactions on Communications*, 68(2):1146–1159, 2019.