Abstract—Federated Learning (FL) and the Internet of Things (IoT) have revolutionized data processing and analysis, overcoming the traditional limitations of cloud computing. However, traditional machine learning strategies lead to increased costs and catastrophic forgetting due to model retraining with new datasets. Continual learning has been proposed to counter this, enabling models to adapt to new data while preserving previous knowledge, which is beneficial for dynamic edge environments. Despite these advantages, the retention of previous knowledge during the continual learning process may lead to the information leakage. To address the inherent challenges of multi-task scenarios, we present a Federated Continual Learning (FCL) framework that integrates the privacy-preserving benefits of Federated Learning (FL) into a continual learning system, ensuring both continual learning and privacy preservation in edge computing data processing and analysis. Specifically, our architecture introduces dedicated fully-connected layers for each task. This architecture ensures that distinctive features pertinent to each task are not only captured but also preserved throughout the model’s lifespan. Within our framework, data is processed via task-specific layers. Subsequently, the final label is determined by associating it with the paramount prediction value, thus capitalizing on the model’s comprehensive knowledge reservoir to bolster prediction accuracy. We subjected our FCL framework to rigorous validation using two benchmark datasets: MNIST and CIFAR-10. Experimental outcomes unequivocally substantiate the efficacy of our proposed methodology.

Index Terms—Federated Learning, Continual Learning, Catastrophic Forgetting, Privacy Preserving Systems

I. INTRODUCTION

With the proliferation of Internet of Things (IoT) devices and the corresponding increase in mobile computing capabilities, edge computing has emerged as a pivotal solution to the challenges of data processing and analysis at the network periphery. Edge computing refers to the processing of data closer to its source of generation, either at the edge of the network or within the terminal devices themselves [1]. This paradigm has the potential to overcome the shortcomings of traditional cloud computing approaches, particularly issues related to high latency and bandwidth limitations [2]. However, the deployment of edge computing is not without potential hazards, including risks pertaining to the security and privacy of data stored within edge devices [3]. In the context of edge computing, training data is often distributed and subjected to temporal changes, which present significant challenges to the training process [4]. For instance, in autonomous driving applications, models must frequently adapt to novel road conditions. In the financial sector, the detection of emerging patterns of criminal activity necessitates continuous learning. In the event of deploying traditional machine learning strategies, the introduction of new datasets necessitates model retraining. This process increases computational costs and reduces efficiency due to the data integration and model retraining [5]. In addition, a recognized issue with this process is catastrophic forgetting, a phenomenon where the model, in the process of learning a new task, forgets or loses understanding of previous tasks, thereby decreasing performance on the previously learnt tasks [6], [7].

Continual learning, also referred to as lifelong learning, has been proposed as an effective solution to the problem of catastrophic forgetting. This method refers to the strategy where machine learning models continuously adapt to new data and learn new tasks during training while preserving the knowledge from previous tasks [8], [9]. Continual learning is characterized by its sequential nature of the learning process, by sequences of tasks [5]. Each task consists of a set of classes that are mutually exclusive with the classes in other tasks,
Continual learning on edge devices facilitates better adaptability to dynamic environments as the models can continuously learn and improve based on real-time data collected by the devices. Moreover, it can effectively manage Non-Independent and Identically Distributed (Non-IID) data with significant variations [12], a common phenomenon in edge computing due to the distinct nature of data collected by different devices. In addition, security challenges persist, such as authentication mechanisms, networking security, and secure computation [13]. Edge computing may provide potential attackers with opportunities to compromise the model’s performance or leak sensitive information, while continual learning retains previous knowledge that could potentially leak data.

To address these issues, we propose a framework that integrates the privacy-preserving benefits of Federated Learning (FL) into a continual learning system, ensuring both continual learning and privacy preservation in edge computing data processing and analysis. FL facilitates training on local devices, eliminating the necessity to store and process sensitive data on a centralized server. This enables the model to learn from a vast array of data from different devices while maintaining data privacy and integrity [14], [15]. Furthermore, the implementation of continual learning in an FL environment allows distributed devices to adapt better to dynamic environments and can help mitigate catastrophic forgetting [4], [16]. Here, each device can learn a local model based on its own data, and these local models are then aggregated into a global model. This setup reduces the chance of catastrophic forgetting because the learning from each device’s local data doesn’t directly overwrite the learning from other devices’ data. This ensures the retention of knowledge from previous tasks while learning new ones, thereby mitigating catastrophic forgetting [17], [18]. In this work, we propose a dynamic multi-head Federated Continual Learning (FCL) framework that integrates the principles of continual learning and FL within a single model architecture. This solution addresses the challenge of catastrophic forgetting issue in continual learning scenarios, while concurrently ensuring privacy preservation. Our dynamic multi-head FCL framework fuses the strengths of FL and continual learning, culminating in a robust, adaptable, and secure system to manage and process data within the demanding environment of edge computing. 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/models 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.

II. 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 approach enables sustained refinement of model performance, thereby obviating the need for extensive model retraining [11], [19]. Such continual adaptability fosters effective knowledge transfer across varying tasks, significantly enhancing the generalization capabilities of reinforcement learning algorithms [20].

Continual Learning paradigms can be broadly classified into three scenarios, as delineated by [21]: Task Incremental Learning (Task-IL), Domain Incremental Learning (Domain-IL), and Class Incremental Learning (Class-IL). Each represents distinct characteristics, differentiated by variations in data distribution, task-specific constraints, and incremental challenges. This study predominantly concentrates on Class-IL, wherein the central challenge is the sequential introduction of new categories within a consistent operational context.

To mitigate the stability-plasticity dilemma prevalent in the Class-IL setting, methodologies primarily fall into three categories: regularization, replay, and parameter isolation [11]. However, due to their effective balance between preserving existing knowledge and integrating new information, replay methods and regularization-based approaches are more commonly employed. Exemplifying the regularization-based techniques are Learning without Forgetting (LwF) [22] and Elastic Weight Consolidation (EWC) [23], while replay-based learning is epitomized by methods such as iCaRL [24]. Despite their inherent challenges, such as heightened computational resource demands, overfitting, and task confusion during large
task disparities, these methods nonetheless represent critical milestones in the evolution of continual learning solutions.

In recent years, to address the challenges of tedious hyperparameter tuning and computational complexity in multi-task learning, meta-learning methods have been proposed [25]. The core idea is to enable models to acquire generic learning rules and strategies by learning across multiple tasks, thereby achieving stronger generalization performance on new tasks [26]. Continual learning and meta-learning represent two distinct approaches to solving multi-task learning and generalization problems. Despite their differences in application domains and methodologies, both methods effectively enhance model adaptability and generalization performance [27]–[29].

**Federated Learning (FL):** was introduced as an advanced solution for responding to the security and privacy vulnerabilities inherent in Distributed Machine Learning (DML) [30], [31]. FL, a specific form of DML, facilitates the construction of global models without necessitating data sharing across individuals or data silos. By permitting the exchange of parameters or intermediate results, it improves data leakage issues during communication [32] and mitigates the impact of industry data silos [33].

Traditional FL endeavours primarily to generate an efficient global model on the server through decentralized training. In contrast, our work aims to secure high-quality performance on both the server and device sides, granting them with continual learning capabilities to adapt to perpetually evolving application scenarios. Prior studies have deployed knowledge distillation for learning and transmitting knowledge across tasks, attenuating catastrophic forgetting [34]. Building upon this, the device model can leverage knowledge from other devices via the server model, enhancing model generality and forestalling overfitting on novel tasks [5], [34]. Additionally, knowledge distillation encourages knowledge interchange amongst local models and aggregation from local to global scales [35]. Furthermore, by partitioning network weights into global federated parameters and sparse task-specific parameters, and allowing each device to receive selective knowledge from peers via a weighted combination of their task-specific parameters, its can optimally minimize interference between incompatible tasks and foster active knowledge exchange among devices [36]. However, these schemes necessitate exchanging information or knowledge from devices for training, thereby posing a substantial risk to data privacy. To this end, Variational Autoencoder (VAE)-based methods have been proposed for the protection of device data privacy [4].

### III. 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, our FL approach ensures that data privacy is maintained. In the FL system, individual devices only access their private datasets. All model training is done locally on each device, and only the parameters are sent to the server for aggregation. This decentralization of data ensures that sensitive information remains local, mitigating the risk of data leakage.

#### A. Class-Incremental Learning

Given a data stream with emerging new classes, Class-IL aims to continually incorporate the knowledge and build a unified classifier [37]. Refer to the definition of Class-IL [38]. denote the sequence of \( b \in \{1, 2, \ldots, B\} \) training sets without overlapping classes as \( \{D^1, D^2, \ldots, D^B\} \), where \( D^b = \{(x_i, y_i)\}_{i=1}^{n^b} \) is the \( b \)-th training set with \( n^b \) samples. A training instance \( x_i \in \mathbb{R}^{D^b} \) belongs to class \( y_i \in Y_b \). \( Y_b \) is the label space of task \( b \), and \( Y_b \cap Y_{b'} = \emptyset \) for \( b \neq b' \). Following the typical Class-IL setting [24], [39], during the \( b \)-th incremental stage, we can only access data from \( D^b \) for model training. The target is to build a unified classifier for all seen classes \( Y_b = Y_1 \cup \cdots \cup Y_b \) continually. In other words, we hope to find a model \( f(x) : X \rightarrow Y_b \) that minimizes the expected loss:

\[
\hat{f} = \arg\min_{f \in H} \mathbb{E}_{(x_i, y_i) \sim \cup D^1 \cup \cdots \cup D^b} [\ell(f(x_i), y_i)].
\] (1)

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

#### B. Multi-Head Federated Continual Learning

We consider an FL system composed of a server and \( k \in \{1, 2, \ldots, K\} \) edge devices, each edge device (or client) can access the task sequence \( B \) and will sequentially execute each task. Then, each device \( k \) uses its local dataset \( D^k \) to train the corresponding task, where \( D^k \) is the private dataset of device \( k \) corresponding to task \( b \). The dataset \( D^k \) of device \( k \) is the union of datasets for all tasks, i.e., \( D_k = \cup_{b \in B} D^k \).
The local loss function of device $k$ utilized for training task $b$ is defined as:
\[
\mathcal{L}^b_k(w) = \frac{1}{w^b_k} \sum \ell \left( f \left( x_{k,i}^b \right), y_{k,i}^b \right),
\]
with
\[
(x_{k,i}^b, y_{k,i}^b) \in \mathcal{D}^b_k
\]
where $n^b_k$ denotes the number of samples on device $k$ associated with task $b$, $\ell(\cdot, \cdot)$ signifies the loss function, and $f(\cdot)$ is the function representing the model’s output.

Consequently, for each device $k$, the local loss function for task $b$ can be expressed as $\mathcal{L}^b_k(w)$. Our goal is to minimize the global loss function in the server-side as follows:
\[
\min_{w} F(w), \text{ with } F(w) \triangleq \frac{1}{N} \sum_{k=1}^{K} \sum_{b=1}^{B} n^b_k \mathcal{L}^b_k(w),
\]
where $N = \sum_{k=1}^{K} \sum_{b=1}^{B} n^b_k$ 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, $\mathcal{L}^b_k(w)$, computed on each edge device $k$ for each task $b$.

C. 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 [40] 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 $k \in \{1, 2, \ldots, K\}$, and a sequential training task set $b \in \{1, 2, \ldots, B\}$. 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 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:

  - **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, 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 $\mathcal{D}_{\text{unknown}} = \{x_1, x_2, \ldots, x_i\}$. We propagate this batch of data through each task-specific layer, for each sample $x_i$ obtaining a set of the prediction values $y_{\text{pred}, i} = \{x_1^i, x_2^i, \ldots, x_b^i\}$ 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

$$y_{\text{pred}} = \underset{\{b \in B\}}{\text{arg max}} f_b(x; w_b)$$

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

IV. EXPERIMENTS

In our work, we concentrate on scenarios involving sequential tasks. We primarily address two pivotal aspects: the performance of the system and its generalization capability for new tasks. Performance is typically gauged by the model’s capacity to predict or classify data accurately. A model’s generalization capability is measured by its proficiency in applying the knowledge it has learned to new tasks. This is especially critical in continual learning settings where models are expected to learn from new data or tasks without forgetting information acquired from previous tasks. To investigate these aspects, we conduct a series of experimental evaluations and engage in comprehensive discussions throughout this section.

A. 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 [41]. 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.

B. Experimental Setup

**Datasets:** We conducted evaluations of our training framework on two image classification tasks: MNIST [42] and CIFAR-10 [43]. 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.

**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. 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 [23], which functions by imposing penalties on crucial parameters to preserve knowledge from previously learned tasks during continuous learning, we used a CNN classification model. The LwF method [22] guides the training of new tasks using knowledge from previous tasks. This approach is also present in current FCL studies [5], [34]. 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 I 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
<table>
<thead>
<tr>
<th>Method</th>
<th>Dataset</th>
<th>Model Architecture</th>
<th>Model Size (MB)</th>
<th>Parameter Size (mill.)</th>
<th>Training Time (h)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EWC</td>
<td>MNIST</td>
<td>1-10C5 2M 10-20C5 2M</td>
<td>D 320D 50D</td>
<td>0.083</td>
<td>0.022</td>
</tr>
<tr>
<td></td>
<td>CIFAR-10</td>
<td>3-6C5 2M 6-16C5 2M</td>
<td>400D 120D 84D</td>
<td>0.24</td>
<td>0.062</td>
</tr>
<tr>
<td>LwF</td>
<td>MNIST</td>
<td>1-6C5 2M 6-16C5 2M</td>
<td>256D 120D 84D</td>
<td>0.169</td>
<td>0.044</td>
</tr>
<tr>
<td></td>
<td>CIFAR-10</td>
<td>3-6C5 2M 6-16C5 2M</td>
<td>400D 120D 84D</td>
<td>0.24</td>
<td>0.062</td>
</tr>
<tr>
<td>Our Method</td>
<td>MNIST</td>
<td>1-6C5 2M 6-16C5 2M</td>
<td>256D 120D 84D</td>
<td>0.79</td>
<td>0.21</td>
</tr>
<tr>
<td></td>
<td>CIFAR-10</td>
<td>3-6C5 2M 6-16C5 2M</td>
<td>400D 120D 84D</td>
<td>1.12</td>
<td>0.29</td>
</tr>
</tbody>
</table>

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.

C. 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 [44]. 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 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 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, it doesn’t involve the computation of weight importance, making it easy to deploy. The test results and discussion are as follows.

Fig. 3. The Accuracy Performance based on MNIST Dataset.

Fig. 4. The Accuracy Performance based on CIFAR-10 Dataset.

Fig. 5. Accuracy on MNIST: Known vs. Unknown Task ID.
The 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 reveals the accuracy of the three models on the MNIST dataset, and Figure 4 showcases their performance on the CIFAR-10 dataset.

For the EWC and LwF strategies, in the MNIST dataset test shown in Figure 3, 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 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 4), 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 5, the average accuracy on the MNIST dataset can rise to over 99%, and as depicted in Figure 6, the average accuracy on the CIFAR-10 dataset can increase to over 83%. Regarding the decrease in accuracy, we believe the potential reasons are as follows:

- Insufficient generalization capability: challenge faced by our model stems from its architectural design: it employs multiple task heads for final label prediction. Ideally, the task head corresponding to the test sample’s label category, having been trained with data from that category, should exhibit strong predictive performance.

However, with each task head being exposed to a limited subset of data categories during training, there’s inherent limitation in their generalization capabilities. This is prominently observed in multifaceted datasets such as CIFAR-10, where each task head gets trained on just two categories. Consequently, during testing, these heads often encounter data from categories they haven’t been trained on, potentially leading to misclassifications.

Our experiments indeed corroborate this observation. As illustrated in the results Figure 5, while task heads exhibited high accuracy for known categories (averaging around 88%), their performance notably deteriorated for unknown ones, reaching an average accuracy of merely 42%. This pronounced difference highlights the inherent challenge and confirms our assertion.

- Weight Conflicts: The shared convolutional layers need to handle information from different tasks. Weight conflicts might occur when the labels of the test data do not align with those of the training data. Different tasks may require the model to discern distinct features. When the layers are shared, they need to strike a balance between these tasks. This compromise can lead to suboptimal weights for any individual task, causing a dip in the model’s performance.

- High Similarity of Training Data: The uniformity of certain datasets can influence the model’s performance on test data. Specifically, considering the MNIST dataset where images are represented as grayscale digits, there is an inherent similarity in features such as digit shape and grayscale intensity. In training on such data, each task head might inadvertently focus on these predominant features, possibly leading to overfitting. This overlap of prominent features could cause confusion during predictions, especially if multiple task heads produce confident outputs for the same input.

The 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 7 illustrates the results derived from the MNIST dataset, while Figure 8 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. Specifically, the forgetting rate on the MNIST dataset oscillated within a range of $\pm 6\%$, and on the CIFAR-10 dataset, it was around $\pm 4.5\%$. Interestingly, as the number of tasks increased, models trained on both datasets showcased a declining trend in their forgetting rates. This not only suggests that the model effectively retained existing knowledge but also potentially enhanced its accuracy on preceding tasks.

However, we observed a significant discrepancy in performance across the two datasets when we either knew or did not know the task ID of the test data. We speculate that the possible reasons might include:

- **Variation in Data Feature Distribution:** When the model knows the task ID of the test data, it can apply specific strategies or settings for that task, helping it to better handle that task’s data distribution. Without this knowledge, the model might struggle to adapt to multiple tasks, making it hard to accurately adjust to the specific data distribution of any given task.

- **Knowledge Retention Approaches:** Knowing the task ID allows the model to use its memory or knowledge-saving methods more efficiently, which can reduce forgetting. In contrast, not knowing the task ID can make it harder for the model to decide which knowledge is most relevant at the time, affecting its ability to retain that knowledge.

- **Dataset Complexity:** Differences between CIFAR-10 and MNIST can also impact the amount they are forgotten. For example, CIFAR-10 images are more complex than those in MNIST, which might lead to a greater degree of forgetting, especially when the task ID isn’t known.

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.

V. 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 [45], [46], object detection [47], semantic segmentation [48], [49], natural language processing [50], [51], robotics [52]–[54], Vision-Language Models [38], and federated semi-supervised learning [55].

The FCL framework that is proposed based on the principles of continual learning presents unique advantages. Foremost among them is its ability to seamlessly integrate new data into the training process at any given point. By eradicating the need to maintain large-scale training datasets, the framework substantially reduces storage and computational overheads. Simultaneously, it addresses the issue of user privacy infringement.

Despite the numerous advantages, training within the FL framework is not without its challenges. Given the Non-IID nature of data on edge devices, concerns arise around handling imbalanced data distribution, countering data drift, and securing training data on these devices. Future research efforts could pivot towards leveraging advanced FL algorithms and strategies - such as federated aggregation algorithms and adaptive learning rate adjustments - to fortify model performance in Non-IID scenarios.

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 [56], [57]. 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 [58], our framework can be applied to distributed financial transaction data. This approach enhances fraud detection and financial analysis without compromising customer privacy [59], [60].

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.

VI. CONCLUSION

FL provides a mechanism for edge devices to train models locally, thereby circumventing the need to transmit raw data to centralized servers, which, in turn, preserves user privacy. Moreover, FL facilitates continual learning and the prompt updating of models in response to local data alterations and real-time requisites. This ensures model adaptability to changing environments and accomplishes timely optimization and refinement of models at the edge. In this paper, we propose a dynamic federated continual learning framework that not only advocates for continual learning but also ensures privacy preservation. Our proposed model assigns a unique, fully connected layer to each task, ensuring that each task is processed by a layer specifically fine-tuned for its particular characteristics. Moreover, during the prediction phase, data is processed by every task-specific layer. The label corresponding to the highest prediction value is deemed the final data label, thereby exploiting the full knowledge spectrum of the model, which consequently enhances prediction accuracy. The novelty of our model lies in its innovative task-specific layers, its capacity for adaptive classifications to amplify model accuracy. These attributes underpin its potential for extensive deployment across a diverse range of tasks.

ACKNOWLEDGEMENT

This work 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.

REFERENCES


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


Xiaohua 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.


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


