Federated Continual Learning with Weighted Inter-client Transfer

Anonymous Authors[1]

Abstract

There has been a surge of interest in continual learning and federated learning, both of which are important in deep neural networks in real-world scenarios. Yet little research has been done regarding the scenario where each client learns on a sequence of tasks from private local data stream. This problem of *federated continual learning* poses new challenges to continual learning, such as utilizing knowledge from other clients, while preventing interference from irrelevant knowledge. To resolve these issues, we propose a novel federated continual learning framework, *Weighted Inter-client Transfer (FedWeIT)*, which decomposes the network weights into global federated parameters and sparse task-specific parameters, and each client receives selective knowledge from other clients by taking a weighted combination of their task-specific parameters. *FedWeIT* minimizes interference between incompatible tasks, and also allows positive knowledge transfer across clients during learning. We validate our *FedWeIT* against existing federated learning and continual learning methods under varying degree of task similarity across clients, and our model significantly outperforms them with large reduction in the communication cost.

1. Introduction

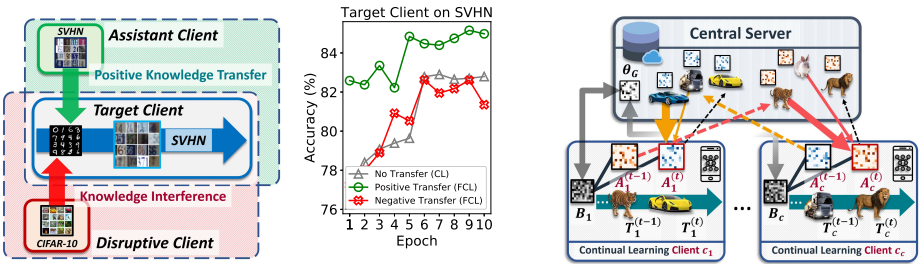
Continual learning (1; 2; 3; 4; 5) describes a learning scenario where a model continuously trains on a sequence of tasks; it is inspired by the human learning process, as a person learns to perform numerous tasks with large diversity over his/her lifespan, making use of the past knowledge to learn about new tasks without forgetting previously learned ones. Continual learning is a long-studied topic since having such an ability leads to the potential of building a general artificial intelligence. However, there are crucial challenges in implementing it with conventional models such as deep neural networks (DNNs), such as *catastrophic forgetting*, which describes the problem where parameters or semantic representations learned for the past tasks drift to the direction of new tasks during training. The problem has been tackled by various prior work (4; 6; 7; 8). More recent works tackle other issues, such as scalability or order-robustness (5; 9).

However, all of these models are fundamentally limited in that the models can only learn from its direct experience - they only learn from the sequence of the tasks they have trained on. Contrarily, humans can learn from *indirect experience* from others, through different means (e.g. verbal communications, books, or various media). Then wouldn't it be beneficial to implement such an ability to a continual learning framework, such that multiple models learning on different machines can learn from the knowledge of the tasks that have been already experienced by other clients? One problem that arises here, is that due to data privacy and communication cost, it may not be possible to communicate data directly between the clients or between the server and clients. Federated learning (10; 11; 12; 13) is a learning paradigm that tackles this issue by communicating the parameters instead of the raw data itself. We may have a server that receives the parameters locally trained on multiple clients, aggregates it into a single model parameter, and sends it back to the clients. Motivated by our intuition on learning from indirect experience, we tackle the problem of *Federated Continual Learning (FCL)* where we perform continual learning with multiple clients trained on private task sequences, which communicate their task-specific parameters via a global server.

Yet, the problem of federated continual learning also brings new challenges. First, there is not only the catastrophic forgetting from continual learning, but also the **threat of potential interference from other clients**. Figure 1 (a) describes this challenge with the results of a simple experiment. Here, we train a model for MNIST digit recognition while communicating the parameters from another client trained on a different dataset. When the knowledge transferred from the other client is relevant to the target task (SVHN), the model starts with high accuracy, converge faster and reach higher accuracy (**green line**), whereas the model underper-

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.



(a) Challenges of Federated Continual Learning (b) Weighted Inter-client Transfer

Figure 1. **(a): Challenge of FCL**. Knowledge interference from other clients hinder optimal training of target clients (*Red*) while positive experience from other clients is beneficial (*Green*). **(b): Overview of FedWeIT**. Each client continuously learns on a private task sequence with inter-client knowledge transfer as selectively utilizing the encoded knowledge of tasks learned at other clients.

forms the base model if the transferred knowledge is from a task highly different from the target task (CIFAR-10, **red line**). Thus, we need to *selective utilize* knowledge from other clients to minimize the *inter-client interference* and maximize *inter-client knowledge transfer*. Another problem with the federated learning is efficient communication, as communication cost could become excessively large when utilizing the knowledge of the other clients, since the communication cost could be the main bottleneck in practical scenarios when working with edge devices. Thus we want the knowledge to be represented as compactly as possible.

To tackle these challenges, we propose a novel framework for federated continual learning, *Federated Weighted Inter-client Transfer (FedWeIT)*, which decomposes the model parameters into a dense global parameter and sparse task-adaptive parameters. As illustrated in Figure 1 (b), FedWeIT reduces the interference between different tasks since the global parameters ($\boldsymbol{\theta}_G$) will encode task-generic knowledge, while the task-specific knowledge will be encoded into the task-adaptive parameters ($\mathbf{A}_c^{(t)}$). When we utilize the generic knowledge, we also want the client to selectively utilize task-specific knowledge obtained at other clients. To this end, we allow each model to take a weighted combination of the task-adaptive parameters broadcast from the server, such that it can select task-specific knowledge helpful for the task at hand. FedWeIT is communication-efficient, since the task-adaptive parameters are *highly sparse* and only need to be communicated once when created. We also perform selective transmission of the parameters to further reduce communication cost.

We validate our method on multiple different scenarios with varying degree of task similarity across clients against various federated learning and local continual learning models. The results show that our model obtains significantly superior performance over all baselines, adapts faster to new tasks, with largely reduced communication cost.

The main contributions of this paper are as follows:

- We introduce **a new problem of Federated Continual Learning (FCL)**, where multiple models continuously learn on distributed clients, which poses new challenges such as prevention of inter-client interference and inter-client knowledge transfer.
- We propose **a novel framework for federated continual learning**, which allows each client to adaptively update the federated parameter and utilize the past knowledge from other clients, by communicating sparse parameters.
- We validate our model under FCL setting with both Overlapped and non-IID task sequences, on which it **largely outperforms** existing federated learning and local continual learning approaches with **significantly reduced communication cost**.

2. Related Work

Continual Learning While continual learning (1; 2; 3) is a long-studied topic with a vast literature, we only discuss recent relevant works. **Regularization-based approaches:** A popular approach for continual learning is to use regularizations that prevent catastrophic forgetting. EWC (4) leverages Fisher Information Matrix to restrict the change of the model parameters such that the model finds solution that is good for both previous and the current task, and IMM (6) proposes to learn the posterior distribution for multiple tasks as a mixture of Gaussians. **Architecture-based approaches:** PGN (14) progressively expands the networks with fixed number of neurons/filters at each layer. DEN (15) tackles this issue by expanding the networks size with minimum number of neurons/filters that are necessary via iterative neuron/filter pruning and splitting, and RCL (16) tackles the same problem using reinforcement learning. APD (9) additively decompose the parameters into shared and task-specific parameters to minimize the increase in the network complexity. **Coreset-based approaches:** VCL (17) performs online variational inference by continuously training

the model while approximating the likelihood for the coreset, and GEM variants (18; 19) minimize the loss on both of actual dataset and stored episodic memory. FRCL (20) memorizes approximated posteriors of previous tasks with sophisticatedly constructed inducing points. To the best of our knowledge, none of the existing approaches considered the communicability for continual learning of deep neural networks, which we tackle. CoLLA (21) aims at solving multi-agent lifelong learning with sparse dictionary learning, but it is not applicable to federated learning or continual deep learning.

Federated learning Federated Learning is a distributed learning framework under differential privacy, which aims to learn a global model on a server while aggregating the parameters learned at the clients on their private data. There are diverse approaches to aggregate the local models. FedAvg (10) aggregates the model trained across multiple clients by computing a weighted average of them based on the number of data points trained. TWAFL (11) and ASO-fed (22) follow weighted averaging in FedAvg while assigning larger weights to newer parameters by leveraging timestamps. FedProx (12) trains the local models with a proximal term which restricts their updates to be close to the global model. FedCurv (23) aims to minimize the model disparity across clients during federated learning by adopting a modified version of EWC (4). Recent works (13; 24) introduce well-designed aggregation policies by leveraging Bayesian non-parametric methods. Another crucial challenge is the reduction of communication cost, as communicating the full network weights may be too costly. TWAFL (11) tackles this problem by performing layer-wise parameter aggregation, where some layers(i.e. shallow layers) are aggregated at every step, but other layers(i.e. deep layers) are aggregated in the last few steps of a loop. Our method also solves the problem of efficient communication by performing weighted inter-client knowledge transfer.

3. Federated Continual Learning with Weighted Inter-client Transfer

Motivated by the human learning process from indirect experiences, we introduce a novel continual learning under federated learning setting, which we refer to as *Federated Continual Learning (FCL)*. FCL assumes that multiple clients are trained on a sequence of tasks from private data stream, while communicating the learned parameters with a global server. We first formally define the problem in *Section* 3.1, and then propose naive solutions that straightforwardly combine the existing federated learning and continual learning methods in *Section* 3.2. Then, following *Section* 3.3 and 3.4, we discuss about two novel challenges that are introduced by federated continual learning, and propose a novel framework, *Federated Weighted Inter-client Transfer (FedWeIT)* which can effectively handle the two problems while also reducing the client-to-sever communication cost.

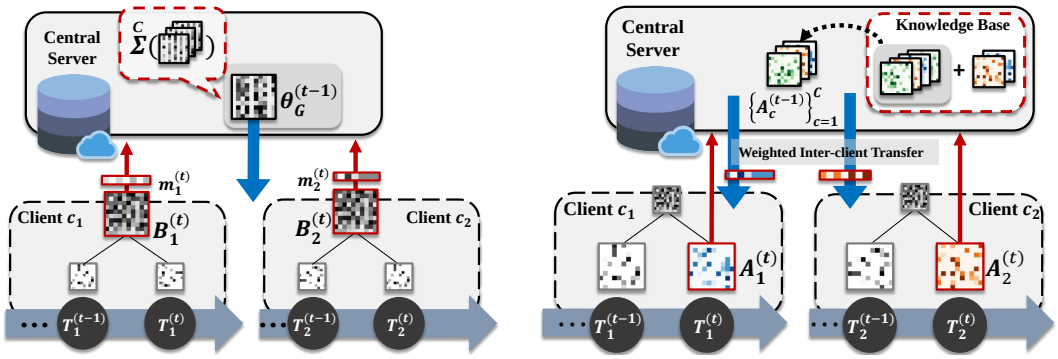
3.1. Problem Definition

In the standard continual learning (on a single machine), the model iteratively learns from a sequence of tasks $\{\mathcal{T}^{(1)}, \mathcal{T}^{(2)}, ..., \mathcal{T}^{(T)}\}$ where $\mathcal{T}^{(t)}$ is a labeled dataset of tth task, $\mathcal{T}^{(t)} = \{\mathbf{x}_i^{(t)}, \mathbf{y}_i^{(t)}\}_{i=1}^{N_t}$, which consists of N_t pairs of instances $\mathbf{x}_i^{(t)}$ and their corresponding labels $\mathbf{y}_i^{(t)}$. Assuming the most realistic situation, we consider the case where the task sequence is a task stream with an unknown arriving order, such that the model is allowed to access $\mathcal{T}^{(t)}$ only at the training period of task t which becomes inaccessible afterwards. Given $\mathcal{T}^{(t)}$ and the model learned so far, the learning objective at task t is as follows: $\text{minimize}_{\boldsymbol{\theta}^{(t)}}\ \mathcal{L}(\boldsymbol{\theta}^{(t)}; \boldsymbol{\theta}^{(t-1)}, \mathcal{T}^{(t)})$, where $\boldsymbol{\theta}^{(t)} \in \mathbb{R}^{N \times M}$ is a set of the parameters in the model at task t.

We now extend the conventional continual learning to the federated learning setting with multiple clients and a global server. Let us assume that we have C clients, where at each client $c_c \in \{c_1, \dots, c_C\}$ trains a model on a *privately accessible* sequence of tasks $\{\mathcal{T}_c^{(1)}, \mathcal{T}_c^{(2)}, ..., \mathcal{T}_c^{(t)}\} \subseteq \mathcal{T}$. Now the goal is to effectively train C continual learning models on their own private task streams, via communicating the model parameters with the global server, which aggregates the parameters sent from each client, and redistributes them to clients.

3.2. Communicable Continual Learning

In conventional federated learning settings, the learning is done with multiple rounds of local learning and parameter aggregation. At each round of communication r, each client c_c and the server s perform the following two procedures: *local parameter transmission* and *parameter aggregation & broadcasting*. In the local parameter transmission step, for a randomly selected subset of clients at round r, $\mathcal{C}^{(r)} \subseteq \{c_1, c_2, ..., c_C\}$, each client c_c sends updated parameters $\boldsymbol{\theta}^{(r)}$ to the server. The server-clients transmission is not done at every client because some of the clients may be temporarily disconnected. Then the server aggregates the parameters $\boldsymbol{\theta}_c^{(r)}$ sent from the clients into a single parameter. The most popular frameworks for this aggregation are FedAvg (10) and FedProx (12). However, naive federated continual learning with these two algorithms on local sequences of tasks may result in catastrophic forgetting. One simple solution is to use a regularization-based, such as Elastic Weight Consolidation (EWC) (4), which allows the model to obtain a solution that is optimal for both the previous and the current tasks. There exist other advanced solutions (14; 15; 16; 17; 19) that successfully prevents catastrophic forgetting. However, the prevention of catastrophic forgetting at the client level is an orthogonal problem from federated learning.



(a) Communication of General Knowledge (b) Communication of Task-adaptive Knowledge

Figure 2. Updates of *FedWeIT*. **(a)** A client sends sparsified federated parameter $\mathbf{B}_c \odot \mathbf{m}_c^{(t)}$. After that, the server redistributes aggregated parameters to the clients. **(b)** The knowledge base stores previous tasks-adaptive parameters of clients, and each client selectively utilizes them with an attention mask.

Thus we focus on challenges that newly arise in this federated continual learning setting. In the federated continual learning framework, the aggregation of the parameters into a global parameter $\boldsymbol{\theta}_G$ allows inter-client knowledge transfer across clients, since a task $\mathcal{T}_i^{(q)}$ learned at client c_i at round q may be similar or related to $\mathcal{T}_j^{(r)}$ learned at client c_j at round r. Yet, using a single aggregated parameter $\boldsymbol{\theta}_G$ may be suboptimal in achieving this goal since knowledge from irrelevant tasks may not to be useful or even hinder the training at each client by altering its parameters into incorrect directions, which we describe as *inter-client interference*. Another problem that is also practically important, is the *communication-efficiency*. Both the parameter transmission from the client to the server, and server to client will incur large communication cost, which will be problematic for the continual learning setting, since the clients may train on possibly unlimited streams of tasks.

3.3. Federated Weighted Inter-client Transfer

How can we then maximize the *knowledge transfer* between clients while minimizing the *inter-client interference*, and communication cost? We now describe our model, *Federated Weighted Inter-client Transfer (FedWeIT)*, which can resolve the these two problems that arise with a naive combination of continual learning approaches with federated learning framework.

The main cause of the problems, as briefly alluded to earlier, is that the knowledge of all tasks learned at multiple clients is stored into a single set of parameters $\boldsymbol{\theta}_G$. However, for the knowledge transfer to be effective, each client should *selectively* utilize only the knowledge of the *relevant* tasks that is trained at other clients. This selective transfer is also the key to minimize the inter-client interference as well as it will disregard the knowledge of irrelevant tasks that may interfere with learning.

We tackle this problem by decomposing the parameters, into three different types of the parameters with different roles: *global parameters* ($\boldsymbol{\theta}_G$) that capture the global and generic knowledge across all clients, *local base parameters* ($\mathbf{B}$) which capture generic knowledge for each client, and *task-adaptive parameters* ($\mathbf{A}$) for each specific task per client. This decomposition scheme is motivated by (9). A set of the model parameters $\boldsymbol{\theta}_c^{(t)}$ for task t at continual learning client c_c is then defined as follows:

$$\boldsymbol{\theta}_c^{(t)} = \mathbf{B}_c^{(t)} \odot \mathbf{m}_c^{(t)} + \mathbf{A}_c^{(t)} + \sum_{i \in \mathcal{C}} \sum_{j<|t|} \alpha_{i,j}^{(t)} \mathbf{A}_i^{(j)} \quad (1)$$

where $\mathbf{B}_c^{(t)} \in \mathbb{R}^{N \times M}$ is the set of base parameters for c^{th} client shared across all tasks in the client, $\mathbf{m}_c^{(t)} \in \mathbb{R}^M$ is a sparse mask which allows to adaptively transform $\mathbf{B}_c^{(t)}$ for the target task, $\mathbf{A}_c^{(t)} \in \mathbb{R}^{N \times M}$ is a sparse matrix of task-adaptive parameters for the task t at client c_c.

The first term allows selective utilization of the global knowledge. We want the base parameter $\mathbf{B}_c^{(t)}$ at each client to capture generic knowledge across all tasks across all clients. In Figure 2 (a), we initialize it at each round t with the global parameter from the previous iteration, $\boldsymbol{\theta}_G^{(t-1)}$ which aggregates the parameters sent from the client. This allows $\mathbf{B}_c^{(t)}$ to also benefit from the *global* knowledge about all the tasks. However, since $\boldsymbol{\theta}_G^{(t-1)}$ also contains knowledge irrelevant to the current task, instead of using it as is, we learn the sparse mask $\mathbf{m}_c^{(t)}$ to select only the relevant parameters for the given task. This sparse parameter selection helps minimize inter-client interference, and also allows for efficient communication. The second term is the task-adaptive parameters $\mathbf{A}_c^{(t)}$. Since we additively decompose the parameters, this will learn to capture knowledge about the task that is not captured by the first term, and thus will capture specific knowledge about the task $\mathcal{T}_c^{(t)}$. The final term describes inter-client knowledge transfer. We have a

Algorithm 1 Federated Weighted Inter-client Transfer

input Dataset $\{\mathcal{D}_c^{(1:t)}\}_{c=1}^{C}$, and Global Parameter $\boldsymbol{\theta}_G^{(0)}$
output $\{\mathbf{B}_c,\ \mathbf{m}_c^{(1:t)},\ \boldsymbol{\alpha}_c^{(1:t)},\ \mathbf{A}_c^{(1:t)}\}_{c=1}^{C}$

Initialize $\mathbf{B}_c$ to $\boldsymbol{\theta}_G^{(0)}$ for all $c \in \mathcal{C} \equiv \{1, ..., C\}$
for task $t = 1, 2, ...$ **do**
  for round $r = 1, 2, ..., R$ **do**
    Select communicable clients $\mathcal{C}^{(r)} \subseteq \mathcal{C}$
    $\mathbf{A}_{c\in\mathcal{C}^{(r)}}^{(t-1,R)}$ and $\hat{\mathbf{B}}_{c\in\mathcal{C}^{(r)}}^{(t,r)}$ are transferred from $\mathcal{C}^{(r)}$ to the central server
    Compute $\boldsymbol{\theta}_G^{(t,r)} \leftarrow \frac{1}{|\mathcal{C}^{(r)}|}\sum_{c\in\mathcal{C}^{(r)}} \hat{\mathbf{B}}_c^{(t,r)}$
    Distribute $\boldsymbol{\theta}_G^{(t,r)}$ and $\{\mathbf{A}_j^{(t-1,R)}\}_{j\in\mathcal{C}^{(r)}}$ to client $c \in \mathcal{C}^{(r)}$
    Minimize Eq. (2) for solving each local CL problems
  end for
end for

set of parameters that are *transmitted* from the server, which contain all task-adaptive parameters from all the clients. To selectively utilizes these indirect experiences from other clients, we further allocate attention $\boldsymbol{\alpha}_c^{(t)}$ on these parameters, to take a weighted combination of them. By learning this attention, each client can select only the relevant task-adaptive parameters that help learn the given task. To reduce a burden of the parameter communication, we send previous task-adaptive parameters at each first training round per task t.

Training. We learn the decomposable parameter $\boldsymbol{\theta}_c^{(t)}$ by optimizing for the following objective:

$$\underset{\mathbf{B}_c^{(t)},\ \mathbf{m}_c^{(t)},\ \mathbf{A}_c^{(1:t)},\ \boldsymbol{\alpha}_c^{(t)}}{\text{minimize}} \ \mathcal{L}\left(\boldsymbol{\theta}_c^{(t)}; \mathcal{T}_c^{(t)}\right) + \lambda_1 \Omega(\{\mathbf{m}_c^{(t)}, \mathbf{A}_c^{(1:t)}\}) + \lambda_2 \sum_{i=1}^{t-1} \|\Delta\mathbf{B}_c^{(t)} \odot \mathbf{m}_c^{(i)} - \Delta\mathbf{A}_c^{(i)}\|_2^2, \tag{2}$$

where $\mathcal{L}$ is a loss function and $\Omega(\cdot)$ is a sparsity-inducing regularization term for the task adaptive parameter and the masking variable (we use ℓ_1-norm regularization), to make them sparse. The final regularization term is used for retroactive update of the past task-adaptive parameters, which helps the task-adaptive parameters to maintain the original solutions for the target tasks, by reflecting the change of the base parameter. Here, $\Delta\mathbf{B}_c^{(t)} = \mathbf{B}_c^{(t)} - \mathbf{B}_c^{(t-1)}$ is the difference between the base parameter at the current and previous timestep, and $\Delta\mathbf{A}_c^{(i)}$ is the difference between the task-adaptive parameter for task i at the current and previous timestep. This regularization is essential for preventing catastrophic forgetting. λ_1 and λ_2 are hyperparameters controlling the effect of the two regularizers.

3.4. Efficient Communication via Sparse Parameters

FedWeIT learns via server-to-client communication. As discussed earlier, a crucial challenge here is to reduce the communication cost. We describe what happens at the client and the server at each step.

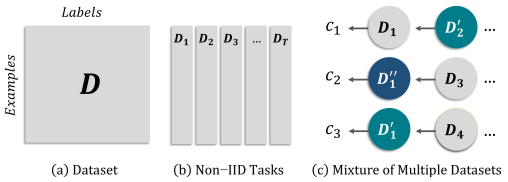


Figure 3. **Configuration of task sequences:** We first split a dataset D into multiple sub-tasks in non-IID manner ((a) and (b)). Then, we distribute them to multiple clients ($C_{\#}$). Mixed tasks from multiple datasets (colored circles) are distributed across all clients ((c)).

Client: At each round r, each client c_c updates its base parameter with the nonzero components of the global parameter sent from the server; that is, $\mathbf{B}_c(i) = \boldsymbol{\theta}_G(i)$ where i is a nonzero element of the global parameter. After training the model using Eq. (2), it obtains a sparsified base parameter $\hat{\mathbf{B}}_c^{(t)} = \mathbf{B}_c^{(t)} \odot \mathbf{m}_c^{(t)}$ and the newly learned task-adaptive parameter $\mathbf{A}_c^{(t)}$. Then, the client sends both $\hat{\mathbf{B}}_c^{(t)}$ and $\mathbf{A}_c^{(t)}$ to the server. Since both parameters are highly sparse, this results in the large reduction of the client-to-server communication cost.

Server: The server first aggregates the *sparsified* base parameters sent from all the clients by taking an weighted average of them: $\boldsymbol{\theta}_G^{(t)} = \frac{1}{\mathcal{C}}\sum_c \hat{\mathbf{B}}_c^{(t)}$. Then, it broadcasts $\boldsymbol{\theta}_G^{(t)}$ along with all task adaptive parameters $\{\mathbf{A}_c^{(t)}\}_{c=1}^{C}$ to all the clients. Algorithm 1 describes our FedWeIT algorithm.

4. Experiments

We validate our **FedWeIT** under different configurations of task sequences against baselines which are namely Overlapped-CIFAR-100 and NonIID-50. **1) Overlapped-CIFAR-100**: We group 100 classes of CIFAR-100 dataset into 20 non-iid superclasses tasks. Then, we randomly sample 10 tasks out of 20 tasks and split instances to create a task sequence for each of the clients with overlapping tasks. **2) NonIID-50**: We use the following eight benchmark datasets: MNIST (25), CIFAR-10/-100 (26), SVHN (27), Fashion-MNIST (28), Not-MNIST (29), FaceScrub (30), and TrafficSigns (31). We split the classes in the 8 datasets into 50 non-IID tasks, each of which is composed of 5 classes that are disjoint from the classes used for the other tasks. This is a large-scale experiment, since we use $280,000$ images of 293 classes from 8 heterogeneous datasets. After generating and processing tasks, we randomly distribute them to multiple clients as illustrated in Figure 3.

Experimental setup We use a modified version of LeNet (25) for the experiments with both Overlapped-

Table 1. Averaged Per-task performance on *NonIID-50* and *Overlapped-CIFAR-100* during FCL with 5 clients. We measured task accuracy and model capacity ratio after completing all learning phases over 3 individual trials.

	NonIID-50 Dataset			Overlapped-CIFAR-100		
Methods	**Accuracy** (%)	**Capacity**	**C2S Cost**	**Accuracy** (%)	**Capacity**	**C2S Cost**
Local-STL	85.78 $\pm$ 0.17	1,000 %	N/A	57.15 $\pm$ 0.07	1,000 %	N/A
Local-EWC (4)	74.30 $\pm$ 0.08	100 %	N/A	44.26 $\pm$ 0.43	100 %	N/A
Local-APD (9)	81.42 $\pm$ 0.72	147 %	N/A	50.82 $\pm$ 0.33	119 %	N/A
FedCurv (23)	72.39 $\pm$ 0.32	100 %	100 %	40.36 $\pm$ 0.44	100 %	100 %
FedCurv-EWC	70.59 $\pm$ 0.39	100 %	100 %	40.59 $\pm$ 0.31	100 %	100 %
FedProx-EWC	68.18 $\pm$ 0.58	100 %	100 %	41.91 $\pm$ 0.47	100 %	100 %
FedProx-APD	81.20 $\pm$ 1.24	130 %	100 %	52.20 $\pm$ 0.41	124 %	100 %
FedWeIT (Ours)	**84.11** $\pm$ **0.27**	**128 %**	**33 %**	**55.16** $\pm$ **0.19**	**126 %**	**33 %**

Table 2. Average Per-task Performance on NonIID-50 during FCL with 20 and 100 clients.

	20 clients		100 clients	
Methods	**Acc.**	**Capa.**	**Acc.**	**Capa.**
Local-APD	46.48%	153%	37.50%	329%
FedWeIT	**50.38%**	155%	**39.58%**	330%

Table 3. FCL results on NonIID-50 dataset with ResNet-18.

	ResNet-18	
Methods	**Accuracy**	**Capacity**
Local-APD	92.44%	110%
FedProx-APD	92.89%	121%
FedWeIT	**94.86%**	**109%**

CIFAR-100 and NonIID-50 dataset. Further, we use ResNet-18 (32) with NonIID-50 dataset. We followed other experimental setups from (33) and (9). We use an Adam optimizer with adaptive learning rate decay, which decays learning rate by a factor of 3 for every 5 epochs that validation loss does not consecutively decrease. For LeNet with 5 clients, we initialize by $1e^{-3} \times \frac{1}{3}$ at the beginning of each new task. Mini-batch size is 100, the rounds per task is 20, an the epoch per round is 1. The setting for ResNet-18 is identical excluding initial learning rate, $1e^{-4}$. In the case of experiments with 20 and 100 clients, we set the same settings except reducing minibatch size from 100 to 10 with an initial learning rate $1e^{-4}$ and exploring client fraction 0.25 and 0.05, respectively. we set $\lambda_1 = [1e^{-1}, 4e^{-1}]$ and $\lambda_2 = 100$ for all experiments. Further, we use $\mu = 5e^{-3}$ for FedProx, $\lambda = [1e^{-2}, 1.0]$ for EWC and FedCurv.

Baselines and our models **1) Local-EWC**: Continual learning with EWC (4). **2) Local-APD**: Continual learning with APD (9). **3) FedCurv**: FCL with FedCurv algorithm (23) which reduces the parameter disparity across clients using Fisher Information Matrix. **4) FedCurv-EWC**: Fedcurv algorithm with EWC. **5) Fed-EWC**: Federated continual learning, that is trained using FedProx (12) algorithm with EWC. **6) Fed-APD**: Federated continual learning with APD using *FedProx* algorithm. **7) FedWeIT**: Our federated weighted inter-client transfer algorithm.

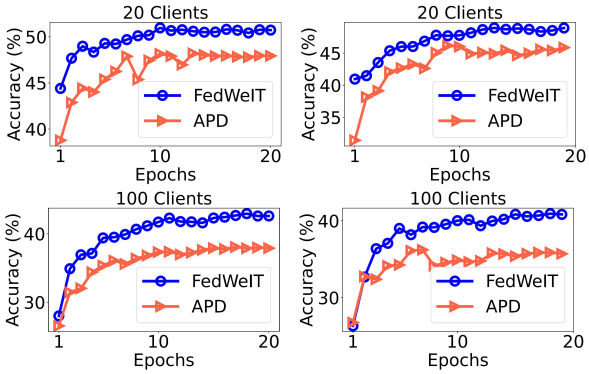
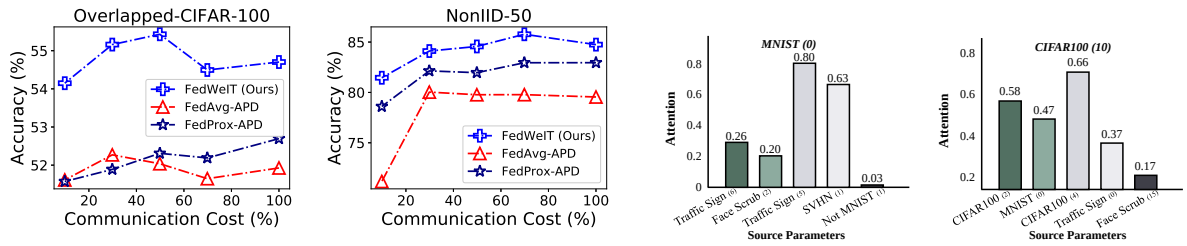


Figure 4. Averaged task adaptation during training last two (9th and 10th) tasks with 20 and 100 clients.

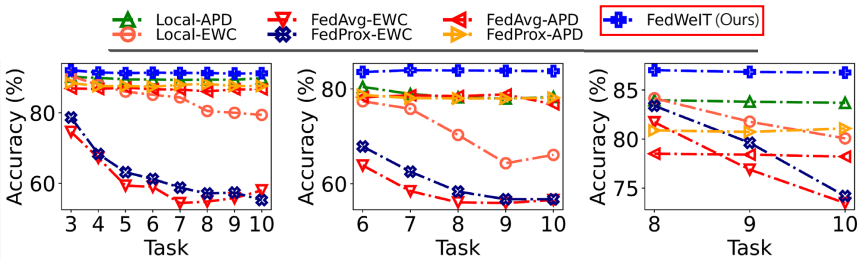
4.1. Experimental Results

We first validate our model on both Overlapped-CIFAR-100 and NonIID-50 task sequences against single task learning (STL), continual learning (EWC, APD), federated learning (FedCurv), and naive federated continual learning (FedCurv, FedProx-based) baselines. Table 1 shows the final average per-task performance after the completion of (federated) continual learning on both datasets. We observe that FedProx-based federated continual learning (FCL) approaches degenerate the performance of continual learning (CL) methods over the same methods without federated learning. This is because the aggregation of all client parameters that are learned on irrelevant tasks results in severe interference in the learning for each task, which leads to catastrophic forgetting and suboptimal task adaptation. While FedCurv reduces inter-task disparity in parameters, it cannot minimize inter-task interference, which results it to underperform single-machine CL methods. On the other hand, FedWeIT significantly outperforms both single-machine CL baselines and naive FCL baselines on both datasets. Even with larger number of clients ($C = 20$, $C = 100$), FedWeIT consistently outperforms Local-APD (Table 2). This improvement largely owes to FedWeIT's ability to selectively utilize the knowledge from other clients to rapidly adapt to the target task, and obtain better final performance (Figure 4). The fast adaptation to new task



(a) Accuracy over client-to-server cost (b) Inter-client Knowledge Transfer of *FedWeIT*

Figure 5. **(a) Accuracy over C2S cost.** We report the relative communication cost to the original network. All results are averaged over the 5 clients. **(b) Inter-client transfer** for NonIID-50. We compare the scale of the attentions at first FC layer which gives the weights on transferred task-adaptive parameters from other clients.



Models	BWT
Local-EWC (4)	-0.045395
Local-APD (9)	-0.004639
FedCurv (23)	-0.191676
FedCurv-EWC	-0.186957
FedProx-EWC	-0.080083
FedProx-APD	-0.002015
FedWeIT (Ours)	**-0.000655**

Figure 6. **Left:** Performance comparison about current task adaptation at 3rd, 6th, and 8th tasks during federated continual learning on *NonIID-50*. **Right:** Forgetting measure using Backward Transfer (BWT).

Table 4. Ablation studies to analyze the effectiveness of parameter decomposition on FedWeIT. All experiments performed on NonIID-50 dataset.

	NonIID-50		
Methods	**Accuracy**	**Capacity**	**C2S Cost**
FedWeIT	**84.11%**	**128%**	**33%**
w/o B comm.	77.88%	115%	2%
w/o A comm.	79.21%	130%	30%
w/o A	65.66%	100%	30%
w/o m	78.71%	143%	104%

is another clear advantage of inter-client knowledge transfer. To further demonstrate the practicality of our method with larger networks, we experiment on Non-IID dtaset with ResNet-18 (Table 3), on which FedWeIT still significantly outperforms the strongest baseline (FedProx-APD) while using fewer parameters.

Efficiency of FedWeIT We also report the accuracy as a function of network capacity in Table 1, 3, which we measure by the number of parameters used. We observe that FedWeIT obtains much higher accuracy while utilizing less number of parameters compared to FedProx-APD. This efficiency mainly comes from the reuse of task-adaptive parameters from other clients, which is not possible with single-machine CL methods or naive FCL methods.

We also examine the communication cost of each method. Table 1 reports the client-to-server communication cost (C2S Cost) and, Figure 5 (a) shows the accuracy as function of communication cost. We observe that FedWeIT is significantly more communication-efficient than naive FCL baselines although it broadcasts task-adaptive parameters, due to high sparsity of the parameters.

Ablation study We perform an ablation study to analyze the role of each component of our FedWeIT. We compare the performance of four different variations of our model. **w/o B communication** describes the model that does not transfer the base parameter **B** and only communicates task-adaptive ones. **w/o A communication** is the model that does not communicate task-adaptive parameters. **w/o A** is the model which trains the model only with sparse transmission of local base parameter, and **w/o m** is the model without the sparse vector mask. As shown in Table 4, without communicating **B** or **A**, the model yields significantly lower performance compared to the full model since they do not benefit from *inter-client knowledge transfer*. The model **w/o A** obtains very low performance due to catastrophic forgetting, and the model **w/o** sparse mask **m** achieves lower accuracy with larger capacity and cost, which demonstrates the importance of performing selective transmission.

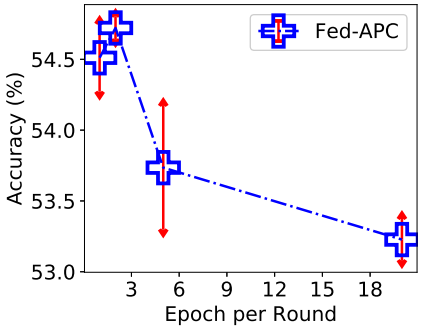
Catastrophic forgetting Further, we examine how the performance of the past tasks change during continual learning, to see the severity of catastrophic forgetting with each method. Figure 6 Left shows the performance of FedWeIT and FCL baselines on the 3rd, 6th, and 8th tasks, at the end of training for later tasks. We observe that naive FCL baselines suffer from more severe catastrophic forgetting than local continual learning with EWC because of the *inter-*

client interference, where the knowledge of irrelevant tasks from other clients overwrites the knowledge of the past tasks. Contrarily, our model shows no sign of catastrophic forgetting. This is mainly due to the selective utilization of the prior knowledge learned from other clients through the global/task-adaptive parameters, which allows it to effectively alleviate *inter-client interference*. FedProx-APD also does not suffer from catastrophic forgetting, but they yield inferior performance due to ineffective knowledge transfer. We also report *Backward Transfer (BWT)*, which is a measure on catastrophic forgetting for all models (more positive the better) and our FedWeIT notably prevents the catastrophic forgetting with a BWT of almost zero.

Weighted inter-client knowledge transfer By analyzing the attention α in Eq. (1), we examine which task parameters from other clients each client selected. Figure 5 (b), shows example of the attention weights that are learned for the 0th split of *MNIST* and 10th split of *CIFAR-100*. We observe that large attentions are allocated to the task parameters from the same dataset (CIFAR-100 utilizes parameters from CIFAR-100 tasks with disjoint classes), or from a similar dataset (MNIST utilizes parameters from Traffic Sign and SVHN). This shows that FedWeIT effectively selects beneficial parameters to maximize *inter-client* knowledge transfer. This is an impressive result since it does not know which datasets the parameters are trained on.

4.2. Effect of the Communication Frequency

We provide an analysis about the effect of the communication frequency of the model, measured by the number of training epochs per communication round. We run the 4 different FedWeIT given $1, 2, 5$, and 20 training epochs per round. Table 5 shows the performance of our FedWeIT variants. As clients frequently update the model parameters through the communication with the central server, the model gets higher performance while maintaining smaller network capacity since the model with a frequent communication efficiently updates the model parameters as transferring the inter-client knowledge. However, it requires much heavier communication costs than the model with sparser communication. For example, the model who trains 1 epochs at each round may need to about 16.9 times larger entire communication cost than the model who trains 20 epochs at each round. Hence, there is a trade-off between model performance of federated continual learning and communication efficiency whereas FedWeIT variants consistently outperform (federated) continual learning baselines.



	Overlapped-CIFAR-100		
Methods	**Accuracy (%)**	**Capacity**	**Epochs / Round**
FedWeIT (Ours)	**54.70 ± 0.24**	**14.8 MB (122%)**	**1**
FedWeIT (Ours)	**54.72 ± 0.08**	15.3 MB (126%)	**2**
FedWeIT (Ours)	53.73 ± 0.44	16.5 MB (136%)	**5**
FedWeIT (Ours)	53.22± 0.14	17.5 MB (144%)	**20**

Table 5. **Average Per-task Performance across the number of training epochs per communication rounds** on *Overlapped-CIFAR-100* for FedWeIT with 5 clients. All models transmit full of local base parameters and highly sparse task-adaptive parameters. All results are the mean accuracies over 5 clients and we run 3 random splits. Gray arrows at each point describes the error bar about the standard deviation of the performance.

5. Conclusion

We tackled a novel problem of federated continual learning, whose goal is to continuously learn local models at each client while allowing it to utilize indirect experience (task knowledge) from other clients. This poses new challenges such as *inter-client knowledge transfer* and prevention of *inter-client interference* between irrelevant tasks. To tackle these challenges, we additively decomposed the model parameters at each client into the global parameters that are shared across all clients, and sparse local task-adaptive parameters that are specific to each task. Further, we allowed each model to selectively update the global task-shared parameters and selectively utilize the task-adaptive parameters from other clients. The experimental validation of our model under various task similarity across clients, against existing federated learning and continual learning baselines shows that our model obtains significantly outperforms baselines with reduced communication cost. We believe that federated continual learning is a practically important topic of large interests to both research communities of continual learning and federated learning, that will lead to new research directions.

References

[1] Sebastian Thrun. *A Lifelong Learning Perspective for Mobile Robot Control*. Elsevier, 1995.

[2] Abhishek Kumar and Hal Daume III. Learning task grouping and overlap in multi-task learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2012.

[3] Paul Ruvolo and Eric Eaton. Ella: An efficient lifelong learning algorithm. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2013.

[4] 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*, page 201611835, 2017.

[5] Jonathan Schwarz, Jelena Luketina, Wojciech M Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. *arXiv preprint arXiv:1805.06370*, 2018.

[6] Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[7] Hanul Shin, Jung Kwon Lee, Jaehon Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[8] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[9] Jaehong Yoon, Saehoon Kim, Eunho Yang, and Sung Ju Hwang. Scalable and order-robust continual learning with additive parameter decomposition. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[10] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*, 2016.

[11] Yang Chen, Xiaoyan Sun, and Yaochu Jin. Communication-efficient federated deep learning with asynchronous model update and temporally weighted aggregation. *arXiv preprint arXiv:1903.07424*, 2019.

[12] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018.

[13] Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Trong Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

[14] Andrei Rusu, Neil Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[15] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[16] Ju Xu and Zhanxing Zhu. Reinforced continual learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2018.

[17] Cuong V. Nguyen, Yingzhen Li, Thang D. Bui, and Richard E. Turner. Variational continual learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[18] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[19] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[20] Michalis K Titsias, Jonathan Schwarz, Alexander G de G Matthews, Razvan Pascanu, and Yee Whye Teh. Functional regularisation for continual learning with gaussian processes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[21] Mohammad Rostami, Soheil Kolouri, Kyungnam Kim, and Eric Eaton. Multi-agent distributed lifelong learning for collective knowledge acquisition. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

[22] Yujing Chen, Yue Ning, and Huzefa Rangwala. Asynchronous online federated learning for edge devices. *arXiv preprint arXiv:1911.02134*, 2019.

[23] 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.

[24] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. In *International Conference on Learning Representations*, 2020.

[25] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

[26] Alex Krizhevsky and Geoffrey E. Hinton. Learning multiple layers of features from tiny images. Technical report, Computer Science Department, University of Toronto, 2009.

[27] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[28] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[29] Yaroslav Bulatov. Not-mnist dataset. 2011.

[30] Hong-Wei Ng and Stefan Winkler. A data-driven approach to cleaning large face datasets. In *2014 IEEE international conference on image processing (ICIP)*, pages 343–347. IEEE, 2014.

[31] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In *The 2011 international joint conference on neural networks*, 2011.

[32] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[33] Joan Serrà, Dídac Surís, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.