

## A IMPLEMENTATION DETAILS

### A.1 Datasets

In our experiments, we used three datasets.

**MNIST** [16] contains 60,000 training images and 10,000 testing images for digits 0 to 9. Each digit has 50000 training and 10000 evaluation samples with the size  $28 \times 28$ .

**CIFAR10** [15] is composed of 60,000 color images from 10 classes. Each class has 50000 training and 10000 evaluation samples with the size  $32 \times 32$ .

**CIFAR100** [15] is composed of 60,000 color images from 100 classes. Each class has 500 training and 100 evaluation samples with the size  $32 \times 32$ .

### A.2 Experimental Settings

For the federated learning setting, there are three clients and one central server in this system. In the beginning, for each dataset, we split the data for each class into three parts using a Dirichlet distribution to ensure that there is data overlap between clients. We ensure that each client has at least 20% of the data for each class by adjusting the scaling factor.

As for the continual learning setting, each client possesses a sequence that consists of five different tasks. Thus, different tasks would include different classes. For MNIST and CIFAR-10, each client exclusively owns two classes that were only accessible to itself and not accessible to others. Therefore, the number of public classes is four. For CIFAR-100, we allow each client to have 25 private classes, resulting in 25 public classes. Each task consists of 10 classes sampled from both private and public classes, with no class overlap between tasks.

In detail, for MNIST and CIFAR-10, we first randomly sample four classes as public classes using different random seeds. Subsequently, we randomly sample two classes from the remaining six classes for each client as private classes. After that, each client owns data from six classes, including four public classes and two private classes. The data of each task is randomly sampled from these six classes, with three classes chosen for each task. So there is an **overlap of classes** between tasks.

As for CIFAR-100, we randomly select 25 classes out of the 100 classes as public classes, and the remaining classes are allocated to each client as their private classes, with each client receiving 25 private classes. Afterward, each client concatenates their private classes with the public classes, shuffles the order, and then selects 10 classes sequentially for their task's data. With this processing, there is **no class overlap** between tasks for each client.

On each dataset, we conducted experiments using three different random seeds (42, 1999, 2002) and averaged the results. We set the number of global epochs to 5 and the number of local epochs to 50. We use an Adam optimizer whose initial learning rate is 0.001 to train all classification models, including baseline methods. The batch size is 32 and each VAE needs to generate 100 pseudo-samples during the selective knowledge fusion on the server side.

### A.3 Comparision Methods

Due to the novelty and complexity of the FCILps scenario, no prior work has explicitly adopted this setting. Therefore, for fair comparisons, we compare our FedAE with several traditional FL methods and conventional FCIL methods, including **FedAvg** [24], **FedAvg+EWC**, **FedProx** [21], **GLFC** [5], **FedSpace** [29], to validate the effectiveness on the heterogeneous knowledge fusion of our proposed FedAE framework. Besides, three novel metrics are employed to evaluate the abilities of fusing heterogeneous knowledge and resisting spatial-temporal catastrophic forgetting of other baseline methods and our proposed FedAE method.

## B OPTIMIZATION PIPELINE

The proposed algorithm is summarized in Algorithm 1. We now further describe our optimization pipeline in detail. Starting from the first incremental task, the first clients group their data of the current task by class. Then, for each class that has appeared in this task, the client establishes a VAE model dedicated to it, serving as an anomaly detection module. If the client's existing global models already have a VAE related to this class, the global VAE will also participate in training for continual personalization. The goal is to personalize the generalized knowledge of this class from the global VAE. If there is no existing global VAE for this class, then regular VAE training will take place to train class-specific VAE.

After each client has established a VAE for each class, the server will begin selective knowledge fusion. Firstly, the server performs a union operation on the classes encountered by each client in this round. This way, the server can determine which classes are included in the client tasks for this round. Subsequently, for each class, the server creates an empty list to store the local models of this class. If the global model from the previous round includes a model related to this class, it is also added to this list. Once all local models and the global model from the previous round related to that class are added to the list, the server will proceed with selective knowledge fusion.

Firstly, all the decoders in this list will generate  $n$  pseudo-samples based on Gaussian noise generated from a normal distribution. Subsequently, these pseudo-samples are used as a training set for the next distillation step to generate a more generalized global model, see Sec. 4.2 for more detail. Once the global model of this class is obtained, the server stores it in the form of the key-value pair, in which the key is the class and the value is the global VAE. When all the class is processed, the server distributes all the global VAE to clients.

## C EXPERIMENTS ON CIFAR-100 DATASET

Table 5 shows the accuracy of local models from clients when tested on the current task's test set after completing the training for the current task. Table 6 shows the accuracy of the global model, generated by aggregating local models from the current task, on the local test set. We can observe that after aggregation, the performance of the global models for all baseline methods decreases compared to the individual local models before aggregation, and it may even drop to 0%. This indicates that these methods have all experienced varying degrees of spatial catastrophic forgetting. Compared to the

**Table 5: The accuracy of local models trained on the current task when tested on the same task.**

Algorithm	Task ID				
	1	2	3	4	5
FedAvg[24]	42.63	38.71	40.95	44.01	41.59
FedAvg+EWC	42.37	37.35	43.55	43.13	42.22
FedProx[21]	51.94	11.86	13.29	12.30	9.92
GLFC[5]	89.45	78.32	98.05	88.88	98.91
FedSpace[29]	58.75	49.92	52.45	49.49	50.08
Ours(FedAE)	61.94	57.99	61.54	58.77	56.97

**Table 6: The accuracy of the global model testing on the local testing set of current task.**

Algorithm	Task ID				
	1	2	3	4	5
FedAvg[24]	10.38	11.53	8.55	13.19	9.76
FedAvg+EWC	5.41	7.62	5.50	7.44	8.09
FedProx[21]	8.92	7.44	1.97	2.49	0.00
GLFC[5]	42.21	51.01	57.67	56.79	63.11
FedSpace[29]	22.19	24.46	20.75	25.34	23.06
Ours(FedAE)	65.16	62.94	65.93	62.77	62.00

accuracy of local model testing, our method shows an improvement in the accuracy of the global model on the local test set after aggregation. This indicates that our selective knowledge fusion

can combine and refine knowledge from different clients about the same class, making the knowledge in the resulting global model more generalized.

**Table 7: The accuracy of local models trained on the current task when tested on the first task.**

Algorithm	Task ID				
	1	2	3	4	5
FedAvg[24]	42.63	0.00	0.00	0.00	0.00
FedAvg+EWC	42.39	0.00	0.00	0.00	0.00
FedProx[21]	51.94	0.00	0.00	0.00	0.13
GLFC[5]	89.78	1.16	0.00	4.15	0.00
FedSpace[29]	58.75	0.00	0.00	0.00	0.00
Ours(FedAE)	62.12	61.77	62.23	61.59	61.64

Table 7 shows the accuracy of local models at different stages when retroactively tested on the local test set of the first task. The results in the first column refer to the accuracy testing on the first test set using the local models trained on the first task. The results clearly show that all the baseline methods cannot alleviate the temporal catastrophic forgetting caused by the class-incremental tasks. Based on the experimental results, it is evident that all baseline methods have not taken into account the temporal forgetting caused by continual learning. Our method shows little to no decline in accuracy, indicating strong resilience against forgetting caused by class-incremental tasks.