Language as an Anchor: Preserving Relative Visual Geometry for Domain Incremental Learning

Shuyi Geng[1,2] Tao Zhou[3] Yi Zhou[1,2*]
[1]School of Computer Science and Engineering, Southeast University, China
[2]Key Laboratory of New Generation Artificial Intelligence Technology and Its Interdisciplinary Applications, Ministry of Education, China
[3]Nanjing University of Science and Technology, China
shuyigeng@seu.edu.cn, yizhou.szcn@gmail.com

Abstract

A key challenge in Domain Incremental Learning (DIL) is to continually learn under shifting distributions while preserving knowledge from previous domains. Existing methods face a fundamental dilemma. On one hand, projecting all domains into a single unified visual space leads to inter-domain interference and semantic distortion, as large shifts may vary with not only visual appearance but also underlying semantics. On the other hand, isolating domain-specific parameters causes knowledge fragmentation, creating "knowledge islands" that hamper knowledge reuse and exacerbate forgetting. To address this issue, we propose ***LAVA*** *(**L**anguage-**A**nchored **V**isual **A**lignment), a novel DIL framework that replaces direct feature alignment with relative alignment driven by a text-based reference anchor. LAVA guides the visual representations of each incoming domain to preserve a consistent relative geometry, which is defined by mirroring the pairwise semantic similarities between the class names. This anchored geometric structure acts as a bridge across domains, enabling the retrieval of class-aware prior knowledge and facilitating robust feature aggregation. Extensive experiments on standard DIL benchmarks demonstrate that LAVA achieves significant performance improvements over state-of-the-arts. Code is available at* https://github.com/ShuyiGeng/LAVA.

1. Introduction

The efficacy of the prevailing pre-training and fine-tuning paradigm faces challenges from real-world distributional shifts, which violate the independent and identically distributed (i.i.d.) assumption and cause severe performance degradation. While traditional domain adaptation (DA) methods [17, 36] can transfer knowledge to a new target domain, it is often at the cost of catastrophically forgetting prior ones. To address this limitation, Domain Incremental Learning (DIL) [28] aims to continuously learn from new domains while preserving knowledge to maintain robust performance across all encountered domains.

Existing DIL approaches can generally be categorized into two major paradigms. The Unified Space Paradigm, shown in Fig. 1(a), attempts to project features from all do-

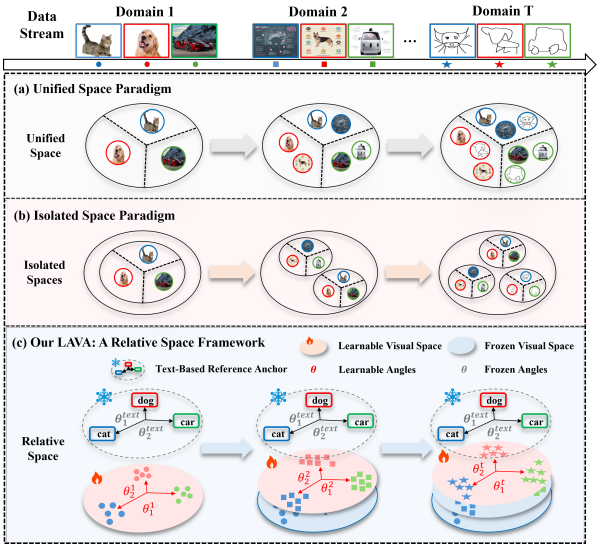


Figure 1. (a) Unified Space Paradigm: Projecting features from all domains into a shared visual space risks interference and semantic distortion. (b) Isolated Space Paradigm: Learning domain-specific subspaces via prompts or adapters leads to fragmented knowledge. **(c) Our LAVA: A Relative Space Framework.** LAVA aligns each domain's relative visual geometry (e.g., the learnable visual angles θ^t) with a stable semantic structure derived from a frozen text-based reference anchor (e.g., the fixed semantic angles θ^{text}), preserving a consistent semantic map across domains.

*Corresponding author.

mains into a single, shared visual embedding space [13, 26]. While promoting knowledge sharing, this strategy struggles with large domain shifts. Visual features may differ considerably—not only in appearance but even in their underlying semantic variations. In such cases, forcing visually inconsistent features to align with each other can cause semantic distortion, inter-class confusion, and indistinct decision boundaries. On the other hand, the Isolated Space Paradigm, depicted in Fig. 1(b), employs parameter-isolation techniques like prompts or adapters [6, 30, 35] to learn domain-specific parameters. Although this effectively prevents inter-domain interference, it leads to knowledge fragmentation, creating isolated knowledge islands that hinder leveraging shared class-aware semantics across domains and exacerbate forgetting.

This raises a critical question: **How can a model bridge these disparate domains to share knowledge without falling into the trap of direct feature unification?** We posit that the key lies in a more stable, invariant reference structure. Language, as a high-density carrier of human prior knowledge, provides a semantic space whose conceptual relations remain consistent across visual domains [2, 25]. Several DA studies [4] have shown that aligning the relative structure between source and target visual spaces through a shared semantic space is more effective than attempting direct feature alignment. The domain-invariant nature of the semantic space inspires our *Language-as-an-Anchor* design, which serves as the foundation for constructing a consistent relative alignment space.

To resolve this trade-off between interference and fragmentation, we propose **LAVA** (**L**anguage-**A**nchored **V**isual **A**lignment), a framework that employs language as a stable anchor to guide the alignment of visual representations based on their relative geometry. LAVA actively aligns each domain's relative visual geometry to the underlying semantic geometry while preserving domain-specific characteristics. This design constructs a consistent relative visual space that not only mitigates inter-domain interference but also facilitates cross-domain communication and knowledge reuse. Specifically, we first introduce the **(a) Vision-Language Relative Structural Alignment (VL-RSA) module**. As conceptualized in Fig. 1(c), LAVA adopts the relative geometry of class semantics as a text-based reference anchor. This anchor is formed by computing the pairwise cosine similarity angles between all classes of embeddings, thus capturing their intrinsic geometric relationships. Similarly, for each new domain, we introduce a learnable visual anchor and compute the relative similarity relationships of each input feature in the same manner. A structural alignment loss then enforces that the resulting relational representation of incoming visual features mirrors the fixed geometry of the text-based anchor. This module forges a domain-invariant relational scaffold, acting as a semantic bridge that maintains consistent geometry and links knowledge across domains. Building on this foundation, to further enhance feature discriminability, we propose the **(b) Class-Aware Cross-Domain Feature Aggregation (CA-CDFA)** module. By leveraging this shared relational structure, its attention mechanism bypasses the comparison of incompatible raw features. Instead, it computes meaningful cross-domain similarities, which directly facilitates robust knowledge reuse and integration to combat forgetting. During inference, to ensure accurate parameter activation and boost final performance, we propose a Multi-Level Feature Integration (MLFI) module for Domain Identification. In summary, our contributions are as follows:

- We explore the fundamental DIL dilemma of inter-domain interference versus knowledge fragmentation by introducing LAVA, a framework that preserves relative visual geometry across domains via a language-anchored semantic bridge, enabling robust knowledge aggregation.
- We design the VL-RSA module to align textual and visual relative structures via a structural alignment loss, and the CA-CDFA module to retrieve prior domain features via similarity-based attention, enabling cross-domain knowledge reuse and class-aware feature aggregation.
- Extensive experiments on four benchmarks demonstrate that LAVA achieves state-of-the-art performance, validating both its effectiveness and generalizability.

2. Related Work

2.1. Domain Incremental Learning

Domain Incremental Learning (DIL) tackles catastrophic forgetting by enabling a model to learn from a sequence of domains with varying data distributions while retaining prior knowledge. Conventional DIL methods can be broadly categorized into three classes: architecture-based [12, 37], regularization-based [9, 16], and replay-based methods [13, 26]. Recently, prompt-based learning [6, 18, 35] has shown promise by preserving domain-specific knowledge via learnable prompts. However, these methods often suffer from several limitations: they are heavily dependent on the initial domain quality [29], highly sensitive to prompt insertion design [6], and poorly adaptable to large domain shifts [35]. In contrast, our method leverages domain-invariant language as an anchor to construct a cross-domain relative alignment space, facilitating flexible class-aware representation aggregation across domains, effectively alleviating the limitations.

2.2. Language Guidance in Domain Adaptation

Vision-language models such as CLIP [25] demonstrate strong zero-shot transferability. However, fine-tuning on specific downstream tasks often degrades their generalization to unseen domains [15, 34]. To mitigate this, recent Do-

main Adaptation (DA) studies has explored leveraging textual information to bridge the domain gap [5, 8, 11, 21, 33]. Among them, [4] proposes an adaptation strategy based on the geometric structure of a language-invariant latent space, encouraging structural consistency across domains by aligning the classifier with the underlying semantic geometry. Inspired by this, we extend the language guidance relative geometry alignment to the more challenging DIL setting, where the model must incrementally adapt to new domains while avoiding catastrophic forgetting.

2.3. Relative Encodings

Recent studies have demonstrated that latent spaces derived from similarly trained networks, while typically misaligned in absolute terms, retain consistent internal geometric relationships [22]. Consequently, representing data points using relative encodings—vectors capturing cosine similarities between each data point and a predefined set of anchors—enables tasks such as zero-shot model stitching [22] and cross-model or cross-modal representation translation [20, 23]. Moreover, predefined invariances can be explicitly integrated into these representations to make them more expressive [1]. Relative encodings have also been successfully employed in downstream tasks, such as action anticipation [3]. Building on this paradigm, LAVA leverages relative encodings within a domain-agnostic, text-based reference anchor to model semantic inter-class relationships and facilitate alignment across diverse domains.

3. Preliminary

3.1. Problem Formulation

Domain Incremental Learning (DIL) focuses on training a unified model sequentially across a series of T domains, denoted by $\mathcal{D} = \{\mathcal{D}_t\}_{t=1}^{T}$, where each domain $\mathcal{D}_t = (\mathcal{X}_t, \mathcal{Z}_t)$ consists of a labeled training set $\mathcal{X}_t$ and a corresponding test set $\mathcal{Z}_t$. During the t-th training stage, the model has access only to the current training data $\mathcal{X}_t$ and is prohibited from accessing any previous training sets $\mathcal{X}_{1\sim t-1} = \bigcup_{\tau=1}^{t-1} \mathcal{X}_\tau$. After completion of training in domain t, the model is evaluated in the union of all test sets seen so far: $\mathcal{Z}_{1\sim t} = \bigcup_{\tau=1}^{t} \mathcal{Z}_\tau$. Each training set $\mathcal{X}_t$ contains N_t labeled examples, i.e., $\mathcal{X}_t = \{(x_{t,i}, y_{t,i})\}_{i=1}^{N_t}$, where $x_{t,i}$ is the i-th input image and $y_{t,i}$ is its associated label. The test set $\mathcal{Z}_t$ follows the same structure. Let $\mathcal{C}_t$ denote the label set of domain t, where $\forall i \in [1, N_t]$, $y_{t,i} \in \mathcal{C}_t$. All domains share a common label space $\mathcal{C}$ with N_c classes, i.e., $\mathcal{C}_1 = \mathcal{C}_2 = \cdots = \mathcal{C}_T = \mathcal{C}$.

3.2. Prompt-based Learning

Following the prompt design in S-Prompts [30], we keep the visual encoder f_θ frozen and maintain a pool of prompts $\mathcal{P} = \{\mathbf{P}_{(t)}\}_{t=1}^{T}$. Each prompt $\mathbf{P}_{(t)} \in \mathbb{R}^{N_p \times D}$ serves as a lightweight adaptor for domain t, where N_p is the prompt length and D matches the hidden dimension of the visual encoder. Given an image x_i from domain t, its corresponding prompt $\mathbf{P}_{(t)}$ is inserted between the patch embeddings $\mathbf{x}_{\text{emb}}$ and the class token $\mathbf{x}_{\text{cls}}$, forming an extended sequence $\mathbf{x}_p = [\mathbf{x}_{\text{cls}}, \mathbf{P}_{(t)}, \mathbf{x}_{\text{emb}}]$, which is then processed by the frozen backbone to yield the visual representation:

$$\mathbf{g}_i = f_\theta(\mathbf{x}_p). \tag{1}$$

This parameter-efficient and modular design allows the model to capture domain-specific knowledge while mitigating catastrophic forgetting, enabling continual adaptation without modifying the shared backbone.

3.3. Text-based Reference Anchor

Following [4], we use a pre-trained text encoder G to generate a set of semantic anchors, $\mathcal{A}^{\text{Text}} \in \mathbb{R}^{N_c \times D_t}$, by encoding the class names of all N_c categories. To capture the fixed geometric structure of the semantic space, we compute a *relative encoding* for each class anchor. Let v be a specific class anchor, corresponding to one of the rows in $\mathcal{A}^{\text{Text}}$. Its relative encoding r^v is computed by measuring its cosine similarity to all anchors in $\mathcal{A}^{\text{Text}}$:

$$r^v = \text{rel}(v, \mathcal{A}^{\text{Text}}) = [\cos(v, \mathcal{A}^{\text{Text}}[1]), \ldots, \cos(v, \mathcal{A}^{\text{Text}}[N_c])], \tag{2}$$

where $\cos(\cdot, \cdot)$ denotes the cosine similarity. The resulting vectors form a static reference structure of inter-class relationships. During training, the model is encouraged to align its relative visual geometry with this reference, thereby preserving consistent class relationships across domains.

4. Method

4.1. Overall Framework

Figure 2 illustrates the overall framework of our proposed LAVA. Prior to training, we use a pre-trained text encoder to embed all class names, thereby establishing a static semantic reference space that captures their inter-class geometric relationships. To effectively capture domain-specific characteristics, we augment the frozen visual encoder with a set of learnable prompts tailored to each domain. Based on these prompt-adapted features, we first employ the Vision-Language Relative Structural Alignment (VL-RSA) module to align the relative structures of the visual and textual feature spaces. Subsequently, the Class-Aware Cross-Domain Feature Aggregation (CA-CDFA) module aggregates relevant features from previously encountered domains to facilitate effective knowledge reuse.

4.2. Vision-Language Relative Alignment

After obtaining the prompt-adapted visual representations (Sec. 3.2), we aim to align their relational structure with a

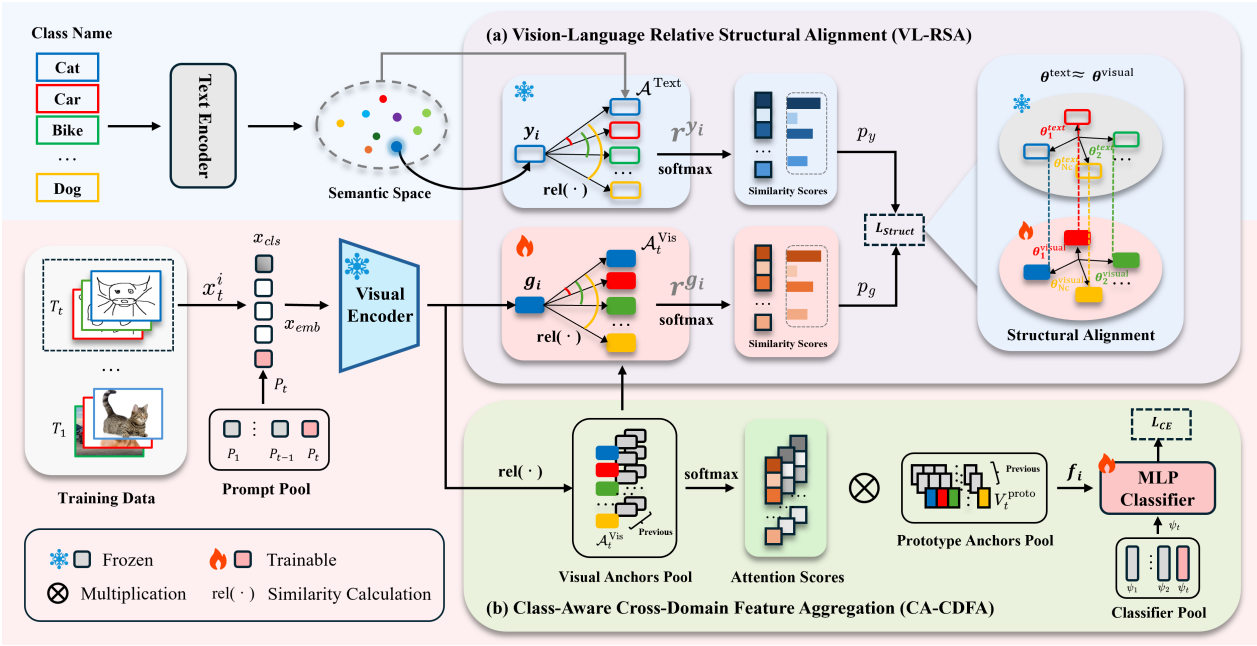


Figure 2. **Overview of the proposed LAVA framework.** LAVA consists of two branches: the text branch establishes a text-based reference anchor from class names and computes relative encodings that capture their pairwise relationships; the visual branch employs domain-specific prompts to extract features from a frozen encoder. These two branches feed into two core modules: (a) The VL-RSA module aligns the visual representations with the text-based geometry via a structural alignment loss, creating a domain-invariant relational space; (b) The CA-CDFA module aggregates class-aware features from previously seen domains using an attention mechanism, enabling effective cross-domain knowledge reuse and aggregation.

fixed semantic geometry anchored by language. To achieve this, we introduce a learnable *Visual Anchor* matrix $\mathcal{A}^{\text{Vis}}_{(t)} \in \mathbb{R}^{N_c \times D_v}$ for each domain t, where N_c is the number of classes and D_v is the output dimension of the visual encoder f_θ. All domain-specific *Visual Anchors* collectively form a *Visual Anchor pool* $\mathcal{A}^{\text{Vis}} = \{\mathcal{A}^{\text{Vis}}_{(1)}, \ldots, \mathcal{A}^{\text{Vis}}_{(T)}\}$, where T is the total number of domains. Similar to the text-based anchor $\mathcal{A}^{\text{Text}} \in \mathbb{R}^{N_c \times D_l}$, each $\mathcal{A}^{\text{Vis}}_{(t)}$ maintains a class-wise structure with one anchor per class. We enforce consistency between their respective *relative encodings*: the input feature's similarity to visual anchors is aligned with its corresponding class name embedding's similarity to the text anchors. Specifically, the textual *relative encoding* is computed from the pre-defined *Text-based Anchor* $\mathcal{A}^{\text{Text}}$ (see Eq. (2)). The semantic reference encoding corresponding to the ground-truth label y_i is defined as:

$$\mathbf{r}^{y_i} = \text{rel}(\mathcal{A}^{\text{Text}}[y_i], \mathcal{A}^{\text{Text}}). \tag{3}$$

Since the pre-trained text encoder G is frozen, $\mathbf{r}^{y_i}$ remains fixed during training. Then, the prompt-adapted feature $\mathbf{g_i}$ is also mapped into a *relative encoding* using cosine similarity with respect to $\mathcal{A}^{\text{Vis}}_{(t)}$:

$$\mathbf{r}^{g_i} = \text{rel}(\mathbf{g_i}, \mathcal{A}^{\text{Vis}}_{(t)}). \tag{4}$$

We encourage the alignment between $\mathbf{r}^{g_i}$ and $\mathbf{r}^{y_i}$ by minimizing the Kullback–Leibler divergence between their softmax-normalized distributions:

$$\mathcal{L}_{\text{Struct}} = \text{KL}(p_g \,\|\, p_y) = \sum_{c=1}^{N_c} p_g^{(c)} \log \frac{p_g^{(c)}}{p_y^{(c)}}, \tag{5}$$

where $p_g = \text{softmax}(\mathbf{r}^{g_i})$ and $p_y = \text{softmax}(\mathbf{r}^{y_i})$.

4.3. Cross-Domain Feature Aggregation

Once the *Visual Anchors* are structurally aligned with the *Text-based Anchor*, the resulting visual geometry space becomes *comparable across domains*. We exploit this relative representation to facilitate effective class-wise aggregation of cross-domain features.

For each domain, we also construct a domain-specific learnable *Prototype Anchor* $V^{\text{proto}}_{(t)} \in \mathbb{R}^{N_c \times D_v}$, where each row represents the latent semantic center of a class in domain t. Collectively, these anchors form a *Prototype Anchor Pool* $\mathcal{V}^{\text{proto}} = \{V^{\text{proto}}_{(1)}, V^{\text{proto}}_{(2)}, \ldots, V^{\text{proto}}_{(T)}\}$, which encapsulates class-level semantic representations across all observed domains. Unlike the structure-aligned *Visual Anchors* $\mathcal{A}^{\text{Vis}}_{(t)}$, which emphasize geometry consistency across domains, these *Prototype Anchors* are designed to capture domain-specific class semantics, thereby enhancing the model's discriminative capability. Each $V^{\text{proto}}_{(t)}$ is trained only on data from its corresponding domain.

We perform a key–value attention mechanism, using the input feature as a query, the *Visual Anchors* as keys, and

the *Prototype Anchors* as values. This design enables attention computation over structure-aligned representations (keys), while retrieving domain-aware class semantics (values). Specifically, given an input feature $\mathbf{g}_i$, we first concatenate all *Visual Anchors* into a global key set: $\mathcal{A}_{(t)}^{\text{global}} = \|_{\tau=1}^{t} \mathcal{A}_{(\tau)}^{\text{Vis}} \in \mathbb{R}^{tN_c \times D_v}$. Then, we compute the relative encoding of $\mathbf{g}_i$ with respect to this global key set:

$$\tilde{\alpha}_i = \text{softmax}\big(\text{rel}(\mathbf{g}_i, \mathcal{A}_{(t)}^{\text{global}})\big) \in \mathbb{R}^{tN_c}, \tag{6}$$

yielding attention scores over class-level anchors across all domains simultaneously. The corresponding global value set is constructed by concatenating all domain-specific *Prototype Anchors*: $V_{(t)}^{\text{global}} = \|_{\tau=1}^{t} V_{(\tau)}^{\text{proto}} \in \mathbb{R}^{tN_c \times D_v}$. Finally, the aggregated feature $\mathbf{f}_i$ is obtained via attention-weighted aggregation followed by a residual connection:

$$\mathbf{f}_i = \tilde{\alpha}_i \, V_{(t)}^{\text{global}} + \mathbf{g}_i. \tag{7}$$

Training Objective. The fused feature $\mathbf{f}_i$ is passed to the current domain classifier $\psi_{(t)}$ to produce the prediction $\hat{y}_i = \psi_{(t)}(\mathbf{f}_i)$. The overall training objective combines cross-entropy loss with structural alignment loss:

$$\mathcal{L} = \mathcal{L}_{\text{CE}}(\hat{y}_i, y_i) + \lambda \mathcal{L}_{\text{Struct}}. \tag{8}$$

where $\mathcal{L}_{\text{CE}}$ denotes the cross-entropy loss, and λ is a hyperparameter that balance the contributions of each term.

During training, all domain-specific prompts, anchors, and classifiers are randomly initialized for each domain and frozen after completion, ensuring stable knowledge accumulation without cross-domain interference.

4.4. Inference via Multi-Level Feature Integration

In the inference phase, the model first identifies the domain of a test image. As shown in Fig. 3, we extract its feature $F(x_{\text{test}})$ and compare it against the pre-computed domain prototypes $\mathcal{M}$. The domain whose prototype has the highest cosine similarity is selected, and the corresponding parameters are activated for final classification.

To facilitate accurate domain identification, we introduce the Multi-Level Feature Integration (MLFI) module. The key intuition is that different layers of a visual encoder capture complementary information (Fig. 6), ranging from low-level textures to high-level semantics. MLFI aggregates this information to form a more discriminative representation, thereby producing more reliable domain prototypes. Specifically, we extract intermediate features $f^{(l)}(x)$ from a fixed set of layers L (detailed in Sec. 5.2), and concatenate them into a unified multi-scale feature:

$$F(x) = \text{Concat}\left(\left[f^{(l)}(x)\right]_{l \in L}\right). \tag{9}$$

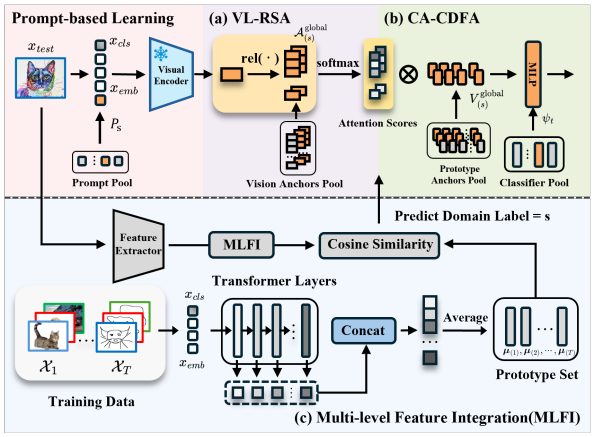


Figure 3. **Overview of the inference pipeline.** The model first identifies the domain of an input image via (c) Multi-Level Feature Integration (MLFI) module (bottom), and then employs the domain-specific modules (top) for the final classification task.

For a given domain t, its prototype $\boldsymbol{\mu}_{(t)}$ is computed by averaging the multi-level features $F(x_i)$ of all its N_t training samples, irrespective of their class labels:

$$\boldsymbol{\mu}_{(t)} = \frac{1}{N_t} \sum_{i=1}^{N_t} F(x_i). \tag{10}$$

Repeating this process for all T domains yields the final prototype set $\mathcal{M} = \{\boldsymbol{\mu}_{(1)}, \ldots, \boldsymbol{\mu}_{(T)}\}$, with all prototypes computed offline after finishing training on each domain.

5. Experiments

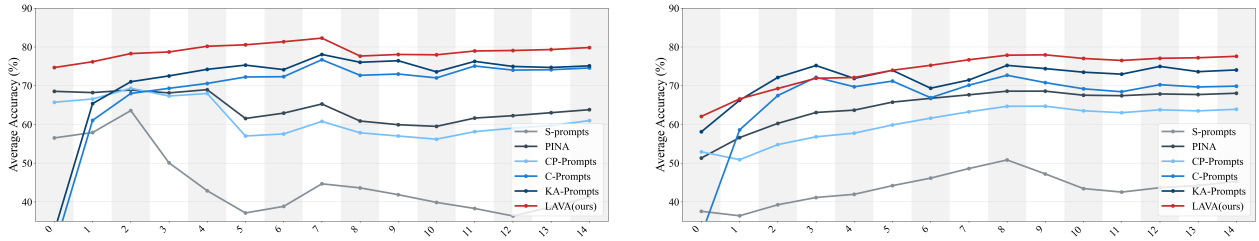
Datasets. Following [35], our experiments are conducted on four DIL benchmarks: DomainNet [24], ImageNet-R [10], ImageNet-C [9] and ImageNet-Mix [35]. Specifically, DomainNet provides 6 domains with 345 classes. ImageNet-R consists of 200 classes across 15 style domains. For ImageNet-C, we use the setup of [35], using 200 classes (identical to ImageNet-R) across 15 corruption types as domains. ImageNet-Mix is a fusion of ImageNet-R and ImageNet-C, comprising 30 domains overall.

Evaluation Metrics. We evaluate our method using three standard DIL metrics: (1) Average Accuracy (A_A), the overall accuracy computed over all test samples from all domains; (2) Average Task Accuracy (A_T), the mean of task accuracies, measuring performance balance; (3) Forgetting Degree (F_T), a measure of catastrophic forgetting, following Core50 [19], where lower values are better. We also report average performance across four DIL benchmarks. See Appendix Sec. 7.1 for metric definitions.

Implementation Details. We employ the AdamW optimizer ($\beta_1 = 0.9$, $\beta_2 = 0.999$) with an initial learning rate of 0.01 and a cosine decay schedule. Additionally, we apply a weight decay of 10^{-4} for regularization to mitigate overfitting. The default training epochs are set to 50. The length

Methods	Publication	DomainNet			ImageNet-R			ImageNet-C			ImageNet-Mix			Average		
		$A_T \uparrow$	$A_A \uparrow$	$F_T \downarrow$	$A_T \uparrow$	$A_A \uparrow$	$F_T \downarrow$	$A_T \uparrow$	$A_A \uparrow$	$F_T \downarrow$	$A_T \uparrow$	$A_A \uparrow$	$F_T \downarrow$	$A_T \uparrow$	$A_A \uparrow$	$F_T \downarrow$
EWC [14]	*NAS 2017*	55.04	51.31	27.52	47.05	45.50	21.30	45.07	45.07	26.69	42.22	43.46	24.75	47.35	46.34	25.07
LwF [16]	*T-PAMI 2017*	59.01	55.02	16.84	54.47	52.89	14.23	42.87	42.87	27.72	50.06	51.54	19.13	51.60	50.58	19.48
L2P [32]	*CVPR 2022*	58.94	55.96	13.35	65.27	65.27	14.93	55.62	55.62	17.35	50.34	44.31	19.72	57.54	55.29	16.34
DualPrompt [31]	*ECCV 2022*	62.68	60.63	13.83	71.10	67.92	7.19	77.25	77.25	2.51	71.23	73.32	7.17	70.57	69.78	7.68
CODA-Prompt [27]	*CVPR 2023*	56.73	52.87	23.88	69.07	66.25	14.12	52.30	52.30	25.15	59.02	52.82	20.22	59.28	56.06	20.85
CPrompt [7]	*CVPR 2024*	60.18	60.62	1.19	74.02	75.27	-0.70	49.71	49.71	**-0.82**	56.86	55.04	-0.83	60.19	60.16	-0.29
S-Prompts [30]	*NeurIPS 2022*	67.51	66.40	0.68	41.23	41.00	2.83	45.26	45.26	0.47	43.84	43.57	0.47	49.46	49.06	1.33
PINA [29]	*ECCV 2024*	73.38	74.18	1.13	63.80	61.41	2.40	68.05	68.05	0.15	65.19	65.38	0.84	67.61	67.26	1.13
CP-Prompt [6]	*MM 2024*	72.16	73.11	0.95	60.98	59.79	2.35	63.91	63.91	0.30	61.46	61.54	1.13	64.63	64.59	1.18
C-Prompt* [18]	*IJCV 2024*	65.03	63.10	1.63	74.57	70.35	**-9.56**	69.89	69.89	2.49	67.90	66.43	**-1.29**	69.35	67.44	**-1.68**
KA-Prompt* [35]	*ICML 2025*	67.06	65.24	2.32	75.12	71.85	-5.83	74.12	74.12	3.97	71.27	71.65	0.17	71.89	70.72	0.16
LAVA-share (Ours)	*This Paper*	73.40	74.64	0.68	80.27	75.10	-0.18	77.00	77.00	0.11	77.73	75.72	0.24	77.00	75.62	0.21
LAVA (Ours)	*This Paper*	**73.84**	**75.13**	**0.58**	**80.89**	**75.58**	-0.34	**77.58**	**77.58**	0.04	**78.45**	**76.56**	0.26	**77.69**	**76.21**	0.14

Table 1. Comparison across four DIL benchmarks. We mark the best results in **bold** and second-best with underlines. (*) C-Prompt and KA-Prompt employ specific mechanisms that can result in negative forgetting (i.e., knowledge gain). The proposed *LAVA-share* denotes a lightweight variant of LAVA that reuses the anchor pool as both keys and values.



(a) ImageNet-R average accuracy on seen domains

(b) ImageNet-C average accuracy on seen domains

Figure 4. Performance progression on the ImageNet-R and ImageNet-C benchmarks.

of prompt token N_p is set to 16. The mini-batch size is set to 128. The training images are resized to 224×224. A temperature parameter $\tau = 0.07$ is applied in the softmax normalization to scale the cosine-similarity scores when constructing relational distributions. All experiments are conducted on a single NVIDIA RTX 4090 GPU.

Baselines. We benchmark LAVA against state-of-the-art methods from both Domain- and Class-Incremental Learning (DIL and CIL). DIL baselines include prompt-based methods (S-Prompts [30], CP-Prompt [6], C-Prompt [18], KA-Prompt [35]) and the adapter-based method PINA [29]. Among these, three methods rely on explicit domain identification at inference to select their domain-specific parameters: S-Prompts and CP-Prompt use a KNN-based strategy, whereas PINA adopts its dedicated Patch Shuffle Selector (PSS) module. We follow the configurations described in their respective papers (see Appendix Sec. 7.2). Ablations on this module are further discussed in Sec. 5.2. For CIL, we consider regularization-based methods (LwF [16], EWC [14]) and prompt-based ones (L2P [32], DualPrompt [31], CODA-Prompt [27], CPrompt [7]). All methods are implemented on a **CLIP (ViT-B/16)** backbone using official or verified re-implementations, and evaluated under the **rehearsal-free** setting without storing past samples.

5.1. Main Results

Comparison with State-of-the-Arts. Our LAVA model demonstrates a significant leap over existing state-of-the-art methods, as detailed in Tab. 1. It achieves a new benchmark with an average Task Accuracy (A_T) of **77.69%** and Average Accuracy (A_A) of **76.21%**, outperforming the previous SOTA, KA-Prompt, by **5.80%** and **5.49%** respectively. LAVA's superiority is most pronounced on benchmarks with severe domain shifts, such as DomainNet (**+6.78%**) and ImageNet-Mix (**+7.18%**), directly validating the robustness of our relative alignment strategy. Beyond conventional accuracy metrics, LAVA demonstrates strong knowledge retention, achieving a relatively low forgetting degree (F_T) of **0.14**. While specialized methods like KA-Prompt and C-Prompt employ explicit reuse mechanisms to obtain numerically lower forgetting, LAVA remains one of the more effective approaches among standard baselines in mitigating knowledge degradation. We further introduce a lightweight variant, **LAVA-share**, which eliminates the separate *Prototype Anchor Pool* and instead reuses the *Visual Anchor Pool* as both keys and values in the CA-CDFA module. In this way, a single anchor pool simultaneously serves as the key–value pair, achieving a **32%** parameter reduction without sacrificing performance (see Sec. 5.3). In addi-

Baseline	VL-RSA	CA-CDFA	MLFI	DomainNet			ImageNet-R			ImageNet-C			ImageNet-Mix		
				$A_T \uparrow$	$A_A \uparrow$	$F_T \downarrow$	$A_T \uparrow$	$A_A \uparrow$	$F_T \downarrow$	$A_T \uparrow$	$A_A \uparrow$	$F_T \downarrow$	$A_T \uparrow$	$A_A \uparrow$	$F_T \downarrow$
✓				67.51	66.40	0.68	41.23	41.00	2.83	45.26	45.26	0.47	43.84	43.57	0.47
✓	✓			73.36	74.38	1.13	78.50	73.84	0.81	75.77	75.77	0.17	75.72	73.15	0.66
✓	✓	✓		73.65	74.86	1.01	79.92	74.90	0.08	77.17	77.17	0.24	77.74	75.71	0.45
✓	✓	✓	✓	**73.84**	**75.13**	**0.58**	**80.89**	**75.58**	**-0.34**	**77.58**	**77.58**	**0.04**	**78.45**	**76.56**	**0.26**

Table 2. Component contributions. The baseline corresponds to S-Prompts with prompt tuning on a frozen backbone, a unified classifier and a KNN-based domain identification strategy. Even without MLFI, LAVA achieves state-of-the-art performance under this setting.

		DomainNet			ImageNet-R		
		$A_T\uparrow$	$A_A\uparrow$	$F_T\downarrow$	$A_T\uparrow$	$A_A\uparrow$	$F_T\downarrow$
	random	73.04	73.95	1.59	77.06	72.47	1.11
ST	IDs	72.98	73.94	1.34	80.22	75.34	-0.07
	class names	73.44	74.45	1.32	80.37	75.54	0.30
CLIP	IDs	73.19	74.14	1.23	79.32	74.51	0.27
	class names	**73.84**	**75.13**	**0.58**	**80.89**	**75.58**	**-0.34**

Table 3. Ablation on alignment reference structures. "ST" and "CLIP" denote text encoders (Sentence Transformer and CLIP); "IDs" and "class names" indicate the semantic input type.

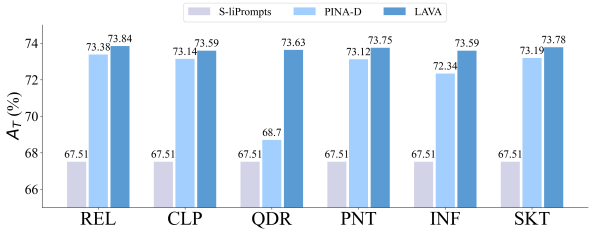


Figure 5. Impact of Domain Order on DomainNet. LAVA shows robust stability and performance across domain orders, including the challenging *Quickdraw*-first order.

tion, the learning process visualizations on ImageNet-R and ImageNet-C (Fig. 4) further confirm LAVA's consistent superiority, particularly in the more challenging domains encountered in the later stages of training.

5.2. Ablation Study

Component Ablations. Table 2 summarizes our component ablation study. We use S-Prompts [30] combined with a standard KNN-based domain identification strategy as our baseline. On top of this baseline, we incrementally add three core components of LAVA: (1) Vision-Language Relative Structural Alignment (VL-RSA), which yields a substantial performance gain by enforcing relative structural alignment between vision and language, validating our relative alignment strategy; (2) Class-Aware Cross-Domain Feature Aggregation (CA-CDFA), which significantly improves cross-domain knowledge reuse and enhances feature discriminability, effectively overcoming knowledge fragmentation; (3) Multi-Level Feature Integration (MLFI), which provides a modest yet consistent improvement by facilitating more accurate domain identification at inference. Notably, even without this module, our method can still achieve state-of-the-art performance using the standard KNN-based domain identification approach.

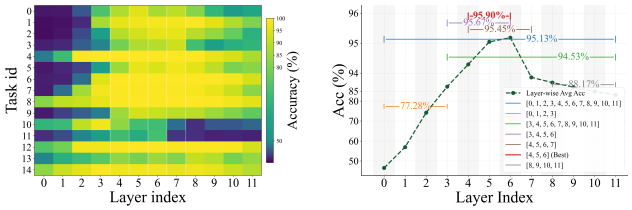
Method	DomainNet			ImageNet-Mix		
	$A_T\uparrow$	$A_A\uparrow$	$F_T\downarrow$	$A_T\uparrow$	$A_A\uparrow$	$F_T\downarrow$
w/o L_{Struct}	73.55	74.56	1.03	77.10	75.39	0.61
L1	73.12	74.12	0.62	76.80	74.71	0.37
L2	73.45	74.54	0.68	77.52	75.35	0.32
KL	**73.84**	**75.13**	**0.58**	**78.45**	**76.56**	**0.26**

Table 4. Ablation of structure alignment loss formulations, including ℓ_1, ℓ_2, KL divergence, and a variant without L_{Struct}.

Impact of Domain Order. We evaluate LAVA's robustness to domain order on DomainNet using various permutations (see Appendix Sec. 7.3). As shown in Fig. 5, LAVA maintains top performance with minimal fluctuation, while PINA drops significantly on the *Quickdraw*-first sequence [29], where LAVA surpasses it by **4.93%** in A_T.

Ablation on Alignment Reference and Objectives. We jointly analyze the effects of different reference structures and alignment objectives. As shown in Tab. 3, we compare random initialization, numerical class IDs, and class names encoded by a Sentence Transformer (ST) [4] or CLIP's text encoder. Using CLIP-encoded class names yields the best results, confirming that its pre-trained semantic geometry is naturally aligned with the visual space. We further evaluate several formulations of the structure alignment loss L_{Struct} (Tab. 4), including ℓ_1, ℓ_2, KL divergence, and a variant without alignment. The KL objective achieves the best accuracy, indicating that penalizing distributional discrepancies enables more stable and discriminative alignment.

Ablation on Layer Selection in MLFI. We conduct a systematic ablation to validate the design of the Multi-Level Feature Integration (MLFI) module. Figure 6(a) presents the per-task accuracy using features from individual layers on ImageNet-C, while (b) shows the average accuracy of each layer and different layer-integration combinations across tasks. The results indicate that different layers encode complementary visual cues—some performing well on certain tasks but less effectively on others. By integrating multiple layers, MLFI exploits these complementarities to achieve more consistent and robust domain identification.

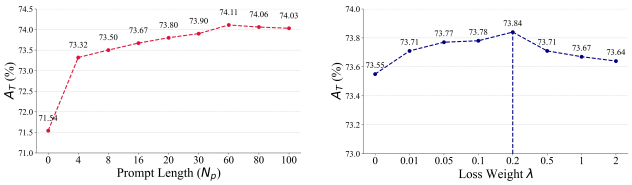


(a) (b)

Figure 6. Ablation on layer selection in the MLFI module on ImageNet-C. (a) Per-layer accuracy across all tasks, illustrating the contribution of each layer. (b) Mean per-layer accuracy (green dashed) and results of different layer-integration combinations.

Method	DomainNet		ImageNet-R		ImageNet-C		ImageNet-Mix	
	$A_{\text{cls}}\uparrow$	$A_A\uparrow$	$A_{\text{cls}}\uparrow$	$A_A\uparrow$	$A_{\text{cls}}\uparrow$	$A_A\uparrow$	$A_{\text{cls}}\uparrow$	$A_A\uparrow$
K-NN	84.03	74.66	56.12	74.90	82.72	77.17	77.52	75.71
NMC	85.02	74.76	57.46	75.47	83.25	77.12	77.74	75.70
PSS	85.07	74.78	53.43	73.76	74.02	75.03	64.11	72.29
MLFI	**86.45**	**75.13**	**57.88**	**75.58**	**95.90**	**77.58**	**88.00**	**76.56**

Table 5. Comparison of domain identification strategies. A_{cls} is the domain identification accuracy, while A_A is the final task Average Accuracy achieved using that method.



(a) (b)

Figure 7. Hyperparameter sensitivity study on the DomainNet dataset. (a) Prompt length N_p. (b) Loss weight λ.

Notably, combining layers $L^* = \{4, 5, 6\}$ yields the highest accuracy on ImageNet-C, surpassing both single-layer and other integration settings. Further details on the layer-selection search space and the associated memory overhead are provided in Appendix Sec. 7.4. Moreover, Table 5 compares MLFI with representative approaches [29], including K-Nearest Neighbors (K-NN), Nearest Mean Classifier (NMC) and Patch Shuffle Selector (PSS) (see Appendix Sec. 7.2). MLFI's superior domain-identification accuracy consistently translates into higher overall performance.

5.3. Further Analysis

Hyperparameter Sensitivity Analysis. As shown in Fig. 7, we analyze LAVA's sensitivity to two key hyperparameters on DomainNet: the prompt length N_p and the alignment loss weight λ. The model remains largely robust to the choice of N_p, with performance improving only marginally as the prompt length increases. For consistency with prior methods, we set the prompt length to N_p=16. For λ, the optimal value differs slightly across datasets but lies within a similar range; for DomainNet we use λ=0.2. Full settings are listed in Appendix Sec. 7.5.

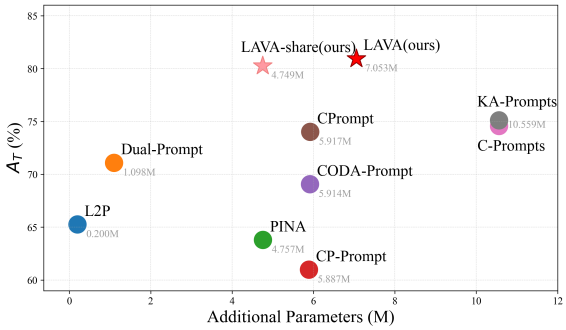


Figure 8. Performance–Parameter comparison on ImageNet-R.

T	Cumulative (M, +%)	Active@Inf. (M, +%)	GPU (GB)
1	0.47 (+0.31%)	**150.092 (+0.31%)**	**2.49**
5	2.35 (+1.57%)		
10	4.70 (+3.14%)		
15	7.05 (+4.71%)		

Table 6. Scalability across domains. Percentages are measured w.r.t. the backbone (149.62M).

Parameter–Performance Trade-off. Figure 8 compares accuracy versus trainable parameters on ImageNet-R (detailed in Appendix Sec. 7.6). LAVA achieves state-of-the-art accuracy with high parameter efficiency, requiring only 7.05M trainable parameters—**33% fewer** than KA-Prompt (the previous SOTA)—while attaining higher task accuracy. The lightweight variant, *LAVA-share*, further reduces the parameter count to 4.75M yet continues to outperform all competing approaches by a substantial margin. These results indicate that LAVA's advantage arises from its architectural efficiency rather than increased model capacity.

Model Scalability. As shown in Tab. 6, the trainable parameters grow linearly with the number of domains, reaching 7.05M at T=15 only—4.7% of the 149.6M backbone. At inference, however, LAVA activates only a fixed set of parameters (150.09M in total), introducing merely ∼**0.3%** additional parameters beyond the backbone and keeping the GPU memory footprint constant at 2.49 GB. This indicates that domain expansion increases only the stored model size, while the inference-time cost remains lightweight and unchanged. A detailed analysis of LAVA's computational overhead and efficiency is provided in Appendix Sec. 7.6.

6. Conclusion and Future Works

In this work, we explore the fundamental dilemma in DIL—the trade-off between inter-domain interference and knowledge fragmentation. We propose LAVA, a novel framework that preserves relative visual geometry across domains via a language-anchored semantic bridge, enabling effective retrieval of prior knowledge and robust feature aggregation. Extensive experiments on multiple DIL benchmarks demonstrate that LAVA consistently outperforms state-of-the-art methods. Beyond empirical success, LAVA

introduces the idea of leveraging an external, stable, and knowledge-rich structure—such as language—as a semantic compass to navigate the drift of internal representations, offering a promising path toward more stable and scalable domain incremental systems. We believe that aligning relative geometry opens promising directions for broader applications, such as few-shot and multi-modal learning.

Acknowledgements. This work is supported in part by the National Natural Science Foundation of China under Grant 62476054, and in part by the Fundamental Research Funds for the Central Universities of China. This research work is supported by the Big Data Computing Center of Southeast University.

References

[1] Irene Cannistraci, Luca Moschella, Marco Fumero, Valentino Maiorca, and Emanuele Rodolà. From bricks to bridges: Product of invariances to enhance latent space communication. *arXiv preprint arXiv:2310.01211*, 2023. 3

[2] Benjamin Devillers, Bhavin Choksi, Romain Bielawski, and Rufin VanRullen. Does language help generalization in vision models? *arXiv preprint arXiv:2104.08313*, 2021. 2

[3] Anxhelo Diko, Danilo Avola, Bardh Prenkaj, Federico Fontana, and Luigi Cinque. Semantically guided representation learning for action anticipation. In *European Conference on Computer Vision*, pages 448–466. Springer, 2024. 3

[4] Anxhelo Diko, Antonino Furnari, Luigi Cinque, and Giovanni Maria Farinella. Laguna: Language guided unsupervised adaptation with structured spaces. *arXiv preprint arXiv:2411.15557*, 2024. 2, 3, 7

[5] Lisa Dunlap, Clara Mohri, Devin Guillory, Han Zhang, Trevor Darrell, Joseph E Gonzalez, Aditi Raghunathan, and Anna Rohrbach. Using language to extend to unseen domains. International Conference on Learning Representations (ICLR), 2023. 3

[6] Yu Feng, Zhen Tian, Yifan Zhu, Zongfu Han, Haoran Luo, Guangwei Zhang, and Meina Song. Cp-prompt: Composition-based cross-modal prompting for domain-incremental continual learning. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 2729–2738, 2024. 2, 6, 1, 3

[7] Zhanxin Gao, Jun Cen, and Xiaobin Chang. Consistent prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 28463–28473, 2024. 6, 3

[8] Tejas Gokhale, Rushil Anirudh, Bhavya Kailkhura, Jayaraman J Thiagarajan, Chitta Baral, and Yezhou Yang. Attribute-guided adversarial training for robustness to natural perturbations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7574–7582, 2021. 3

[9] Dan Hendrycks and Thomas G Dietterich. Benchmarking neural network robustness to common corruptions and surface variations. *arXiv preprint arXiv:1807.01697*, 2018. 2, 5

[10] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8349, 2021. 5

[11] Zeyi Huang, Andy Zhou, Zijian Ling, Mu Cai, Haohan Wang, and Yong Jae Lee. A sentence speaks a thousand images: Domain generalization through distilling clip with language guidance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11685–11695, 2023. 3

[12] Ching-Yi Hung, Cheng-Hao Tu, Cheng-En Wu, Chien-Hung Chen, Yi-Ming Chan, and Chu-Song Chen. Compacting, picking and growing for unforgetting continual learning. *Advances in neural information processing systems*, 32, 2019. 2

[13] Kishaan Jeeveswaran, Elahe Arani, and Bahram Zonooz. Gradual divergence for seamless adaptation: A novel domain incremental learning method. *arXiv preprint arXiv:2406.16231*, 2024. 2

[14] 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*, 114(13):3521–3526, 2017. 6

[15] Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. *arXiv preprint arXiv:2202.10054*, 2022. 2

[16] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017. 2, 6

[17] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International conference on machine learning*, pages 6028–6039. PMLR, 2020. 1

[18] Zichen Liu, Yuxin Peng, and Jiahuan Zhou. Compositional prompting for anti-forgetting in domain incremental learning. *International Journal of Computer Vision*, 132(12): 5783–5800, 2024. 2, 6, 3

[19] Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. In *Conference on robot learning*, pages 17–26. PMLR, 2017. 5

[20] Valentino Maiorca, Luca Moschella, Antonio Norelli, Marco Fumero, Francesco Locatello, and Emanuele Rodolà. Latent space translation via semantic alignment. *Advances in Neural Information Processing Systems*, 36:55394–55414, 2023. 3

[21] Seonwoo Min, Nokyung Park, Siwon Kim, Seunghyun Park, and Jinkyu Kim. Grounding visual representations with texts for domain generalization. In *European conference on computer vision*, pages 37–53. Springer, 2022. 3

[22] Luca Moschella, Valentino Maiorca, Marco Fumero, Antonio Norelli, Francesco Locatello, and Emanuele Rodolà. Relative representations enable zero-shot latent space communication. *arXiv preprint arXiv:2209.15430*, 2022. 3

[23] Antonio Norelli, Marco Fumero, Valentino Maiorca, Luca Moschella, Emanuele Rodola, and Francesco Locatello. Asif: Coupled data turns unimodal models to multimodal without training. *Advances in Neural Information Processing Systems*, 36:15303–15319, 2023. 3

[24] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1406–1415, 2019. 5

[25] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021. 2

[26] Haizhou Shi and Hao Wang. A unified approach to domain incremental learning with memory: Theory and algorithm. *Advances in Neural Information Processing Systems*, 36:15027–15059, 2023. 2

[27] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt 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*, pages 11909–11919, 2023. 6, 3

[28] Gido M Van de Ven, Tinne Tuytelaars, and Andreas S Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 4(12):1185–1197, 2022. 1

[29] Qiang Wang, Yuhang He, Songlin Dong, Xinyuan Gao, Shaokun Wang, and Yihong Gong. Non-exemplar domain incremental learning via cross-domain concept integration. In *European Conference on Computer Vision*, pages 144–162. Springer, 2024. 2, 6, 7, 8, 1, 3

[30] Yabin Wang, Zhiwu Huang, and Xiaopeng Hong. S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning. *Advances in Neural Information Processing Systems*, 35:5682–5695, 2022. 2, 3, 6, 7, 1

[31] 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 *European Conference on Computer Vision*, pages 631–648. Springer, 2022. 6, 3

[32] 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*, pages 139–149, 2022. 6, 3

[33] Zhenbin Wang, Lei Zhang, Lituan Wang, and Minjuan Zhu. Landa: Language-guided multi-source domain adaptation. *arXiv preprint arXiv:2401.14148*, 2024. 3

[34] Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, et al. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7959–7971, 2022. 2

[35] Kunlun Xu, Xu Zou, Gang Hua, and Jiahuan Zhou. Componential prompt-knowledge alignment for domain incremental learning. *arXiv preprint arXiv:2505.04575*, 2025. 2, 5, 6, 3

[36] Shiqi Yang, Joost Van de Weijer, Luis Herranz, Shangling Jui, et al. Exploiting the intrinsic neighborhood structure for source-free domain adaptation. *Advances in neural information processing systems*, 34:29393–29405, 2021. 1

[37] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*, 2017. 2

Language as an Anchor: Preserving Relative Visual Geometry for Domain Incremental Learning

Supplementary Material

7. Supplementary

7.1. Details of the Evaluation Metrics

To provide a comprehensive assessment, our evaluation framework is built upon three core metrics, each chosen to target a distinct pillar of continual learning performance: overall effectiveness, balanced performance across domains, and resistance to catastrophic forgetting.

Average Accuracy (A_A) This metric measures the overall classification performance across all test samples from all domains after the model has been fully trained on all T domains. It is defined as:

$$A_A = \frac{\sum_{i=1}^{T} c_{T,i}}{\sum_{i=1}^{T} |\mathcal{Z}_i|}$$

where $c_{T,i}$ denotes the number of correctly classified samples in the test set $\mathcal{Z}_i$ by the final model M_T, and $|\mathcal{Z}_i|$ is the total number of samples in that test set.

Average Task Accuracy (A_T) This metric emphasizes balanced performance across domains, computing the mean of per-domain accuracies. It is calculated as:

$$A_T = \frac{1}{T} \sum_{i=1}^{T} a_{T,i}$$

where $a_{T,i}$ is the accuracy of the final model M_T on the test set of the i-th domain, $\mathcal{Z}_i$.

Forgetting Degree (F_T) This metric quantifies catastrophic forgetting by measuring how performance on previously learned domains changes as new domains are introduced. It is calculated by averaging the per-domain backward transfer (BWT_i) across all previously learned domains:

$$BWT_i = \frac{1}{T-i} \sum_{j=i+1}^{T} (a_{j,i} - a_{i,i})$$

The overall Forgetting Degree is then defined as:

$$F_T = \frac{1}{T-1} \sum_{i=1}^{T-1} BWT_i$$

Here, $a_{j,i}$ denotes the accuracy on domain i after training up to domain j, and $a_{i,i}$ is the accuracy on domain i immediately after it was learned.

Since model performance on previous domains typically decreases as new domains are learned (i.e., $a_{j,i} < a_{i,i}$), the raw forgetting degree defined above is generally negative. For clearer interpretation, our tables report its negation while still using the symbol F_T. Under this convention, larger positive values indicate stronger forgetting, whereas smaller values reflect better knowledge retention. Notably, when F_T becomes negative, it implies positive backward transfer, meaning that learning later domains unexpectedly improves performance on earlier ones.

7.2. Details of Domain Identification Strategies

Several baselines in our comparison rely on explicit domain identification during inference. For example, S-Prompts [30] and CP-Prompts [6] employ a K-NN-based strategy, whereas PINA [29] adopts its dedicated Patch Shuffle Selector (PSS). For completeness, we summarize these strategies below.

K-Nearest Neighbors (K-NN). These methods use the frozen visual encoder to extract features for each training domain and apply K-Means clustering to obtain a small set of domain prototypes. During inference, a test sample is encoded with the same frozen encoder and matched to the nearest prototype (typically via L2 distance); the identified domain determines which domain-specific parameters are used for inference.

Patch Shuffle Selector (PSS). PSS [29] constructs domain prototypes by applying random patch shuffling to each training image before feature extraction, suppressing class-dependent structure while retaining domain-level appearance statistics. The prototype for each domain is computed as the mean feature of its shuffled samples. At inference, the test sample is encoded without shuffling and assigned to the nearest prototype using cosine similarity; the identified domain determines the domain-specific parameters used for inference.

Nearest Mean Classifier (NMC). Consistent with PINA [29], we also include NMC as a comparison method. NMC computes a single mean feature for each domain directly from the frozen-encoder embeddings of all training samples, without clustering or patch shuffling. During inference, the test sample is assigned to the domain whose mean prototype yields the highest cosine similarity.

As shown in Tab. 5, we compare these three strategies—K-NN, PSS, and NMC—under a unified evaluation protocol. For all baselines, we follow the configurations reported in their respective papers.

Order	1st	2nd	3rd	4th	5th	6th
1	real	quickdraw	painting	sketch	infograph	clipart
2	clipart	infograph	painting	quickdraw	real	sketch
3	quickdraw	infograph	clipart	painting	sketch	real
4	painting	sketch	clipart	real	infograph	quickdraw
5	infograph	clipart	painting	quickdraw	real	sketch
6	sketch	clipart	painting	quickdraw	real	infograph

Table 7. The six domain order permutations for the DomainNet dataset used in our experiments.

Dataset	Selected Layers
DomainNet	$\{9, 10, 11\}$
ImageNet-R	$\{10, 11\}$
ImageNet-C	$\{4, 5, 6\}$
ImageNet-Mix	$\{6, 11\}$

Table 8. Selected MLFI layers for each benchmark.

Dataset	MLFI Storage	K-NN Storage
DomainNet	0.05 MB	0.09 MB
ImageNet-R	0.10 MB	0.22 MB
ImageNet-C	0.13 MB	0.22 MB
ImageNet-Mix	0.18 MB	0.44 MB

Table 9. Memory cost of MLFI and K-NN across different benchmarks.

Dataset	Optimal λ
DomainNet	0.2
ImageNet-R	1.0
ImageNet-C	0.5
ImageNet-Mix	1.0

Table 10. Optimal alignment-loss weight λ across benchmarks.

7.3. Domain Order Permutations on DomainNet

To rigorously evaluate the robustness of our model, LAVA, against variations in domain order, we constructed six distinct domain permutations for the DomainNet dataset.

The first sequence follows the methodology of [35], where domains are sorted by decreasing data size to simulate a challenging Domain Incremental Learning (DIL) scenario. The second sequence adopts the default DomainNet order, which is consistent with the experimental setups in several baseline methods [6, 29, 30].

The design of the remaining four permutations is primarily motivated by the evaluation strategy in PINA [29]. Acknowledging the significant domain gaps among the six domains in DomainNet (*Real*, *Quickdraw*, *Painting*, *Sketch*, *Infograph*, and *Clipart*), the initial domain can substantially impact subsequent model adaptation. Therefore, mirroring PINA's setup, we ensure that each of the six domains serves as the starting point in one of the permutations.

However, to create a more demanding continual learning testbed, we introduce an additional layer of complexity. While PINA's setting focuses on varying the initial domain, we further randomize the sequence of the subsequent domains. Specifically, after designating a unique starting domain for each permutation, we also randomly shuffle the order of the remaining five domains. This approach not only varies the initial domain exposure but also alters the entire sequence of domain transitions, enabling a more comprehensive and challenging assessment of model robustness.

The six resulting domain order permutations are detailed in Tab. 7, and their impact on model performance is illustrated in Fig. 5.

7.4. Best Layer Searching and Memory Analysis

To identify the most informative layers for MLFI, we first evaluate the domain-prediction accuracy of each Transformer layer individually. Based on these scores, we perform an accuracy-guided greedy search: starting from the top-performing layers, we enumerate compact combinations and retain only those that further improve validation accuracy. This lightweight procedure effectively finds layer subsets that provide strong complementary cues while avoiding exhaustive search. The final selected layers are reported in Tab. 8.

Comparison with K-NN Storage. For reference, we also estimate the storage cost of the K-NN domain selector used in prior work. In contrast to MLFI—which stores only a few averaged multi-layer features—K-NN must maintain several high-dimensional centroids in the full embedding space, leading to noticeably higher memory usage. The comparison is summarized in Tab. 9.

7.5. Alignment Loss Hyperparameters

Table 10 presents the optimal alignment-loss weight λ for each benchmark. The preferred values vary slightly across datasets, suggesting that a moderate level of alignment tends to work well in practice.

Method	Params (Trainable/Total, M)	GPU Mem (GB)	Batch Time (s)	A_T (Δ)
L2P [32]	0.200 / 149.82	16.56	0.21	65.27 (+15.62)
Dual-Prompt [31]	1.098 / 150.72	18.39	0.61	71.10 (+9.79)
CODA-Prompt [27]	5.914 / 155.53	18.16	0.60	69.07 (+11.82)
CPrompt [7]	5.917 / 155.54	40.27	1.34	74.02 (+6.87)
S-Prompts [30]	0.307 / 149.93	14.10	0.38	41.23 (+39.66)
PINA [29]	4.757 / 154.38	13.99	0.29	63.80 (+17.09)
CP-Prompt [6]	5.887 / 155.51	12.60	0.26	60.98 (+19.91)
C-Prompts [18]	10.559 / 160.18	9.79	0.32	74.57 (+6.32)
KA-Prompts [35]	10.559 / 160.18	19.54	0.60	75.12 (+5.77)
LAVA-share (ours)	4.749 / 154.37	16.45	0.46	80.27 (+0.62)
LAVA (ours)	7.053 / 156.68	15.33	0.48	80.89 (Best)

Table 11. Comparison of the number of parameters, overhead, and performance with the state-of-the-art on ImageNet-R. "Params" reports trainable/total parameters (in M) on top of a frozen backbone with 149.62M parameters; "GPU Mem" is the peak GPU memory in GB (MB/1024); "Batch Time" is the per-batch running time in seconds; A_T is the final average task accuracy, and Δ denotes the accuracy gap w.r.t. LAVA, computed as $\Delta = A_T(\text{LAVA}) - A_T(\text{method})$.

Method	Active Params@Inf. (Extra/Total, M, +%)	GPU Mem (GB)	Latency (s)
L2P [32]	0.200 / 149.82 (+0.13%)	2.00	0.04
Dual-Prompt [31]	1.098 / 150.72 (+0.73%)	4.04	0.36
CODA-Prompt [27]	5.914 / 155.53 (+3.95%)	4.07	0.32
CPrompt [7]	2.688 / 152.31 (+1.80%)	2.23	0.10
S-Prompts [30]	0.020 / 149.64 (+0.01%)	3.68	0.90
PINA [29]	0.317 / 149.94 (+0.21%)	2.95	1.23
CP-Prompt [6]	0.385 / 150.01 (+0.26%)	2.52	0.45
C-Prompts [18]	10.559 / 160.18 (+7.06%)	3.71	0.32
KA-Prompts [35]	10.559 / 160.18 (+7.06%)	5.25	0.32
LAVA-share (ours)	0.317 / 149.94 (+0.21%)	2.44	0.37
LAVA (ours)	0.470 / 150.09 (+0.31%)	2.49	0.38

Table 12. Inference efficiency comparison with the state-of-the-art on ImageNet-R. "Active Params@Inf." reports, for each method, the extra/total active parameters (in M) on top of a frozen backbone with 149.62M parameters, and the percentage denotes the relative increase over the backbone; "GPU Mem" is the peak GPU memory usage in GB; "Latency" is the per-batch inference time in seconds.

7.6. Computational Overhead and Efficiency

To better understand the computational profile of LAVA, we provide a detailed comparison of training and inference overhead on ImageNet-R in Tab. 11 and Tab. 12. These results complement the main paper by quantifying how LAVA trades parameter cost and runtime for accuracy.

Training-time parameter and memory cost. Table 11 reports the number of trainable parameters, peak GPU memory, and per-batch training time for each method, together with the final average task accuracy A_T. On ImageNet-R, LAVA achieves the best performance with $A_T = 80.89\%$, while introducing only 7.05M trainable parameters on top of a frozen 149.62M backbone. This represents a substantial reduction in trainable parameters compared to parameter-heavy methods such as C-Prompts and KA-Prompts, both of which require 10.56M trainable parameters but still lag behind LAVA by **+6.32%** and **+5.77%** in A_T, respectively. Even the more compact *LAVA-share* variant, with 4.75M trainable parameters, already outperforms all baselines, reaching 80.27% and remaining only **0.62%** behind the full LAVA model. Overall, LAVA and *LAVA-share* attain these accuracy gains while keeping peak GPU memory and per-batch training time competitive with or more efficient than existing prompt-based baselines.

Inference-time efficiency. Table 12 further evaluates the runtime footprint at inference. Here, we report the number of active parameters, including both the extra parameters introduced by each method and the resulting total parameter count at test time, as well as the peak GPU memory and per-batch latency. Despite using 7.05M trainable parameters during training, LAVA activates only 0.47M extra

Algorithm 1 LAVA Training and Inference Algorithm

Input: Training sets $\mathcal{X}_1, \cdots, \mathcal{X}_T$, test sets $\mathcal{Z}_1, \cdots, \mathcal{Z}_T$, class names for label set $\mathcal{C}$.
Output: The predicted labels of test images.

// Initialization Phase
Pre-compute static Text-based Anchors $\mathcal{A}^{\text{Text}}$ using the frozen text encoder G.
Initialize learnable pools with random parameters: Prompts $\mathcal{P}$, Visual Anchors $\mathcal{A}^{\text{Vis}}$, Prototype Anchors $\mathcal{V}^{\text{proto}}$, and Classifiers Ψ.
// Training Phase
for t in $[1, T]$ **do**
  for each (x_i, y_i) in $\mathcal{X}_t$ **do**
    Compute the prompt-adapted feature $\mathbf{g}_i$ using prompt $\mathbf{P}_{(t)}$, per Eq. (1).
    Compute structural loss $\mathcal{L}_{\text{Struct}}$ via the VL-RSA module, using Eqs. (2) to (5).
    Compute aggregated feature $\mathbf{f}_i$ via the CA-CDFA module, per Eqs. (6) and (7).
    Optimize learnable parameters for domain t ($\mathbf{P}_{(t)}, \ldots, \psi^{(t)}$) by minimizing the combined loss $\mathcal{L}$ from Eq. (8).
  end for
  Using the MLFI module, compute the domain prototype $\boldsymbol{\mu}_{(t)}$ for domain t on $\mathcal{X}_t$ per Eqs. (9) and (10).
end for
// Inference Phase
Get the test set $\mathcal{Z}_{1\sim T} = \bigcup_{\tau=1}^{T} \mathcal{Z}_\tau$.
for z in $\mathcal{Z}_{1\sim T}$ **do**
  Compute multi-level feature $F(z)$ using the MLFI extractor per Eq. (9).
  Identify domain label $s = \arg\max_{1\le t\le T} \cos(F(z), \boldsymbol{\mu}_t)$.
  Compute prompt-adapted feature $\mathbf{g}_z$ using components from domain s, per Eq. (1).
  Compute aggregated feature $\mathbf{f}_z$ using components up to domain s, per Eqs. (6) and (7).
  Predict label $\hat{y} = \psi^{(s)}(\mathbf{f}_z)$ using the classifier for domain s.
end for

parameters at inference, corresponding to a **0.31%** increase over the backbone. This overhead is an order of magnitude smaller than that of C-Prompts and KA-Prompts, which each require 10.56M extra active parameters (**+7.06%**). The *LAVA-share* variant is even more lightweight, with just 0.32M extra active parameters (**+0.21%**) while preserving most of the performance. Consequently, both LAVA and *LAVA-share* maintain a lightweight inference-time footprint, with GPU memory usage and latency that are on par with or better than other strong baselines, while still delivering a clear accuracy advantage.

Overall parameter efficiency and accuracy–cost trade-off. A key aspect of LAVA's design is its *per-stage* efficiency: although the total number of trainable parameters grows with the number of domains, the parameters updated at each incremental stage remain modest compared to other high-performing, parameter-expanding methods. Practically, this leads to improved training efficiency, since only a small subset of the overall parameters needs to be updated when a new domain arrives. Overall, these results demonstrate that LAVA's state-of-the-art performance is not achieved by merely increasing parameter counts or computation: LAVA uses a moderate training-time parameter budget together with an extremely small inference-time overhead, and converts these resources into substantial accuracy gains, achieving a superior balance between accuracy, memory footprint, and runtime cost across both training and inference.

7.7. Further Details of the Training and Inference

Algorithm 1 introduces how to train and test our Language-Anchored Visual Alignment (LAVA) framework. The symbols correspond to those used in the Problem Formulation subsection and the Method section.

7.8. Visualization Results

In addition to the main paper, we also provide the performance trends on seen domains for two additional benchmarks, i.e., DomainNet and ImageNet-Mix, as illustrated in Fig. 10 and Fig. 9.

As shown in Fig. 10, LAVA sets a new state-of-the-art on the DomainNet benchmark, outperforming the prior SOTA method, PINA. This result is particularly compelling as the experiment utilizes the *Real*-first domain sequence, a setting known to be most favorable for PINA due to its architectural reliance on the base domain. Despite this, LAVA secures a **+0.46%** advantage. The performance gap widens significantly in other permutations; for instance, LAVA's lead in-

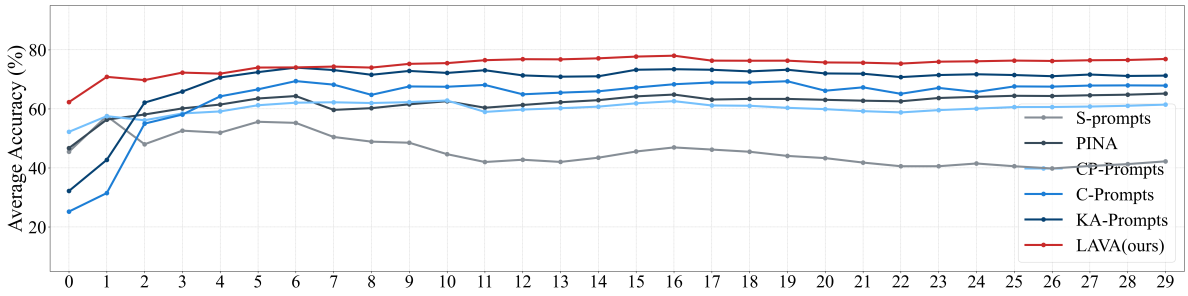


Figure 9. ImageNet-Mix average accuracy on seen domains.

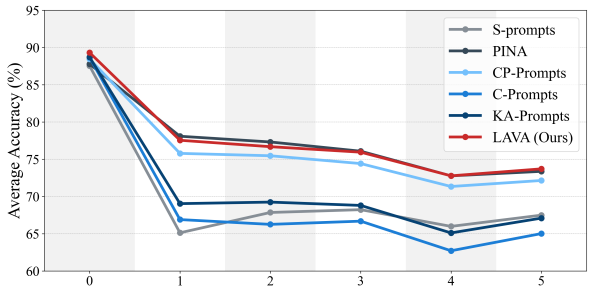


Figure 10. DomainNet average accuracy on seen domains.

creases to a substantial **+4.93%** when *Quickdraw* is the initial domain.

LAVA's robustness is further demonstrated on the more demanding ImageNet-Mix benchmark (Fig. 9), which comprises 30 sequential domain shifts. Here, LAVA establishes a decisive and consistent lead over all baseline methods throughout the entire learning process. This significant performance margin underscores LAVA's superior generalization and robustness in complex, long-sequence continual learning scenarios.

7.9. Limitations and Future Directions

While LAVA performs strongly across diverse benchmarks, several limitations remain. First, the method implicitly assumes a certain degree of semantic structural consistency across domains, as the relative alignment mechanism relies on semantic relationships in the embedding space remaining reasonably stable. In practice, this assumption generally holds for natural-image benchmarks; however, if the model is applied to domains with extreme discrepancies in visual style, statistical properties, or semantic ontology—such as highly stylized or artificially constructed domains with little correspondence to natural-image semantics—the shared language-anchored structure may become less reliable, potentially reducing adaptation effectiveness and degrading performance on those domains.

Second, although only a small number of parameters are updated at each stage, the cumulative parameter count still grows linearly with the number of domains. Reducing redundancy by merging pools for semantically similar domains or reusing pools when domain similarity is high represents a promising direction for improving long-horizon scalability.

Looking ahead, future work could pursue more robust domain-invariant structures and more scalable parameterization strategies that better leverage cross-domain commonality. Such developments may enhance the deployment of incremental models in dynamic environments and deepen our understanding of stability–plasticity trade-offs in large-scale vision–language systems.