When Prompt-based Incremental Learning Does Not Meet Strong Pretraining

Yu-Ming Tang[1,3] Yi-Xing Peng[1,3] Wei-Shi Zheng[1,2,3]*

[1]School of Computer Science and Engineering, Sun Yat-sen University, China
[2]Peng Cheng Laboratory, Shenzhen, China
[3]Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education, China
{tangym9, pengyx23}@mail2.sysu.edu.cn wszheng@ieee.org

Abstract

Incremental learning aims to overcome catastrophic forgetting when learning deep networks from sequential tasks. With impressive learning efficiency and performance, prompt-based methods adopt a fixed backbone to sequential tasks by learning task-specific prompts. However, existing prompt-based methods heavily rely on ***strong*** *pretraining (typically trained on ImageNet-21k), and we find that their models could be trapped if the potential gap between the pretraining task and unknown future tasks is large. In this work, we develop a learnable* ***A****daptive* ***P****rompt* ***G****enerator (APG). The key is to unify the prompt retrieval and prompt learning processes into a learnable prompt generator. Hence, the whole prompting process can be optimized to reduce the negative effects of the gap between tasks effectively. To make our APG avoid learning ineffective knowledge, we maintain a knowledge pool to regularize APG with the feature distribution of each class. Extensive experiments show that our method significantly outperforms advanced methods in exemplar-free incremental learning without (strong) pretraining. Besides, under strong pretraining, our method also has comparable performance to existing prompt-based models, showing that our method can still benefit from pretraining. Codes can be found at* https://github.com/TOM-tym/APG

1. Introduction

Deep neural networks (DNNs) have become powerful tools in various fields [5, 7, 12, 15, 28, 56]. However, when facing sequential training tasks, DNNs learn new tasks along with severe performance degradation on previous tasks in the absence of old data, which is the notorious catastrophic forgetting [11, 41, 49]. Incremental learning aims to overcome catastrophic forgetting in DNNs, pushing DNNs toward complex real-world applications, *e.g.* AI robotics [4, 10, 42, 52] or self-driving [13, 16, 43, 45]. Previous works usually maintain a memory buffer with a handful of old samples for rehearsal when learning new tasks [18, 26, 48, 54, 58, 62, 63, 65]. Since keeping old data may be infeasible due to privacy/storage concerns, another branch of work [64, 66, 67] explores exemplar-free incremental learning, which tunes the network based on the introduced priors but the performance still far falls behind those of rehearsal-based methods.

Recently, an appealing development [59–61] based on prompting [21, 27, 29, 34, 35] manages to encode knowledge into sets of prompts to steer a frozen backbone for handling sequential tasks. In addition to impressive performance, it has several benefits. (1) The catastrophic forgetting prob-

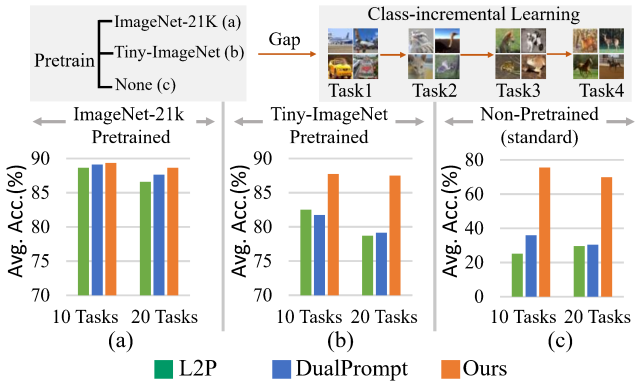


Figure 1. Experimental comparison between our adaptive prompting scheme and other prompted-based methods [60, 61] on the CIFAR100 [25] dataset. 'Non-pretrained' (a standard protocol in incremental learning) means the data from the first task is used to pretrain the backbone. (a) With an intensive pretrained backbone, all three methods perform well. (b) When swapping to Tiny-ImageNet (200 classes) pretrained weights, the performance of other methods clearly drops, while ours does not. (c) Our method significantly outperforms other methods in the presence of a large semantic gap between pretraining task and unknown future tasks.

*Corresponding author

lem is effectively alleviated since the backbone is fixed; (2) learning prompts instead of backbone significantly reduces training costs and improves learning efficiency; (3) prompt-based methods are free from keeping exemplars. To employ prompts for task-agnostic class-incremental learning, a crucial step is to select task-specific prompts given any input images. Existing methods maintain a prompt pool and retrieve prompts by directly computing the similarity between the image feature extracted by the pretrained model and the prompts in the pool, which is simple yet effective with a strong pretrained model. However, as the pretrained model dominates the retrieval, such a non-learnable retrieval process will be problematic because the future tasks are unknown and the gap between the pretraining task and the unknown future tasks could be large. As in Fig. 1 (c), when the first task in incremental learning is used for pretraining, the classes in the pretraining task are totally different from other tasks, which we refer to as a semantic gap that degenerates existing models. Although the semantic gap between domains is also studied in some works such as transfer learning [14, 44, 53], their works do not consider the forgetting problem in sequential tasks. It is necessary to emphasize that the intention of this work is **NOT** to refuse pretraining but to propose a more general method that does not rely heavily on strong pretraining and can benefit from it if task-related pretraining is available. For more experiments regarding the necessity of our work, please refer to the supplementary materials.

In this work, we develop a learnable **A**daptive **P**rompt **G**enerator (APG) to effectively bridge the potential gap between pretraining tasks and unknown future tasks. The core of our method is to unify the prompt retrieval and the prompt learning process into a learnable prompt generator. In this way, the whole prompting process can be optimized to reduce the negative effects of the gap between tasks effectively. Besides, rather than retrieving prompts from a fixed-size prompt pool, learning to generate prompts enhances the expression ability of prompts. As a result, the APG can be applied to a model without strong pretraining, and notably, the employment of APG does not discount the effort on overcoming forgetting since the backbone is still fixed.

For incremental learning, APG holds an extendable prompt candidate list for aggregating knowledge from seen tasks into a group of prompts. To adaptively prompt the backbone, the knowledge aggregation in APG is conditioned on the immediate feature from the backbone. In addition, we form a knowledge pool to summarize the knowledge encoded in the feature space. The summarized knowledge is further used to regularize the AGP to prevent it from learning ineffective knowledge.

In summary, our contributions are as follows. (1) We propose a learnable adaptive prompt generator (APG) to reduce the negative effects of the gap between the pretraining task and unknown future tasks, which is critical but ignored by previous work. Our adaptive prompting eases the reliance on intensive pretraining. (2) To regularize APG, we propose the knowledge pool, which retains the knowledge effectively with only the statistics of each class. (3) The extensive experiments show that our method significantly outperforms advanced exemplar-free incremental learning methods without pretraining. Besides, under strong pretraining, our method also achieves comparable satisfactory performance to existing prompt-based models.

2. Related Work

Non-pretrained Class-incremental Learning. Rehearsal-based class incremental learning methods [2,3,8,18,19,22, 38, 54, 55] have access to a handful of old samples. With the retained samples, several works propose to distill old knowledge to the current network [2,8,18,19,54] or to maintain the old feature space [22,55]. Different from rehearsal-based methods, we propose to get rid of the memory buffer and bridge the gap between old and new tasks with a learnable module. A straightforward idea to handle incremental learning is to dynamically expand the network for each task [1, 23, 39, 40, 46, 47, 50, 58, 63]. Although these methods are more intuitive, their methods often require careful design of network architecture and often require a memory bank like rehearsal-based methods. There also exists exemplar-free methods aiming to learn sequentially tasks without saving any images. These works propose to estimate the semantic drift [64], or to keep knowledge in class prototypes [66, 67]. Although exemplar-free methods proposed an appealing prospect for incremental learning, they typically introduce manually designed priors to the learning of new tasks, and human-designed priors are less generalizable, which makes their results unsatisfactory.

Prompt-based Class-incremental Learning. Inspired by prompting in natural language processing, some methods investigate prompt-based incremental learning and achieve great success [59–61]. These methods are generally inherited directly from NLP, utilizing a pretrained transformer as a base. S-prompt [59] focuses on domain-incremental learning and learns different prompts across domains. For class incremental learning, L2P [61] and DualPrompt [60] first propose to learn a pool of prompts and query the prompts based on the feature extracted by the pretrained backbone. DualPrompt [60] further proposes to attach complementary prompts to divide old knowledge into the general one and the expert one. The main idea of prompt-based methods is to encode knowledge from old tasks into sets of vectors (*i.e.* prompts) and retrieve them to instruct the backbone when the old data is not accessible. These methods are in need of a strong pretrained backbone to assist such a retrieval process. Different from their methods, we propose a unified prompting scheme that eases the reliance on intensive pretraining and makes it suitable for both pretrained and non-pretrained scenarios.

3. Adaptive-prompting for Incremental Learning

3.1. Preliminary

Class-incremental learning. For a deep model $\Phi(f(\cdot))$ consisting of a classifier $\Phi(\cdot)$ and the backbone $f(\cdot)$, the goal of class-incremental learning is to train the model $\Phi(f(\cdot))$ on tasks $\{\mathcal{T}_t\}_{t=1}^{n}$ sequentially so that the model can classify testing samples from any task. Specifically, at the t-th task $\mathcal{T}_t$, the training set is $\mathcal{D}_t = \{x_m^t, y_m^t\}_{m=1}^{N_t}$, where x_m^t is the m-th image at task $\mathcal{T}_t$ with label y_m^t. And the label space $\mathcal{Y}_t$ of task $\mathcal{T}_t$ is disjoint with other tasks, *i.e.* $\bigcap_{t=1}^{n} \mathcal{Y}_t = \emptyset$. Once the model finishes learning on task $\mathcal{T}_t$, the corresponding training set $\mathcal{D}_t$ will be dropped and becomes inaccessible when learning from $\mathcal{T}_{t+1}$, which raises a risk of forgetting old tasks when learning new ones.

Vision transformers and prompting. Different from convolutional neural networks, Vision Transformers (ViTs) treat the image as a sequence of tokens. Specifically, the image x_m^t will be converted into a sequence $\mathbf{E} \in \mathbb{R}^{N_E \times d}$ through patch embedding layers, where N_E is the total number of image patches (*i.e.* tokens) and d is the embedding dimension. An extra token called *class token* $e_{cls} \in \mathbb{R}^d$ will be concatenated to $\mathbf{E}$ to gather information from patches. The resultant token sequence $\mathbf{X} = [e_{cls}; \mathbf{E}]$ will be fed into transformer layers for feature extraction, and the core is self-attention operation:

$$\begin{aligned}\text{SelfAttn}(\mathbf{X}) &= \text{Attn}(\mathbf{X}\mathbf{W}^Q, \mathbf{X}\mathbf{W}^K, \mathbf{X}\mathbf{W}^V),\\ \text{Attn}(Q,K,V) &= \text{Softmax}(\frac{QK^T}{\sqrt{d}})V,\end{aligned} \tag{1}$$

where $\mathbf{W}^Q$, $\mathbf{W}^K$ and $\mathbf{W}^V$ are learnable projections.

To facilitate visual feature extraction, a common prompting [21, 30, 32, 33, 35, 36] technique is to insert a set of learnable extra tokens $\mathbf{P} \in \mathbb{R}^{N_P \times d}$ into the original sequence as $[e_{cls}; \mathbf{P}; \mathbf{E}]$. In this work, we only discuss vanilla ViTs [7,56] due to their simplicity and versatility. We detail our method as follows, and an overview is in Fig. 2.

3.2. Adaptive Prompt Generation

Adaptive prompt generator. We propose an adaptive prompt generator (APG) for adaptively aggregating old and new knowledge. Assume there are N_L transformer layers denoted as $\{L_l\}_{l=1}^{N_L}$, and the output of the l-th transformer layer is $\mathbf{X}_l = L_l(L_{l-1}(\cdots L_1(x))) \in \mathbb{R}^{(N_E+1)\times d}$, where x[1] is the input image. The APG takes the intermediate feature as input, which is denoted as $v_l = \mathbf{X}_l[0,:]$, and outputs the class-specific prompts to facilitate feature extraction in deeper layers:

$$\mathbf{P} = \text{APG}(v_l) \in \mathbb{R}^{N_P \times d}. \tag{2}$$

Specifically, in the APG, we keep updating a prompt candidate list to maintain different types of knowledge. Starting from an empty list denoted as $\mathbf{I}_0 = []$, the list is extended at each task, and the list at task $\mathcal{T}_t$ is denoted as:

$$\mathbf{I}_t = \left[\mathbf{I}_{t-1}; \widehat{\mathbf{P}}_1, \ldots, \widehat{\mathbf{P}}_c, \ldots, \widehat{\mathbf{P}}_{|\mathcal{Y}_t|}\right], \tag{3}$$

where $\widehat{\mathbf{P}}_c \in \mathbb{R}^{N_g \times d}$ represents a group of prompts candidates, and there are totally $N_g \times |\mathcal{Y}_t|$ prompt candidates are extended at $\mathcal{T}_t$ to form $\mathbf{I}_t$. In order to generate prompts with class-specific knowledge adaptively for v_l, we firstly apply a projection module M_{in} to v_l, denoted as $z = M_{in}(v_l)$, and then adopt a cross-attention operation between the projection z and the candidate list $\mathbf{I}_t$:

$$\widetilde{\mathbf{P}} = \text{CrossAttn}(z, \mathbf{I}_t) = \text{MMHA}(z, \mathbf{I}_t, \mathbf{I}_t), \tag{4}$$

where the MMHA denotes Multi-output Multi-head Attention, an extension of the Multi-head attention (MHA) [57]. While the original MHA outputs the same number of tokens with the input query, in our case, the query is a single token z and we want to obtain N_P prompts after cross-attention. To avoid restricting $N_P = 1$, we directly extend the MHA and form the MMHA as follows. Firstly, with the input z and $\mathbf{I}_t$, we define the j-th token computed from the h-th attention head as:

$$r_h^j = \text{Attn}(z\mathbf{W}_{h,j}^Q, \mathbf{I}_t\mathbf{W}_{h,j}^K, \mathbf{I}_t\mathbf{W}_{h,j}^V). \tag{5}$$

Assuming the number of heads is n_h, we concatenate results from every head as the j-th token $\mathbf{R}_j$, and pass the token through a projection module as the output of MMHA:

$$\begin{aligned}\mathbf{R}_j &= \text{Concat}(r_1^j, \ldots, r_h^j, \ldots, r_{n_h}^j),\\ \widetilde{\mathbf{P}} &= \text{MMHA}(z, \mathbf{I}_t, \mathbf{I}_t) = [\mathbf{R}_1, \ldots, \mathbf{R}_{N_P}]\,\mathbf{W}^o.\end{aligned} \tag{6}$$

We further put the $\widetilde{\mathbf{P}}$ into an output projection module M_{out}, and the obtained tokens $\mathbf{P} = M_{out}(\widetilde{\mathbf{P}}) \in \mathbb{R}^{N_P \times d}$ are the adaptive prompts for current image x.

The generated prompts $\mathbf{P}$ are further integrated with the original output $\mathbf{X}_l$ of the l-th layer, and the token sequence for the next layer becomes $\mathbf{X}_l = [e_{cls,l}; \mathbf{P}; \mathbf{E}_l]$. we feed $\mathbf{X}_l$ into the remaining deeper layers $\{L_i\}_{i=l+1}^{N_L}$. The final output of the network with APG is denoted as $\mathbf{X}_{N_L} = [e_{cls,N_L}; \mathbf{P}_{N_L}; \mathbf{E}_{N_L}]$. We take the class token

[1]In the later part of Sec. 3.2, we will omit the task label t and the index m of image x_m^t for simplicity.

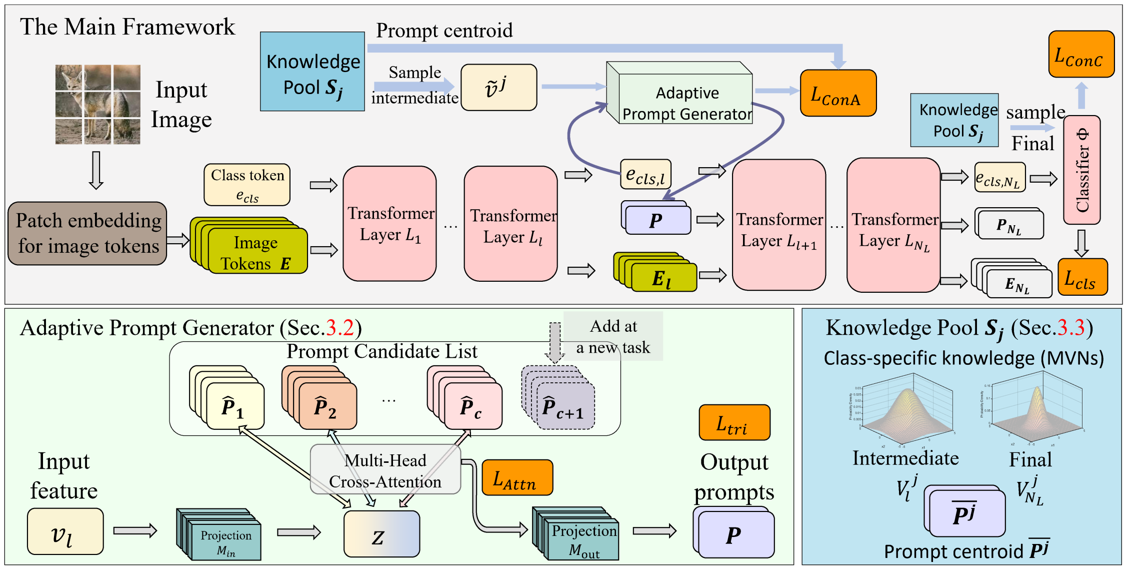


Figure 2. Overview of the proposed method. The Main framework is shown at the top part of the figure. The adaptive-prompting conducts in the middle of Layer L_l and L_{l+1}. With the help of the adaptive prompt generator (APG) which adaptively aggregates old and new knowledge, deeper layers are instructed by the old knowledge. A corresponding illustration of the APG is shown in the lower-left corner of the figure. To facilitate the knowledge aggregation, the knowledge pool (shown in the lower-right corner) consists of class-specific knowledge and the prompt centroid is used to constrain the APG and the classifier Φ of the old knowledge.

e_{cls,N_L} as the image feature. The classifier $\Phi(\cdot)$ is composed of a fully-connected layer and a softmax function and outputs the classification probability $\Phi(e_{cls,N_L})$.

It is worth mentioning that all components inside the APG are trainable, including the projection module M_{in}, M_{out}, the prompt candidate list $\mathbf{I}_t$ and the parameters in cross-attention operation ($\mathbf{W}^o$, $\{\mathbf{W}^{Q/K/V}_{h,j}\}$).

Optimization of the APG. The generated prompts are supposed to contain sufficient knowledge so that the model can handle the current task. Hence, we adopt a classification loss to learn new knowledge from the current task:

$$\mathcal{L}_{cls} = -\log \Phi(e_{cls,N_L})^{(y)}, \tag{7}$$

where $\Phi(e_{cls,N_L})^{(y)}$ is the y-th element of $\Phi(e_{cls,N_L})$.

Since $\mathcal{L}_{cls}$ is evaluated on the ultimate feature representation e_{cls,N_L}, we further directly apply constraints on the generated prompts and the prompt candidate list for more effective learning. For the prompt candidates, its information is aggregated through the cross-attention operation in Eq. 5. When computing the j-th token r_h^j in the h-th attention head, the attention score (Eq. 5) is calculated as:

$$A_h^j = \text{Softmax}(\frac{(z\mathbf{W}^Q_{h,j})(\mathbf{I}_t\mathbf{W}^K_{h,j})^T}{\sqrt{d}}), \tag{8}$$

where each element $A_h^{j,(c)}$ indicates the score between the projection z and the c-th prompt in the candidate list $\mathbf{I}_t$ under the projection $\mathbf{W}^Q_{h,j}, \mathbf{W}^K_{h,j}$. The prompt candidates should contain diverse knowledge from each class, otherwise, the APG can not generate prompts adaptively to various images. In other words, if the candidates only learn the knowledge from part of the classes, APG can not handle images from the other classes. Thus, we explicitly guide distinctive groups of prompt candidates to learn class-specific knowledge. We apply a class-specific constraint on the attention score:

$$L_{attn} = -\sum_{j=1}^{n_p}\sum_{h=1}^{n_h}\sum_{c\in\widehat{C}_y} \log(A_h^{j,(c)}), \tag{9}$$

where $\widehat{C}_y$ is the set of indexes of the prompts in $\widehat{P}_y$.

Moreover, since the images of the same class have similar characteristics, such as appearance, it is natural that the network can be prompted in a similar way when extracting features for different images in a class. To this end, we further constrain the relations between prompts through a triplet loss, encouraging the APG to learn the common knowledge of each class. Assume (x_1, x_2, x_3) is a triplet of images and their labels satisfy $y_1 = y_2, y_1 \neq y_3$. The corresponding prompts generated by the APG are $\mathbf{P}_1$, $\mathbf{P}_2$ and $\mathbf{P}_3$. Since x_1 and x_2 are from the same class while x_3 is from another class, we constrain the distance between $\mathbf{P}_1$ and $\mathbf{P}_2$ should be smaller than that between $\mathbf{P}_1$ and $\mathbf{P}_3$. Formally, the cosine distance between $\mathbf{P}_1$ and $\mathbf{P}_2$ is denoted as $d_p = \cos(\mathbf{P}_1, \mathbf{P}_2)$, and similarly we have $d_n = \cos(\mathbf{P}_1, \mathbf{P}_3)$. The triplet loss with margin is adopted:

$$\mathcal{L}_{tri} = [d_p - d_n + \alpha]_+, \tag{10}$$

where $[\cdot]_+$ denotes the hinge loss, and α is the margin.

Discussion. The proposed APG has the following advantages. First, APG continuously learns to generate diverse prompts conditioned on the input of the network. Hence, APG breaks the limitation of selecting prompts from a fixed-size prompt pool and alleviates the dependence on the pretrained model that is used for querying discrete prompts from a pool. Second, the prompt candidate list is extendable to handle growing knowledge. This enables the model capable of long incremental learning (see Sec. 4.2 and Sec. 4.3) as the APG can aggregate knowledge from different tasks by the cross-attention operation (Eq. 5).

3.3. Anti-forgetting Learning: A Knowledge Pool

Construction of the knowledge pool. Any trainable module faces the forgetting (*i.e.* old knowledge degradation) problem when learning from sequential tasks, including the proposed APG and the classifier $\Phi(\cdot)$. Hence, we develop a knowledge pool to better maintain old knowledge.

Once finishing learning at task $\mathcal{T}_t$, we summarize the knowledge in the training set $\mathcal{D}_t$ into the knowledge pool before dropping the dataset. Specifically, denote all the images in class c as $\mathcal{D}_t^c = \{x_m^t, y_m^t | x_m^t \in \mathcal{D}_t, y_m^t = c\}$. We feed these images into the backbone and get a set of features $V_c^l = \{v_m^l | x_m^t \in \mathcal{D}_t^c\}$ from the l-th transformer layer. Then the class centroid μ_c^l is then calculated by averaging features and the and Σ_c^l is a matrix where each element $\Sigma_c^{l,(i,k)}$ is the covariance between the i-th feature and the k-th feature. We use the class-specific statistics to form a multivariate normal distribution $\mathcal{N}_c^l = \mathcal{N}(\mu_c^l, \Sigma_c^l)$ for each class. Besides, to better constrain the APG, the feature centroid μ_c^l is used to extract corresponding prompts by feeding the centroid into the APG, *i.e.* $\overline{\mathbf{P}^c} = \text{APG}(\mu_c)$.

Finally, we form the knowledge pool for class c as $\mathcal{S}_c = (\mathcal{N}_c^l, \mathcal{N}_c^{N_L}, \overline{\mathbf{P}^c})$, where $\mathcal{N}_c^{N_L}$ is computed from the features from the last layer. Generally, we store the statistics for each class $\{\mathcal{S}_c\}_{c=1}^{N_t^{cls}}$, where $N_t^{cls} = \sum_t |\mathcal{Y}_t|$ is the number of seen classes so far.

Anti-forgetting learning. For exemplar-free incremental learning, we can only access the current training set $\mathcal{D}_t$ during task $\mathcal{T}_t$. The knowledge pool $\{\mathcal{S}_c\}_{c=1}^{N_{t-1}^{cls}}$ described above is used to maintain old knowledge. To alleviate forgetting in the APG, we sample from $\mathcal{N}_c^l$, and get intermediate knowledge vector $\widetilde{v}^c \sim \mathcal{N}_c^l$. With the vector and the corresponding prompt centroid $\overline{\mathbf{P}^j}$, we apply a constraint on APG:

$$\mathcal{L}_{ConA} = \phi(\text{APG}(\widetilde{v}^c), \overline{\mathbf{P}^c}), \tag{11}$$

where ϕ is a distance metric and we find that the $L1$ loss is useful. With the constraint, APG can be reminded of old knowledge and perform aggregation on them.

The classifier $\Phi(\cdot)$ is extended after each task, and needs to be updated during every task. Thus forgetting also happens at the classifier. Similar to the constraint on the APG, we sample the knowledge vector from $\widetilde{v}^c \sim \mathcal{N}_c^{N_L}$. Then a simple but effective cross-entropy loss is adopted:

$$\mathcal{L}_{ConC} = -\log \Phi(\widetilde{v}_m^c)^{(c)}, \tag{12}$$

where $\Phi(\widetilde{v}^c)^{(c)}$ is the c-th element of $\Phi(\widetilde{v}^c)$.

Training objective for the whole model. The objective function for optimizing the model and APG is summarized as follows:

$$\mathcal{L} = \mathcal{L}_{attn} + \mathcal{L}_{tri} + \mathcal{L}_{cls} + \mathcal{L}_{conA} + \mathcal{L}_{conC}. \tag{13}$$

For the backbone $f(\cdot)$, it is only trained on the first task and then is fixed in the case of without pretraining, and otherwise it is fixed all the time. The classifier $\Phi(\cdot)$ and APG are optimized across all tasks.

4. Experiments

4.1. Experiment Setup

Datasets. We conduct experiments on three datasets: CIFAR100 [25], ImageNet-Subset [6], and ImageNet-R [17]. CIFAR100 [25] contains 100 classes. There are 50,000 images for training and 10,000 images for evaluation. ImageNet [6] is a large-scale dataset containing 1,000 classes, 1300 images per class. We follow prior works [8, 18, 19, 26, 54, 58, 62, 63, 65] and use ImageNet-Subset which contains 100 classes. ImageNet-R [17] dataset contains newly collected data that are different styles of the original ImageNet classes. We follow prior work [60] to split the dataset with 80% samples for training and the rest 20% samples are used for evaluation.

Training and testing protocols. For the training and testing protocols, we follow previous works [8, 18, 19, 54, 58, 60, 61, 63] to conduct class-incremental learning. After every training stage, we evaluate the model by testing on the union of all testing sets of seen tasks. The classification accuracies in every task are averaged, and we report the *average accuracy* in all experiments [8, 19, 48, 54, 60–63]. When comparing with L2P [61] and DualPrompt [60], we also report the *forgetting* metric following their papers.

We do not save any images as a memory buffer and only keep the class statics of old classes on the feature level, following the exemplar-free class-incremental learning proposed in PASS [66] and SSRE [67].

Implementation details. For all experiments below, we use the vanilla ViT as our backbone. For the standard non-pretrained setting, we adjust the embedding dimension and number of heads to match the number of parameters of the widely used ResNet-18 [15]. For more information about the adjusted backbone, please refer to the appendix. We

Methods	ImageNet-Sub B50-T10		ImageNet-Sub B30-T14		CIFAR100 B50-T10		CIFAR100 B40-T20	
	Avg. Acc.↑	Forgetting↓	Avg. Acc.↑	Forgetting↓	Avg. Acc.↑	Forgetting↓	Avg. Acc.↑	Forgetting↓
L2P	25.11	1.72	29.92	3.03	36.55	2.61	18.84	3.02
DualPrompt	35.82	4.53	30.33	4.16	26.84	4.68	11.86	1.80
Ours	**75.52**	5.63	**69.87**	6.87	**66.68**	5.42	**62.41**	6.72

Table 1. Comparison with state-of-the-art prompt-based methods. Since existing prompt-based methods rely on a pretrained backbone, we treat the first task (*i.e.* 50 classes for the B50 setting) as the pretraining task and we conduct class-incremental learning on the rest classes using their methods. The best results are marked in **bold**. Results are averaged across three trials.

Methods	Mem Per cls	B50-T10	B30-T14	B50-T50
		Avg. Acc.(%)		
Imagine [54]	20	76.76	-	-
PODNet [8]	20	74.58	68.83	62.48
UCIR [18]	20	67.82	-	57.25
DDE [19]	20	75.41	65.11	-
CwD [51]	20	76.91	-	-
AFC [22]	20	75.75	72.47	71.97
Foster [58]	20	75.72	75.28	69.76
Imagine [54]	10	74.94	-	-
PODNet [8]	10	70.40	60.18	41.47
UCIR [18]	10	64.04	-	-
DDE [19]	10	73.00	-	-
AFC [22]	10	74.85	70.46	63.28
Foster [58]	10	71.46	65.91	58.79
EWC [24]	0	20.40	18.43	5.9
LwF_MC [31]	0	31.18	13.22	-
MUC [37]	0	35.07	-	-
SDC [64]	0	61.12	-	-
PASS [66]	0	61.80	-	-
SSRE [67]	0	67.69	-	-
Ours	0	**75.52**	**69.87**	**75.70**

Table 2. Comparison with state-of-the-art methods trained from scratch on ImageNet-SubSet. We list rehearsal-based methods with 20 images per class and their variant with 10 images. The best results among exemplar-free methods are marked in **bold**. Results are averaged across three trials.

follow the training strategy proposed in DeiT(without distillation) [56], using AdamW as the optimizer with initial learning of 5e-4. For class-incremental learning with a pretrained backbone, we follow prior works [60,61] using ViT-Base as our backbone.

Setting notation. We denote class-incremental learning settings in form of: 'Bx-Ty', which means the first task contains x classes, and the rest classes are evenly divided into y tasks to incrementally learn. For example, 'B50-T10' means the first task contains 50 classes and the rest classes are divided into 10 tasks evenly.

4.2. Non-pretrained Class-incremental Learning

In this section, we evaluate our method under the standard non-pretrained incremental learning setting [8, 9, 18, 19, 22, 24, 31, 37, 51, 58, 63, 64, 66, 67].

Comparison with prompted-based methods. In the popular non-pretrained scenario, we first compare our method with prompted-based methods L2P [61] and DualPrompt [60]. Since existing prompt-based methods rely on a pretrained backbone to conduct incremental learning, for the non-pretrained setting, we treat the first task during incremental learning as a 'pretraining' task and train a backbone in a supervised manner. Later, we re-implement L2P and DualPrompt according to the official code and conduct incremental learning on the rest tasks based on the pretrained weight. In this way, we can study the situation when future tasks are totally disjoined from the pre-trained task.

The experiment results are shown in Table 1. In terms of average accuracy, our proposed method outperforms L2P and DualPrompt by a large margin under different settings of the number of tasks. The forgetting metric of L2P and DualPrompt is incredibly low. We found they basically maintain the first task's performance and perform poorly on new tasks. Therefore the forgetting metric (which can be regarded as an average *maximum accuracy drop*) is low. Nevertheless, our method still achieves competitive results on forgetting compared to these two methods. We validate this observation by varying the number of classes at the first task and the total number of tasks on two different datasets.

Comparison with exemplar-free methods. To compare the proposed method with pioneering exemplar-free [31, 64, 66, 67] works fairly, we conduct experiments on ImageNet-Subset and CIFAR100 under different numbers of tasks. The detailed results on ImageNet-Subset are listed in Table 2. For the 10-task setting, we find that our method outperforms other exemplar-free methods by a large margin. Specifically, our method achieves 75.52% average accuracy, which is 7.95%, 13.84%, 14.52% higher than the state-of-the-art methods SSRE [67], PASS [66] and SDC [64]. On CIFAR100, our method still performs better than other exemplar-free methods(see Table 3). Specifically, our method outperforms the exemplar-free method SSRE by 2.56%, 1.65%, and 0.66% on average accuracy under 5, 10 and 20 task settings. **Comparison with rehearsal-based methods.** To show the effectiveness of our adaptive-prompting scheme, we further report the results of rehearsal-based methods on ImageNet-Subset and CIFAR100 under different numbers of tasks and different sizes of memory budgets.

The comparison results of different settings on ImageNet-Subset are shown in Table 2. Our method (w/o

Methods	Mem Per cls	B50-T10	B50-T5	B40-T20	B30-T14
		Avg. Acc.(%)			
Imagine [6]	20	66.47	68.01	-	-
PODNet [8]	20	66.70	62.65	58.92	53.91
UCIR [18]	20	62.77	56.75	42.88	45.07
CwD [51]	20	70.30	-	-	-
AFC [22]	20	64.98	66.49	62.19	72.47
DDE [19]	20	65.42	64.12	-	-
Foster [58]	20	77.54	67.69	56.87	53.80
Imagine [54]	10	64.41	67.08	-	-
PODNet [8]	10	60.09	62.03	55.78	55.93
UCIR [18]	10	56.95	59.95	42.88	45.07
AFC [22]	10	61.77	63.98	57.06	58.81
DDE [19]	10	64.41	61.47	-	-
Foster [58]	10	57.70	60.51	33.18	54.57
EWC [24]	0	24.48	21.20	15.89	15.89
LwF_MC [31]	0	45.93	27.43	20.07	21.05
MUC [37]	0	49.42	30.19	21.27	-
SDC [64]	0	56.77	57.00	58.90	-
PASS [66]	0	63.47	61.84	58.09	56.61
SSRE [67]	0	65.88	65.04	61.70	-
Ours	0	**66.68**	**66.83**	**62.41**	**57.57**

Table 3. Comparison with state-of-the-art methods trained from scratch on CIFAR100. We list rehearsal-based methods with 20 images per class and their variant with 10 images. The best results are marked in **bold**. Results are averaged across three trials.

Methods	CIFAR100 B0-5		CIFAR100 B0-10		CIFAR100 B0-20	
	A$\uparrow$	F$\downarrow$	A$\uparrow$	F$\downarrow$	A$\uparrow$	F$\downarrow$
ImageNet-21K Pretraining						
Upper Bound	90.80	-	91.22	-	91.53	-
FT	43.30	51.02	28.38	69.31	17.63	84.32
L2P [61]	88.65	88.64	88.64	7.35	86.57	9.31
DualPrompt [60]	**89.56**	**89.12**	89.12	**5.16**	87.61	6.92
Ours	88.48	89.35	**89.35**	6.01	**88.64**	**6.51**
Tiny-ImageNet Pretraining						
Upper Bound	89.42	-	89.37	-	88.44	-
FT	43.16	50.95	28.36	69.61	17.65	84.25
L2P [61]	84.68	9.49	82.51	7.78	78.69	8.12
DualPrompt [60]	84.54	8.25	81.71	9.70	79.13	11.18
Ours	**87.72**	**7.56**	**87.61**	**7.16**	**87.50**	**7.49**

Table 4. Comparison to state-of-the-art prompt-based methods with ImageNet-21k/Tiny-IamgeNet pretrained weights on CIFAR100. The best and the second best results are marked in **bold**. 'A' means Avg. Acc and 'F' indicates the forgetting metric. Our method effectively alleviates reliance on the strong pretraining (Tiny-ImageNet pretraining) and can still benefit from pretraining (ImageNet-21k pretraining).

exemplars) achieves 75.52% Avg. Acc. at B50-T10 setting, outperforming PODNet [8] and AFC [22] (both 20 exemplars per class) by 1.06% and 0.23%. To explore the effectiveness of our proposed method under a long incremental process, we also conduct experiments of B50-T50 (*i.e*. 50-task incremental learning). When the learning process gets longer from 10 tasks to 50 tasks, our performance maintains around 75% and outperforms other methods by a large margin. This is mainly because our proposed extendable candidate list in the APG enables our method to learn in long incremental tasks while other methods suffer greatly from the forgetting problem. When we tighten the memory budget, rehearsal-based methods face a drastic performance drop. Compared with Foster [58] and AFC [22] (both kept 10 images per class) under the B50-T10 setting, our method achieves 4.56% and 0.67% higher average accuracy. Even for a method like Imagine [54] which requires an extra auxiliary dataset to diversify the limited memory, our method still outperforms it by 0.7% accuracy when it kept 10 images per class. More results on CIFAR100 are in Table 3.

4.3. Evaluation with Pretrained Backbone

Incremental learning with intensive pretrained backbone. To demonstrate the flexibility of the proposed method, we further extend our method to the setting with pretrained weight and conduct experiments with the pretrained weight as previous methods [60, 61] did. We follow previous works [60, 61] to load an ImageNet 21k pretrained weight [56] for the vanilla ViT-Base/16. We conduct experiments on two different datasets CIFAR100 and ImageNet-R following DualPrompt [60]. For each dataset, we conduct 5, 10, and 20 incremental tasks to validate different methods. Furthermore, we also conduct an upper bound and a finetuning baseline (FT) for reference. An upper bound is constructed by treating all training samples available across tasks and turning the pretrained ViT backbone using supervised training strategies. The FT baseline is to turn the whole backbone (also initialized with the pretrained weight) without any constraint on the old knowledge during the learning.

As shown in Table 4, our method makes comparable performance when the number of tasks is 5. When the learning sequence becomes longer, our method outperforms L2P and DualPrompt by a larger margin. Specifically, our proposed method outperforms DualPrompt and L2P by 1.03% and 2.07% on average accuracy on 20-task incremental learning on CIFAR100. A similar conclusion can be observed under a more challenging dataset ImageNet-R in Table 5. As the training procedure gets longer, the superiority of our model begins to emerge. On 20-task incremental learning on ImageNet-R, we achieve 0.74% and 3.06% improvement compared with DualPrompt at average accuracy and forgetting respectively.

It is worth mentioning that our method does not rely on a complex and delicate prompting pattern (*i.e*. focus on where or how to insert prompts), only utilizing the vanilla way [21] to insert the prompts. Experiments in this section show that adaptive-prompting is an effective design and the static selection process from a prompt pool in previous methods [60, 61] can be the limitation of prompt-based methods.

Further analysis on different pretraining weights. To investigate the effect of the gap between pretrained knowl-

Methods	ImageNet-R B0-T5		ImageNet-R B0-T10		ImageNet-R B0-T20	
	A↑	F↓	A↑	F↓	A↑	F↓
Upper Bound	77.47	-	77.87	-	78.90	-
FT	37.73	46.23	24.76	63.14	18.66	72.35
L2P [61]	66.86	5.02	66.61	9.37	64.08	8.64
DualPrompt [60]	**73.02**	3.73	72.57	4.68	70.48	7.47
Ours	72.36	6.37	**73.27**	8.59	**71.22**	7.39

Table 5. Comparison with state-of-the-art methods with ImageNet-21k pretrained weights on ImageNet-R. The best results are marked in **bold**. 'A' means Avg. Acc and 'F' indicates the forgetting metric. Even with the use of strong pretraining (Table 2), our method also achieves comparable performance to other methods.

Configs	w/o APG		w/ APG			Avg.
	$\mathcal{L}_{cls}$	$\mathcal{L}_{conC}$	$\mathcal{L}_{conA}$	$\mathcal{L}_{attn}$	$\mathcal{L}_{tri}$	Acc.(%)
c-1	√					19.37
c-2	√	√				71.84
c-3	√	√	√			73.46
c-4	√	√	√	√		74.70
c-5	√	√	√		√	75.59
c-6	√	√		√	√	73.60
Full	√	√	√	√	√	**75.94**

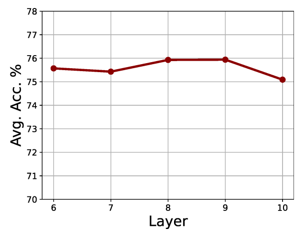
Table 6. Effectiveness of each loss function of the proposed method. 'c-x' denotes different training configs. 'Full' indicates the default full model. Experiments are conducted on ImageNet-Sub under non-pretrained 10-task setting. The best result is marked in **bold**.

edge and future tasks, we further conduct experiments on CIFAR100 using TinyImageNet pretrained weights [20]. As shown in Table 4, we can observe that our method outperforms L2P by 3.04%, 5.10% and 8.81% on Avg. Acc. under 5, 10 and 20 tasks. Across different pretraining strategies (ImageNet-21k and TinyImageNet in Table 4 and standard non-pretrained in Table 1), our method achieves 1.03%, 8.37% and 39.54% higher Avg. Acc. compared with the DualPrompt under the 20-task setting. It is mainly because the pretrained knowledge dominates the incremental performance for existing prompt-based methods, making their methods fail when the semantic gap between the pretrained task and incremental tasks is relatively large. By contrast, our method breaks the reliance on the strong pretrained backbone since the APG can effectively learn to reduce the negative effect of the semantic gap.

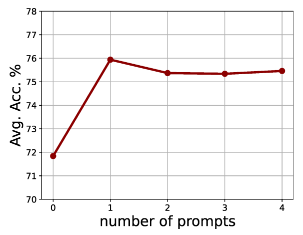
4.4. Ablation Studies

In this section, we discuss the effectiveness of each loss function proposed in Sec. 3. We perform ablation studies on the ImageNet-Subset with a vanilla non-pretrained ViT backbone under the 10-task incremental learning setting.

The effectiveness of knowledge pool. Our knowledge pool is served in two aspects: one is the constraint on the APG ($\mathcal{L}_{conA}$) and the second is to guide the classifier ($\mathcal{L}_{conC}$). In Table 6, c-1 indicates we only apply the classification loss



(a) The impact of which layers that APG augments into.



(b) The impact of the number of generated prompts.

Figure 3. We conduct experiments on the impact of the layer of equipping the APG (a) and the impact of the number of generated prompts (b). All experiments are conducted on ImageNet-SubSet at non-pretrained 10-task incremental learning setting.

$\mathcal{L}_{cls}$ and the performance is far from satisfactory because the network is inaccessible to any old knowledge. We can observe that with the constraint $\mathcal{L}_{conC}$ on the classifier, the unbalance problem of the classifier [62] can be greatly alleviated (Avg. Acc. from 19.37% to 71.84%). This indicates that the knowledge vector can represent the original classes well, which can effectively overcome forgetting. With the help of $\mathcal{L}_{conA}$, the APG is able to output effective prompts for the old classes. The average accuracy is boosted from 71.84% to 73.46%. This is mainly because the trainable module APG is facing severe forgetting problems during learning which can be effectively addressed by $\mathcal{L}_{conA}$.

The effectiveness of the APG and the associated losses. To further alleviate the forgetting problem, we design two losses in Sec. 3 to guide the APG to learn class-specific knowledge while training. The first one is $\mathcal{L}_{attn}$, which guides the attention operation in the APG to be more class-related. With the $\mathcal{L}_{attn}$ loss, the performance increases from 73.46% to 74.70%. The triplet $\mathcal{L}_{tri}$ serves as a constraint that avoids the APG to generate degraded prompts and narrows the distance between two prompts with the same classes, bringing a performance boost of 1.24% (compares c-4 with the full model).

The impact of the location of the APG. As stated in Sec. 3, the proposed APG takes the intermediate feature from layer L_l and generates the class-specific prompts. In order to investigate the impact of the location l, we conduct experiments on $l = \{6, 7, 8, 9, 10\}$, and results are shown in Fig. 3(a). It is shown that our method performs consistently when the APG is inserted to layers $l \in \{6, 7, 8, 9\}$ but drops at layer $l = 10$. It is mainly because the deeper level of the backbone encodes more task-specific information and it is not suitable for the APG for aggregating knowledge. Our model performs best when $l = 9$, and we choose $l = 9$ as our default setting to balance the complexity and the performance.

The impact of the number of prompts. As stated in Sec. 3, the Multi-head Self Attention (MSA) is referred to perform the cross-attention operation. Such a multi-head property allows us to generate any number of prompts to insert into deep layers. To explore the impact of the number of prompts, we conduct experiments on $N_P \in \{0, 1, 2, 3, 4\}$, whose results are in Fig. 3(b). Compared with the baseline ($N_P = 0$), we achieve 4.1% accuracy improvement when generating 1 prompt. When the number of prompts increases, the performance stays around 75%. This indicates that with the trainable APG, the generated prompt is powerful enough for a single prompt to instruct deeper layers. Therefore, we choose $N_P = 1$ as the default.

5. Conclusion

In this work, we reveal the fact that existing prompt-based models achieve impressive performance in class-incremental learning but unfortunately they are affected by strong pretrained backbones. The potential gap between the pretraining task and future tasks could be a stumbling block to existing prompting schemes in incremental learning. We therefore contribute a learnable adaptive-prompting scheme. The extensive experiments show that our method obtains satisfactory performance without pretraining and also achieves comparable performance to other models under strong pretraining. In addition to adopting the knowledge from the pretraining task to other tasks, our work also shows that the prompt-based model can adaptively learn new knowledge when current knowledge is not sufficient for future tasks, which makes prompt-based models possible for more general incremental learning (e.g, more complicated tasks, multi-modality data).

6. Acknowledgment

This work was supported partially by the NSFC (U21A20471, U1911401, U1811461), Guangdong NSF Project (No. 2023B1515040025, 2020B1515120085). The authors would like to thank Kun-Yu Lin for insightful discussions. The authors also thank three anonymous reviewers for their constructive suggestions.

References

[1] Davide Abati, Jakub Tomczak, Tijmen Blankevoort, Simone Calderera, Rita Cucchiara, and Babak Ehteshami Bejnordi. Conditional channel gated networks for task-aware continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 2

[2] Arjun Ashok, KJ Joseph, and Vineeth N Balasubramanian. Class-incremental learning with cross-space clustering and controlled transfer. In *Proceedings of the European Conference on Computer Vision*, 2022. 2

[3] Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 2

[4] Michel Breyer, Jen Jen Chung, Lionel Ott, Siegwart Roland, and Nieto Juan. Volumetric grasping network: Real-time 6 dof grasp detection in clutter. In *Conference on Robot Learning*, 2020. 1

[5] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017. 1

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2009. 5, 7

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 1, 3

[8] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *Proceedings of the European Conference on Computer Vision*, 2020. 2, 5, 6, 7

[9] Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dynamic token expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 6

[10] Hao-Shu Fang, Chenxi Wang, Minghao Gou, and Cewu Lu. Graspnet-1billion: A large-scale benchmark for general object grasping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 1

[11] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 1999. 1

[12] Ziteng Gao, Limin Wang, Bing Han, and Sheng Guo. Adamixer: A fast-converging query-based object detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 1

[13] Abhishek Gupta, Alagan Anpalagan, Ling Guan, and Ahmed Shaharyar Khwaja. Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues. *Array*, 2021. 1

[14] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019. 2

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016. 1, 5

[16] David Held, Sebastian Thrun, and Silvio Savarese. Learning to track at 100 fps with deep regression networks. In *Proceedings of the European Conference on Computer Vision*, 2016. 1

[17] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu,

Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021. 5

[18] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. 1, 2, 5, 6, 7

[19] Xinting Hu, Kaihua Tang, Chunyan Miao, Xian-Sheng Hua, and Hanwang Zhang. Distilling causal effect of data in class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 2, 5, 6, 7

[20] Ethan Huynh. Vision transformers in 2022: An update on tiny imagenet. *arXiv preprint arXiv:2205.10660*, 2022. 8

[21] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *Proceedings of the European Conference on Computer Vision*, 2022. 1, 3, 7

[22] Minsoo Kang, Jaeyoo Park, and Bohyung Han. Class-incremental learning by knowledge distillation with adaptive feature consolidation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 2, 6, 7

[23] Zixuan Ke, Bing Liu, and Xingchang Huang. Continual learning of a mixed sequence of similar and dissimilar tasks. In *Advances in Neural Information Processing Systems*, 2020. 2

[24] 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*, 2017. 6, 7

[25] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Technical report, University of Toronto*, 2009. 1, 5

[26] Kibok Lee, Kimin Lee, Jinwoo Shin, and Honglak Lee. Overcoming catastrophic forgetting with unlabeled data in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. 1, 5

[27] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021. 1

[28] Feng Li, Hao Zhang, Shilong Liu, Jian Guo, Lionel M Ni, and Lei Zhang. Dn-detr: Accelerate detr training by introducing query denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 1

[29] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021. 1

[30] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021. 3

[31] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 2017. 6, 7

[32] Lingbo Liu, Bruce XB Yu, Jianlong Chang, Qi Tian, and Chang-Wen Chen. Prompt-matched semantic segmentation. *arXiv preprint arXiv:2208.10159*, 2022. 3

[33] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021. 3

[34] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 2023. 1

[35] Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*, 2021. 1, 3

[36] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. *arXiv:2103.10385*, 2021. 3

[37] Yu Liu, Sarah Parisot, Gregory Slabaugh, Xu Jia, Ales Leonardis, and Tinne Tuytelaars. More classifiers, less forgetting: A generic multi-classifier paradigm for incremental learning. In *Proceedings of the European Conference on Computer Vision*, 2020. 6, 7

[38] Yaoyao Liu, Yuting Su, An-An Liu, Bernt Schiele, and Qianru Sun. Mnemonics training: Multi-class incremental learning without forgetting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 2

[39] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the European Conference on Computer Vision*, 2018. 2

[40] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018. 2

[41] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of learning and motivation*, 1989. 1

[42] Arsalan Mousavian, Clemens Eppner, and Dieter Fox. 6-dof graspnet: Variational grasp generation for object manipulation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019. 1

[43] Filipe Mutz, Vinicius Cardoso, Thomas Teixeira, Luan FR Jesus, Michael A Golçalves, Rânik Guidolini, Josias Oliveira, Claudine Badue, and Alberto F De Souza. Following the leader using a tracking system based on pre-trained deep neural networks. In *2017 international joint conference on neural networks (IJCNN)*, 2017. 1

[44] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 2010. 2

[45] Evan Prianto, MyeongSeop Kim, Jae-Han Park, Ji-Hun Bae, and Jung-Su Kim. Path planning for multi-arm manipulators using deep reinforcement learning: Soft actor–critic with hindsight experience replay. *Sensors*, 2020. 1

[46] Jathushan Rajasegaran, Munawar Hayat, Salman Khan, Fahad Shahbaz Khan, and Ling Shao. Random path selection for incremental learning. *Advances in Neural Information Processing Systems*, 2019. 2

[47] Jathushan Rajasegaran, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Mubarak Shah. itaml: An incremental task-agnostic meta-learning approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 2

[48] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017. 1, 5

[49] Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 1995. 1

[50] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, 2018. 2

[51] Yujun Shi, Kuangqi Zhou, Jian Liang, Zihang Jiang, Jiashi Feng, Philip HS Torr, Song Bai, and Vincent YF Tan. Mimicking the oracle: An initial phase decorrelation approach for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 6, 7

[52] Martin Sundermeyer, Arsalan Mousavian, Rudolph Triebel, and Dieter Fox. Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021. 1

[53] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27*, 2018. 2

[54] Yu-Ming Tang, Yi-Xing Peng, and Wei-Shi Zheng. Learning to imagine: Diversify memory for incremental learning using unlabeled data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 1, 2, 5, 6, 7

[55] Rishabh Tiwari, Krishnateja Killamsetty, Rishabh Iyer, and Pradeep Shenoy. Gcr: Gradient coreset based replay buffer selection for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 2

[56] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, 2021. 1, 3, 6, 7

[57] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, 2017. 3

[58] Fu-Yun Wang, Da-Wei Zhou, Han-Jia Ye, and De-Chuan Zhan. Foster: Feature boosting and compression for class-incremental learning. In *Proceedings of the European Conference on Computer Vision*, 2022. 1, 2, 5, 6, 7

[59] Yabin Wang, Zhiwu Huang, and Xiaopeng Hong. S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning. *arXiv preprint arXiv:2207.12819*, 2022. 1, 2

[60] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *Proceedings of the European Conference on Computer Vision*, 2022. 1, 2, 5, 6, 7, 8

[61] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 1, 2, 5, 6, 7, 8

[62] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. 1, 5, 8

[63] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 1, 2, 5, 6

[64] Lu Yu, Bartlomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 1, 2, 6, 7

[65] Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C-C Jay Kuo. Class-incremental learning via deep model consolidation. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, 2020. 1, 5

[66] Fei Zhu, Xu-Yao Zhang, Chuang Wang, Fei Yin, and Cheng-Lin Liu. Prototype augmentation and self-supervision for incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 1, 2, 5, 6, 7

[67] Kai Zhu, Wei Zhai, Yang Cao, Jiebo Luo, and Zheng-Jun Zha. Self-sustaining representation expansion for non-exemplar class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 1, 2, 5, 6, 7