Isolation and Integration: A Strong Pre-trained Model-Based Paradigm for Class-Incremental Learning

Wei Zhang[1][0009−0009−8202−6186], Yuan Xie[1], Zhizhong Zhang[1], and Xin Tan[1]

[1]East China Normal University, Shanghai, China
weizhang_01@stu.ecnu.edu.cn, {yxie,zzzhang,xtan}@cs.ecnu.edu.cn

Abstract. Continual learning aims to effectively learn from streaming data, adapting to emerging new classes without forgetting old ones. Conventional models without pre-training are constructed from the ground up, suffering from severely catastrophic forgetting. In recent times, pre-training has made significant strides, opening the door to extensive pre-trained models for continual learning. To avoid obvious stage learning bottlenecks in traditional single-backbone networks, we propose a brand-new stage-isolation based class incremental learning framework, which leverages parameter-efficient tuning technique to finetune the pre-trained model for each task, thus mitigating information interference and conflicts among tasks. Simultaneously, it enables the effective utilization of the strong generalization capabilities inherent in pre-trained networks, which can be seamlessly adapted to new tasks. Then, we fuse the features acquired from the training of all backbone networks to construct a unified feature representation. This amalgamated representation retains the distinctive features of each task while incorporating the commonalities shared across all tasks. Finally, we use the selected exemplars to compute the prototype as the classifier weights to make final prediction. We conduct extensive experiments on different class incremental learning benchmarks and settings, results indicate that our method consistently outperforms other methods with a large margin.

Keywords: Continual Learning · Class-Incremental Learning · Pre-Trained Models · Parameter-Efficient Tuning.

1 Introduction

Continual learning, also known as incremental learning or lifelong learning, aims to learn effectively from streaming data, that is adapt to emerging new class without forgetting old ones [19, 38]. Current efforts are mainly based on the premise of learning from scratch. However, with the rapid development of pre-training techniques, recent advancements in pre-training have significantly facilitated the utilization of pre-trained models for downstream tasks[8]. These pre-trained models are typically trained with massive data, resulting in strong generalizability[41, 18]. Therefore, continual learning with pre-trained models is

emerging as a promising direction and is attracting increasing attention [41, 36, 31, 42, 22, 32, 29].

When confronted with a growing number of new tasks, conventional single-backbone models often encounter several significant challenges. Firstly, as new tasks are introduced, the existing model may grapple with knowledge conflicts stemming from disparities between the feature distributions of new and old tasks, resulting in a decline in the model's performance when handling novel tasks [11]. Secondly, upon learning a new task, the existing model might undergo catastrophic forgetting [7, 16], wherein the feature representations of old tasks are overwritten by the newly acquired knowledge, causing performance degradation on the former tasks. Lastly, as the number of new tasks increases, the model may suffer from a decline in generalization performance, *i.e.*, its ability to generalize to new tasks deteriorates [37].

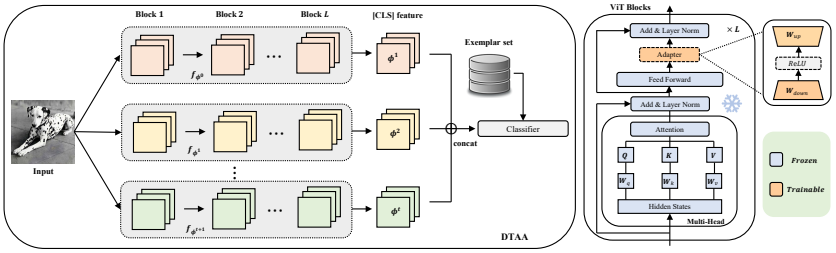


Fig. 1: Illustration of DTAA. **Left:** the architecture of DTAA. At each task, we dynamically tune the pre-trained model with adapter to learn new concepts and keep the pre-trained backbone frozen, then we fuse the features acquired from the training of all backbone networks to construct a unified feature representation and feed it to the prototype-based classifier. **Right:** the framework of ViT blocks with adapter. Red modules in the figure are trainable, while blue ones are frozen.

To tackle these challenges, drawing inspiration from techniques like DER [35], we employ a novel approach, Dynamically Tuning pre-trained model to Adapt and Aggregate new concepts (DTAA), as shown in Figure 1. Specifically, we introduce an additional pre-trained backbone network, equipped with adapters [10], for each new task. Each of these backbone networks is tasked with fine-tuning the feature representation specific to its associated task. This approach offers distinct advantages: it ensures that feature representations for different tasks are learned independently, thereby averting information interference and conflicts between tasks. Simultaneously, it enables the effective utilization of the strong generalization capabilities inherent in pre-trained networks, which can be seamlessly adapted to new tasks.

Subsequently, we fuse the features acquired from the training of all backbone networks to construct a unified feature representation. This amalgamated representation retains the distinctive features of each task while incorporating

the commonalities shared across all tasks. Such a holistic feature representation not only preserves the task-specific characteristics but also assimilates the overarching patterns common to all tasks. This integration enhances the model's adaptability and generalization prowess.

During the classification phase, we employ a unified classifier that caters to all tasks. This classifier takes the concatenated feature representation generated by the multi-backbone network as its input and conducts classification on this comprehensive feature set. This approach offers the distinct advantage of enabling the classifier to learn from the features of all tasks, eliminating the need for a separate classifier for each task. Consequently, it enhances the model's scalability and reduces computational overhead.

Upon the completion of learning for each task, we curate a subset of 20 instances for each category within that task, selecting those instances that exhibit the greatest similarity to the task's prototype. Subsequently, we recompute the prototype representations for the known categories using the exemplar samples drawn from all categories. These recalculated prototype representations then serve as the parameters for the classifier's existing categories in subsequent task learning endeavors. The resultant prototype representations encapsulate additional information derived from the fine-tuned model, encompassing task-specific features that enhance recognition performance, thereby ensuring a higher degree of generalization and adaptability.

Experiments showcase the exceptional performance of our approach in incremental learning scenarios. It proficiently handles the introduction of new tasks, the preservation of existing tasks, and simultaneously sustains the learning progress of established tasks during the incorporation of new ones. This underscores our method as an efficient and reliable solution to the multi-task incremental learning challenge.

To sum up, the main contributions of this work can be summarized as follows:

(1) We propose DTAA, which dynamically expand and tune the pre-trained network backbone with adapter for each task to solve the continual learning problem.

(2) We conduct extensive experiments to show that, DTAA reaches a new state-of-the-art performance in the different class incremental learning benchmarks and settings.

(3) We also compare DTAA to a set of strong continual learning baseline methods in a fair way using the same pre-trained backbone for all methods.

2 Realeated work

Class-Incremental Learning(CIL) Class-Incremental Learning, a paradigm focused on continuous learning of new classes while retaining knowledge of the old ones, has seen the development of various algorithms, broadly categorized into three groups. Regularization-based methods [1, 4, 26] devise strategies like knowledge distillation or parameter regularization terms to alleviate catastrophic forgetting. LwF [14] was a pioneering success in applying knowledge distillation

to CIL by enforcing consistent predicted probabilities among old classes. EWC [11] maintains an importance matrix to assess parameter significance, keeping vital ones static to preserve prior knowledge.Rehearsal-based methods [2, 17, 34, 28] store and employ exemplars from old classes to recover previous knowledge, displaying remarkable versatility and resilience. iCaRL [19] extends LwF with exemplar sets, aiding in better recall of past knowledge during the learning process.Structure-based methods [21, 27, 20] maintain learned parameters while allocating new ones or introducing additional networks to grasp novel concepts. DER [35], a prime example of this approach, expands a new backbone upon the arrival of a new task and aggregates features using a larger classifier.

CIL with pre-trained model The rapid evolution of pre-training techniques has significantly streamlined incremental learning by harnessing pre-trained models. DyTox [6] pioneered the exploration of pre-trained Vision Transformers (ViTs) [5] in CIL, expanding solely task tokens for each new task, thereby demanding considerably less memory compared to saving the entire backbone. L2P [32] and DualPrompt [31] also leverage pre-trained ViTs to progressively fine-tune models using adaptable parameters known as "prompts". In L2P, the pre-trained model remains fixed during the learning process, with the model solely optimizing prompts within the prompt pool to accommodate new concepts. DualPrompt extends this approach by incorporating complementary prompts into the pre-trained model, enabling the learning of both task-invariant and task-specific information. CODA-Prompt [22] further advances prompt search through the incorporation of attention mechanisms. Additionally, S-Prompts [30] capitalizes on the pre-trained language-vision model CLIP [18] for incremental learning, simultaneously mastering language and visual prompts.

Adapter tuning for pre-trained model Parameter-efficient tuning encompasses techniques and methodologies aimed at refining pre-trained models while conserving computational resources and minimizing the count of task-specific parameters. In [10], the concept of integrating an adapter layer into the Transformer architecture is introduced for the purpose of model fine-tuning. The adapter layer's design is uncomplicated, comprising a downward projection onto a reduced dimension, followed by a stratum of nonlinear activation, and subsequently an upward projection to restore the initial dimension. Furthermore, there is an intrinsic residual connection linking the input and output of the complete adapter layer. This adapter can be incorporated into diverse layers of a model, affording adaptability in tailoring the model for different tasks, as elucidated in [3].

3 Method

We begin by describing the problem setup and, along the way, introduce the notations in Sec.3.1. Then we present a minimum feasible prototype-based framework which serves as our baseline model in Sec.3.2. In Sec.3.3, we introduce the proposed method, which builds upon the baseline model.

3.1 Problem Setting

Class-incremental learning aims to learn from an evolving data stream with new classes to build a unified classifier. There is a sequence of B training tasks $\{\mathcal{D}^1, \mathcal{D}^2, ..., \mathcal{D}^B\}$, where $\mathcal{D}^b = \{(x_i^b, y_i^b)\}_{i=1}^{n_b}$ is the b-th incremental step with n_b instances. Here, the training instance $x_i^b \in \mathbb{R}^{W\times H\times C}$ belongs to class $y_i \in Y_b$, where Y_b is the label space of task b. $Y_b \cap Y_{b'} = \varnothing$ for $b \neq b'$. During the b-th training stage, we can only access data from $\mathcal{D}^b$ for model updating. This paper focuses on the exemplar-based CIL setting, where little historical data can be fetched for rehearsal. For example, 20 instances each class. The goal of CIL is to incrementally build a unified model for all seen classes, i.e., acquiring knowledge from new classes and meanwhile preserving knowledge from former ones. The model's capability is evaluated over all seen classes $\mathcal{Y}_b = Y_1 \cup ...Y_b$ after each incremental task. In general, a neural network at session t can be decoupled into an embedding function $f_{\phi^t}(\cdot) : \mathbb{R}^{W\times H\times C} \to \mathbb{R}^d$ and a classifier $g_{\psi^t}(\cdot) : \mathbb{R}^d \to \mathbb{R}^k$ that are parameterized by ϕ^t and ψ^t, respectively. Thus, the target is to fit a model $F(x) : X \to \mathcal{Y}_b$ that minimizes the empirical risk across all testing datasets:

$$\sum\nolimits_{(x_i,y_i)\in\mathcal{D}_t^1\cup...\mathcal{D}_t^b} l(g_{\psi^t}(f_{\phi^t}(x_j)), y_j) \tag{1}$$

where $l(\cdot,\cdot)$ measures the discrepancy between prediction and ground-truth label. $\mathcal{D}_t^b$ denotes the testing set of task b. A good CIL model satisfying Eq.1 has discriminability among all classes, which strikes a balance between learning new classes and remembering old ones.

Same as [41] , we use a pre-trained model (*e.g.*, a ViT) on ImageNet as the initialization of $F(x)$. In a plain ViT, the input encoding layer transforms the image into a sequence-like output features $x_e \in \mathbb{R}^{L\times d}$, where L is the sequence length. We assume the first token in x_e as the [CLS] token to simplify notation. x_e is then fed into the subsequent layers (*i.e.*, multi-head self-attention and feed forward layer) to produce the final embeddings. We treat the embedded [CLS] token as $\phi(x)$ for ViT.

3.2 A Simple Baseline

As indicated by [41] , pre-trained models are born with generalizability, which can be transferred to downstream tasks. They define a simple baseline, SimpleCIL, to transfer pretrained models for incremental tasks. With the embedding function $f_\phi(x)$ $frozen$ throughout the learning process, we extract average embedding (*i.e.*, prototype) of each class:

$$p_i = \frac{1}{K}\sum\nolimits_{j=1}^{|\mathcal{D}|} \mathbb{I}(y_j = i) f_\phi(x_j) \tag{2}$$

where $K = \sum_{j=1}^{|\mathcal{D}|} \mathbb{I}(y_j = i)$, $\mathbb{I}(\cdot)$ is the indicator function, $f_\phi(\cdot)$ is initialized by a pre-trained ViT. SimpleCIL sets the classifier weights to the average embeddings of each new class for classification, and outperforms many state-of-the-art methods even without any tuning on these downstream tasks. In this paper, we also adopt SimpleCIL as our baseline model.

3.3 Dynamically Adaption and Aggregation

In class-incremental learning, given the scarcity of data from previous tasks, it is challenging for single-models trained through gradient descent to maintain good feature representations for these tasks because the parameters of the previous task are overwritten by those learned by the new task, causing catastrophic forgetting. However, the forgetting of the previously learned feature representation is the core problem of CIL. Therefore, inspired by the model expansion approach, we fine-tune the pre-trained model with adapter and save a fine-tuned backbone network for each task. It ensures that feature representations for different tasks are learned independently, thereby averting information interference and conflicts between tasks. Simultaneously, it enables the effective utilization of the strong generalization capabilities inherent in pre-trained networks, which can be seamlessly adapted to new tasks. The overview of our proposed method DTAA is shown in Figure 1.

Denote the input of the Feed Forward Layer (FFL) in ViT as x_l, the output of the adapted FFL is formatted as:

$$\mathrm{FFL}(x_l) + \mathrm{ReLU}(x_l W_{down}) W_{up} \tag{3}$$

where W_{down} is down-projection operation to reduce the feature dimension, W_{up} is up-projection operation which projects it back to the original dimension, $ReLU$ is non-linear activation function.

At time step $t+1$, the models fine-tuned in previous tasks, $f_{\phi^i}(x)$, $i \leq t$, are frozen and a new model, $f_{\phi^{t+1}}(x)$, is learned from $\mathcal{D}_{t+1}$. The input x is then processed by all models, to produce a sequence of feature vectors

$$\phi^i(x) \leftarrow f_{\phi^i}(x),\ i = 1, 2, ..., t+1 \tag{4}$$

where $\phi^i(x)$ is the feature representation (i.e., embedded [CLS] token) of x under model $f_{\phi^i}(x)$.

Since each model $f_{\phi^i}(x)$ is fine-tuned on a large dataset $\mathcal{D}_i$ of the i-th task and then fixed, input x from the classes of task i can still be well represented by feature vector $\phi^i(x)$, after step $t+1$. Hence by combining all $\phi^i(x)$, $i \leq t+1$, the CIL model can obtain all necessary information to represent any input. This combination is usually implemented with the feature concatenation operation:

$$\phi(x) = \phi^0(x) \oplus \phi^1(x) \oplus \phi^2(x) \oplus ... \oplus \phi^{t+1}(x) \tag{5}$$

where $\phi^0(\cdot)$ is the original pre-trained model.

Then, we use a unified classifier for all tasks. This classifier receives the concatenated feature representation (i.e., $\phi(x)$) of the multi-backbone network as input and performs classification on this unified feature representation. Similar to SimpleCIL, we also use class prototypes as parameters to the classifier. The difference is that after learning each task, we save 20 examplar samples from each class that are closest to the prototype using the rehearsal algorithm:

$$g_\psi \leftarrow p_i = \frac{1}{K} \sum_{j=1}^{|\widetilde{\mathcal{D}}|} \mathbb{I}(y_j = i) f_\phi(x_j) \tag{6}$$

where $K = \sum_{j=1}^{|\widetilde{\mathcal{D}}|} \mathbb{I}(y_j = i)$, $\widetilde{\mathcal{D}}$ is the examplar set.

After the model is fine-tuned on each task, the prototype representation of the class is re-computed on the examplar set as the current classifier parameter. The resultant prototype representations encapsulate additional information derived from the fine-tuned model, encompassing task-specific features that enhance recognition performance, thereby ensuring a higher degree of generalization and adaptability.Finally, we use the Eq.(1) to make the final prediction.

4 Experiments

In this section, we present a series of experiments to investigate the performence of DTAA under different CIL settings. First, we detail the benchmark used for the experiments and discuss our implementation. Second, we compare DTAA with state-of-the-art methods on benchmark datasets to show the superiority.

4.1 Experimental Setups

Benchmark: Similar to L2P [32] and DualPrompt [31] , we evaluate CIL methods on the CIFAR-100 [12], ImageNet-R [9] and CUB-200 [24] benchmarks. CIFAR-100 consists of 60k images with a size of 32×32 from 100 classes, each class consists of 500 training and 100 testing samples. ImageNet-R contains 200 classes that are hard examples from ImageNet [13] or newly collected data of different styles (e.g. cartoon, graffiti, origami), split into 24k and 6k images with size of 224×224 for training and testing (similar ratio for each class), respectively. CUB-200 dataset, the most widely used dataset for fine-grained visual categorization tasks, contains a total of 11,788 image samples from 200 different bird species, of which the training set contains about 5,994 images and the test set contains about 5,794 images.

Dataset split: Adhering to the benchmark configuration outlined in [19, 38], we also adopt two types of dataset splits, *i.e.*, training from half and training from scratch. We unify them as "Base/B m, Inc n", which means the first incremental session contains m classes, and each following session contains n classes. $m = 0$ means the total classes are equally divided into each task. All classes are randomly shuffled with the same random seed before splitting for fair comparison. The evaluation process employs the original testing set to provide a comprehensive assessment.

Comparison methods: Following [23], we first compare to the current state-of-the-art pre-trained model based methods L2P [32], DualPrompt [31], CODA-Prompt [22], ADAM-Adapter [41]. Also, we conduct experiments on typical class-incremental learning algorithms, including DER [35], FOSTER [25], iCaRL [19], Coil [40] and MEMO [39], within the context of pre-trained models to make a comparison.

Training details: We implement all methods in PyTorch with a single NVIDIA RTX 3090 GPU, and choose the same network backbone ViT-B/16-IN1K for all compared methods for fair comparison. The model is trained with a

batch size of 48 for 20 epochs on CIFAR-100 and a batch size of 16 for 10 epochs on ImageNet-R, respectively. We use SGD with momentum for optimization, and the learning rate starts from 0.01 and decays with cosine annealing. For some rehearsal methods, we use the herding algorithm [33] to select 20 exemplars per class for rehearsal after each task training.

Evaluation metric: Assuming there are T tasks in total, we report the accuracy from the end session as $Acc = A_T$, and report the average accuracy over all sessions as $Acc = \frac{1}{T}\sum_{i=1}^{T} A_i$.

4.2 Comparison with State of the Art

In this section, we conduct a comparative analysis of DTAA against state-of-the-art approaches compatible with the Transformer architecture on the CIFAR-100 and ImageNet-R datasets. As depicted in Table 2, our comparison is conducted under the training from half and training from scratch configuration for CIFAR-100 and ImageNet-R benchmarks, respectively. The second column in the table represent the methods with and without exemplars. In the case of "FineTune", this approach involves continual training of a pre-trained model on new tasks, necessitating updates to all parameters and rendering it susceptible to severe catastrophic forgetting, and we involve it to make a comparison.

Table 1: Comparison of the methods with different settings on CIFAR-100 and ImageNet-R benchmarks. Avg and Last denote the average and final performance, respectively. The second column represent the methods with and without exemplars. The best performance is shown in bold.

Method	Exemplar	CIFAR B0-Inc10		IN-R B0-Inc20		CIFAR B50-Inc10		IN-R B100-Inc20	
		Avg	Last	Avg	Last	Avg	Last	Avg	Last
Coil	✓	87.33	79.84	77.85	71.75	85.26	80.27	79.40	74.37
DER	✓	88.79	78.72	80.95	75.22	86.67	81.36	80.59	77.83
iCaRL	✓	89.58	78.33	72.33	60.93	87.84	80.30	70.83	63.28
MEMO	✓	86.74	76.89	74.80	66.10	86.60	81.30	73.85	68.30
FOSTER	✓	91.61	87.18	82.13	76.02	90.41	87.69	**80.47**	76.77
L2P	✗	89.32	84.61	77.81	71.45	86.84	80.52	73.18	66.63
FineTune	✗	80.11	69.09	72.23	60.83	80.81	72.45	72.58	62.15
SimpleCIL	✗	82.32	76.21	67.05	61.28	78.67	76.21	63.51	61.28
DualPrompt	✗	87.38	82.30	74.35	68.67	86.08	80.92	70.15	65.08
CODA-Prompt	✗	91.31	86.93	78.68	74.72	88.13	83.63	76.61	73.03
ADAM-Adapter	✗	90.92	85.81	79.12	72.88	91.44	89.14	78.58	76.08
Ours	✓	**93.24**	**89.28**	**82.69**	**77.52**	**92.01**	**89.85**	80.17	**78.33**

Notably, as shown in second row, even when replacing the backbone of the traditional method with an existing pre-trained model, it exhibits a performance gap compared to our approach. DTAA achieves better results under different settings, outperforming alternative methods with rehearsal by a substantial margin. To provide specific insights, DTAA surpasses the second-place method, FOSTER

Table 2: Comparison of the methods on CUB-200 benchmarks. Avg and Last denote the average and final performance, respectively. The second column represent the methods with and without exemplars. The best performance is shown in bold.

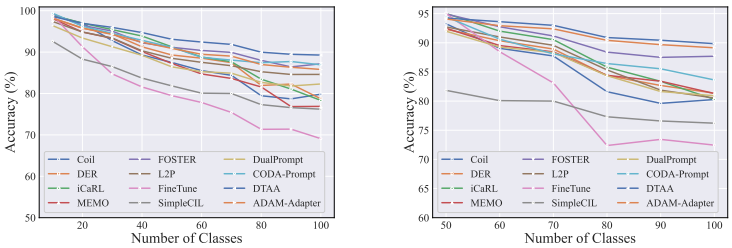
Method	Exemplar	CUB-200	B0-Inc10
		Avg	Last
Coil	✓	32.79	14.33
DER	✓	89.89	84.52
iCaRL	✓	88.69	81.38
MEMO	✓	89.32	84.61
FOSTER	✓	85.21	81.26
L2P	✗	73.19	60.90
FineTune	✗	64.89	43.55
SimpleCIL	✗	90.95	85.16
DualPrompt	✗	78.93	67.13
CODA-Prompt	✗	74.32	65.99
ADAM-Adapter	✗	90.91	85.20
Ours	✓	**91.08**	**86.13**

[25], by 1.63% and 0.56% in average accuracy on the CIFAR-100 and ImageNet-R datasets under the setting of training form scratch, respectively. On the other hand, it is only slightly lower than FOSTER [25] by about 0.3% average accuracy on the training from half setting of the Imagenet-R dataset and leads FOSTER [25] by 1.6% average accuracy on the CIFAR-100 dataset. It's worth noting that FOSTER [25], a powerful competitive method, dynamically expands new modules to adapt to the residuals between the target model and the original model's output, is similar to which our method is based. Furthermore, it's worth noting that approaches excelling on the CIFAR-100 dataset may not perform as effectively on the ImageNet-R dataset due to significant domain differences. However, structure-based methods (*e.g.*, DER [35], FOSTER [25] and ours) consistently delivers strong performance on the ImageNet-R dataset, maintaining its position as the top performer.

As shown in the third row, Our method also shows considerable advantages compared with some recent methods based on pre-trained models, which are based on prompt or adapter and do not require exemplars. For example, on the training from scratch setting, DTAA leads the CODA-Prompt [22] by about 1.93% average accuracy on the CIFAR-100 data set, and leads the ADAM-Adapter [41] by about 3.57% average accuracy on the ImageNet-R data set. Finally, we show incremental performance for each step of all the compared methods in Figure 2. For more complex incremental tasks, *e.g.* 20-step on CUB-200 benchmarks, we can see a similar case, which further proves the validity of our approach.

We attribute this excellent performance to the design of the isolation and integration: isolation ensures that feature representations for different tasks are learned independently, thereby averting information interference and conflicts between tasks. Simultaneously, it enables the effective utilization of the strong generalization capabilities inherent in pre-trained networks, which can be seam-

lessly adapted to new tasks. On the other hand, integration fuses the features acquired from the training of all backbone networks to construct a unified feature representation. This amalgamated representation retains the distinctive features of each task while incorporating the commonalities shared across all tasks. Such a holistic feature representation not only preserves the task-specific characteristics but also assimilates the overarching patterns common to all tasks. This underscores the robustness and stability of our approach.



(a) CIFAR B0-Inc10 (b) CIFAR B50-Inc10

(c) ImageNet-R B0-Inc20 (d) ImageNet-R B100-Inc20

Fig. 2: The incremental performance for each step with ViT-B/16-IN1K as the backbone. (a) and (c) are under the setting of training from scratch, (b) and (d) are in the 5-step setting which half of the total classes are base classes.

4.3 Ablation Study

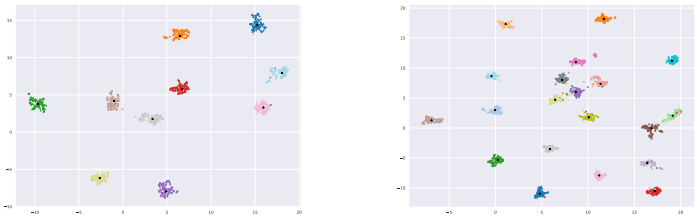
In this section, we conduct an ablation study to evaluate the contribution of our method.

The effect of adapter. Table 3 summarizes the results of our ablative experiments, which explore the impact of the location and number of adapters on experimental performance in the CIFAR-100 B0-Inc10 setting. Blocks-ID is "None" means that only the prototype classifier is used without using the adapter to fine-tune the network. Experimental results demonstrate that the performance of adapters in various locations (*i.e.*, front, middle, and rear) within the 12-blocks

ViT network, or the adjustment of their numbers, remains largely consistent. However, the network performs best when each block is fine-tuned using an adapter. We believe that placing adapters near the input layer may enable the model to adapt to task-specific features in the early stages, while placing them in deeper positions may capture more abstract and general features. This is because when adapters are placed at every location, the network achieves the best performance.

Table 3: Comparison of how the location and number of adapters impact accuracy performance in the CIFAR-100 B0-Inc10 setting. "Blocks-ID" indicates which block uses the adapter for fine-tuning. Avg and Last denote the average and final performance, respectively. The best performance is shown in bold.

Blocks-ID	CIFAR B0-Inc10	
	Avg	Last
None	82.32	76.21
1-2	92.70	88.44
5-6	92.71	88.37
10-11	92.67	88.36
1-4	92.66	88.30
5-8	92.71	88.43
9-12	92.65	88.10
1-12	**93.24**	**89.28**



(a) 1st stage on CIFAR B0-Inc10 (b) 2nd stage on CIFAR B0-Inc10

Fig. 3: The visualization of the learned decision boundaries between two incremental sessions.

Visualization of incremental stages. We visualize the learned decision boundaries with t-SNE [15] on CIFAR-100 dataset between two incremental sessions, as shown in Figure 3. The classes from the first and second incremental

tasks are in colorful markers. Correspondingly, the class prototypes are represented by asterisks with black. Based on these visualizations, it is clear that models fine-tuned with adapters exhibit strong performance. They proficiently group instances into their respective classes. The central placement of class prototypes confirms their representativeness in the recognition process. Notably, when transitioning the model from the initial stage to the following stage, DTAA exhibits commendable performance across both previously learned and newly introduced classes. These visual representations offer compelling evidence of DTAA's robust generalization and adaptability capabilities.

5 Conclusion

In this paper, we propose a strong pre-trained model-based paradigm which dynamically tuning the pre-trained model with adapter to learn new concepts and keep the pre-trained backbone frozen to retain its good generalization ability, and then we introduce a prototype based classifier which receives the concatenated feature representation to make a final prediction. We conduct exhaustive experiments on the two major incremental classification benchmarks and compare traditional and recent state-of-the-art methods within the context of pre-trained models. The experimental results show that our method consistently performs better than other methods with a sizable margin.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grants 62222602, 62302167, 62176224, 62106075, 61972157 and U23A20343, in part by Science and Technology Commission under Grant 21511100700, in part by CCF-Tencent Rhino-Bird Young Faculty Open Research Fund under Grant RAGR20230121.

References

1. Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., Tuytelaars, T.: Memory aware synapses: Learning what (not) to forget. In: Proceedings of the European conference on computer vision (ECCV). pp. 139–154 (2018)
2. Buzzega, P., Boschini, M., Porrello, A., Abati, D., Calderara, S.: Dark experience for general continual learning: a strong, simple baseline. Advances in neural information processing systems **33**, 15920–15930 (2020)
3. Chen, S., Ge, C., Tong, Z., Wang, J., Song, Y., Wang, J., Luo, P.: Adaptformer: Adapting vision transformers for scalable visual recognition. Advances in Neural Information Processing Systems **35**, 16664–16678 (2022)
4. Dhar, P., Singh, R.V., Peng, K.C., Wu, Z., Chellappa, R.: Learning without memorizing. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5138–5146 (2019)

5. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
6. Douillard, A., Ramé, A., Couairon, G., Cord, M.: Dytox: Transformers for continual learning with dynamic token expansion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9285–9295 (2022)
7. French, R.M., Chater, N.: Using noise to compute error surfaces in connectionist networks: A novel means of reducing catastrophic forgetting. Neural computation **14**(7), 1755–1769 (2002)
8. Han, X., Zhang, Z., Ding, N., Gu, Y., Liu, X., Huo, Y., Qiu, J., Yao, Y., Zhang, A., Zhang, L., et al.: Pre-trained models: Past, present and future. AI Open **2**, 225–250 (2021)
9. Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., et al.: The many faces of robustness: A critical analysis of out-of-distribution generalization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8340–8349 (2021)
10. Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., Gelly, S.: Parameter-efficient transfer learning for nlp. In: International Conference on Machine Learning. pp. 2790–2799. PMLR (2019)
11. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al.: Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences **114**(13), 3521–3526 (2017)
12. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
13. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems **25** (2012)
14. Li, Z., Hoiem, D.: Learning without forgetting. IEEE transactions on pattern analysis and machine intelligence **40**(12), 2935–2947 (2017)
15. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(11) (2008)
16. Nguyen, C.V., Achille, A., Lam, M., Hassner, T., Mahadevan, V., Soatto, S.: Toward understanding catastrophic forgetting in continual learning. arXiv preprint arXiv:1908.01091 (2019)
17. Prabhu, A., Torr, P.H., Dokania, P.K.: Gdumb: A simple approach that questions our progress in continual learning. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16. pp. 524–540. Springer (2020)
18. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021)
19. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: icarl: Incremental classifier and representation learning. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 2001–2010 (2017)
20. Rusu, A.A., Rabinowitz, N.C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., Hadsell, R.: Progressive neural networks. arXiv preprint arXiv:1606.04671 (2016)

21. Serra, J., Suris, D., Miron, M., Karatzoglou, A.: Overcoming catastrophic forgetting with hard attention to the task. In: International conference on machine learning. pp. 4548–4557. PMLR (2018)
22. Smith, J.S., Karlinsky, L., Gutta, V., Cascante-Bonilla, P., Kim, D., Arbelle, A., Panda, R., Feris, R., Kira, Z.: Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11909–11919 (2023)
23. Sun, H.L., Zhou, D.W., Ye, H.J., Zhan, D.C.: Pilot: A pre-trained model-based continual learning toolbox. arXiv preprint arXiv:2309.07117 (2023)
24. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The caltech-ucsd birds-200-2011 dataset (2011)
25. Wang, F.Y., Zhou, D.W., Ye, H.J., Zhan, D.C.: Foster: Feature boosting and compression for class-incremental learning. In: European conference on computer vision. pp. 398–414. Springer (2022)
26. Wang, L., Zhang, M., Jia, Z., Li, Q., Bao, C., Ma, K., Zhu, J., Zhong, Y.: Afec: Active forgetting of negative transfer in continual learning. Advances in Neural Information Processing Systems **34**, 22379–22391 (2021)
27. Wang, L., Zhang, X., Li, Q., Zhu, J., Zhong, Y.: Coscl: Cooperation of small continual learners is stronger than a big one. In: European Conference on Computer Vision. pp. 254–271. Springer (2022)
28. Wang, L., Zhang, X., Yang, K., Yu, L., Li, C., Hong, L., Zhang, S., Li, Z., Zhong, Y., Zhu, J.: Memory replay with data compression for continual learning. arXiv preprint arXiv:2202.06592 (2022)
29. Wang, R., Duan, X., Kang, G., Liu, J., Lin, S., Xu, S., Lü, J., Zhang, B.: Attriclip: A non-incremental learner for incremental knowledge learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3654–3663 (2023)
30. Wang, Y., Huang, Z., Hong, X.: S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning. Advances in Neural Information Processing Systems **35**, 5682–5695 (2022)
31. Wang, Z., Zhang, Z., Ebrahimi, S., Sun, R., Zhang, H., Lee, C.Y., Ren, X., Su, G., Perot, V., Dy, J., et al.: Dualprompt: Complementary prompting for rehearsal-free continual learning. In: European Conference on Computer Vision. pp. 631–648. Springer (2022)
32. Wang, Z., Zhang, Z., Lee, C.Y., Zhang, H., Sun, R., Ren, X., Su, G., Perot, V., Dy, J., Pfister, T.: Learning to prompt for continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 139–149 (2022)
33. Welling, M.: Herding dynamical weights to learn. In: Proceedings of the 26th Annual International Conference on Machine Learning. pp. 1121–1128 (2009)
34. Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y., Fu, Y.: Large scale incremental learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 374–382 (2019)
35. Yan, S., Xie, J., He, X.: Der: Dynamically expandable representation for class incremental learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3014–3023 (2021)
36. Zhang, G., Wang, L., Kang, G., Chen, L., Wei, Y.: Slca: Slow learner with classifier alignment for continual learning on a pre-trained model. arXiv preprint arXiv:2303.05118 (2023)

37. Zheng, Z., Ma, M., Wang, K., Qin, Z., Yue, X., You, Y.: Preventing zero-shot transfer degradation in continual learning of vision-language models. arXiv preprint arXiv:2303.06628 (2023)
38. Zhou, D.W., Wang, Q.W., Qi, Z.H., Ye, H.J., Zhan, D.C., Liu, Z.: Deep class-incremental learning: A survey. arXiv preprint arXiv:2302.03648 (2023)
39. Zhou, D.W., Wang, Q.W., Ye, H.J., Zhan, D.C.: A model or 603 exemplars: Towards memory-efficient class-incremental learning. arXiv preprint arXiv:2205.13218 (2022)
40. Zhou, D.W., Ye, H.J., Zhan, D.C.: Co-transport for class-incremental learning. In: Proceedings of the 29th ACM International Conference on Multimedia. pp. 1645–1654 (2021)
41. Zhou, D.W., Ye, H.J., Zhan, D.C., Liu, Z.: Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need. arXiv preprint arXiv:2303.07338 (2023)
42. Zhou, D.W., Zhang, Y., Ning, J., Ye, H.J., Zhan, D.C., Liu, Z.: Learning without forgetting for vision-language models. arXiv preprint arXiv:2305.19270 (2023)