# Continual Federated Learning Based on Knowledge Distillation

**Yuhang Ma**[1] , **Zhongle Xie**[1] , **Jue Wang**[1] , **Ke Chen**[1] and **Lidan Shou**[1,2]
[1]College of Computer Science and Technology, Zhejiang University
[2]State Key Laboratory of CAD&CG, Zhejiang University
{myh0032, xiezl, zjuwangjue, chenk, should}@zju.edu.cn

## Abstract

Federated learning (FL) is a promising approach for learning a shared global model on decentralized data owned by multiple clients without exposing their privacy. In real-world scenarios, data accumulated at the client-side varies in distribution over time. As a consequence, the global model tends to forget the knowledge obtained from previous tasks while learning new tasks, showing signs of "catastrophic forgetting". Previous studies in centralized learning use techniques such as data replay and parameter regularization to mitigate catastrophic forgetting. Unfortunately, these techniques cannot adequately solve the non-trivial problem in FL. We propose Continual Federated Learning with Distillation (CFeD) to address catastrophic forgetting under FL. CFeD performs knowledge distillation on both the clients and the server, with each party independently having an unlabeled surrogate dataset, to mitigate forgetting. Moreover, CFeD assigns different learning objectives, namely learning the new task and reviewing old tasks, to different clients, aiming to improve the learning ability of the model. The results show that our method performs well in mitigating catastrophic forgetting and achieves a good trade-off between the two objectives.

## 1 Introduction

Federated Learning (FL) [McMahan *et al.*, 2017] is proposed as a solution to learn a shared model using decentralized data owned by multiple clients without disclosing their private data. Figure 1 illustrates an example of a FL task to infer the usage habits of mobile device users. The sensitive data, namely a stream of the temporal histogram of screen time, is collected on mobile devices (clients) and used to train a local model. Meanwhile, a global model is built on a central server, which leverages the local models submitted by the clients without accessing any client's private data. During each round of training, the server first broadcasts the global model to clients. Then, each client independently uses its local data to update the model and submits the model update to the server. Finally, the server aggregates these updates to produce a new global model for the next round.

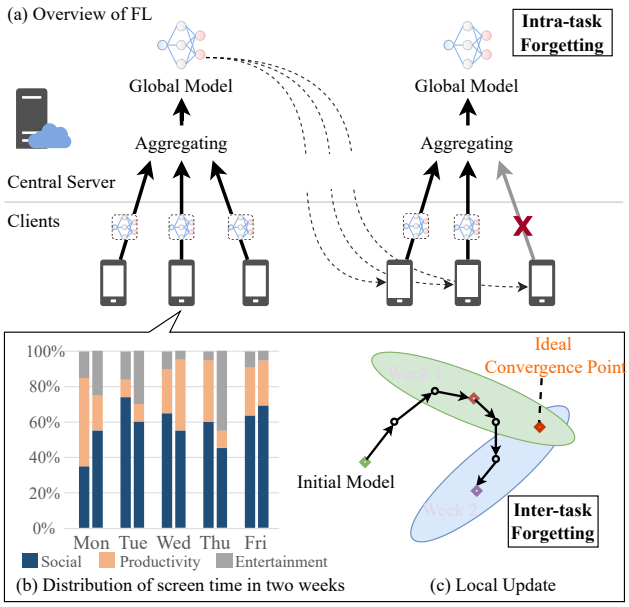


Figure 1: (a) An overview of a FL system to predicate the usage habits on mobile device. (b) Distribution of screen time in two weeks for a specific user. (c) Local update suffers catastrophic forgetting while learning from new data.

Although FL can protect data privacy well, its performance is at risk in practice due to the following issues. First, clients participating in one round of training may become unavailable in the next round due to network failure, causing variation in the training data distribution between consecutive rounds. Second, the data accumulated at the client-side may vary over time, and can even be considered as a *new task* with different data distribution or different labels, which poses significant challenges to the adaptability of the model. Furthermore, due to the inaccessibility of the raw data, minimizing the loss on the new task may increase the loss on old tasks. These issues all lead to underperformance of the global model, a phenomenon known as "catastrophic forgetting".

Specifically, catastrophic forgetting in FL system is observed in two main categories, namely *intra-task forgetting* and *inter-task forgetting*. (1) Intra-task forgetting occurs when two different subsets of clients are involved in two con-

secutive rounds. In Fig. 1(a), since a client does not participate in a training round, the new global model may forget knowledge obtained from this client in previous rounds and thus performs poorly on its local data. (2) Inter-task forgetting occurs when clients accumulate new data with different domains or different labels. As shown in Fig. 1(c), due to the different distribution of the screen time data in week 2, the performance of the new global model on week 1 data is degraded. It should be noted that the Non-IID issue brings more challenges to both kinds of forgetting.

Catastrophic forgetting in FL is a non-trivial problem because the conventional approaches to catastrophic forgetting, namely Continual Learning (CL), cannot be easily applied in FL due to privacy and resource constraints. Some recent attempts on this topic, such as [Shoham *et al.*, 2019; Usmanova *et al.*, 2021], do not solve the problem adequately since they are designed to address either intra-task forgetting or inter-task forgetting.

In this paper, we propose a framework called **C**ontinual **Fe**derated Learning with **D**istillation (CFeD) to mitigate catastrophic forgetting on both intra-task and inter-task categories when learning new tasks. Specifically, CFeD leverages *knowledge distillation* in two ways based on unlabeled public datasets called the *surrogate datasets*. First, while learning the new task, each client transfers the knowledge of old tasks from the model converged on the last task into the new model via its own surrogate dataset to mitigate inter-task forgetting. Meanwhile, CFeD assigns the two objectives to different clients to improve the performance, called *clients division mechanism*. Second, the server also maintains another independent surrogate dataset to fine-tune the aggregated model in each round by distilling the knowledge learned in the current and last rounds into the new aggregated one, called *server distillation*.

The main contributions of this paper are as follows:

- We extend continual learning to the federated learning scenario and define Continual Federated Learning (CFL) to address catastrophic forgetting when learning a series of tasks. (Section 3)
- We propose a CFL framework called CFeD, which employs knowledge distillation based on surrogate datasets to mitigate catastrophic forgetting both at the server-side and client-side. In each round, the inter-task forgetting is mitigated by assigning clients to learning the new task or to reviewing the old tasks. The intra-task forgetting is mitigated by applying a distillation scheme at the server-side. (Section 4)
- We evaluate two scenarios of CFL by varying the data distribution and adding categories on text and image classification datasets. CFeD outperforms existing FL methods in overcoming forgetting without sacrificing the ability to learn new tasks. (Section 5)

## 2 Related Work

### 2.1 Continual Learning

Continual Learning (CL) aims to solve *stability-plasticity* dilemma when learning a sequence of tasks [Delange *et al.*, 2021]. Models with strong *stability* forget little but perform poorly on the new task. In contrast, models with better *plasticity* can adapt quickly to the new task but tend to forget old ones. Existing CL methods can generally be divided into three categories: data replay, parameter separation and parameter regularization.

The core idea of *data replay* methods [Chaudhry *et al.*, 2018] is to store and replay raw data from old tasks to mitigate forgetting. However, storing old data directly violates privacy and storage restrictions. *Parameter separation* methods [Hung *et al.*, 2019] overcome catastrophic forgetting by assigning different subsets of the model parameters to dealing with each task. However, separation methods will result in an infinite increase of parameters with the arrival of new tasks, which can quickly become unacceptable in FL.

The regularization-based methods limit the updating process by punish the updates on important parameters [Kirkpatrick *et al.*, 2017] or adding knowledge distillation [Hinton *et al.*, 2015] loss to the objective function [Li and Hoiem, 2017; Lee *et al.*, 2019; Zhang *et al.*, 2020] to learn the knowledge from the old model. However, the importance is difficult to be precisely evaluated. Some distillation-based methods perform distillation based on the new task data, but its efficacy drops significantly when domains vary greatly. The others that leverage unlabeled external data solve difficulties above. CFeD adopts knowledge distillation with public datasets and proposes a client division mechanism to reduce the cost of time and computation.

### 2.2 Federated Learning

Recent studies have considered introducing Continual Learning into FL to improve the performance of models under Non-IID data [Shoham *et al.*, 2019]. [Yoon *et al.*, 2021] proposed Federated Continual Learning and focused on multiple continual learning agents that use each other's indirect experience to enhance the continual learning performance of their local models, rather than to jointly train a better global model. Therefore, the purpose of their study is to obtain a collection of local models for the participating clients. While our work looks literally similar, our research problem is very different, as we aim at learning a better global model. Thus we decide not to compare with it in our experiment study.

Based on FedAvg, [Jeong *et al.*, 2018] proposed Federated Distillation framework to enhance communication efficiency without comprising performance. There are also several works leveraging additional datasets constructed from public dataset [Li and Wang, 2019; Lin *et al.*, 2020] or original local data [Itahara *et al.*, 2020] to aid distillation. Compared with the above works, our proposed method to mitigate catastrophic forgetting is orthogonal and could work with them together.

## 3 Problem Definition

In FL, a central server and $K$ clients cooperate to train a model on task $\mathcal{T}$ through $R$ rounds of communication. The optimization objective can be written as follows:

$$\min_{\boldsymbol{\theta}} \sum_{k=1}^{K} \frac{n_k}{n_a} \mathcal{L}(\mathcal{T}^k; \boldsymbol{\theta}) \tag{1}$$

where $\mathcal{T}^k$ refers to a collection of $n_k$ training samples at $k$-th client and $n_a$ is the sum of all $n_k$.

Here we define *Continual Federated Learning* (CFL), which aims to train a global model via iterative communication between the server and clients on a series of tasks $\{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \cdots\}$ accumulated at the client-side.

When the $t$-th task arrives, data of previous tasks will become unavailable for training. We define the global model parameters obtained from the previous task as $\boldsymbol{\theta}_{t-1}$, and the new task as $\mathcal{T}_t = \bigcup_{k=1}^{K} \mathcal{T}_t^k$, where $\mathcal{T}_t^k$ contains newly collected data at each client.

The goal is to train a global model with a minimized loss on the new task $\mathcal{T}_t$ as well as old tasks $\{\mathcal{T}_1, \cdots, \mathcal{T}_{t-1}\}$. The optimization objective is achieved by minimizing the losses of $K$ clients on all local tasks up to time $t$ through iterative server-client communication. The global model parameters can be obtained as follows:

$$\boldsymbol{\theta}_t = \arg\min_{\boldsymbol{\theta}} \sum_{k=1}^{K} \frac{n_k}{n_a} \sum_{i=1}^{t} \mathcal{L}(\mathcal{T}_i^k; \boldsymbol{\theta}) \tag{2}$$

The global model at task $\mathcal{T}_t$ is expected to achieve no higher loss on historical tasks than that at time $t-1$. That is, $\sum_{i=1}^{t-1} \mathcal{L}(\mathcal{T}_i; \boldsymbol{\theta}_t) \leq \sum_{i=1}^{t-1} \mathcal{L}(\mathcal{T}_i; \boldsymbol{\theta}_{t-1})$.

However, in real-world scenarios, due to the limitation on accessing previous data, CFL suffers catastrophic forgetting at "intra-task" and "inter-task" levels. Formally, intra-task forgetting means that after the updates of $r$-th round, the global model gets a higher loss than the ($r$-1)-th round on the local dataset of $k$-th client: $\mathcal{L}(\mathcal{T}_t^k; \boldsymbol{\theta}_{t,r}) > \mathcal{L}(\mathcal{T}_t^k; \boldsymbol{\theta}_{t,r-1})$, especially when the distribution across clients is Non-IID. And then, inter-task forgetting is that the loss of the model at time $t$ on old tasks is higher than that at time $t-1$: $\sum_{i=1}^{t-1} \mathcal{L}(\mathcal{T}_i^k; \boldsymbol{\theta}_t) > \sum_{i=1}^{t-1} \mathcal{L}(\mathcal{T}_i^k; \boldsymbol{\theta}_{t-1})$.

# 4 Proposed Method

To tackle the catastrophic forgetting in FL, we propose a framework named CFeD (**C**ontinual **Fe**derated learning with **D**istillation). As shown in Figure 2, the core idea is to use the model of the last task to predict the surrogate dataset, and treat the outputs as pseudo-labels to perform knowledge distillation to review the knowledge of unavailable data. To improve the learning ability of the global model and fully utilize computation resources, learning the new task and reviewing old tasks can be assigned to different clients and those clients without enough new task data could only review the old tasks. Moreover, a server distillation mechanism is proposed to mitigate the intra-task forgetting in Non-IID data. The aggregated global model is finetuned to mimic the outputs of the global model on the last round and local models on the current round.

The surrogate dataset should be collected from public datasets for privacy and cover as many features as possible or be similar to the old tasks to ensure the effectiveness of distillation. Since the model parameters do not increase, there is no additional communication cost compared with FedAvg.

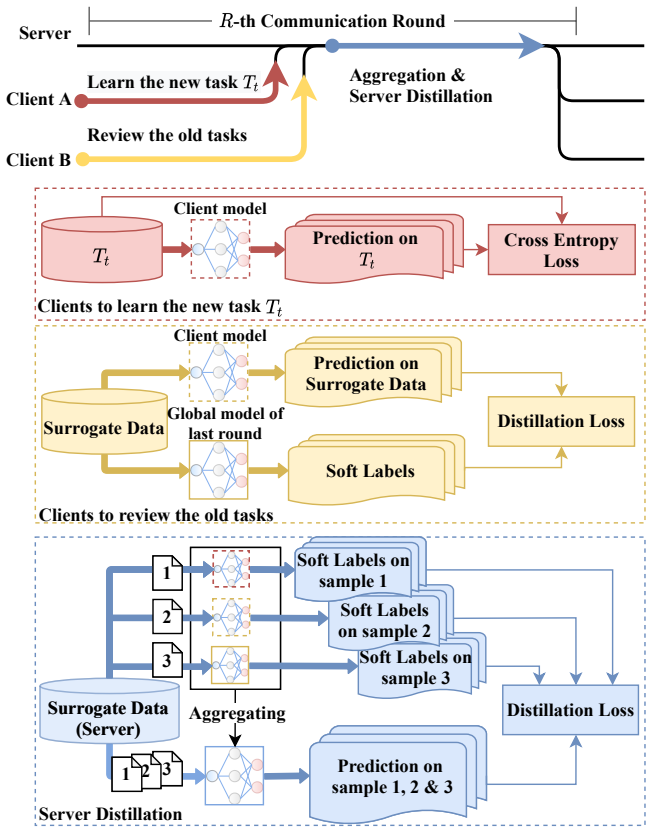


Figure 2: Continual federated learning with knowledge distillation.

## 4.1 Clients Distillation

The distillation process of CFeD assumes that there is a surrogate dataset $X_s^k$ at the $k$-th client. For each sample $\mathbf{x}_s \in X_s^k$, its label at time $t$ is $\mathbf{y}_{s,t} = f(\mathbf{x}_s; \boldsymbol{\theta}_{t-1})$. Thus we obtain a per-client set of sample pairs $\mathcal{S}_t^k = \{(\mathbf{x}_s, \mathbf{y}_{s,t}), \forall \mathbf{x}_s \in X_s^k\}$. A distillation term $\mathcal{L}_d(\mathcal{S}_t^k; \boldsymbol{\theta})$ is added to approximate the loss on old tasks $\sum_{i=1}^{t-1} \mathcal{L}(\mathcal{T}_i^k; \boldsymbol{\theta})$. For a specific surrogate sample pair $(\mathbf{x}_s, \mathbf{y}_{s,t})$ at time $t$, the distillation loss can be formalized as a modified version of cross-entropy loss:

$$\mathcal{L}_d((\mathbf{x}_s, \mathbf{y}_{s,t}); \boldsymbol{\theta}) = -\sum_{i=1}^{l} y_{s,t}'^{(i)} \log \hat{y}_{s,t}'^{(i)} \tag{3}$$

where $l$ is the number of target classes, $y_{s,t}'^{(i)}$ is the modified surrogate label, and $\hat{y}_{s,t}'^{(i)}$ is the modified output of the model on surrogate sample $\mathbf{x}_s$. The latter two are defined as:

$$y_{s,t}'^{(i)} = \frac{(y_{s,t}^{(i)})^{1/T}}{\sum_{j=1}^{l} (y_{s,t}^{(j)})^{1/T}}, \quad \hat{y}_{s,t}'^{(i)} = \frac{(\hat{y}_{s,t}^{(i)})^{1/T}}{\sum_{j=1}^{l} (\hat{y}_{s,t}^{(j)})^{1/T}} \tag{4}$$

where $\hat{y}_{s,t}^{(i)}$ is the $i$-th element of $f(\mathbf{x}_s; \boldsymbol{\theta})$, $T$ is the temperature of distillation and a greater $T$ value could amplify minor logits so as to increase the information provided by the teacher model.

Based on the above notions, CFeD computes the total loss on all clients at time $t$ by substituting the unknown losses on

old tasks in Equation 2 with the distillation losses on the per-client surrogate datasets. Formally, the total loss is:

$$\begin{aligned} &\sum_{k=1}^{K} \frac{n_k}{n_a}(\mathcal{L}(\mathcal{T}_t^k;\boldsymbol{\theta}) + \sum_{i=1}^{t-1}\mathcal{L}(\mathcal{T}_i^k;\boldsymbol{\theta})) \\ \propto &\sum_{k=1}^{K} \frac{n_k}{n_a}(\mathcal{L}(\mathcal{T}_t^k;\boldsymbol{\theta}) + \lambda_d \mathcal{L}_d(\mathcal{S}_t^k;\boldsymbol{\theta})) \end{aligned} \tag{5}$$

where $\lambda_d$ (default 0.1) is a pre-defined parameter to weight the distillation loss.

### 4.2 Clients Division Mechanism

In real-world scenarios, different clients accumulate data at different speeds. While some clients are ready to learn a new task, others may not have gained enough data for the new task and thus cannot be effectively utilized in learning it.

To leverage the under-utilized computation resources, CFeD treats *learning the new task* and *reviewing old tasks* as two individual objectives. The framework assigns one of the two objectives to each client so that some clients can only perform reviewing. Further, this kind of division may improve the exploration of the model on different objectives and help the model depart from previous local minima. Formally, regarding the division mechanism, Equation 5 can be expanded and rewritten as:

$$\sum_{k=1}^{N} \frac{n_k}{n_a}\mathcal{L}_d(\mathcal{S}_t^k;\boldsymbol{\theta}) + \sum_{k=N+1}^{K} \frac{n_k}{n_a}\mathcal{L}(\mathcal{T}_t^k;\boldsymbol{\theta}) \tag{6}$$

where $N = \lceil \alpha K \rceil, \alpha \in [0, 1]$. We introduce a factor $\alpha$ to describe the proportion of clients involved in reviewing per round.

### 4.3 Server Distillation

Although the clients division mechanism allows spare computing resources to be utilized, we must note that some active clients are not learning the new task now. This may harm the performance on the new task and lead to severe intra-task forgetting, especially in the Non-IID scenario. The reason is that, since the labels of the data on different clients may not intersect each other, the global model fitting one client (learning the new task) may easily exhibit forgetting in another (reviewing the old tasks, or learning the new task on data of different labels).

To mitigate such performance degradation on the new task, a client may take a naive solution to increase its local training iterations. However, increasing the number of epochs of local updates on Non-IID data could easily cause overfitting and unstablize the performance of the global model.

Motivated by the paradigm of mini-batch iterative updates in centralized training [Toneva *et al.*, 2018], we propose *Server Distillation* (SD) to mitigate the intra-task forgetting and stabilize the performance of the aggregated model at the server-side. In our approach, the server also maintains a lightweight, unlabeled public dataset $X_s$, similar to the surrogate datasets on the clients. After the $r$-th round of aggregating the $K$ local models collected from the clients, the server divides $X_s$ into $K + 1$ batches, assigns $K$ of them to the received local models and one to the global model of the last round, and collects outputs $\hat{Y}_{s,r}$ of above-mentioned models on their assigned batches to construct a labeled set of sample pairs for the server distillation, denoted as $\mathcal{S}_r = \{(\mathbf{x}_s, \mathbf{y}_{s,r}), \forall \mathbf{x}_s \in X_s\}$. Next, the aggregated global model $\boldsymbol{\theta}_r$ will be iteratively updated with a distillation loss $\mathcal{L}_d(\mathcal{S}_r;\boldsymbol{\theta})$.

With server distillation, the global model is able to further retrieve the knowledge from the local models and the previous global model, thus to mitigate intra-task forgetting.

## 5 Experiments

In this section, we evaluate CFeD and baseline approaches extended from traditional continual learning methods on text and image classification tasks.

### 5.1 Datasets and Tasks

We consider both text and image classification datasets: **THUCNews** [Li *et al.*, 2006] contains 14 categories of Chinese news data collected from Sina News RSS between 2005 and 2011. **SogouCS** [SogouLabs, 2012] contains 18 categories of 511218 Chinese news data in 2012. **Sina2019** contains 30533 Chinese news data in 2019 crawled from the Sina News by ourselves. **NLPIR Weibo Corpus** [NLPIR, 2017] consists of 230000 samples obtained from Sina Weibo and Tencent Weibo, two Chinese social media sites. We use it as a surrogate dataset across different tasks. **CIFAR-10**[Krizhevsky, 2009] contains 60000 images with 10 classes. **CIFAR-100**[Krizhevsky, 2009] contains 60000 images with 100 classes. **Caltech-256**[Griffin *et al.*, 2007] contains 30608 images with 256 classes as the surrogate dataset in image classification. [1]

We design task sequences to be learned in two different scenarios: Domain-IL indicates the case where the input distributions continually vary in the sequence; Class-IL indicates the case where new classes incrementally emerge in the sequence. Tasks on the text dataset are denoted by 'Tx' while those on the image dataset are denoted by 'Ix', where 'x' is the task sequence ID. Most task sequences in the experiments are short, containing two or three classification tasks.

### 5.2 Compared Methods

We choose the following approaches for evaluation:

(1) **Finetuning:** A naive method that trains the model on tasks sequentially. (2) **FedAvg:** A FL method that each client learns tasks in sequence and the server aggregates local models from clients. (3) **MultiHead:** A CL method training individual classifiers for each task, requiring task labels to specify the output during the inferring phase. **FedMultiHead** denotes FedAvg with MultiHead applied to clients. (4) **EWC:** a regularization-based method [Kirkpatrick *et al.*, 2017] that uses Fisher information matrix to estimate the importance of parameters. **FedEWC** denotes FedAvg with EWC applied to clients. (5) **LwF:** A distillation-based method. Instead of unlabeled data, LwF leverages new task data to perform

[1] Our code and datasets are publicly available at https://github.com/lianziqt/CFeD.

| | **Domain-IL scenario** | | **Class-IL scenario** | | | |
|---|---|---|---|---|---|---|
| | Domain-T1 | Domain-T2 | Class-T1 | Class-T2 | Class-I1 | Class-I2 |
| MultiHead* | 94.66 | 93.40 | 95.84 | 95.66 | 51.81 | 57.04 |
| Finetuing | 85.96 | 91.32 | 48.00 | 32.43 | 36.74 | 30.58 |
| EWC | 87.42 | 91.98 | 47.96 | 32.42 | 36.35 | **30.72** |
| LwF | 90.58 | 92.01 | 48.59 | 39.10 | 35.31 | 26.32 |
| DMC | - | - | 48.37 | 41.92 | 37.81 | 27.37 |
| CFeD(C)$_{\text{lr=1e-6}}$ | 82.48 | 83.20 | **71.21** | **63.45** | **37.88** | 28.36 |
| CFeD(C)$_{\text{lr=1e-3}}$ | **94.49** | **92.95** | 62.22 | 34.26 | 33.75 | 22.21 |
| FedMultiHead* | 81.83 | 91.04 | 96.26 | 96.07 | 56.97 | 60.43 |
| FedAvg | 86.50 | 92.60 | 48.25 | 32.61 | 32.53 | 29.28 |
| FedEWC | 84.76 | 92.06 | 48.24 | 32.51 | 30.37 | 28.62 |
| FedLwF | 87.35 | 92.41 | 59.39 | 44.76 | 33.97 | 23.46 |
| FedDMC | - | - | 46.58 | 10.46 | 16.50 | 8.33 |
| FedDMC$_{\text{full}}$ | - | - | 56.95 | 50.87 | 40.13 | 29.18 |
| CFeD | **92.34** | **94.15** | **85.81** | **83.80** | **40.51** | **32.33** |

Table 1: The *average accuracy on learned tasks* (%) of different methods (global models for FL). The top 7 rows are from centralized methods, while the bottom 7 rows are from FL methods. $*$ indicates methods with additional information (task labels). DMC is only suitable for the Class-IL scenario.

distillation. FedLwF denotes FedAvg with LwF applied to clients. (6) **DMC:** Deep Model Consolidation [Zhang *et al.*, 2020], a Class-IL CL method that first trains a separate model only for the new task, and then leverages an unlabeled public dataset to distill the knowledge of the old tasks and the new task to obtain a new combined model. **FedDMC** denotes FedAvg with DMC applied to clients. (7) **CFeD:** our method and **CFeD(C)** denotes the centralized version of our method CFeD.

Note that MultiHead and FedMultiHead require task labels during inference to know which task the current input belongs to. Moreover, multiple classifiers inevitably bring more parameters in the Domain-IL scenario. Owing to these additional information, their performance can be seen as a target for other methods.

## 5.3 Experimental Settings

We use TextCNN [Kim, 2014] or ResNet-18 [He *et al.*, 2016] followed by fully-connected layers for classification. Each task trains the model for $R = 20$ rounds. For the local updating in each client, the learning epoch is 10 in Domain-IL or 40 in Class-IL. Unless otherwise stated, the constraint factor $\lambda$ of the EWC method is set to 100000. The temperature of distillation is set to 2 as default.

For the configuration of FL, we assume that there are 100 clients, and only random $10\%$ clients are sampled to participate in each training round. The training dataset and surrogate dataset are both divided into 200 shards randomly (IID) or sorted by the category (Non-IID). In each experiment, every client selects two shards of data on each task as the local dataset and also two shards of the surrogate dataset as the local surrogate. In particular, the server also selects two shards for server distillation in the Non-IID distribution. All above selections are conducted randomly.

For each task, we select 70% of data as the training set, $10\%$ as the validation set and the rest as the test set. The global model is evaluated on the test set at the end of each training round. All experiments are repeated for 3 runs with different random seeds.

## 5.4 Experimental Results

**Effect on Mitigating Inter-task Forgetting**

We evaluate all methods in both centralized and FL scenarios. Table 1 summarizes the average accuracy on ever learned task after the model learns the second and third tasks sequentially, both in Domain-IL (left) and Class-IL (right). By comparing the results of different methods, it can be seen that CFeD exceeds other baselines on average accuracy.

In Domain-IL scenario (left half of Table 1), CFeD exceeds FedAvg, FedLwF, FedEWC and FedDMC methods on the average accuracy, being close to FedMultiHead. Moreover, the average accuracy of all methods improves after the model continually learns the Domain-T2. The result implies that the new task of Domain-T2 may cover some features of old tasks, which help the models review the old knowledge, and notably, CFeD still outperforms the other baselines.

In the Class-IL scenario (right half of Table 1), we can see that the average accuracy of FedAvg and FedEWC both drop significantly. The reason is that the labels of the old task are not available, and the model quickly overfits the new task. In contrast, CFeD outperforms other baselines, indicating the benefit of leveraging the surrogate dataset to get reasonable soft labels for old tasks.

We notice that the performance of FedDMC drops significantly in Class-IL. Since the model consolidation of DMC only uses surrogate data for distillation, i.e. no new task data, to learn new tasks and review old tasks, its performance is significantly limited by the surrogate dataset size (in our case, each client only has 2300 surrogate samples). To verify this, we construct a variant FedDMC$_{\text{full}}$ where every client has access to the entire surrogate dataset. Under such a setting, FedDMC$_{\text{full}}$ achieves considerable improvement as each

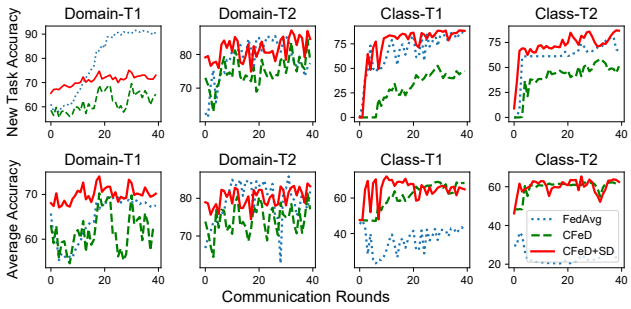


Figure 3: The performance of FedAVG(blue), CFeD(green) and CFeD+SD(red) on both the Domain-IL and Class-IL scenarios under Non-IID distribution

| | Class-T1 | | Class-T2 | |
|---|---|---|---|---|
| | Avg | New | Avg | New |
| $\text{CFeD(C)}_{\text{lr=1e-6}}$ | 71.21 | 59.40 | 63.45 | 66.60 |
| $\text{CFeD(C)}_{\text{lr=1e-3}}$ | 62.22 | 93.90 | 34.26 | 93.34 |
| $\text{CFeD}_{\text{lr=1e-6}}$ | **85.81** | **95.30** | **83.80** | **97.22** |

Table 2: The *average accuracy on learned tasks* (Avg, %) and *test accuracy on the new task* (New, %) under different learning rates.

client has more data. However, our approach still outperforms it, showing the robustness of CFeD to the surrogate dataset size.

**Effect on Mitigating Intra-task Forgetting**

To illustrate the effect of our proposed approach against intra-task forgetting, we compare three methods: FedAVG, CFeD, and CFeD with Server Distillation (namely CFeD+SD) both in Domain-IL and Class-IL scenarios with Non-IID distribution. Figure 3 shows the accuracy on new tasks and the average accuracy on learned tasks of the model during the learning process. The results show that CFeD+SD improves the performance on mitigating without sacrificing the ability to learn the new task. Moreover, the performance of all methods in Non-IID distribution degrades significantly, but CFeD+SD is more stable than the other two methods. Owing to clients division and server distillation, CFeD+SD achieves higher average accuracy without sacrificing plasticity.

**Clients Division Mechanism**

To evaluate the effect of the clients division mechanism, Table 2 shows more detailed results of both the accuracy on new tasks and the average accuracy to illustrate the trade-off of CFeD between stability and plasticity (See Section 2.1). It can be seen that, in Class-IL, CFeD(C) also suffers the dilemma between plasticity and stability: $\text{CFeD(C)}_{\text{lr=1e-6}}$ cannot learn well on the new tasks and $\text{CFeD(C)}_{\text{lr=1e-3}}$ performs poorly on the average accuracy. In contrast, CFeD strikes a good balance between plasticity and stability owing to the clients division mechanism.

**Varying Surrogate Data Size**

To see how the surrogate data size affects the performance of CFeD, we vary the number of shards ($\beta$) of surrogate data

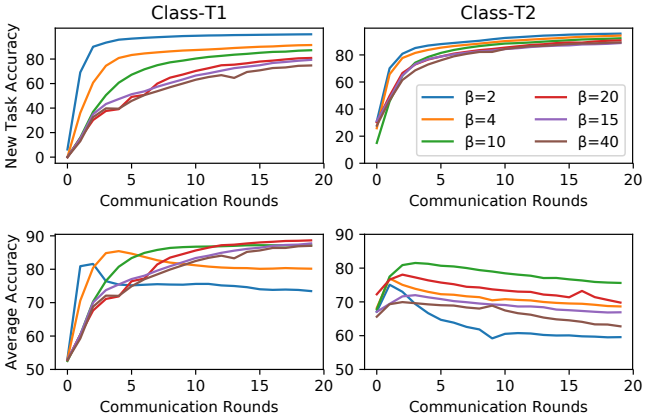


Figure 4: Results of varying surrogate-ratio ($\beta$ denotes the number of selected shards)

assigned to each client from 2 (default) to 40. To reduce the experiment time, we set the local number of epochs to 10.

The experiment results are shown in Figure 4. Generally, the performance on new tasks reduces when $\beta$ increases. However, it is not the case for the old tasks. When varying $\beta$ from 2 to 40, the performance on old tasks improves first and then decreases. The optimal value is around 10 for T1 and 20 for T2. It is worth noting that when $\beta$ is small, the average accuracy of both tasks reaches a peak value and then diminishes slowly as learning proceeds (bottom left subfigure). This indicates that the model learns the new task quickly (reaching the peak) and then gradually forgets the old tasks, which offsets the performance gained from the new task. The forgetting is apparently postponed in task sequence T1 when we enlarge $\beta$. But the effect of postponing is not obvious due to the large number of tasks in T2.

## 6 Conclusions

In this paper, we tackle the problem of catastrophic forgetting in federated learning of a series of tasks. We proposed a Continual Federated Learning framework named CFeD, which leverages knowledge distillation based on surrogate datasets, to address the problem. Our approach allows clients to review the knowledge learned in the old model by optimizing the distillation loss based on their own surrogate datasets. The server also performs distillation to mitigate intra-task forgetting. To further improve the learning ability of the model, the clients could be assigned to either learning the new task or reviewing the old tasks separately. The experiment results showed that our proposed approach outperforms baselines in mitigating catastrophic forgetting and achieves a good trade-off between stability and plasticity. For future work, we will further enhance our approach to overcome the intra-task forgetting in Non-IID data and reduce its training costs.

## Acknowledgments

This work was supported by the Key Research and Development Program of Zhejiang Province of China (No. 2021C01009), the National Natural Science Foundation of

China (Grant No. 62050099), and the Fundamental Research Funds for the Central Universities.

## References

[Chaudhry *et al.*, 2018] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018.

[Delange *et al.*, 2021] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021.

[Griffin *et al.*, 2007] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. Technical report, California Institute of Technology, 2007.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Hinton *et al.*, 2015] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

[Hung *et al.*, 2019] Ching-Yi Hung, Cheng-Hao Tu, Cheng-En Wu, Chien-Hung Chen, Yi-Ming Chan, and Chu-Song Chen. Compacting, picking and growing for unforgetting continual learning. In *Advances in Neural Information Processing Systems*, pages 13647–13657, 2019.

[Itahara *et al.*, 2020] Sohei Itahara, Takayuki Nishio, Yusuke Koda, Masahiro Morikura, and Koji Yamamoto. Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data. *arXiv preprint arXiv:2008.06180*, 2020.

[Jeong *et al.*, 2018] Eunjeong Jeong, Seungeun Oh, Hyesung Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *arXiv preprint arXiv:1811.11479*, 2018.

[Kim, 2014] Yoon Kim. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882, 2014.

[Kirkpatrick *et al.*, 2017] 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.

[Krizhevsky, 2009] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[Lee *et al.*, 2019] Kibok Lee, Kimin Lee, Jinwoo Shin, and Honglak Lee. Overcoming catastrophic forgetting with unlabeled data in the wild. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 312–321, 2019.

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

[Li and Wang, 2019] Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*, 2019.

[Li *et al.*, 2006] Jingyang Li, Maosong Sun, and Xian Zhang. A comparison and semi-quantitative analysis of words and character-bigrams as features in chinese text categorization. In *Proceedings of the International Conference on ACL*, pages 545–552, 2006.

[Lin *et al.*, 2020] Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems*, 33:2351–2363, 2020.

[McMahan *et al.*, 2017] 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.

[NLPIR, 2017] NLPIR. Nlpir weibo corpus. http://www.nlpir.org/, 2017. Accessed: 2017-12-03.

[Shoham *et al.*, 2019] Neta Shoham, Tomer Avidor, Aviv Keren, Nadav Israel, Daniel Benditkis, Liron Mor-Yosef, and Itai Zeitak. Overcoming forgetting in federated learning on non-iid data. *arXiv preprint arXiv:1910.07796*, 2019.

[SogouLabs, 2012] SogouLabs. Sohu news data(sogoucs). http://www.sogou.com/labs/resource/cs.php, 2012. Accessed: 2020-06-22.

[Toneva *et al.*, 2018] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*, 2018.

[Usmanova *et al.*, 2021] 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.

[Yoon *et al.*, 2021] 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.

[Zhang *et al.*, 2020] 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 *The IEEE Winter Conference on Applications of Computer Vision*, pages 1131–1140, 2020.