BPG: Balancing Plasticity and Generalization for Domain Incremental Learning

Qiang Wang, Songlin Dong, Shaokun Wang, Jizhou Han, Xiang Song, Chenhao Ding, Yuhang He, and Yihong Gong, *Fellow, IEEE*

arXiv:2608.10804v1 [cs.CV] 11 Aug 2026

***Abstract*—Deep neural networks excel in various tasks but struggle to generalize across evolving data distributions, leading to significant performance degradation under domain shifts. Domain incremental learning (DIL) addresses this challenge by enabling models to continuously adapt while retaining prior knowledge. Among existing DIL approaches, the parameter-isolation paradigm achieves state-of-the-art performance. However, these methods often adopt a one-size-fits-all approach to adapt to new domains, resulting in either insufficient learning capacity or redundant parameters. In this work, we propose BPG, a unified framework that addresses both challenges through two complementary components: BPG-Adapter, which dynamically determines each domain's adapter hidden dimension based on domain-specific feature separability, and BPG-Inference, a soft domain mixture strategy that integrates multiple domain-specific models at test time, mitigating domain ID misselection. Experimental results on DomainNet, CDDB, and CORe50 demonstrate that BPG consistently outperforms uniform adapter-based approaches and hard domain selection strategies, achieving state-of-the-art average accuracy while reducing forgetting to as low as 0.22% on DomainNet.**

***Index Terms*—Domain Incremental Learning, Continual Learning, Parameter-Efficient Fine-Tuning, Adapter**

I. Introduction

DEEP neural networks (DNNs) have achieved remarkable success in visual recognition tasks such as image classification [1]–[5], object detection [6]–[9], and semantic segmentation [10], [11]. However, these models typically assume that training and test data share the same distribution, making them prone to catastrophic forgetting [12]–[14] in non-stationary environments where lighting conditions [15], sensor characteristics, or visual styles change [16], [17] over time. This is especially concerning in safety-critical applications such as autonomous driving [18], medical imaging [19], and surveillance systems [20], [21].

Domain Incremental Learning (DIL) [22]–[29] addresses this challenge: a model is sequentially exposed to domains that exhibit distinct visual characteristics, and must incorporate new domain knowledge while preserving performance on all previously learned domains. Existing DIL approaches fall into three paradigms: rehearsal-based methods [30]–[32] that replay stored past samples but incur storage and privacy costs, regularization-based methods [33]–[38] that constrain parameter updates to protect old knowledge but limit the absorption of new knowledge, and parameter-isolation methods [22], [39]–[42] that freeze a shared backbone and learn lightweight domain-specific modules, achieving state-of-the-art performance by explicitly preventing cross-domain interference. While parameter-isolation methods have shown strong results, they struggle to balance the fundamental tension between *plasticity* (the ability to learn effectively from new domains) and *generalization* (the capacity to maintain robust performance across all encountered domains).

The Plasticity Challenge: One Size Does Not Fit All. Most existing methods assign fixed-capacity modules (e.g., uniform adapters or prompts) to every domain, overlooking the substantial variation in domain difficulty. As illustrated in Fig. 1 (a-b), applying a pre-trained backbone to three DomainNet domains yields markedly different feature-space separability, ranging from well-separated clusters in the Painting domain to heavily entangled ones in the Quickdraw domain. This disparity has direct implications for model design. As shown in Fig. 1 (c), easier domains attain peak performance at small adapter dimensions and may even degrade when granted excessive capacity through overfitting, whereas harder domains continue to benefit from larger capacity. Consequently, a uniform adapter design is suboptimal: allocating too few parameters to a hard domain undermines plasticity, while allocating too many to an easy domain induces overfitting and impairs generalization.

The Generalization Challenge: Hard Selection is Brittle. Maintaining separate parameters per domain requires a domain identification step at inference: given a test sample, the model selects a single domain ID and applies only the corresponding parameters. This hard selection is inherently fragile. As illustrated in Fig. 1 (d), samples near domain boundaries are prone to misassignment, causing a catastrophic mismatch between the test sample and the applied domain-specific model. The problem worsens as more domains are learned, and our analysis shows that domain ID errors in hard selection can degrade early-domain accuracy by over seven points.

We propose **BPG**, a unified framework that addresses both challenges through complementary mechanisms. For plasticity, we introduce **BPG-Adapter**, an adaptive capacity allocation strategy that matches adapter parameters to each domain's intrinsic difficulty: domains already well separated by the pre-trained backbone need only lightweight adaptation, while domains with entangled representations require richer parameterization. Guided by a qualitative risk decomposition, BPG-Adapter operationalizes this insight through a simple inverse-proportionality rule that directs capacity where it is most

Qiang Wang, Jizhou Han, Yuhang He, and Yihong Gong are with the College of Artificial Intelligence, Xi'an Jiaotong University, Xi'an, China. Songlin Dong is with Shenzhen University of Advanced Technology, Shenzhen, China. Shaokun Wang is with Harbin Institute of Technology, Shenzhen, China. Xiang Song and Chenhao Ding are with the School of Software Engineering, Xi'an Jiaotong University, Xi'an, China.

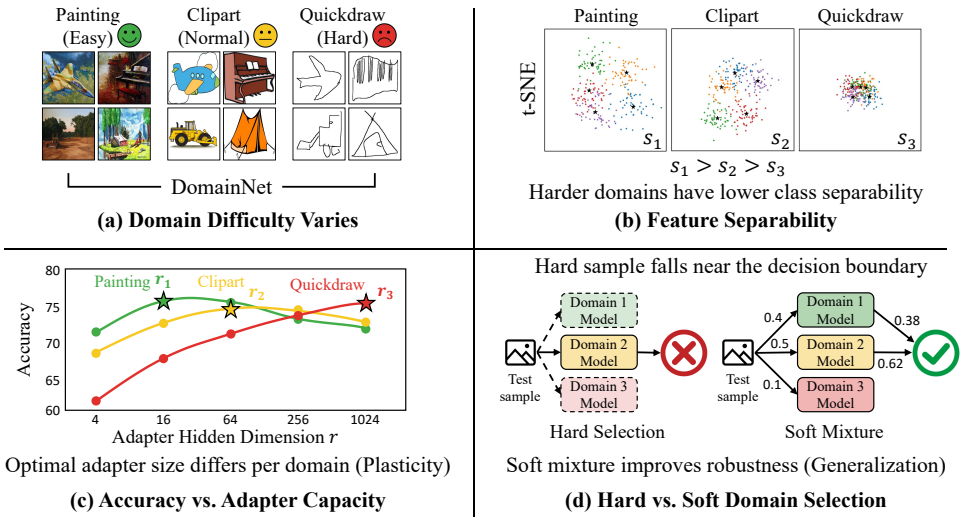


Fig. 1. **Motivation of BPG.** (a) Domains from DomainNet exhibit varying levels of visual complexity. (b) t-SNE visualizations of pre-trained backbone (ViT-B/16) features confirm that harder domains have lower feature separability scores ($s_1 > s_2 > s_3$). (c) The optimal adapter dimension differs across domains: easier domains peak at smaller capacities while harder domains benefit from larger ones, motivating our adaptive allocation strategy (BPG-Adapter). (d) Hard domain selection at inference can misassign ambiguous samples near decision boundaries, whereas our soft mixture strategy (BPG-Inference) aggregates predictions from multiple domain-specific models weighted by confidence, improving robustness.

needed. For generalization, we introduce **BPG-Inference**, a soft domain mixture strategy that replaces hard selection with confidence-weighted aggregation over multiple domain-specific models. Rather than committing to a single domain, it estimates a distribution over all learned domains for each test sample, prunes unlikely candidates, and fuses the surviving predictions, drawing on complementary domain knowledge when a sample is ambiguous. Together, these components reduce average forgetting to as low as 0.22% on DomainNet while attaining state-of-the-art average accuracy. The main contributions are summarized below.

- We identify the plasticity-generalization tension in parameter-isolation DIL and propose BPG, a unified framework that addresses both challenges through complementary mechanisms.
- We introduce BPG-Adapter, which dynamically allocates adapter capacity by feature separability, supported by a qualitative risk decomposition that motivates the inverse-proportionality allocation rule.
- We present BPG-Inference, a soft domain mixture strategy that mitigates the fragility of hard domain ID selection and serves as a plug-and-play module for other parameter-isolation methods.
- Extensive experiments on three multi-domain benchmarks (DomainNet, CDDB, and CORe50) show that BPG consistently achieves state-of-the-art performance in both average accuracy and forgetting rate.

II. Related Work

A. Domain Incremental Learning

Domain Incremental Learning (DIL) methods fall into three paradigms: rehearsal-based, regularization-based, and parameter-isolation.

Rehearsal-based methods replay stored or generated past samples during subsequent training, via exemplar selection [43], [44], prototype preservation [45], or generative replay [46], [47]. On frozen pre-trained models, these ideas resurface in a data-free form for class-incremental learning: RanPAC [48] and APER [49] build prototype-based classifiers over frozen or first-session-adapted features, while SLCA [50] models each class as a Gaussian and replays sampled pseudo-features to debias the classifier. However, their performance degrades as the exemplar budget shrinks [51], [52], and storing raw data raises privacy concerns [53].

Regularization-based methods add loss terms that constrain parameter updates to protect learned representations, e.g., EWC [33] penalizes changes to parameters important for earlier domains and distillation variants [38] enforce feature-space consistency. For vision-language models, ZSCL [54] follows the same regularization philosophy, distilling against the initial CLIP model and averaging weights to prevent zero-shot transfer degradation. Although they avoid data storage, the finite capacity of a single model makes them overly conservative, struggling to absorb substantially different domains while retaining old knowledge [55].

Parameter-isolation methods instead maintain domain-specific parameters to prevent cross-domain interference. S-Prompts [22] learns per-domain prompts on a frozen transformer and selects domains by K-NN matching over K-Means centroids, and MoP-CLIP [39] extends this to CLIP-based prompt mixtures. Later works differ mainly in the isolated module: PINA [40] adds a per-domain alignment module with a Patch Shuffle Selector, C-Prompt [41] pools learnable prompts for cross-domain compositionality, ISPSL [42] decouples domain-specific and domain-shared low-rank subspaces, and KA-Prompt [56] aligns knowledge across prompts component-wise. Pre-trained-model methods for class-incremental learning follow the same spirit with per-task modules, e.g., LAE [57] ensembles online and offline expert adapters, and EASE [58] learns a per-task adapter to form expandable subspaces. Despite strong performance, two limitations persist: a uniform-capacity module is attached to

every domain, ignoring the wide variation in domain difficulty; and hard domain selection at inference is fragile near domain boundaries. Our BPG tackles both by allocating adapter capacity according to domain difficulty and replacing hard selection with a soft mixture over domain-specific experts.

B. Adaptive Parameter-Efficient Fine-Tuning

Parameter-efficient fine-tuning (PEFT) methods, such as adapters [59] and LoRA [60], adapt pre-trained models by updating only a small fraction of parameters, but typically fix the rank or hidden dimension across all weight matrices and tasks. Since performance is highly sensitive to this choice [61], a uniform budget overlooks that different matrices and tasks demand varying adaptation capacity. To address this, recent works adaptively allocate capacity, e.g., by pruning or budgeting ranks according to importance or intrinsic dimensionality [62]–[65], or by gating and selecting adapters at the layer, token, or module level [66]–[68]. However, these methods target single- or multi-task fine-tuning and allocate capacity across layers within a task; none explicitly links adapter capacity to the intrinsic difficulty of the target distribution, leaving open how to determine the right capacity per domain in incremental learning. Our BPG-Adapter fills this gap with a data-driven separability metric that guides hidden-dimension allocation across domains, motivated by a qualitative risk decomposition.

C. Test-Time Domain Identification

Parameter-isolation methods for DIL must decide, at inference, which domain-specific parameters to activate for a given test sample, and the design of this mechanism critically affects performance. Most existing methods rely on *hard* selection. S-Prompts [22] stores per-domain K-Means centroids during training and assigns a single domain label via K-NN matching in the frozen feature space. Later methods refine this selection: PINA [40] introduces a Patch Shuffle Selector that suppresses class-dependent cues and outperforms K-NN and Nearest Mean classifiers, while ESN [69] selects the most confident domain via a temperature-controlled energy metric over stage classifiers. C-Prompt [41] instead pools and recomposes prompts across domains, implicitly softening the domain boundary. Nevertheless, committing to a single domain remains fragile for samples near domain boundaries.

A related paradigm is Mixture-of-Experts (MoE), which softly routes inputs across expert sub-networks via a learned gating network. In continual learning, Lifelong-MoE [70] adds experts for emerging distributions while regularizing experts and gates to retain old knowledge, and MoE-Adapters [71], [72] treat adapters as experts within CLIP, using a Distribution Discriminative Auto-Selector to route between adapted and zero-shot models. Although these methods confirm the benefit of soft routing over hard selection, they train the gating mechanism jointly with the experts, incurring extra training cost and risking forgetting in the router itself.

In contrast, our BPG-Inference is a training-free, plug-and-play soft domain mixture. For each test sample, it estimates a confidence-weighted distribution over all learned domains, prunes unlikely candidates, and fuses predictions from the surviving domain-specific models. This retains the robustness of soft routing without any additional trainable module or test-time parameter update, and can be applied on top of existing parameter-isolation DIL methods.

III. Method

A. Problem Formulation

We consider a domain incremental learning scenario with T domains, where the dataset is written as $\mathcal{D} = \{\mathcal{D}_t\}_{t=1}^{T}$. Each domain dataset $\mathcal{D}_t = (\mathcal{X}_t, \mathcal{Z}_t)$ consists of a training set $\mathcal{X}_t$ and a test set $\mathcal{Z}_t$. Following the DIL protocol, during the t-th training session only samples in $\mathcal{X}_t$ are accessible, while data from other domains is not stored or replayed. After completing session t, the model is evaluated on previously seen test sets

$$\mathcal{Z}_{\le t} = \mathcal{Z}_1 \cup \mathcal{Z}_2 \cup \cdots \cup \mathcal{Z}_t, \tag{1}$$

and the performance across sessions is summarized by standard metrics, i.e., average accuracy and forgetting. The details of metrics will be provided in Section IV-A.

B. Vanilla Adapter for Domain-Specific Knowledge

Before introducing BPG-Adapter, we describe the vanilla adapter-based architecture that serves as the building block for our framework. This subsection explains how domain-specific adapters are inserted into a ViT encoder and how they interact with prompt tokens and classification heads.

a) Tokenization and Encoder Backbone: Let x denote an input image from domain t. x is first divided into non-overlapping image patches, which are linearly projected to patch tokens and added with positional embeddings. We denote the resulting sequence of N patch tokens as $\mathbf{x} = [\mathbf{x}_1, \ldots, \mathbf{x}_N] \in \mathbb{R}^{N\times d}$, where d is the embedding dimension.

For domain t, we introduce domain-specific prompt tokens $\mathbf{p}_t \in \mathbb{R}^{M\times d}$ that are prepended to the patch tokens and a class token $\mathbf{x}_{\text{cls}} \in \mathbb{R}^{1\times d}$ that aggregates image-level information. The input sequence to the encoder at layer $\ell = 0$ is

$$\mathbf{z}_t^{(0)} = [\mathbf{x}_{\text{cls}},\, \mathbf{x}_1, \ldots, \mathbf{x}_N,\, \mathbf{p}_t] \in \mathbb{R}^{(N+M+1)\times d}. \tag{2}$$

We employ a stack of L transformer encoder layers with frozen parameters θ. Each layer consists of a multi-head self-attention (MHSA) block and a feed-forward multi-layer perceptron (MLP) block with residual connections and layer normalization (LN), as shown in Fig. 2 (b). Let $\mathbf{z}_t^{(\ell)}$ be the input to layer ℓ. The standard ViT update is

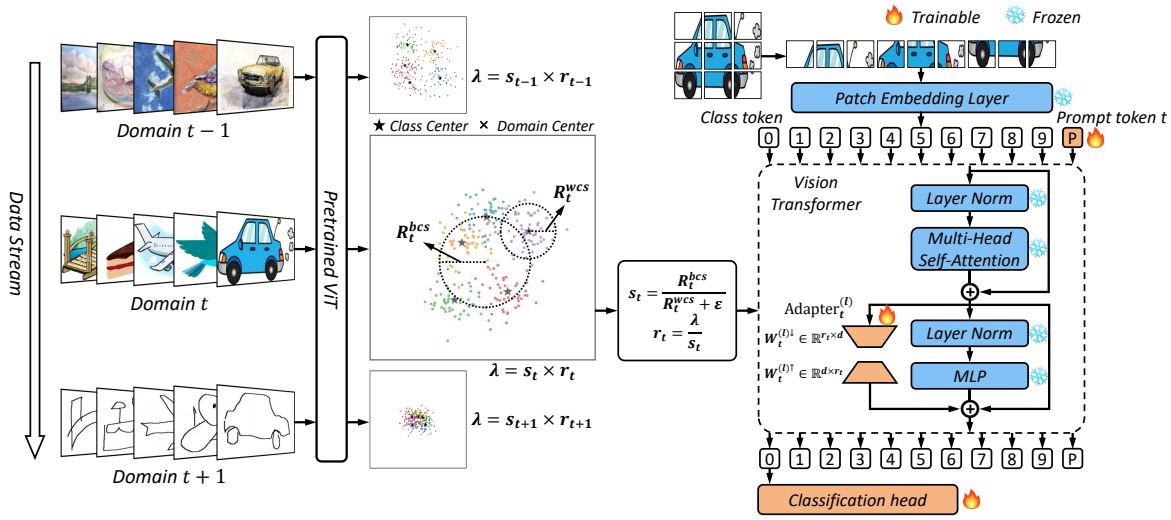
$$\mathbf{h}_t^{(\ell)} = \mathbf{z}_t^{(\ell)} + \mathrm{MHSA}\big(\mathrm{LN}(\mathbf{z}_t^{(\ell)}); \theta_{\text{att}}^{(\ell)}\big), \tag{3}$$

$$\mathbf{z}_t^{(\ell+1)} = \mathbf{h}_t^{(\ell)} + \mathrm{MLP}\big(\mathrm{LN}(\mathbf{h}_t^{(\ell)}); \theta_{\text{mlp}}^{(\ell)}\big), \tag{4}$$

where $\theta_{\text{att}}^{(\ell)}$ and $\theta_{\text{mlp}}^{(\ell)}$ are frozen pre-trained parameters.

b) Adapter Structure: To capture domain-specific characteristics with a small number of trainable parameters, we insert an adapter module as an additional residual branch in each encoder layer, running in parallel to the feed-forward MLP block. For domain t and layer ℓ, the adapter is a two-layer bottleneck MLP

$$\mathrm{Adapter}_t^{(\ell)}(\mathbf{u}) = \sigma\Big(\mathbf{u}\,\mathbf{W}_t^{(\ell)\downarrow\top}\Big)\,\mathbf{W}_t^{(\ell)\uparrow\top}, \tag{5}$$



(a) Compute Feature Separability s_t and Hidden Dimension r_t (b) Training Pipeline

Fig. 2. Overall architecture of the proposed BPG framework. An input image is tokenized by the patch embedding layer and concatenated with trainable prompt tokens and a class token. The token sequence is processed by a stack of frozen vision transformer layers equipped with domain-specific adapters. For each domain t, the adapter hidden dimension r_t is determined by the feature separability score s_t.

where $\mathbf{u} \in \mathbb{R}^{(N+M+1)\times d}$ stacks all token embeddings as rows (shared with the MLP branch), $\mathbf{W}_t^{(\ell)\downarrow} \in \mathbb{R}^{r_t \times d}$ and $\mathbf{W}_t^{(\ell)\uparrow} \in \mathbb{R}^{d \times r_t}$ are the down- and up-projection matrices, r_t is the adapter hidden dimension of domain t, and $\sigma(\cdot)$ is a non-linear activation function (ReLU). Each matrix multiplication is applied independently to every token row, yielding an output in $\mathbb{R}^{(N+M+1)\times d}$. Because the adapter output is added to the same residual path as the MLP output and the adapter is initialized to produce near-zero outputs, the overall layer recovers the frozen pre-trained parameters at initialization, which stabilizes training.

Concretely, for domain t, the attention sub-layer in (3) is unchanged. We then apply layer normalization and branch into two parallel paths: a frozen MLP branch and a trainable adapter branch:

$$\mathbf{m}_t^{(\ell)} = \mathrm{MLP}\big(\mathrm{LN}(\mathbf{h}_t^{(\ell)}); \theta_{\mathrm{mlp}}^{(\ell)}\big), \tag{6}$$

$$\mathbf{a}_t^{(\ell)} = \mathrm{Adapter}_t^{(\ell)}\Big(\mathrm{LN}(\mathbf{h}_t^{(\ell)})\Big). \tag{7}$$

The output of layer ℓ is obtained by adding the two branches to the residual connection:

$$\mathbf{z}_t^{(\ell+1)} = \mathbf{h}_t^{(\ell)} + \mathbf{m}_t^{(\ell)} + \mathbf{a}_t^{(\ell)}. \tag{8}$$

When the adapter branch is disabled ($\mathbf{a}_t^{(\ell)} = \mathbf{0}$), (8) reduces to the standard ViT update in (4). All backbone parameters $\theta = \{\theta_{\mathrm{att}}^{(\ell)}, \theta_{\mathrm{mlp}}^{(\ell)}\}_{\ell=1}^{L}$ are frozen, whereas adapter parameters $\{\mathbf{W}_t^{(\ell)\downarrow}, \mathbf{W}_t^{(\ell)\uparrow}\}_{\ell=1}^{L}$ and the prompt token $\mathbf{p}_t$ are trainable for domain t. This parallel-residual design creates an isolated parameter set per domain while fully reusing the same pre-trained transformer layer.

c) Domain-Specific Classification Head: After passing through L encoder layers, we obtain the final token sequence $\mathbf{z}_t^{(L)}$. We use the output of the class token (the first token in the sequence) as the image representation, denoted as $\mathbf{h}_t = \mathbf{z}_t^{(L)}[0] \in \mathbb{R}^d$. Each domain t is associated with its own linear classifier

$$\mathbf{y}_t = g_t(\mathbf{h}_t; \eta_t) = \mathbf{W}_t \mathbf{h}_t + \mathbf{b}_t, \tag{9}$$

where $\eta_t = \{\mathbf{W}_t, \mathbf{b}_t\}$ and $\mathbf{y}_t \in \mathbb{R}^C$ are the class logits. For domain t, only $\mathbf{p}_t$, the adapter parameters and classifier η_t are updated, while parameters from previous domains remain fixed. In the next subsections, we describe how BPG allocates the adapter capacity across domains and how it performs soft inference at test time.

C. BPG-Adapter: Domain-wise Capacity Allocation

The vanilla adapter in Section III-B uses the same hidden dimension for all domains, i.e., $\forall t \in [1, T], r_t = r$. However, domains may exhibit different levels of difficulty: some align well with the pre-trained backbone, while others contain more confusing or fine-grained categories. Allocating an equal number of parameters to all domains may thus fail to model hard domains and over-parameterize easy ones. BPG-Adapter addresses this issue by dynamically determining the hidden dimension r_t for each domain according to a *feature separability* score.

a) Domain-wise Feature Separability: Let $t \in \{1, \ldots, T\}$ index domains, $c \in \{1, \ldots, C\}$ index classes, and $i \in \{1, \ldots, N_{t,c}\}$ index training samples from class c in domain t. We use a pre-trained feature extractor to extract a d-dimensional feature vector $\mathbf{x}_{t,c,i} \in \mathbb{R}^d$ for each sample. The class mean and domain mean are

$$\boldsymbol{\mu}_{t,c} = \frac{1}{N_{t,c}} \sum_{i=1}^{N_{t,c}} \mathbf{x}_{t,c,i}, \quad \boldsymbol{\mu}_t = \frac{1}{C} \sum_{c=1}^{C} \boldsymbol{\mu}_{t,c}. \tag{10}$$

We define the *between-class scatter* (BCS) and *within-class scatter* (WCS) of domain t as

$$R_t^{\mathrm{bcs}} = \frac{1}{C} \sum_{c=1}^{C} \big\|\boldsymbol{\mu}_{t,c} - \boldsymbol{\mu}_t\big\|_2^2, \tag{11}$$

$$R_t^{\mathrm{wcs}} = \frac{1}{C} \sum_{c=1}^{C} \frac{1}{N_{t,c}} \sum_{i=1}^{N_{t,c}} \big\|\mathbf{x}_{t,c,i} - \boldsymbol{\mu}_{t,c}\big\|_2^2, \tag{12}$$

where $\|\cdot\|_2$ is the Euclidean norm. Intuitively, a large R_t^{bcs} and a small R_t^{wcs} indicate that classes in domain t are well

separated and compact, which is favorable for classification. A clear example is shown in Fig. 2 (a). Then we combine these quantities into a scalar feature separability score

$$s_t = \frac{R_t^{\mathrm{bcs}}}{R_t^{\mathrm{wcs}} + \varepsilon}, \tag{13}$$

where $\varepsilon = 10^{-6}$ prevents division by zero. A higher s_t implies better separability and thus an easier domain.

b) Capacity Allocation Rule: To balance plasticity and generalization, we assume that each domain is allocated a certain amount of trainable capacity through its adapter hidden dimension r_t. We desire an inverse relationship between s_t and r_t: hard domains (low s_t) should receive more parameters to improve plasticity, while easy domains (high s_t) can be modeled with smaller adapters to avoid unnecessary capacity. The detailed justification and derivation are provided in Section III-E. We impose a simple multiplicative constraint $\lambda = s_t r_t$, $t = 1, \ldots, T$, where λ is a constant controlling the overall capacity budget. In practice, instead of tuning λ directly, we use a reference domain with its separability s_0 and a reference hidden dimension r_0, and set $\lambda = s_0 r_0$. Substituting into this constraint yields

$$r_t = \frac{s_0}{s_t} r_0. \tag{14}$$

We treat the ImageNet dataset [73] as the reference domain because it is the pre-training dataset of the backbone and typically exhibits relatively high separability. By using Eq. (14), BPG-Adapter automatically assigns more trainable parameters to challenging domains and less to easier ones, without manual per-domain tuning.

D. BPG-Inference: Soft Mixture for More Generalization

The previous subsections describe how BPG learns a collection of domain-specific adapters and classifiers. At inference time, a crucial question is how to leverage these domain-specific models for an unseen test sample. Most existing methods perform *hard* domain selection: a domain-ID predictor is first used to choose one domain, and only the corresponding adapter and classifier are applied. This strategy is brittle: when the predicted domain is wrong or when the sample lies near the boundary of multiple domains, the final prediction can be severely degraded. BPG-Inference replaces hard selection with a *soft domain mixture*. Each domain-specific model is treated as an expert, and their logits are combined according to similarities between the test sample and each domain.

a) Prototype-based Domain Affinity: For each domain t, we run k-means clustering ($k = 5$) on the frozen feature representations of its training samples (the same features used in Eqs. (11) and (12)). This yields k prototypes $\{\boldsymbol{\nu}_{t,j}\}_{j=1}^{k}$ that summarize the feature distribution of domain t. Given a test image, we extract its feature vector $\mathbf{z} \in \mathbb{R}^d$ using the frozen backbone and compute the distance to domain t as the nearest-prototype distance

$$a_t = \min_{j \in \{1,\ldots,k\}} \left\| \mathbf{z} - \boldsymbol{\nu}_{t,j} \right\|_2. \tag{15}$$

Smaller a_t indicates that the sample is closer to domain t in the feature space and thus more likely to be well handled by its adapter and classifier.

We convert distances into normalized domain confidences via

$$w_t = \frac{\exp(-a_t)}{\sum_{i=1}^{T} \exp(-a_i)}, \quad t = 1, \ldots, T. \tag{16}$$

This softmax over negative distances assigns larger weights to domains whose prototypes are closer to the test sample.

b) Confidence-Guided Sparse Domain Mixture: To suppress noisy contributions from irrelevant domains, we apply a parameter-free pruning step on the confidence scores $\{w_t\}_{t=1}^{T}$. We treat the uniform distribution over domains as a threshold: domains whose confidence is lower than this threshold are considered unlikely to be helpful for the current sample and their weights are set to zero. The remaining domains keep their original confidences, which are then rescaled as w_t' so that the surviving weights sum to one. In this way, each test sample is associated with a small set of high-confidence domains whose classifiers are linearly combined, using the rescaled confidences as mixture weights, to produce the final logits, i.e., $\mathbf{y}_{\mathrm{final}} = \sum_{t=1}^{T} w_t' \mathbf{y}_t$. The final prediction is obtained by applying the softmax function to $\mathbf{y}_{\mathrm{final}}$. This soft domain mixture allows multiple domains to jointly contribute to decision making when appropriate and reduces the risk of catastrophic errors caused by incorrect hard domain selection.

E. Theoretical Insight for Capacity Allocation

We now provide a qualitative theoretical motivation for the capacity allocation rule in Eq. (14) and discuss how it helps balance plasticity and generalization. The goal is not to derive a tight bound or claim global optimality; rather, we use this analysis as intuitive guidance for why assigning larger adapters to harder domains and smaller adapters to easier domains is a sensible design choice.

a) A Toy Risk Decomposition: Consider the expected classification risk on domain t as a function of its adapter dimension r_t:

$$\mathcal{R}_t(r_t) = \underbrace{\mathcal{R}_t^{\mathrm{approx}}(r_t)}_{\text{approximation error}} + \underbrace{\mathcal{R}_t^{\mathrm{gen}}(r_t)}_{\text{estimation error}}. \tag{17}$$

The approximation error $\mathcal{R}_t^{\mathrm{approx}}(r_t)$ decreases as r_t increases, because a larger adapter offers higher expressive power to fit domain-specific variations. The estimation error $\mathcal{R}_t^{\mathrm{gen}}(r_t)$, on the other hand, typically increases with r_t, since a more complex model tends to have larger capacity and thus higher generalization error on finite samples [74], [75]. For analytical convenience, we adopt the following simple parametric forms:

$$\mathcal{R}_t^{\mathrm{approx}}(r_t) \approx A_t \exp(-\kappa_t r_t), \tag{18}$$

$$\mathcal{R}_t^{\mathrm{gen}}(r_t) \approx B \sqrt{\frac{r_t}{N_t}}, \tag{19}$$

where $A_t > 0$ and $\kappa_t > 0$ describe how fast the approximation error decays with r_t for domain t, $B > 0$ is a constant, and N_t is the number of training samples. We emphasize that these two expressions are illustrative surrogates rather than exact laws: the exponential form captures the diminishing returns of adding capacity, echoing the geometric error decay of expressive models in classical approximation arguments, while the $\sqrt{r_t/N_t}$ term mirrors standard capacity-based generalization

bounds [74], [75]. Importantly, the derivation that follows uses only their qualitative shape, namely an approximation error that decreases in r_t and an estimation error that increases in r_t, so the same conclusion holds for any pair of terms sharing this monotone structure. The parameter κ_t reflects domain difficulty: when classes are well separated (large s_t), the error can be reduced quickly with small r_t (larger κ_t); when classes are highly entangled (small s_t), more parameters are needed, corresponding to smaller κ_t.

b) Qualitative Allocation Under a Capacity Budget: Suppose that the total adapter capacity is constrained by a budget $\sum_{t=1}^{T} r_t \leq R_{\text{tot}}$. We approximate the average risk by

$$\bar{\mathcal{R}} = \frac{1}{T}\sum_{t=1}^{T} \mathcal{R}_t(r_t). \tag{20}$$

Minimizing $\bar{\mathcal{R}}$ under the budget constraint can be approached via the Lagrangian

$$\mathcal{L} = \frac{1}{T}\sum_{t=1}^{T}\Big(A_t e^{-\kappa_t r_t} + B\sqrt{\tfrac{r_t}{N_t}}\Big) + \alpha\Big(\sum_{t=1}^{T} r_t - R_{\text{tot}}\Big), \tag{21}$$

where $\alpha \geq 0$ is a Lagrange multiplier. We differentiate $\mathcal{L}$ with respect to r_t and set $\frac{\partial \mathcal{L}}{\partial r_t} = 0$, which yields the stationarity condition

$$\underbrace{A_t \kappa_t e^{-\kappa_t r_t^\star}}_{\text{marginal plasticity gain}} = \underbrace{\frac{B}{2\sqrt{N_t r_t^\star}} + \alpha T}_{\text{marginal generalization / budget cost}}. \tag{22}$$

Eq. (22) is precisely the plasticity–generalization balance we seek: at the optimum, the marginal reduction of approximation error obtained by enlarging the adapter (left-hand side, *plasticity*) is equated with the marginal increase of estimation error together with the shadow price of the capacity budget (right-hand side, *generalization*). The allocation is thus determined by the equality of the two competing terms, rather than by discarding either of them. Although Eq. (22) admits no elementary closed-form solution, its structure is informative. The left-hand side decays exponentially in r_t, whereas the right-hand side, $g_t(r_t) := \frac{B}{2\sqrt{N_t r_t}} + \alpha T$, varies only slowly (algebraically) with r_t. Solving for $r_t^\star$ therefore gives

$$r_t^\star \approx \frac{1}{\kappa_t}\ln\left(\frac{A_t \kappa_t}{g_t(r_t^\star)}\right), \tag{23}$$

where $\ln$ denotes the natural logarithm. Because the generalization/budget cost g_t enters only through the logarithm, it acts as a slowly varying factor and is retained rather than neglected: a stronger generalization penalty (larger B or smaller sample size N_t) or a tighter budget (larger α) uniformly shrinks every $r_t^\star$, while the *relative* allocation across domains is governed by the fast exponential term. Consequently, $r_t^\star$ scales roughly like $1/\kappa_t$ up to this slowly varying logarithmic factor: harder domains (smaller κ_t) are assigned larger adapter dimensions, whereas easier domains (larger κ_t) can be modeled with smaller adapters.

The remaining step is to connect the abstract decay rate κ_t to the observable separability score s_t. Crucially, obtaining the desired trend does *not* require an exact functional relationship: since $r_t^\star$ in Eq. (23) decreases with κ_t, *any* monotonically increasing map $\kappa_t = \varphi(s_t)$ already yields $r_t^\star$ decreasing in s_t, i.e., more separable (easier) domains receive smaller adapters. This monotone relationship is corroborated by the empirical behavior reported in Section IV-F: the least separable domains (e.g., Infograph, s_2=0.124, and Quickdraw, s_4=0.151) keep benefiting from larger r up to r=1024, whereas the most separable one (Real, s_5=0.530) already peaks at a small r and degrades under excess capacity; uniformly enlarging r therefore helps hard domains at the cost of more severe degradation on easier ones, indicating a faster error decay (larger κ_t) for easier domains. Adopting the simplest such map, $\kappa_t \propto s_t$, Eq. (23) then reduces to the clean trend

$$r_t^\star \propto \frac{1}{s_t}, \tag{24}$$

which is consistent with our capacity allocation rule in Eq. (14). Therefore, BPG-Adapter can be viewed as a practical heuristic that approximates an optimal allocation of limited capacity across domains with different difficulties.

c) Implications for Plasticity and Generalization: The analysis above highlights the trade-off between plasticity and generalization. Allocating too few parameters to a hard domain results in high approximation error and thus poor plasticity; allocating too many parameters to an easy domain inflates the total capacity, which can harm global generalization under a fixed budget. By adapting r_t according to $1/s_t$, BPG-Adapter increases plasticity where it is most needed and saves parameters where the backbone already captures the domain well. Together with BPG-Inference, which promotes cross-domain sharing at test time via soft mixtures, the BPG framework achieves a more balanced trade-off between learning new domains and preserving performance on past ones.

IV. Experiments

A. Experimental Settings

a) Datasets: To evaluate our method, we conduct experiments on three widely used multi-domain datasets: DomainNet, CDDB, and CORe50. DomainNet [76] is a large-scale dataset designed for domain adaptation and domain incremental learning. It consists of six distinct domains with significant inter-domain variations, each containing 345 categories. The training set comprises 409,832 images, while the test set includes 176,743 images. CDDB [77] is tailored for continual deepfake detection, encompassing multiple deepfake techniques. Following previous works, we adopt the Hard track, which includes GauGAN, BigGAN, WildDeepfake, WhichFaceReal, and SAN as representative deepfake generation methods. CORe50 [78] is an object recognition dataset structured into 11 domains, with eight designated for training and three for testing. Each domain contains 50 classes and approximately 15,000 images.

b) Evaluation Metrics: To assess the performance of DIL methods, we utilize two key evaluation metrics. We denote by $B \in \mathbb{R}^{T\times T}$ a lower triangular matrix, where $B_{i,j}$ is the test accuracy on the j-th domain after training on the i-th domain. (1) Average Accuracy (A_T) measures the classification performance after training on the T-th domain; it is computed as the overall accuracy over all test images from

TABLE I
BETWEEN-CLASS SCATTER R_t^{bcs}, WITHIN-CLASS SCATTER R_t^{wcs}, AND FEATURE SEPARABILITY SCORE s_t ON THE DOMAINNET DATASET.

t	0	1	2	3	4	5	6
R_t^{bcs}	0.392	0.120	0.072	0.192	0.029	0.298	0.088
R_t^{wcs}	0.518	0.543	0.579	0.656	0.193	0.562	0.568
s_t	0.756	0.221	0.124	0.293	0.151	0.530	0.156

TABLE II
BETWEEN-CLASS SCATTER R_t^{bcs}, WITHIN-CLASS SCATTER R_t^{wcs}, AND FEATURE SEPARABILITY SCORE s_t ON THE CDDB DATASET.

t	0	1	2	3	4	5
R_t^{bcs}	0.392	0.007	0.004	0.002	0.004	0.009
R_t^{wcs}	0.518	0.832	0.892	0.440	0.547	0.842
s_t	0.756	0.008	0.004	0.005	0.008	0.011

the T learned domains. (2) Average Forgetting (F_T), which measures the retention of knowledge from previous domains, is computed as:

$$F_T = \frac{1}{T-1}\sum_{j=1}^{T-1}\frac{1}{T-j}\sum_{i=j+1}^{T}(B_{j,j} - B_{i,j}). \quad (25)$$

A higher A_T and a lower F_T indicate better performance of a DIL method.

c) Backbones: Following [22], we adopt pre-trained ViT-B/16 [79] and CLIP [80] as backbones. ViT extracts image features with a classifier trained from scratch; CLIP additionally encodes class-name text features and uses image-text similarity as classification logits.

d) Implementation Details: We train the model for 30, 50, and 20 epochs on the DomainNet, CDDB, and CORe50 datasets, respectively. The model is optimized using SGD with an initial learning rate of 0.01, following a cosine decay schedule. The batch size is set to 128, and all training images are resized to 224×224. Each experiment is repeated three times with different random seeds, and we report the mean and standard deviation. The between-class scatter (R_t^{bcs}), within-class scatter (R_t^{wcs}), and feature separability score (s_t) for each domain in these datasets are presented in Tables I to III, where $t = 0$ denotes ImageNet as the reference domain for computing r_t. The reference hidden dimension r_0 is set to 64 for DomainNet and CORe50, and to 1 for CDDB. All experiments are conducted on a workstation with an Intel Xeon Gold 6226R CPU, 320 GB RAM, and NVIDIA RTX 4090 GPUs; each run requires a single GPU with approximately 18–22 GB of GPU memory.

B. Main Results

a) Compared Methods: We compare BPG with representative continual learning methods across three paradigms: rehearsal-based (iCaRL [96], LUCIR [97], DyTox [81]), rehearsal-free regularization (EWC [33], LwF [82]), and rehearsal-free parameter isolation, including prompt-based methods (L2P [83], DualPrompt [84], S-Prompts [22], CODA-P [85], C-Prompt [41]) and recent approaches (ICON [92], DUCT [86], CONEC-LoRA [89], DCE [90], PC [91], PINA [40]). On the CLIP backbone, we additionally include MoP-CLIP [39], HiDe-Prompt [93], and CP-Prompt [94]. BPG is a rehearsal-free method that requires no replay buffer. We also report the Upper Bound accuracy, which assumes oracle domain IDs at test time, as a performance ceiling.

TABLE III
BETWEEN-CLASS SCATTER R_t^{bcs}, WITHIN-CLASS SCATTER R_t^{wcs}, AND FEATURE SEPARABILITY SCORE s_t ON THE CORE50 DATASET.

t	0	1	2	3	4	5	6	7	8
R_t^{bcs}	0.392	0.205	0.208	0.178	0.162	0.219	0.167	0.199	0.185
R_t^{wcs}	0.518	0.355	0.331	0.347	0.368	0.336	0.363	0.388	0.307
s_t	0.756	0.579	0.629	0.513	0.441	0.650	0.460	0.513	0.604

TABLE IV
EXPERIMENTAL RESULTS ON THE DOMAINNET DATASET. "UPPER BOUND" DENOTES AN IDEAL SCENARIO WHERE THE DOMAIN ID OF THE TEST SET IS KNOWN. * RESULTS REPORTED IN THE ORIGINAL PAPER. **BOLD**/UNDERLINE: BEST/SECOND-BEST.

Backbone	Method	Buffer (↓)	A_T (↑)	F_T (↓)
ViT	DyTox [81]	50/class	62.94	-
	EWC [33]	0/class	47.62±1.35	12.85±1.03
	LwF [82]		49.19±1.13	5.01±0.59
	L2P [83]		40.15±2.73	2.25±0.51
	DualPrompt [84]		43.79±1.55	2.03±0.89
	S-iPrompts [22]		50.62±0.15	2.85±0.28
	CODA-P [85]		47.42±0.78	3.46±0.89
	PINA [40]		54.86±0.71	2.24±0.84
	DUCT* [86]		67.01±1.35	-
	C-Prompt [41]		58.68±1.28	1.34±0.36
	DualCP [87]		60.13±2.16	1.96±0.75
	SOYO [88]		65.25±0.92	<u>1.26±0.41</u>
	CONEC-LoRA* [89]		66.42±0.38	-
	KA-Prompt* [56]		62.91±0.14	1.93±0.26
	DCE [90]		63.50±0.50	-
	PC [91]		58.82±0.46	1.27±0.18
	ICON [92]		<u>67.95±1.87</u>	8.18±1.80
	BPG (ours)	0/class	**72.19±0.86**	**0.22±0.12**
	Upper Bound		74.21±0.25	0.00
CLIP	S-liPrompts [22]	0/class	67.78±0.90	1.64±0.68
	MoP-CLIP* [39]		69.70	-
	PINA [40]		69.06±1.61	<u>1.59±0.33</u>
	HiDe-Prompt* [93]		60.15	-
	CP-Prompt* [94]		<u>73.35</u>	-
	BPG (ours)	0/class	**75.72±0.15**	**0.59±0.03**
	Upper Bound		77.53±0.39	0.00

b) Results on DomainNet: Table IV summarizes results on DomainNet, which comprises six visually distinct natural image domains. On ViT, BPG achieves A_T=72.19% and F_T=0.22%, surpassing the second-best rehearsal-free method ICON [92] (67.95%) by 4.24% in accuracy and SOYO [88] (1.26%) by 1.04% in forgetting. Without any replay, BPG also exceeds DyTox [81] (50 samples/class; 62.94%) by 9.25%. The near-zero forgetting indicates that BPG-Inference mitigates errors from hard domain ID selection by softly combining domain-specific experts. BPG further narrows the gap to the Upper Bound (74.21%) to 2.02%. On CLIP, BPG attains 75.72% and 0.59% in A_T and F_T, respectively, exceeding CP-Prompt [94] (73.35%) by 2.37% while maintaining low forgetting.

c) Results on CDDB: Table V reports results on CDDB, a continual deepfake detection benchmark where each domain corresponds to a distinct generation method. On ViT, BPG achieves the best A_T (88.55%) and F_T (0.68%). Without stor-

TABLE V
EXPERIMENTAL RESULTS ON THE CDDB DATASET. "UPPER BOUND" DENOTES AN IDEAL SCENARIO WHERE THE DOMAIN ID OF THE TEST SET IS KNOWN. * RESULTS REPORTED IN THE ORIGINAL PAPER. **BOLD**/UNDERLINE: BEST/SECOND-BEST.

Backbone	Method	Buffer (↓)	A_T (↑)	F_T (↓)
ViT	LRCIL [95]	100/class	76.39	4.39
	iCaRL [96]		79.76	8.73
	LUCIR [97]		82.53	5.34
	LRCIL [95]	50/class	74.01	8.62
	iCaRL [96]		73.98	14.50
	LUCIR [97]		80.77	7.85
	DyTox [81]		86.21	1.55
	EWC [33]	0/class	50.59±3.79	42.62±1.83
	LwF [82]		60.94±1.69	13.53±0.50
	DyTox [81]		51.27±5.07	45.85±1.85
	L2P [83]		61.28±0.52	9.23±0.23
	DualPrompt [84]		64.80±1.82	8.74±0.73
	S-iPrompts [22]		74.51±1.96	1.30±1.12
	CODA-P [85]		70.54±0.37	5.53±0.94
	PINA [40]		77.35±1.46	0.98±0.22
	C-Prompt [41]		78.44±0.87	1.55±0.15
	KA-Prompt [56]		80.78±1.21	0.96±0.26
	DualCP [87]		82.16±1.53	0.73±0.25
	CONEC-LoRA* [89]		88.21±0.88	-
	DUCT* [86]		85.10±0.52	-
	DCE* [90]		71.80±4.20	-
	BPG (ours)	0/class	**88.55±0.32**	**0.68±0.09**
	Upper Bound		89.17±0.40	0.00
CLIP	S-liPrompts [22]	0/class	88.65±0.64	0.69±0.26
	MoP-CLIP* [39]		88.54	0.79
	PINA [40]		85.71±1.99	0.51±0.27
	HiDe-Prompt* [93]		84.32	2.61
	CP-Prompt* [94]		93.65	0.25
	BPG (ours)	0/class	**93.91±0.39**	**0.12±0.06**
	Upper Bound		94.40±0.25	0.00

TABLE VI
EXPERIMENTAL RESULTS ON THE CORE50 DATASET. SINCE THE TRAINING AND TESTING DOMAINS DO NOT OVERLAP, AVERAGE FORGETTING AND UPPER BOUND ACCURACY ARE NOT APPLICABLE. * RESULTS REPORTED IN THE ORIGINAL PAPER. **BOLD**/UNDERLINE: BEST/SECOND-BEST.

Backbone	Method	Buffer (↓)	A_T (↑)
ViT	ER [98]	50/class	80.10
	GDumb [99]		74.92
	BiC [100]		79.28
	DER++ [101]		79.70
	Co^{2}L [102]		79.75
	DyTox [81]		79.21
	L2P [83]		81.07
	EWC [33]	0/class	74.82±1.81
	LwF [82]		75.45±1.27
	L2P [83]		78.33±2.12
	DualPrompt [84]		80.25±1.09
	S-iPrompts [22]		83.13±1.62
	ICON [92]		74.98±0.03
	CODA-P [85]		85.68±0.31
	C-Prompt [41]		85.31±1.87
	KA-Prompt [56]		85.61±1.64
	PINA [40]		86.74±0.82
	DCE* [90]		84.80±0.30
	DualCP [87]		88.10±0.89
	CONEC-LoRA* [89]		90.24±1.64
	SOYO [88]		90.77±0.53
	PC [91]		91.35±0.39
	BPG (ours)	0/class	**91.87±0.31**
	Upper Bound		N/A
CLIP	S-liPrompts [22]	0/class	89.06±1.44
	MoP-CLIP* [39]		92.29
	PINA [40]		87.38±1.92
	HiDe-Prompt* [93]		80.81±0.76
	CP-Prompt* [94]		90.67±0.55
	BPG (ours)	0/class	**92.46±0.42**
	Upper Bound		N/A

ing past samples, it outperforms all rehearsal-based methods, including DyTox [81] (50 samples/class; 86.21%) and LUCIR [97] (100 samples/class; 82.53%). Its forgetting is 0.05% lower than DualCP [87] (0.73%). On CLIP, BPG reaches 93.91% with 0.12% forgetting, surpassing CP-Prompt [94] (93.65%) and approaching the Upper Bound (94.40%) within 0.49%.

d) Results on CORe50: Table VI reports results on CORe50, where training and test domains do not overlap; consequently, F_T and the Upper Bound are not applicable. BPG-Inference therefore softly aggregates predictions from all learned domain-specific models according to feature-space affinity, rather than committing to a single domain ID. On ViT, BPG attains 91.87%, outperforming the second-best method PC [91] (91.35%) and exceeding L2P [83] with a 50-sample buffer (81.07%) by 10.80%. On CLIP, BPG reaches 92.46%, surpassing MoP-CLIP [39] (92.29%) by 0.17%. These results confirm that BPG generalizes effectively to entirely unseen test domains.

C. Ablation Study

a) Component-wise Analysis: Table VII presents the ablation results, where the baseline trains only the domain prompt and classifier on top of the frozen backbone and performs hard domain selection at inference. Both BPG-Adapter and BPG-Inference independently improve accuracy while reducing forgetting. By adaptively allocating larger capacity to harder domains, BPG-Adapter attains 68.33% accuracy with 1.28% forgetting on DomainNet (ViT), and BPG-Inference is particularly effective at reducing forgetting.

b) Generality of BPG-Inference: To demonstrate the broad applicability of our soft mixture strategy, we integrate BPG-Inference into several representative prompt-based continual learning methods, as shown in Table VIII. BPG-Inference consistently improves all baseline methods across all three benchmarks. For instance, when applied to S-iPrompts, it yields +5.42% accuracy gain on DomainNet with 0.74% forgetting reduction. Similar improvements are observed for PINA (+4.48% on DomainNet), C-Prompt (+4.83% on DomainNet), and KA-Prompt (+2.52% on DomainNet). These results confirm that BPG-Inference is a general and effective strategy that can enhance existing parameter-isolation methods by mitigating the brittleness of hard domain selection at inference time.

D. Additional Analysis on BPG-Adapter

a) Effect of BPG-Adapter and Hidden Dimension: To evaluate the impact of the adapter's hidden dimension on model performance, we conducted experiments on the Do-

TABLE VII

ABLATION STUDY ON BPG COMPONENTS. WE EVALUATE THE INDIVIDUAL AND COMBINED CONTRIBUTIONS OF BPG-ADAPTER (ADAPTIVE CAPACITY ALLOCATION) AND BPG-INFERENCE (SOFT MIXTURE STRATEGY) USING BOTH VIT-B/16 AND CLIP-VIT-B/16 BACKBONES. "BASELINE" TRAINS ONLY THE DOMAIN PROMPT AND CLASSIFIER (WITHOUT ADAPTERS) AND USES HARD DOMAIN SELECTION AT INFERENCE. BOTH COMPONENTS CONSISTENTLY IMPROVE PERFORMANCE, AND THEIR COMBINATION ACHIEVES THE BEST RESULTS ACROSS ALL BENCHMARKS.

Method	Backbone ViT	Backbone CLIP	BPG-Adapter	BPG-Inference	DomainNet A_T (↑)	DomainNet F_T (↓)	CDDB A_T (↑)	CDDB F_T (↓)	CORe50 A_T (↑)
Baseline	✓				61.69±1.68	1.86±0.46	80.52±1.21	0.93±0.07	83.23±1.07
Baseline + BPG-Adapter	✓		✓		68.33±0.89	1.28±0.17	86.94±1.93	0.85±0.42	90.15±1.09
Baseline + BPG-Inference	✓			✓	66.25±1.53	1.54±0.30	83.84±2.07	0.77±0.51	84.55±0.58
BPG (ours)	✓		✓	✓	**72.19±0.86**	**0.22±0.12**	**88.55±0.32**	**0.68±0.09**	**91.87±0.31**
Baseline		✓			67.85±0.72	1.43±0.26	91.07±1.49	0.30±0.44	86.87±1.12
Baseline + BPG-Adapter		✓	✓		73.59±0.79	1.15±0.41	93.42±0.32	0.26±0.21	92.09±1.06
Baseline + BPG-Inference		✓		✓	70.46±0.47	1.21±0.23	91.76±1.38	0.31±0.29	87.50±1.91
BPG (ours)		✓	✓	✓	**75.72±0.15**	**0.59±0.03**	**93.91±0.39**	**0.12±0.06**	**92.46±0.42**

TABLE VIII

GENERALITY OF BPG-INFERENCE. WE INTEGRATE OUR SOFT MIXTURE INFERENCE STRATEGY INTO EXISTING PROMPT-BASED CONTINUAL LEARNING METHODS. BPG-INFERENCE CONSISTENTLY IMPROVES ALL BASELINES ACROSS BENCHMARKS, DEMONSTRATING ITS BROAD APPLICABILITY AS A PLUG-AND-PLAY ENHANCEMENT. NUMBERS IN PARENTHESES INDICATE IMPROVEMENTS OVER THE ORIGINAL METHODS.

Method	DomainNet A_T (↑)	DomainNet F_T (↓)	CDDB A_T (↑)	CDDB F_T (↓)	CORe50 A_T (↑)
S-iPrompts [22]	50.62	2.85	74.51	1.30	83.13
S-iPrompts + BPG-Inference	56.04 (+5.42)	2.11 (-0.74)	77.69 (+3.18)	1.10 (-0.20)	83.80 (+0.67)
PINA [40]	54.86	2.24	77.35	0.98	86.74
PINA + BPG-Inference	59.34 (+4.48)	1.72 (-0.52)	79.49 (+2.14)	0.84 (-0.14)	87.35 (+0.61)
C-Prompt [41]	58.68	1.34	78.44	1.55	85.31
C-Prompt + BPG-Inference	63.51 (+4.83)	1.03 (-0.31)	80.57 (+2.13)	1.30 (-0.25)	86.45 (+1.14)
KA-Prompt [56]	62.91	1.93	80.78	0.96	85.61
KA-Prompt + BPG-Inference	65.43 (+2.52)	1.19 (-0.74)	83.52 (+2.74)	0.75 (-0.21)	86.02 (+0.41)

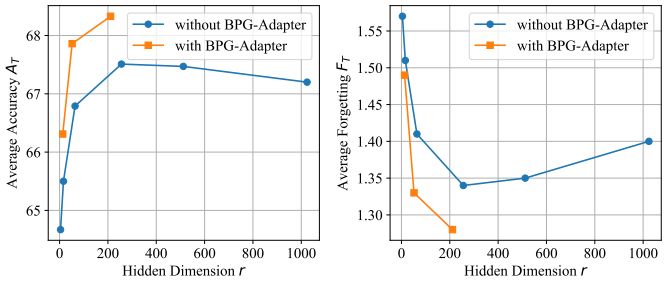


Fig. 3. Ablation study on the hidden dimension r and BPG-Adapter. The two curves compare models with and without BPG-Adapter under varying hidden dimensions, reporting both average accuracy A_T (left) and average forgetting F_T (right) on DomainNet.

TABLE IX

ABLATION ON DIFFERENT FINE-TUNING METHODS BASED ON VIT. BPG-A: BPG-ADAPTER; BPG-I: BPG-INFERENCE.

Dataset	BPG-A	BPG-I	Adapter A_T (↑)	Adapter F_T (↓)	LoRA A_T (↑)	LoRA F_T (↓)
DomainNet			61.69	1.86	61.69	1.86
	✓		68.33	1.28	68.04	1.39
	✓	✓	**72.19**	**0.22**	**71.83**	**0.27**
CDDB			80.52	0.93	80.52	0.93
	✓		86.94	0.85	86.92	0.88
	✓	✓	**88.55**	**0.68**	**88.48**	**0.73**
CORe50			83.23	–	83.23	–
	✓		90.15	–	90.36	–
	✓	✓	**91.87**	–	**92.14**	–

mainNet dataset, as illustrated in Fig. 3. For models without BPG-Adapter, a standard adapter with a uniform hidden dimension is assigned to each domain. We set the hidden dimension r to 4, 16, 64, 256, 512, and 1024, measuring both average accuracy and average forgetting. For models equipped with BPG-Adapter, we set the base hidden dimension r_0 to 4, 16, and 64. Since BPG-Adapter dynamically determines the actual hidden dimension for each domain according to feature separability, the resulting dimension is generally larger than r_0. To enable a fair comparison, we report the equivalent hidden dimension in this figure. The results reveal two key findings. First, BPG-Adapter consistently outperforms the uniform counterpart in both accuracy and forgetting across comparable hidden dimensions, confirming that adaptively allocating capacity according to domain complexity is more effective than assigning a fixed size to all domains. Second, with BPG-Adapter, a small base dimension ($r_0 = 64$, corresponding to an equivalent dimension around 256) already achieves near-optimal accuracy while maintaining the lowest forgetting, demonstrating that our method enables efficient parameter utilization by assigning larger adapters only to domains that genuinely require additional capacity.

b) Different Tuning Methods: In existing parameter-isolation DIL methods, two common parameter-efficient fine-tuning approaches are Adapter and LoRA. Our primary experiments are conducted using Adapter. To evaluate the generalizability of our method, we also present experimental results using LoRA in Table IX. The results demonstrate that the proposed adaptive capacity allocation transfers well to both fine-tuning paradigms. Adapter-based tuning yields slightly stronger results on DomainNet, where domain shifts are more pronounced, whereas LoRA remains competitive on CDDB and CORe50.

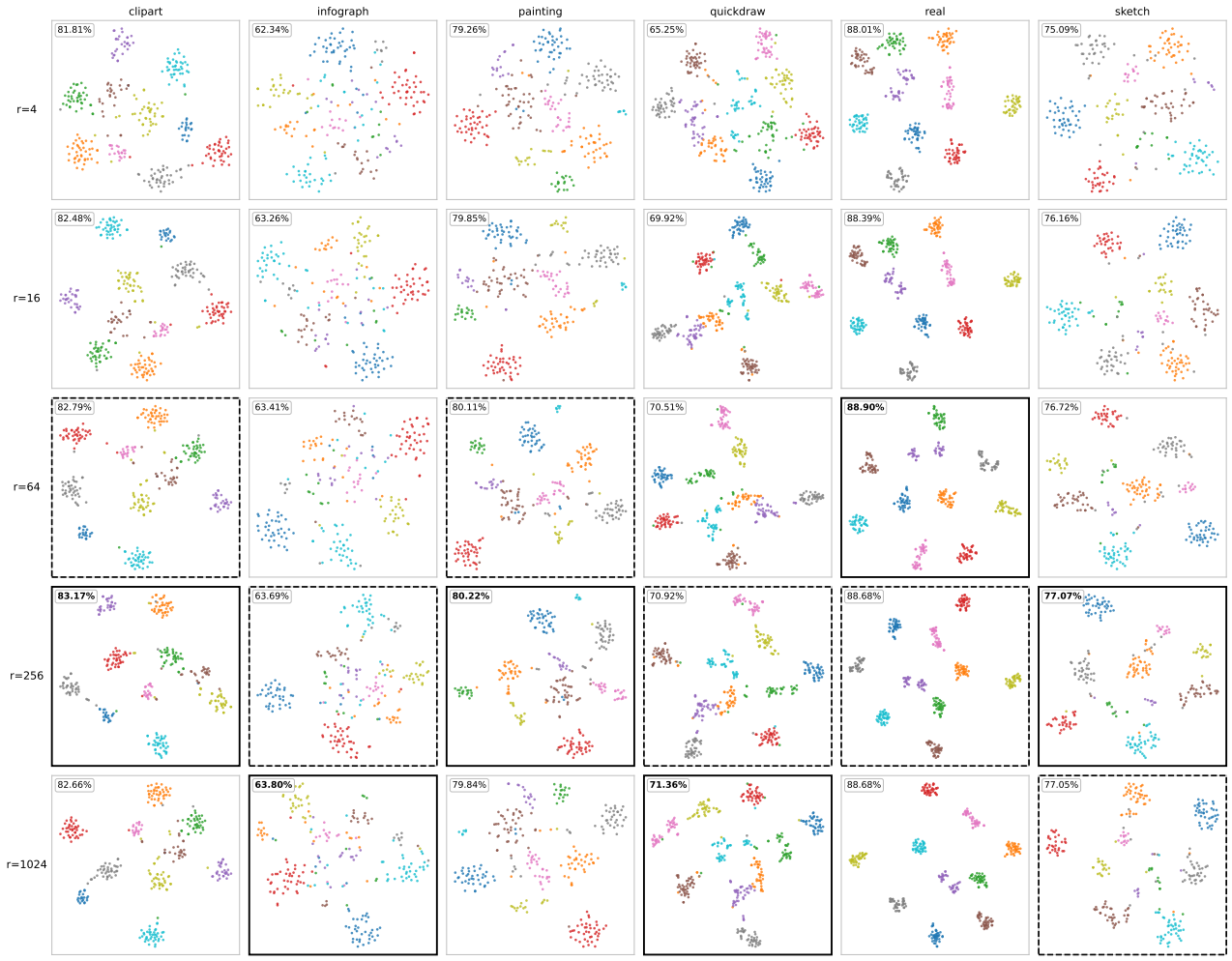


Fig. 4. t-SNE visualizations of domain-specific features on DomainNet under uniform adapter hidden dimensions. Each column is a domain and each row a fixed r; solid/dashed boxes mark the highest and second-highest accuracy among tested r per column.

E. Additional Analysis on BPG-Inference

a) Detailed Analysis on BPG-Inference: Table X compares BPG-Adapter with hard domain selection (*BPG-Adapter only*) against the full BPG framework on DomainNet. Hard selection causes Domain 1 accuracy to drop from 78.49% to 71.06% as more domains are added, reflecting growing domain-ID misselection. BPG-Inference mitigates this via multi-expert logit fusion. After training on the third domain, the accuracy of Domain 1 improves from 73.64% to 75.71% and eventually recovers to 76.83%.

b) Ablation on Weighting Methods in BPG-Inference: We further ablate logit fusion on the BPG-Adapter-only baseline (Table XI). Naive averaging (Mean) and max pooling (Max) underperform hard selection, whereas threshold-filtered fusion with L1 normalization (Thres+L1, our default) improves A_T by 3.86% and reduces F_T to 0.22%, confirming that ambiguous samples benefit from selective multi-domain aggregation.

TABLE X
PER-DOMAIN ACCURACY MATRIX ON DOMAINNET: BPG-ADAPTER ONLY (HARD SELECTION) VS. FULL BPG. D_1 REPRESENTS DOMAIN 1, AND SO FORTH.

BPG-Adapter only	A_T (↑): 68.33 F_T (↓): 1.28					
	Test D_1	Test D_2	Test D_3	Test D_4	Test D_5	Test D_6
Train D_1	78.49					
Train D_2	74.95	43.04				
Train D_3	73.64	42.84	67.16			
Train D_4	72.81	42.43	66.85	69.31		
Train D_5	71.95	41.80	66.29	69.31	78.16	
Train D_6	71.06	40.58	65.70	69.13	77.94	63.94
BPG (full)	A_T (↑): 72.19 F_T (↓): 0.22					
	Test D_1	Test D_2	Test D_3	Test D_4	Test D_5	Test D_6
Train D_1	78.49					
Train D_2	74.95	43.04				
Train D_3	75.71	42.99	70.44			
Train D_4	76.09	43.50	71.56	70.82		
Train D_5	76.31	44.04	71.07	70.81	82.92	
Train D_6	76.83	44.10	70.85	70.57	83.03	68.31

F. Visualization

Fig. 4 complements the ablation in Fig. 3 with t-SNE visualizations of learned features on DomainNet, varying a uniform adapter hidden dimension $r \in \{4, 16, 64, 256, 1024\}$ across all domains (test accuracy annotated per subplot). Domain-wise optima are heterogeneous: easier domains such as Real (s_5=0.530) peak at smaller r and degrade when given excess capacity, whereas the hardest domains, Infograph

TABLE XI
ABLATION ON BPG-INFERENCE WEIGHTING. HARD SELECTION: BPG-ADAPTER ONLY; MEAN: SIMPLE AVERAGE OF ALL LOGITS; MAX: SELECT THE HIGHEST LOGIT VALUE ACROSS ALL DOMAINS AS THE CLASS PREDICTION; THRES: CONFIDENCE THRESHOLDING; L1/L2: WEIGHT NORMALIZATION.

	Hard selection	Mean	Max	Thres+Mean	Thres+L1	Thres+L2
A_T (↑)	68.33	56.87	66.67	71.25	72.19	72.24
F_T (↓)	1.28	N/A	2.08	0.49	0.22	0.26
ΔA_T	0	-11.46	-1.66	+2.92	+3.86	+3.91

TABLE XII
PARAMETER BUDGET ON DOMAINNET (VIT-B/16, $r_0 = 64$). ADAPTER PARAMS. ARE COMPUTED AS $2Ldr_t + Ld + Lr_t$ WITH $L = 12$ AND $d = 768$. PER-DOMAIN TOTAL INCLUDES ADAPTERS, PROMPT TOKENS, AND THE LINEAR CLASSIFIER.

Domain t	s_t	r_t	Adapter (M)	Domain total (M)
1	0.221	219	4.05	4.32
2	0.124	390	7.20	7.48
3	0.293	165	3.05	3.33
4	0.151	320	5.91	6.18
5	0.530	91	1.69	1.96
6	0.156	310	5.73	6.00
BPG (adaptive, $r_0 = 64$)			27.63	**29.26**
Uniform ($r = 64$)			7.14	8.77
Uniform ($r = 256$)			28.39	30.02

(s_2=0.124) and Quickdraw (s_4=0.151), continue to improve up to r=1024; the remaining domains reach their best accuracy at intermediate values (e.g., r=256). Crucially, uniformly increasing r is not globally optimal. Although hard domains benefit from larger capacity, the same setting increasingly over-parameterizes easier domains and more severely impairs performance on the others, as also evidenced by the rising forgetting in Fig. 3. This per-domain asymmetry motivates difficulty-aware capacity allocation in BPG-Adapter.

G. Efficiency Analysis

We analyze parameter budget, training overhead, and inference cost on DomainNet (ViT-B/16, L=12, d=768, C=345, T=6). Per domain, trainable parameters comprise adapters ($2Ldr_t$+Ld+Lr_t), prompt tokens (Md), and a linear classifier (Cd+C). BPG-Adapter reallocates capacity via Eq. (14) instead of fixing r_t=r.

a) Parameter Budget: Table XII reports per-domain s_t, r_t, and trainable parameter counts. At r_0=64, harder domains receive larger adapters (e.g., Infograph: r_2=390) and easier ones smaller capacities (e.g., Real: r_5=91). Total capacity ($\sum_t r_t$=1495, $\bar{r}$≈249) matches a uniform adapter at r=249 (Fig. 3). With 29.3 M parameters, BPG surpasses a uniform r=64 baseline (8.8 M; A_T=66.79%, Fig. 3) at A_T=68.33%. Matched to uniform r=256 (30.0 M), it still wins on both A_T (68.33% vs. 67.51%) and F_T (1.28% vs. 1.34%), showing gains from difficulty-aware reallocation over uniform scaling.

b) Training Overhead: BPG follows the same sequential protocol: each session updates only the current domain's prompt, adapter, and classifier for 30 epochs. Table XIII reports wall-clock time on DomainNet (ViT-B/16). All methods share a one-time prototype cost of 0.34 h (k-means, k=5); BPG adds 0.38 h for separability scoring (one frozen-backbone forward pass per new domain), less than a single training epoch. Larger adapters on harder domains slightly extend main training (22.30 h vs. 21.62 h at r=64 and 21.83 h at r=256). Overall, BPG finishes in 23.02 h, only +4.8% over uniform r=64 (21.96 h) and +3.8% over r=256 (22.17 h). Crucially, BPG introduces no learnable router and therefore incurs no joint router-training overhead.

TABLE XIII
WALL-CLOCK TRAINING TIME ON DOMAINNET (VIT-B/16, 30 EPOCHS PER DOMAIN, $T = 6$), IN HOURS ON A SINGLE GPU AGGREGATED OVER INCREMENTAL SESSIONS. TRAINING: MAIN EPOCH-WISE OPTIMIZATION; SEPARABILITY: ONE-TIME FEATURE SEPARABILITY SCORING; PROTOTYPES: ONE-TIME DOMAIN PROTOTYPE BUILDING VIA k-MEANS.

Method	Training	Separability	Prototypes	Total
Uniform ($r = 64$)	21.62	0	0.34	21.96
Uniform ($r = 256$)	21.83	0	0.34	22.17
BPG (adaptive, $r_0 = 64$)	22.30	0.38	0.34	23.02

c) Inference Latency: BPG-Inference fuses logits from all T domain-specific models at test time; hard selection uses only the highest-confidence expert. On DomainNet (ViT-B/16, T=6, single GPU), average per-image latency increases from 11.5 ms to 74.2 ms. The overhead is confined to inference and does not alter the training schedule (Table XIII). For offline DIL deployments that prioritize robustness over throughput, 74.2 ms per image remains modest. Unlike MoE-based soft routing, BPG achieves comparable logit aggregation without a learnable gating module or joint router training. When peak throughput is required, hard selection provides a simple latency-accuracy knob within the same framework.

V. CONCLUSION

We address two key bottlenecks in domain incremental learning: insufficient plasticity from uniform adapter capacity and brittle generalization from hard domain selection at inference. We propose BPG, a unified framework comprising BPG-Adapter, which adaptively allocates adapter hidden dimensions from feature separability, and BPG-Inference, which soft-mixes logits from multiple domain-specific models. Experiments on DomainNet, CDDB, and CORe50 show that BPG achieves state-of-the-art average accuracy with near-zero forgetting, validating the need to jointly adapt plasticity and generalization in lifelong learning.

Limitations and Future Work: BPG operates within the parameter-isolation paradigm for rehearsal-free DIL, learning domain-specific parameters sequentially without storing past samples. When a rehearsal buffer is available and privacy constraints permit, combining BPG with selective replay may yield further gains. BPG-Inference increases per-image latency by aggregating logits from all domain experts; this overhead is confined to inference, requires no learnable router, and hard domain selection remains available when peak throughput is critical.

REFERENCES

[1] Z. Zhang, D. Chang, R. Zhu, X. Li, Z. Ma, and J.-H. Xue, "Query-aware cross-mixup and cross-reconstruction for few-shot fine-grained image classification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 35, no. 2, pp. 1276–1286, 2024.

[2] J. Li, S. Wang, B. Qian, Y. He, X. Wei, and Y. Gong, "Dynamic integration of task-specific adapters for class incremental learning," *arXiv preprint arXiv:2409.14983*, 2024.

[3] C. Ding, S. Dong, Z. Zhou, J. Han, Q. Wang, Y. He, and Y. Gong, "Class-independent increment: An efficient approach for multi-label class-incremental learning," *arXiv preprint arXiv:2503.00515*, 2025.

[4] J. Han, C. Ding, Y. He, S. Dong, Q. Wang, X. Gao, and Y. Gong, "Learn by reasoning: Analogical weight generation for few-shot class-incremental learning," *IEEE Transactions on Circuits and Systems for Video Technology*, 2025.

[5] S. Lu, Y. He, J. Li, Q. Wang, and Y. Gong, "Gfpl: Generative federated prototype learning for resource-constrained and data-imbalanced vision task," *arXiv preprint arXiv:2602.21873*, 2026.

[6] B.-H. Chen, L.-F. Shi, and X. Ke, "A robust moving object detection in multi-scenario big data for video surveillance," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 4, pp. 982–995, 2018.

[7] X. Song, Y. He, J. Li, Q. Wang, and Y. Gong, "Learning endogenous attention for incremental object detection," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 30 354–30 364.

[8] J. Zhao, C. Ding, S. Dong, J. Li, Q. Wang, Y. He, and Y. Gong, "Shared & domain self-adaptive experts with frequency-aware discrimination for continual test-time adaptation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 40, no. 34, 2026, pp. 28 697–28 705.

[9] Z. Shi, Z. Wu, J. Wen, M. Li, X. Fan, Y. Wang, and L. Shen, "Decenter: Density-center guided perception enhancement for uav object detection," *IEEE Transactions on Circuits and Systems for Video Technology*, 2026.

[10] X. Li, Y. Liu, G. Sun, M. Wu, L. Zhang, and C. Zhu, "Towards open-vocabulary video semantic segmentation," *IEEE Transactions on Multimedia*, 2025.

[11] Y. Tang, T. Chen, X. Jiang, Y. Yao, G.-S. Xie, and H.-T. Shen, "Holistic prototype attention network for few-shot video object segmentation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 34, no. 8, pp. 6699–6709, 2023.

[12] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," in *Psychology of learning and motivation*. Elsevier, 1989, vol. 24, pp. 109–165.

[13] C. Ding, X. Gao, S. Dong, Y. He, Q. Wang, A. Kot, and Y. Gong, "Lobg: less overfitting for better generalization in vision-language model," *arXiv preprint arXiv:2410.10247*, 2024.

[14] J. Han, C. Ding, S. Dong, Y. He, S. Wang, Q. Wang, and Y. Gong, "Goal: Geometrically optimal alignment for continual generalized category discovery," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 40, no. 6, 2026, pp. 4565–4573.

[15] J. Li, C. Wan, S. Dong, C. Ding, Q. Wang, Z. Ma, and Y. Gong, "Trajectory-diversity-driven robust vision-and-language navigation," *arXiv preprint arXiv:2603.15370*, 2026.

[16] J. Han, S. Wang, Y. He, C. Ding, Q. Wang, X. Gao, S. Dong, and Y. Gong, "Consistent supervised-unsupervised alignment for generalized category discovery," *arXiv preprint arXiv:2507.04725*, 2025.

[17] S. Dong, C. Ding, J. Li, J. Han, Q. Wang, Y. He, and Y. Gong, "Beyond clip generalization: Against forward&backward forgetting adapter for continual learning of vision-language models," *arXiv preprint arXiv:2505.07690*, 2025.

[18] R. Yang, T. Yang, Z. Yan, T. Krajnik, and Y. Ruichek, "Preventing catastrophic forgetting in continuous online learning for autonomous driving," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 5505–5512.

[19] M. Perkonigg, J. Hofmanninger, C. J. Herold, J. A. Brink, O. Pianykh, H. Prosch, and G. Langs, "Dynamic memory to alleviate catastrophic forgetting in continual learning with medical imaging," *Nature communications*, vol. 12, no. 1, p. 5678, 2021.

[20] K. Doshi and Y. Yilmaz, "Continual learning for anomaly detection in surveillance videos," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020, pp. 254–255.

[21] Q. Wang, X. Gao, S. Dong, J. Han, J. Li, Y. He, and Y. Gong, "Vdc-agent: When video detailed captioners evolve themselves via agentic self-reflection," *arXiv preprint arXiv:2511.19436*, 2025.

[22] Y. Wang, Z. Huang, and X. Hong, "S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 5682–5695, 2022.

[23] D. S. Tan, Y.-X. Lin, and K.-L. Hua, "Incremental learning of multi-domain image-to-image translations," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 4, pp. 1526–1539, 2020.

[24] X. Gao, S. Dong, Y. He, Q. Wang, and Y. Gong, "Beyond prompt learning: Continual adapter for efficient rehearsal-free continual learning," in *European Conference on Computer Vision*. Springer, 2024, pp. 89–106.

[25] Y. Hu, J. Hou, X. Liu, X. Sun, and W. Guo, "Video domain incremental learning for human action recognition in home environments," in *International Conference on Image and Graphics*. Springer, 2025, pp. 316–327.

[26] Z. Tao, L. Yu, H. Yao, S. Huang, and C. Xu, "Class incremental learning for light-weighted networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 34, no. 12, pp. 12 210–12 220, 2024.

[27] L. Ding, X. Song, Y. He, C. Wang, S. Dong, X. Wei, and Y. Gong, "Domain incremental object detection based on feature space topology preserving strategy," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 34, no. 1, pp. 424–437, 2023.

[28] Y.-S. Liang and W.-J. Li, "Inflora: Interference-free low-rank adaptation for continual learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 23 638–23 647.

[29] L. Li, S. Wang, C. Li, Y. Yuan, and G. Wang, "Dc-lora: Domain correlation low-rank adaptation for domain incremental learning," *High-Confidence Computing*, p. 100270, 2025.

[30] D. Isele and A. Cosgun, "Selective experience replay for lifelong learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[31] D. Rolnick, A. Ahuja, J. Schwarz, T. Lillicrap, and G. Wayne, "Experience replay for continual learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[32] H. Zhao, H. Wang, Y. Fu, F. Wu, and X. Li, "Memory-efficient class-incremental learning for image classification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 10, pp. 5966–5977, 2021.

[33] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[34] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *International conference on machine learning*. PMLR, 2017, pp. 3987–3995.

[35] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars, "Memory aware synapses: Learning what (not) to forget," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 139–154.

[36] A. F. Akyürek, E. Akyürek, D. T. Wijaya, and J. Andreas, "Subspace regularizers for few-shot class incremental learning," *arXiv preprint arXiv:2110.07059*, 2021.

[37] H. Liu, L. Gu, Z. Chi, Y. Wang, Y. Yu, J. Chen, and J. Tang, "Few-shot class-incremental learning via entropy-regularized data-free replay," in *European Conference on Computer Vision*. Springer, 2022, pp. 146–162.

[38] Y. Shi, D. Shi, Z. Qiao, Z. Wang, Y. Zhang, S. Yang, and C. Qiu, "Multi-granularity knowledge distillation and prototype consistency regularization for class-incremental learning," *Neural Networks*, vol. 164, pp. 617–630, 2023.

[39] J. Nicolas, F. Chiaroni, I. Ziko, O. Ahmad, C. Desrosiers, and J. Dolz, "Mop-clip: A mixture of prompt-tuned clip models for domain incremental learning," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 1762–1772.

[40] Q. Wang, Y. He, S. Dong, X. Gao, S. Wang, and Y. Gong, "Non-exemplar domain incremental learning via cross-domain concept integration," in *European Conference on Computer Vision*. Springer, 2024, pp. 144–162.

[41] Z. Liu, Y. Peng, and J. Zhou, "Compositional prompting for anti-forgetting in domain incremental learning," *International Journal of Computer Vision*, pp. 1–18, 2024.

[42] S. Wang, C. Li, J. Tang, X. Gong, Y. Yuan, and G. Wang, "Importance-aware shared parameter subspace learning for domain incremental learning," in *Proceedings of the 32nd ACM International Conference on Multimedia*, 2024, pp. 8874–8883.

[43] C. D. Kim, J. Jeong, and G. Kim, "Imbalanced continual learning with partitioning reservoir sampling," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*. Springer, 2020, pp. 411–428.

[44] C. Ding, X. Gao, S. Dong, Y. He, Q. Wang, X. Song, A. Kot, and Y. Gong, "Space rotation with basis transformation for training-free test-time adaptation," *arXiv preprint arXiv:2502.19946*, 2025.

[45] M. Zhang, T. Wang, J. H. Lim, G. Kreiman, and J. Feng, "Variational prototype replays for continual learning," *arXiv preprint arXiv:1905.09447*, 2019.

[46] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," *Advances in neural information processing systems*, vol. 30, 2017.

[47] G. M. Van De Ven, Z. Li, and A. S. Tolias, "Class-incremental learning with generative classifiers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3611–3620.

[48] M. D. McDonnell, D. Gong, A. Parvaneh, E. Abbasnejad, and A. Van den Hengel, "Ranpac: Random projections and pre-trained models for continual learning," *Advances in Neural Information Processing Systems*, vol. 36, pp. 12 022–12 053, 2023.

[49] D.-W. Zhou, Z.-W. Cai, H.-J. Ye, D.-C. Zhan, and Z. Liu, "Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need," *International Journal of Computer Vision*, vol. 133, no. 3, pp. 1012–1032, 2025.

[50] G. Zhang, L. Wang, G. Kang, L. Chen, and Y. Wei, "Slca: Slow learner with classifier alignment for continual learning on a pre-trained model," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 19 148–19 158.

[51] P. Buzzega, M. Boschini, A. Porrello, D. Abati, and S. Calderara, "Dark experience for general continual learning: a strong, simple baseline," *Advances in neural information processing systems*, vol. 33, pp. 15 920–15 930, 2020.

[52] J. Li, C. Ding, S. Dong, Q. Wang, J. Zhao, Y. He, and Y. Gong, "Is parameter isolation better for prompt-based continual learning?" *arXiv preprint arXiv:2601.20894*, 2026.

[53] T. Verma, L. Jin, J. Zhou, J. Huang, M. Tan, B. C. M. Choong, T. F. Tan, F. Gao, X. Xu, D. S. Ting *et al.*, "Privacy-preserving continual learning methods for medical image classification: a comparative analysis," *Frontiers in Medicine*, vol. 10, p. 1227515, 2023.

[54] Z. Zheng, M. Ma, K. Wang, Z. Qin, X. Yue, and Y. You, "Preventing zero-shot transfer degradation in continual learning of vision-language models," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 19 125–19 136.

[55] G. M. Van de Ven, N. Soures, and D. Kudithipudi, "Continual learning and catastrophic forgetting," *arXiv preprint arXiv:2403.05175*, 2024.

[56] K. Xu, X. Zou, G. Hua, and J. Zhou, "Componential prompt-knowledge alignment for domain incremental learning," in *International Conference on Machine Learning*. PMLR, 2025, pp. 70 032–70 046.

[57] Q. Gao, C. Zhao, Y. Sun, T. Xi, G. Zhang, B. Ghanem, and J. Zhang, "A unified continual learning framework with general parameter-efficient tuning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 11 483–11 493.

[58] D.-W. Zhou, H.-L. Sun, H.-J. Ye, and D.-C. Zhan, "Expandable subspace ensemble for pre-trained model-based class-incremental learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 23 554–23 564.

[59] S.-A. Rebuffi, H. Bilen, and A. Vedaldi, "Learning multiple visual domains with residual adapters," *Advances in neural information processing systems*, vol. 30, 2017.

[60] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen *et al.*, "Lora: Low-rank adaptation of large language models." *ICLR*, vol. 1, no. 2, p. 3, 2022.

[61] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for nlp," in *International conference on machine learning*. PMLR, 2019, pp. 2790–2799.

[62] Q. Zhang, M. Chen, A. Bukharin, N. Karampatziakis, P. He, Y. Cheng, W. Chen, and T. Zhao, "Adalora: Adaptive budget allocation for parameter-efficient fine-tuning," *arXiv preprint arXiv:2303.10512*, 2023.

[63] M. Valipour, M. Rezagholizadeh, I. Kobyzev, and A. Ghodsi, "Dylora: Parameter efficient tuning of pre-trained models using dynamic search-free low-rank adaptation," *arXiv preprint arXiv:2210.07558*, 2022.

[64] A. Ed-dib, Z. Datbayev, and A. M. Aboussalah, "Gelora: Geometric adaptive ranks for efficient lora fine-tuning," *arXiv preprint arXiv:2412.09250*, vol. 3, 2024.

[65] H. U. K. Shinwari and M. Usama, "Ard-lora: Dynamic rank allocation for parameter-efficient fine-tuning of foundation models with heterogeneous adaptation needs," *IEEE Transactions on Artificial Intelligence*, 2025.

[66] J. Guo, Z. Zhang, L. Xu, B. Chen, and E. Chen, "Adaptive adapters: An efficient way to incorporate bert into neural machine translation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1740–1751, 2021.

[67] X. Zhou, D. Liang, W. Xu, X. Zhu, Y. Xu, Z. Zou, and X. Bai, "Dynamic adapter meets prompt tuning: Parameter-efficient transfer learning for point cloud analysis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 14 707–14 717.

[68] Z.-H. Qi, D.-W. Zhou, Y. Yao, H.-J. Ye, and D.-C. Zhan, "Adaptive adapter routing for long-tailed class-incremental learning," *Machine Learning*, vol. 114, no. 3, pp. 1–20, 2025.

[69] Y. Wang, Z. Ma, Z. Huang, Y. Wang, Z. Su, and X. Hong, "Isolation and impartial aggregation: A paradigm of incremental learning without interference," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 8, 2023, pp. 10 209–10 217.

[70] W. Chen, Y. Zhou, N. Du, Y. Huang, J. Laudon, Z. Chen, and C. Cui, "Lifelong language pretraining with distribution-specialized experts," in *International Conference on Machine Learning*. PMLR, 2023, pp. 5383–5395.

[71] J. Yu, Y. Zhuge, L. Zhang, P. Hu, D. Wang, H. Lu, and Y. He, "Boosting continual learning of vision-language models via mixture-of-experts adapters," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 23 219–23 230.

[72] J. Yu, Z. Huang, Y. Zhuge, L. Zhang, P. Hu, D. Wang, H. Lu, and Y. He, "Moe-adapters++: Towards more efficient continual learning of vision-language models via dynamic mixture-of-experts adapters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[73] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[74] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[75] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning*. MIT press, 2018.

[76] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1406–1415.

[77] C. Li, Z. Huang, D. P. Paudel, Y. Wang, M. Shahbazi, X. Hong, and L. Van Gool, "A continual deepfake detection benchmark: Dataset, methods, and essentials," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 1339–1349.

[78] V. Lomonaco and D. Maltoni, "Core50: a new dataset and benchmark for continuous object recognition," in *Conference on robot learning*. PMLR, 2017, pp. 17–26.

[79] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[80] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

[81] A. Douillard, A. Ramé, G. Couairon, and M. Cord, "Dytox: Transformers for continual learning with dynamic token expansion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 9285–9295.

[82] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 12, pp. 2935–2947, 2017.

[83] Z. Wang, Z. Zhang, C.-Y. Lee, H. Zhang, R. Sun, X. Ren, G. Su, V. Perot, J. Dy, and T. Pfister, "Learning to prompt for continual learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 139–149.

[84] Z. Wang, Z. Zhang, S. Ebrahimi, R. Sun, H. Zhang, C.-Y. Lee, X. Ren, G. Su, V. Perot, J. Dy *et al.*, "Dualprompt: Complementary prompting for rehearsal-free continual learning," in *European Conference on Computer Vision*. Springer, 2022, pp. 631–648.

[85] J. S. Smith, L. Karlinsky, V. Gutta, P. Cascante-Bonilla, D. Kim, A. Arbelle, R. Panda, R. Feris, and Z. Kira, "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*, 2023, pp. 11 909–11 919.

[86] D.-W. Zhou, Z.-W. Cai, H.-J. Ye, L. Zhang, and D.-C. Zhan, "Dual consolidation for pre-trained model-based domain-incremental learn-

ing," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 20 547–20 557.

[87] Q. Wang, Y. He, S. Dong, X. Song, J. Han, H. Luo, and Y. Gong, "Dualcp: Rehearsal-free domain-incremental learning via dual-level concept prototype," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 20, 2025, pp. 21 198–21 206.

[88] Q. Wang, X. Song, Y. He, J. Han, C. Ding, X. Gao, and Y. Gong, "Boosting domain incremental learning: Selecting the optimal parameters is all you need," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 4839–4849.

[89] N. Paeedeh, M. Pratama, W. Ding, J. Cao, W. Mayer, and R. Kowalczyk, "Continual knowledge consolidation lora for domain incremental learning," *arXiv preprint arXiv:2510.16077*, 2025.

[90] L. Li, D.-W. Zhou, H.-J. Ye, and D.-C. Zhan, "Addressing imbalanced domain-incremental learning through dual-balance collaborative experts," in *International Conference on Machine Learning*. PMLR, 2025, pp. 36 974–36 992.

[91] Y. Dai, X. Hong, Y. Wang, Z. Ma, D. Jiang, and Y. Wang, "Prompt customization for continual learning," *IEEE Transactions on Artificial Intelligence*, 2025.

[92] M.-Y. Park, J.-H. Lee, and G.-M. Park, "Versatile incremental learning: Towards class and domain-agnostic incremental learning," in *European Conference on Computer Vision*. Springer, 2024, pp. 271–288.

[93] L. Wang, J. Xie, X. Zhang, M. Huang, H. Su, and J. Zhu, "Hierarchical decomposition of prompt-based continual learning: Rethinking obscured sub-optimality," *Advances in Neural Information Processing Systems*, vol. 36, pp. 69 054–69 076, 2023.

[94] Y. Feng, Z. Tian, Y. Zhu, Z. Han, H. Luo, G. Zhang, and M. Song, "Cp-prompt: Composition-based cross-modal prompting for domain-incremental continual learning," in *Proceedings of the 32nd ACM International Conference on Multimedia*, 2024, pp. 2729–2738.

[95] L. Pellegrini, G. Graffieti, V. Lomonaco, and D. Maltoni, "Latent replay for real-time continual learning," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 10 203–10 209.

[96] F. Marra, C. Saltori, G. Boato, and L. Verdoliva, "Incremental learning for the detection and classification of gan-generated images," in *2019 IEEE international workshop on information forensics and security (WIFS)*. IEEE, 2019, pp. 1–6.

[97] S. Hou, X. Pan, C. C. Loy, Z. Wang, and D. Lin, "Learning a unified classifier incrementally via rebalancing," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 831–839.

[98] A. Chaudhry, M. Rohrbach, M. Elhoseiny, T. Ajanthan, P. K. Dokania, P. H. Torr, and M. Ranzato, "On tiny episodic memories in continual learning," *arXiv preprint arXiv:1902.10486*, 2019.

[99] A. Prabhu, P. H. Torr, and P. K. Dokania, "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*. Springer, 2020, pp. 524–540.

[100] Y. Wu, Y. Chen, L. Wang, Y. Ye, Z. Liu, Y. Guo, and Y. Fu, "Large scale incremental learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 374–382.

[101] P. Buzzega, M. Boschini, A. Porrello, D. Abati, and S. Calderara, "Dark experience for general continual learning: a strong, simple baseline," *Advances in neural information processing systems*, vol. 33, pp. 15 920–15 930, 2020.

[102] H. Cha, J. Lee, and J. Shin, "Co2l: Contrastive continual learning," in *Proceedings of the IEEE/CVF International conference on computer vision*, 2021, pp. 9516–9525.