Supplementary Material for "Text-Enhanced Data-free Approach for Federated Class-Incremental Learning"

Minh-Tuan Tran[1], Trung Le[1], Xuan-May Le[2], Mehrtash Harandi[1], Dinh Phung[1]
[1]Monash University, [2]University of Melbourne
{tuan.tran7,trunglm,mehrtash.harandi,dinh.phung}@monash.edu
xuanmay.le@student.unimelb.edu.au

A. Experimental Detail

A.1. Evaluation Metric

In this works, we used two commonly used evaluation metric in FCIL including the averaging accuracy and averaging forgetting score.

- Last Incremental Averaging Accuracy (Acc): It is the accuracy of server model after training all tasks.
- Averaging Forgetting ($\mathcal{F}$): $\mathcal{F}^t$ of task t is defined as the difference between the highest accuracy of the model on task t and its performance at the end of the training. Therefore, we can evaluate the average forgetting $\mathcal{F}$ by averaging all the $\mathcal{F}^t$ for task 1 to T - 1 at the end of task T.

A.2. Client Training Details

For a fair comparison, in accordance with [10], we employed ResNet18 [4] as the backbone for all experiments. Each client was trained with a batch size of 128 for 100 communication rounds, with 2 local training epochs per communication round. For CIFAR-100, we used the SGD optimizer with a learning rate of 0.04, a momentum of 0.9, and a weight decay of 5e-4. In the case of Tiny-ImageNet and ImageNet, a learning rate of 0.1, weight decay of 2e-4, and a multi-step scheduler were applied, reducing the learning rate by 10 at the 50th and 75th communication rounds.

A.3. Generator Network Training Details

The generator architecture for CIFAR-100 and Tiny-ImageNet is detailed in Table 1, while the generator architecture for ImageNet is presented in Table 2. We utilize the Adam optimizer with a learning rate of 2e-3 to optimize the generator. For CIFAR-100 and Tiny-ImageNet, we configure the synthetic batch size to be 256, whereas for ImageNet, the synthetic batch size is set to 128 to ensure that GPU memory remains below 24GB.

Table 1. Generator Network ($\mathcal{G}$) Architecture for CIFAR-100 and Tiny-ImageNet.

Output	**Size Layers**
256	Input
$128 \times h/4 \times w/4$	Linear, BatchNorm1D, Reshape
$128 \times h/4 \times w/4$	SpectralNorm (Conv (3×3)), BatchNorm2D, LeakyReLU
$128 \times h/2 \times w/2$	UpSample ($2\times$)
$64 \times h/2 \times w/2$	SpectralNorm (Conv (3×3)), BatchNorm2D, LeakyReLU
$64 \times h \times w$	UpSample ($2\times$)
$3 \times h \times w$	SpectralNorm (Conv (3×3)), Sigmoid, BatchNorm2D

Table 2. Generator Network ($\mathcal{G}$) Architecture for ImageNet.

Output	**Size Layers**
256	Input
$128 \times h/16 \times w/16$	Linear, BatchNorm1D, Reshape
$128 \times h/16 \times w/16$	SpectralNorm (Conv (3×3)), BatchNorm2D, LeakyReLU
$128 \times h/8 \times w/8$	UpSample ($2\times$)
$128 \times h/8 \times w/8$	SpectralNorm (Conv (3×3)), BatchNorm2D, LeakyReLU
$128 \times h/4 \times w/4$	UpSample ($2\times$)
$64 \times h/4 \times w/4$	SpectralNorm (Conv (3×3)), BatchNorm2D, LeakyReLU
$64 \times h/2 \times w/2$	UpSample ($2\times$)
$64 \times h/2 \times w/2$	SpectralNorm (Conv (3×3)), BatchNorm2D, LeakyReLU
$64 \times h \times w$	UpSample ($2\times$)
$3 \times h \times w$	SpectralNorm (Conv (3×3)), Sigmoid, BatchNorm2D

A.4. Hyperparameters Tuning

In this section, we present the hyperparameter tuning approaches and results used in our experiments. The hyperparameters for CIFAR-100, Tiny-ImageNet, and ImageNet datasets are summarized in Table 3.

Table 3. The hyperparameters for LANDER applied to CIFAR-100, Tiny-ImageNet, and ImageNet are detailed below. Specifically, α_{cur} and α_{pre} are the base factors for training the client model, while λ_{cls}, λ_{bn}, and λ_{adv} are the hyperparameters associated with generative model training. The variables g represent the training steps to optimize the generator, and I is the number of rounds for generating synthetic images.

	α_{cur}	α_{pre}	λ_{bn}	λ_{oh}	λ_{ltc}	g	I
CIFAR-100	0.2	0.4	1	0.5	5	40	40
Tiny-ImageNet						100	200
ImageNet						200	400

Scale Factor Tuning. Scale factor hyperparameters play a crucial role in algorithm performance. In our experiments, we demonstrate the sensitivity of the final performance to

each hyperparameter. Due to the computational expense of hyperparameter tuning in this setting, we systematically vary one parameter at a time while keeping others fixed. The final accuracy for CIFAR-100 datasets with five tasks and a Dirichlet parameter set at 0.5 is reported in Table 4.

Table 4. We examine the impact of various hyperparameters on the final accuracy for CIFAR-100. Here, α_{cur} and α_{pre} denote the scale factors for client model training, while λ_{bn} and λ_{oh} serve as the scale factors for generator training. Additionally, λ_{ltc} is utilized for both client and generator training.

α_{cur}	Acc	α_{pre}	Acc	λ_{bn}	Acc	λ_{oh}	Acc	λ_{ltc}	Acc
0.1	47.12	0.1	46.37	0.01	46.25	0.1	47.79	0.5	47.93
0.2	48.23	0.2	47.24	0.1	47.68	0.5	48.23	1	48.12
0.4	46.92	0.4	48.23	1	48.23	1	48.12	5	48.23
0.6	44.23	0.6	47.82	10	47.31	5	48.07	10	48.07

Generation Training Steps g. Due to the distinct challenges and image resolutions of CIFAR-100, Tiny-ImageNet, and ImageNet, we assess the impact of different generator training steps, denoted as g, for each dataset. Our findings reveal that the most crucial impact of g lies in distilling knowledge from the server to the client. This is evaluated through the distilling accuracy of an additional student (discriminator) in generator training after the first task. Table 5 illustrates that student accuracy perfectly aligns with the final accuracy across different g values. Consequently, we propose using this metric to expedite parameter tuning.

Table 5 shows that a low value of g has negative effects on the performance of our work. Increasing the value of g improves performance; however, an excessively high g does not guarantee higher performance.

Table 5. The impact of different generator training steps, denoted as g, on the final accuracy for three datasets. Here, "1st Acc" represents the distilling accuracy of the student model after the first task, while "Acc" signifies the final accuracy of the model.

CIFAR100			Tiny-ImageNet			ImageNet		
g	1st Acc	Acc	g	1st Acc	Acc	g	1st Acc	Acc
30	72.12	49.21	50	49.12	27.47	100	58.12	40.38
40	**74.61**	**51.78**	**100**	**52.82**	**28.21**	**200**	**61.86**	**41.75**
50	74.51	51.71	150	52.19	28.12	300	61.37	41.65
60	74.27	51.74						

Data Synthetic Rounds I. Building on the findings from the last section, we also utilize the distilling accuracy of the student model at the first task as the evaluation metric for tuning the different data synthetic rounds, denoted as I, for three datasets. The results from Table 6 indicate that with a low volume of synthetic data, our method's accuracy fails to provide sufficient information for effective learning from previous tasks. Increasing the data volume effectively mitigates the forgetting phenomenon and enhances performance. However, a continuous increase in data volumes does not yield a significant improvement in the model's performance. It's crucial to highlight that the data volume alone does not guarantee the effectiveness of synthetic data in enhancing machine learning models.

Table 6. The effect of different data synthetic rounds, denoted as I, on the final accuracy for three datasets. In this context, "1st Acc" represents the distilling accuracy of the student model after the first task, and "SynData" denotes the total number of synthetic data. Please note that we set the synthetic batch size at 256 for CIFAR-100 and Tiny-ImageNet, while the synthetic batch size is 128 for ImageNet to keep the GPU memory below 24GB.

CIFAR100			Tiny-ImageNet			ImageNet		
I	SynData	1st Acc	I	SynData	1st Acc	I	SynData	1st Acc
20	5120	73.31	100	25600	48.12	200	25600	57.42
40	**10240**	**74.61**	150	38400	50.95	300	38400	61.16
60	15360	74.47	**200**	**51200**	**52.82**	**400**	**51200**	**61.86**
80	20480	74.32	250	64000	51.43	500	64000	60.82

B. Extended Results

Discussing about Local Training Epoch. We conducted experiments with different client training epochs in our methods and found that our approach is effective with epochs higher than 1. For optimal speed, we chose $l = 2$, making our methods typically twice as fast as TARGET (4.25 to 9.87 hours), which requires local training epochs of 5. This demonstrates our method's superior performance and substantial reduction in training time compared to TARGET.

Table 7. Accuracies and training time of our LANDER over different local training epoches. This experiments is conducted using a single NVIDIA RTX 4090 in CIFAR-100 dataset with NIID(0.5).

		l = 1	l = 2	l = 3	l = 4	l = 5	l = 6
Method	Time (hours)	2.35	4.25	6.13	8	9.87	11.75
TARGET	Acc	27.94	29.19	30.27	32.18	33.33	33.26
LANDER	Acc	43.24	48.23	48.23	48.23	48.23	48.23

Comparison with Different Number of Clients. We conducted additional experiments to evaluate our method with a higher number of clients. Table 8 demonstrates that even with an increased number of clients, our method maintains good performance. Notably, with 50 clients, our method significantly outperforms state-of-the-art methods (28.42% compared to 12.72%).

Table 8. Comparison with Different Number of Clients

	m=5	m=10	m=20	m=30	m=40	m=50
TARGET	36.31	27.48	21.98	19.8	15.12	12.75
LANDER	**52.60**	49.98	40.24	35.64	28.92	28.42

C. Privacy of LANDER

Numerous attacks prevalent in federated learning, either in general or specifically in FedAvg [8], include data poisoning, model poisoning, backdoor attacks, and gradient inversion attacks [3, 5–7]. In general, our approach does not introduce additional privacy concerns and maintains the same privacy issues as FedAvg. Therefore, our method is

Figure 1. The top row displays real data, while the middle and bottom rows illustrate randomly generated synthetic data from our LANDER for ImageNet datasets.

also compatible with any defense solutions used for FedAvg, such as secure aggregation [2] or noise injection before aggregation [9].

In contrast to several existing FCIL approaches [11], where clients need to share a locally trained generative model or perturbed private data, LANDER's generative model training relies on the weights of the global model, already shared with all clients in the FedAvg scenario. Anchoring the label-text embedding can enhance the data-free generation process while still preserving the privacy setting. Figure 1 illustrates several examples of synthetic images. From these examples, it is evident that the synthetic images exhibit significant differences compared to real data. However, they still encapsulate common knowledge and can be viewed as abstract visualizations of classes, thereby enhancing knowledge transfer.

Furthermore, we introduce Learnable Data Stats (LDS) to bolster the data privacy of the FCIL setting. Unlike prior methods [1, 10, 11] that demand training data statistics, such as the mean and variance of all training data, for effective synthetic data generation, our approach with LDS achieves comparable performance without relying on these statistics. By eliminating the need for training data stats, our method enhances data privacy in the FCIL setting.

D. Does LANDER Work in Meaningless Label Datasets?

The paper proposes employing label-text embedding (LTE) while acknowledging its potential drawbacks in datasets lacking meaningful labels, such as those related to chemical compounds. Despite these challenges, our approach exclusively considers LTE as the optimal choice for anchoring each class. In instances of datasets with meaningless labels, our method can utilize the label index instead of the label description to generate the prompt, as demonstrated by `"a class of {class_index}"`. Consequently, our method consistently outperforms the current state-of-the-art approach when using only the label, as illustrated in Section 5.4 of the main paper.

Furthermore, our method can be compatible with any pretrained language model, thereby expanding the applications of our work in various real-world scenarios.

E. Limitations and Future Works

In our method, a primary limitation is the need to store and transfer a large amount of synthetic data from the server to the client, thereby increasing communication costs and the training load for the client. Consequently, reducing the volume of required synthetic data emerges as a potential direction for future work.

References

[1] Sara Babakniya, Zalan Fabian, Chaoyang He, Mahdi Soltanolkotabi, and Salman Avestimehr. A data-free approach to mitigate catastrophic forgetting in federated class incremental learning for vision tasks. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 3

[2] K Bonawitz, V Ivanov, B Kreuter, A Marcedone, HB McMahan, S Patel, D Ramage, A Segal, and K Seth. Practical secure aggregation for federated learning on user-held data. arxiv 2016. *arXiv preprint arXiv:1611.04482*. 3

[3] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. Local model poisoning attacks to {Byzantine-Robust} federated learning. In *29th USENIX security symposium (USENIX Security 20)*, pages 1605–1622, 2020. 2

[4] 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. 1

[5] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021. 2

[6] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine*, 37(3):50–60, 2020. 2

[7] Lingjuan Lyu, Han Yu, and Qiang Yang. Threats to federated learning: A survey. *arXiv preprint arXiv:2003.02133*, 2020. 2

[8] Aman Priyanshu, Mudit Sinha, and Shreyans Mehta. Continual distributed learning for crisis management. *arXiv preprint arXiv:2104.12876*, 2021. 2

[9] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 15:3454–3469, 2020. 3

[10] Jie Zhang, Chen Chen, Weiming Zhuang, and Lingjuan Lyu. Target: Federated class-continual learning via exemplar-free distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4782–4793, 2023. 1, 3

[11] Martin Zong, Zengyu Qiu, Xinzhu Ma, Kunlin Yang, Chunya Liu, Jun Hou, Shuai Yi, and Wanli Ouyang. Better teacher better student: Dynamic prior knowledge for knowledge distillation. In *The Eleventh International Conference on Learning Representations*, 2023. 3