Task-Aware Prompt Gradient Projection for Parameter-Efficient Tuning Federated Class-Incremental Learning

Supplementary Material

Symbols	Meaning
$p^{t^k_r}$	The prompt for the k-th client in the r-th round of t-th task
P^{t-1}	The prompt for task $t-1$ comes from the server
p_{u-1} , P_{u-1}	The prompt for the u-th layer of the model
e_{u-1}	A collection of image patch embeddings for the u-th layer of the model
$\hat{V}_{u-1}$	Projection matrix for the u-th layer of the model
$\mathcal{C}_k^t$	The model of the k-th client in the task t
$\mathcal{S}^{t-1}$	The model of the server in the task $t-1$
$\mathcal{D}_k^t$	The private data of the k-th client in the task t
$\mathcal{M}^{t-1}$	The synthetic data of the server in the task $t-1$

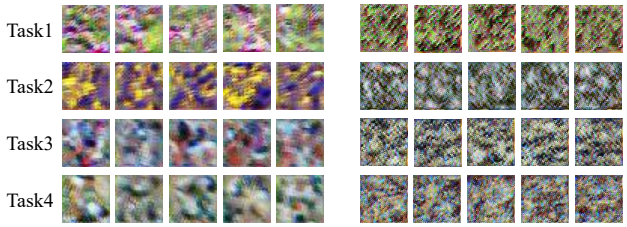
Table 1. The meaning of symbols.

	Method	Task1	Task2	Task3	Task4	Task5
IID	TARGET [3]	77.16	55.32	45.67	36.19	31.83
	LANDER [2]	77.32	65.42	56.34	48.82	43.24
	TPPR	92.27	83.77	78.44	72.79	65.72
NIID(1)	TARGET [3]	76.78	54.74	42.67	31.19	29.83
	LANDER [2]	76.91	63.35	54.25	45.35	41.75
	TPPR	90.93	81.18	74.61	65.78	61.62

Table 2. The Average Accuracy (%) on ImageNet for 5 tasks.

	Method	Task1	Task2	Task3	Task4	Task5
NIID(0.5)	I	96.4	86.72	80.65	76.9	73.53
	II	96.4	86.88	82.22	77.91	73.84

Table 3. Gradient projection analysis on different components. The experiments are carried out on task quantity $T = 5$ and Dirichlet parameter $\beta = 0.5$.



(a) CIFAR-100 (b) Tiny-ImageNet

Figure 1. Visualization of generated data for CIFAR-100 and Tiny-ImageNet datasets.

A. Other experiment results

A.1. Results on the ImageNet dataset

To further validate the effectiveness of our method, we conduct experiments on the ImageNet dataset with task quantity $T = 5$ under both IID and NIID (1) scenarios. Table 2 presents the average accuracy across all tasks after training each one. The results demonstrate that even in a challenging setting like ImageNet, which contains a large number of categories and high training difficulty, our method significantly enhances performance. Notably, compared to the baseline, our approach achieves a substantial 20% improvement, further highlighting its effectiveness.

A.2. Gradient projection analysis on different components.

We employ the data-free knowledge transfer framework for training. On the client side, we address knowledge forgetting by leveraging synthetic data $\mathcal{M}$ and the server model $\mathcal{S}$ from the old task to distill knowledge to the current task. Simultaneously, current private data is utilized to train the model for new tasks. Consequently, the model's gradient is primarily composed of two components: the gradient from distillation learning and the gradient from new task learning. As shown in Table 3, we evaluate the effectiveness of two gradient projection strategies: (I) applying projection only to the gradient from new task learning and (II) applying projection to the sum of both gradients. The results indicate minimal difference between the two approaches. Our analysis suggests that when projection is applied only to the new task gradient, adding the distillation gradient afterward may alter the overall gradient direction to some extent, potentially disrupting its orthogonality and leading to a slight performance drop.

A.3. Privacy analysis of generated data

As illustrated in Figure 1, we showcase the virtual data synthesized by the server-side model on two widely used datasets, CIFAR-100 and Tiny-ImageNet. The generated images, as depicted in the figure, primarily consist of abstract lines and indistinct contours, which exhibit a noticeable disparity when compared to real images in terms of clarity and detail. This intentional design ensures that the synthetic data does not closely resemble actual images, thereby preserving the privacy-centric nature of federated learning. By leveraging such low-fidelity representations, we effectively mitigate the risk of catastrophic forgetting in the model without compromising the confidentiality of the original data. The results demonstrate that our method successfully balances the dual objectives of preserving privacy and maintaining model performance, making it a viable solution for privacy-sensitive applications.

B. Loss Function

We adopt a loss function similar to the SOTA method LANDER [2], which is based on Data-Free Knowledge Transfer (DFKT). The key to DFKT lies in generating high-quality data related to the previous task on the server side, thereby mitigating knowledge forgetting. LANDER [2] proposes using label-text embedding (LTE) as an anchor, ensuring that the features of both generated data and real data are centered around the LTE. This approach reduces the distance between the generated data and the real data, improving the quality of the generated data. We begin by constructing the label text Y_y for each class y using a predefined template: "A photo of a {class_name}", where {class_name} corresponds to the class label. For instance, if the label is "cat", the resulting prompt would be "A photo of a cat." Next, we compute the label text embedding z_y using a pretrained language model $\mathcal{E}$ (i.e., text encoder of CLIP) as follows:

$$z_y = \mathcal{E}(Y_y), \quad \forall y \in \mathcal{Y}. \tag{1}$$

Notably, the embedding z_y is computed once and stored in the LTE pool $\mathcal{P}$, remaining unchanged throughout the entire training process, with no further fine-tuning of the pretrained language model $\mathcal{E}$. In order to bridge the gap between generated data and real data, we use the following Bounding Loss:

$$\mathcal{B}(f_c, z_y) = \max\left(0, \|z_y - \mathcal{W}(f_c)\|^2 - r\right), \tag{2}$$

$\mathcal{W}(\cdot)$ represents a linear projection layer to align the dimensions between visual features f_c and text embedding z_y. r represents a distance constraint, allowing features to cluster around z_y without converging entirely to it. This helps maintain feature diversity and prevents homogenization.

B.1. The loss function of the clients

The client learns new tasks while utilizing models and generated data from old tasks for knowledge distillation. For the new task data, we employ the cross-entropy (CE) loss to optimize the client model using real training data. Additionally, we impose constraints on f_c via the Bounding Loss, as defined in Eq. (2). The objective function for the current task is formulated as:

$$\mathcal{L}_{\text{cur}} = \text{CE}(y_c, y) + \lambda_{\text{lte}} B(f_c, e_y), \tag{3}$$

where y_c and f_c are derived from $\mathcal{C}_k^t(x)$, with $(x, y) \in \mathcal{D}_k^t$.

To retain knowledge from previous tasks, we perform knowledge distillation using synthetic data $\mathcal{M}^{t-1}$. The corresponding loss function is given by:

$$\mathcal{L}_{\text{pre}} = \text{KL}(\hat{y}_c, \hat{y}_s) + \text{MSE}(\hat{f}_c, \hat{f}_s), \tag{4}$$

where the synthetic data-label pair $(\hat{x}, \hat{y})$ is sampled from $\mathcal{M}^{t-1}$. Here, $\hat{y}_c$ and $\hat{f}_c$ are obtained from $\mathcal{C}_k^t(\hat{x})$, while $\hat{y}_s$ and $\hat{f}_s$ originate from $\mathcal{S}^{t-1}(\hat{x})$. $\hat{y}_s$ represents the output of the classifier for $\hat{f}_s$. The Kullback-Leibler (KL) divergence term is employed to transfer the logits from the previous model to the current one, ensuring knowledge retention.

By combining the above losses, the overall loss function for the client model C_k^t is defined as:

$$\mathcal{L}_{\mathcal{C}} = \alpha_{\text{cur}}^t \mathcal{L}_{\text{cur}} + \alpha_{\text{pre}}^t \mathcal{L}_{\text{pre}}. \tag{5}$$

To address the increasing challenge of retaining previous knowledge as the ratio of old classes to new classes grows, the scaling factors α_{cur}^t and α_{pre}^t are dynamically adjusted as follows:

$$\alpha_{\text{cur}}^t = \frac{1 + 1/\kappa}{\delta} \alpha_{\text{cur}}, \quad \alpha_{\text{pre}}^t = \kappa \delta \alpha_{\text{pre}}, \tag{6}$$

where $\kappa = \log_2\left(\frac{|\mathcal{Y}^t|}{2} + 1\right)$, $\delta = \sqrt{\frac{|\mathcal{Y}^{1:t-1}|}{|\mathcal{Y}^t|}}$. Here, $|\mathcal{Y}^t|$ denotes the number of classes in task t, while α_{cur} and α_{pre} represent the base scaling factors.

B.2. The loss function of the server

The loss function on the server is mainly used to facilitate the generation of old task data. Firstly, we randomly sample a pseudo label $\hat{y}$ from a categorical distribution and retrieve the corresponding LTE $z_{\hat{y}}$ from the $\mathcal{P}$ (i.e., $z_{\hat{y}} \sim \mathcal{P}$), which serves as the input to the noisy layer $\mathcal{Z}$. Subsequently, the representation $\mathcal{Z}(z_{\hat{y}})$ is fed into the generator $\mathcal{G}$ to produce synthetic images $\hat{x}$:

$$\hat{x} = \mathcal{G}(\mathcal{Z}(z_{\hat{y}}) - \mu)/\sigma. \tag{7}$$

where $\mathcal{Z}$ consists of a Batch Normalization (BatchNorm) layer followed by a single Linear layer. μ and σ are learnable mean and standard deviation.

To effectively facilitate knowledge transfer, the synthetic data must satisfy two key properties: similarity and diversity.

Similarity. The synthetic data $\hat{x}$ should closely resemble real training data. However, since direct access to client data is unavailable, we enforce this similarity by aligning the logits of the previous model $\mathcal{S}^{t-1}$ with the pseudo label $\hat{y}$. This alignment is achieved by minimizing the cross-entropy (CE) loss between the prediction of $\mathcal{S}^{t-1}$ on $\hat{x}$ and the pseudo label $\hat{y}$, formulated as follows:

$$\mathcal{L}_g^{\text{oh}} = \text{CE}(\hat{y}_{\mathcal{S}^{t-1}}, \hat{y}), \tag{8}$$

where $\hat{y}_{\mathcal{S}^{t-1}}$ denotes the predicted label of $\mathcal{S}^{t-1}$ for the synthetic sample $\hat{x}$.

Diversity. To mitigate the risk of generating highly similar images, an additional discriminator, denoted as the student network $\mathcal{Q}$ is introduced. Specifically, for the synthetic images $\hat{x}$, $\mathcal{Q}$ is trained to minimize the discrepancy between its predictions and those of the server (teacher) model:

$$\mathcal{L}_{\mathcal{Q}} = \text{KL}(\hat{\boldsymbol{y}}_{\mathcal{Q}}, \hat{\boldsymbol{y}}_{\mathcal{S}^{t-1}}) + \text{MSE}(\hat{\boldsymbol{f}}_{\mathcal{Q}}, \hat{\boldsymbol{f}}_{\mathcal{S}^{t-1}}), \tag{9}$$

where $\hat{\boldsymbol{y}}_{\mathcal{Q}}$ and $\hat{\boldsymbol{f}}_{\mathcal{Q}}$ are derived from $\mathcal{Q}(\hat{\boldsymbol{x}})$.

To encourage the generator to produce diverse images of the previous server model, the negative KL loss is minimized:

$$\mathcal{L}_{\mathcal{G}}^{adv} = -\omega \mathrm{KL}(\hat{\boldsymbol{y}}_{\mathcal{Q}}, \hat{\boldsymbol{y}}_{\mathcal{S}^{t-1}}), \tag{10}$$

where $\omega = \mathbb{1}(\mathrm{argmax}(\hat{\boldsymbol{y}}_{\mathcal{S}^{t-1}}) \neq \mathrm{argmax}(\hat{\boldsymbol{y}}_{\mathcal{Q}}))$, $\mathbb{1}(P)$ is an indicator function that returns 1 if P is true and 0 otherwise.

The Bounding Loss is used to ensure synthetic images remain within the LTE area:

$$\mathcal{L}_{\mathcal{G}}^{lte} = \mathrm{B}(\hat{\boldsymbol{f}}_{\mathcal{S}^{t-1}}, \boldsymbol{z}_{\hat{\boldsymbol{y}}}). \tag{11}$$

The final objective for the generator $\mathcal{G}$, noisy layer $\mathcal{Z}$, and learnable parameters μ and σ is:

$$\mathcal{L}_{\mathcal{G},\mathcal{Z},\mu,\sigma} = \mathcal{L}_{\mathcal{G}}^{adv} + \lambda_{oh}\mathcal{L}_{\mathcal{G}}^{oh} + \lambda_{lte-g}\mathcal{L}_{\mathcal{G}}^{lte}, \tag{12}$$

with $\lambda_{oh} = 0.5$, and $\lambda_{lte-g} = 5$ in all experiments.

C. Theoretical foundations

As previously discussed in the main text, knowledge forgetting can be mitigated when the gradient of the current task maintains orthogonality with respect to the knowledge of previous tasks. From a modeling perspective, the prevention of knowledge forgetting essentially requires that for a given input e^t, the model's output after learning a new task t+1 should remain consistent with its output immediately following the learning of task t. This relationship can be formally expressed by the following equation:

$$f_\theta(p^{t+1}, e^t) = f_\theta(p^t, e^t) \tag{13}$$

where e^t denotes the feature embeddings from the old task t, p^t and p^{t+1} denotes the prompts trained at task t and $t+1$, respectively. θ is the model parameter. This formulation ensures that the acquisition of new knowledge does not interfere with the retention of previously learned information, thereby maintaining the model's performance across sequential tasks. To achieve Eq. (13), we begin with the implementation of prompt-based continual learning (PCL). In this framework, after training task $t+1$, we concatenate the prompts p^{t+1} with the embedding sequences e^t (i.e., inputs from task t) along the embedding dimension: $Z_t^{t+1} = \begin{bmatrix} p^{t+1} \\ e^t \end{bmatrix}$. Using the weight matrices W_q, W_k, and W_v, PCL leverages a transformer architecture to compute the query $Q_t^{t+1} = W_q Z_t^{t+1}$ and key $K_t^{t+1} = W_k Z_t^{t+1}$. The attention matrix is then computed as:

$$A_t^{t+1} = \mathrm{softmax}\left(\frac{Q_t^{t+1} K_t^{t+1^T}}{\sqrt{d/h}}\right). \tag{14}$$

Here, the denominator serves as a normalization factor, so our focus shifts to the numerator $Q_t^{t+1} K_t^{t+1^T}$, which can be expanded as $W_q Z_t^{t+1} Z_t^{t+1^T} W_k^T$. Notably, the visual encoder weights W_q and W_k remain frozen during training, leaving the trainable parameters as:

$$Z_t^{t+1} \cdot Z_t^{t+1^T} = \begin{bmatrix} p^{t+1} \\ e^t \end{bmatrix} \begin{bmatrix} p^{t+1^T} & e^{t^T} \end{bmatrix} = \begin{bmatrix} p^{t+1}p^{t+1^T} & p^{t+1}e^{t^T} \\ e^t p^{t+1^T} & e^t e^{t^T} \end{bmatrix}. \tag{15}$$

In contrast, the previous embedding Z_t^t is constructed by concatenating the prompts trained on task t with the embedding sequences e^t:

$$Z_t^t \cdot Z_t^{t^T} = \begin{bmatrix} p^t \\ e^t \end{bmatrix} \begin{bmatrix} p^{t^T} & e^{t^T} \end{bmatrix} = \begin{bmatrix} p^t p^{t^T} & p^t e^{t^T} \\ e^t p^{t^T} & e^t e^{t^T} \end{bmatrix}. \tag{16}$$

To satisfy Eq. (13), that is, let $Z_t^t \cdot Z_t^{t^T} = Z_t^{t+1} \cdot Z_t^{t+1^T}$, the following system of equations be obtained:

$$\begin{cases} p^{t+1}p^{t+1^T} = p^t p^{t^T}, \\ e^t p^{t+1^T} = e^t p^{t^T}, \\ p^{t+1}e^{t^T} = p^t e^{t^T}. \end{cases} \tag{17}$$

we decompose p^{t+1} into p^t and an update term Δp, where Δp represents the gradient of prompts during task $t+1$ training. Expanding the first condition:

$$\begin{aligned} p^{t+1}p^{t+1^T} &= (p^t + \Delta p)(p^t + \Delta p)^T \\ &= p^t p^{t^T} + p^t \Delta p^T + \Delta p p^{t^T} + \Delta p \Delta p^T. \end{aligned} \tag{18}$$

Neglecting the higher-order infinitesimal term $\Delta p \Delta p^T$, the condition $p^{t+1}p^{t+1^T} = p^t p^{t^T}$ holds if $p^t \Delta p^T = 0$. Similarly, transforming the second condition:

$$e^t p^{t+1^T} = e^t (p^{t^T} + \Delta p^T) = e^t p^{t^T} + e^t \Delta p^T = e^t p^{t^T}. \tag{19}$$

Eliminating $e^t p^{t^T}$ from both sides gives $e^t \Delta p^T = 0$. Notably, this condition also satisfies the third term in Eq. (17) since $e^t p^{t+1^T}$ is the transpose of $p^{t+1} e^{t^T}$. Thus, the key observation is that constraining the prompt gradients using the following conditions effectively mitigates forgetting:

$$\begin{cases} e^t \Delta p^T = 0, \\ p^t \Delta p^T = 0. \end{cases} \tag{20}$$

Strictly speaking, let $(e^t + p^t)\Delta p^T = s^t \Delta p^T = 0$ is not entirely equivalent to Eq. (20). However, based on the conclusions of method PGP [1], enforcing $s^t \Delta p^T = 0$ can, to some extent, approximate the effect of Eq. (20). Our analysis suggests that while $s^t \Delta p^T = 0$ does not directly derive Eq. (20), both share the same fundamental objective: ensuring the prompt gradient remains orthogonal to prior knowledge (e^t and p^t), thereby preventing interference. Experimental results confirm that this approach can mitigate

catastrophic forgetting to some degree. A more rigorous proof remains an open direction for further exploration. It is worth noting that the above proof is for ordinary continuous learning scenarios. In Federated class-incremental learning, due to data being scattered across clients, we redefine s^t using synthetic data and aggregated prompts in the server.

References

[1] Jingyang Qiao, Xin Tan, Chengwei Chen, Yanyun Qu, Yong Peng, Yuan Xie, et al. Prompt gradient projection for continual learning. In *ICLR*, 2023. 3

[2] Minh-Tuan Tran, Trung Le, Xuan-May Le, Mehrtash Harandi, and Dinh Phung. Text-enhanced data-free approach for federated class-incremental learning. In *CVPR*, pages 23870–23880, 2024. 1, 2

[3] Jie Zhang, Chen Chen, Weiming Zhuang, and Lingjuan Lyu. Target: Federated class-continual learning via exemplar-free distillation. In *ICCV*, pages 4782–4793, 2023. 1