Boosting Vision-Language Models Towards Cross-Domain Incremental Object Detection

Supplementary Material

In this appendix, we provide: (1) Additional experiment details. (2) Additional method details. (3) More comparison and ablation results of our method. (4) Visualizations of task-specific subspaces and our group shared subspace.

A. Additional Experimental Details

This section presents: (i) detailed statistics of the datasets, (ii) comprehensive description of the benchmark protocol construction, and (iii) a brief overview of the evaluated methods.

A.1. Dataset Details

To evaluate the continual learning capabilities of Vision-Language Models (VLMs) across a wider range of downstream task scenarios, we constructed the Cross-Domain Class-Incremental Object Detection (CDIOD) benchmark. This benchmark leverages datasets from three different domains, encompassing common natural scenes alongside remote sensing and underwater datasets that exhibit significant distributional gaps from typical pre-training data.

- **DIOR** [11] is a large-scale remote sensing object detection dataset. It contains 20 classes (e.g., `golffield`, `bridge`, `stadium`), with 18,463 training images and 5,000 validation images. Classes primarily feature architectural and infrastructural objects.
- **Pascal VOC 2012** [5] is a widely recognized natural scenes object detection benchmark. It comprises 20 classes (e.g., `car`, `person`, `cat`), with 13,690 training images and 3,422 validation images. Classes focus on common everyday objects.
- **RUOD** [6] is a dataset specifically curated for underwater object detection. It consists of 10 classes (e.g., `turtle`, `diver`, `starfish`), with 9,800 training images and 4,200 validation images. Classes include various marine life.

For related Incremental Object Detection (IOD) and Incremental Vision-Language Object Detection (IVLOD) benchmarks, we utilize:

- **COCO** [14] is a large-scale natural scenes dataset for object detection, segmentation. It features 80 classes (e.g., `person`, `car`, `dog`), with 118,000 training images and 5,000 validation images.
- **ODinW13** [10] is a comprehensive benchmark designed to evaluate zero-shot object detection performance. It aggregates 13 different sub-datasets, including: Aerial Maritime Drone (Ae), Aquarium (Aq), Cottontail Rabbits (Co), Egohands (Eg), Mushrooms (Mu), Packages (Pa), Pascal VOC (Pv), Pistols (Pi), Pothole (Po), Raccoon (Ra), Shellfish (Sh), Thermal Dogs and People (Th), and Vehicles (Ve).

A.2. Benchmark construction details

Extending IOD for Modern VLMs. Modern VLM-based Detectors, such as GLIP [12] and Grounding DINO [15], are designed to generalize across diverse domains. Accordingly, their generalization capability is commonly evaluated on heterogeneous benchmarks (e.g., ODinW35), which cover a wide range of domains, imaging conditions, and visual concepts. Motivated by this, we construct CDIOD not merely as a dataset extension of conventional IOD, but as a generalized IOD setting tailored to modern VLMs.

We interpret incremental learning as a continuation of large-scale pre-training, where the detector progressively expands its knowledge capacity through exposure to new domains. Under this view, we integrate multiple datasets from distinct domains into a unified continual learning protocol, enabling the model to acquire both intra-domain and cross-domain continual learning capability. The ultimate goal is to obtain a unified detector that generalize to diverse downstream tasks without retraining from scratch.

Dataset Split and CDIOD Protocol. We first verify that the adopted datasets (DIOR, Pascal VOC, and RUOD) exhibit no label-level overlap. Although some categories may appear semantically related at a high level (e.g., *vehicle* in DIOR vs. *car* in Pascal VOC), they correspond to distinct visual concepts in practice due to substantial differences in annotation definitions and imaging conditions. Specifically, DIOR adopts a coarse-grained *vehicle* category covering diverse transportation types captured from remote-sensing viewpoints, whereas Pascal VOC defines *car* as a fine-grained category observed from natural scenes. Combined with large gaps in viewpoint, scale, resolution, and scene context (remote, natural, underwater), these categories are not interchangeable. Therefore, no category merging or exclusion is required.

All categories are mapped into a unified global label space with 50 classes in total (e.g., DIOR: 1–20, VOC: 21–40, RUOD: 41–50), and all datasets are converted into a unified COCO-format annotation with global label indexing applied consistently during training and evaluation. To simulate continual learning, we follow the standard class-IOD protocol within each domain. Each dataset is partitioned into multiple incremental tasks with disjoint class subsets, where each task introduces five new classes (e.g., Pascal

(a) DIOR (b) Pascal VOC (c) RUOD

Figure 1. Dataset visualization of Cross-Domain Incremental Object Detection

VOC is split into four sequential tasks). After completing all incremental stages within one dataset, the model is transferred to the next dataset and continues training under the same protocol. After the full CDIOD process, the model is evaluated on the validation sets of all datasets. This protocol captures both **intra-domain continual learning** (within a single dataset) and **cross-domain continual learning** (sequential adaptation across datasets).

In our experiments, we report the average performance after three runs with shuffled training orders. In practice, to ensure fair comparisons, these orders were defined as **Order 1** (DIOR $\rightarrow$ Pascal VOC $\rightarrow$ RUOD), **Order 2** (Pascal VOC $\rightarrow$ RUOD $\rightarrow$ DIOR) and **Order 3** (RUOD $\rightarrow$ Pascal VOC $\rightarrow$ DIOR). Specifically, we treat the 50 classes from DIOR (20 classes), Pascal VOC (20 classes), and RUOD (10 classes) as a single, complete continual learning process. We take **Order 1** to explain the training process: Classes $1 \sim 20$ correspond to DIOR, $21 \sim 40$ to Pascal VOC, and $41 \sim 50$ to RUOD. On this basis, we consider the following two configurations:

- **0-10 (5 phases):** Each phase introduces 10 new classes, followed by evaluation on all learned classes. The training and testing process is as follows:
  - Phase 1: DIOR (classes $1 \sim 10$)
  - Phase 2: DIOR (classes $11 \sim 20$)
  - Phase 3: Pascal VOC (classes $21 \sim 30$)
  - Phase 4: Pascal VOC (classes $31 \sim 40$)
  - Phase 5: RUOD (classes $41 \sim 50$)
  - Test: DIOR + Pascal VOC + RUOD (classes $1 \sim 50$)
- **0-5 (10 phases):** Each phase introduces 5 new classes, with evaluation conducted after each phase across all accumulated classes.

Furthermore, to test the model's robustness against extreme task sequence variation, we introduce an additional setting **'Shuffled'**, where all tasks across domains are randomly ordered. The results for all settings are summarized in Tab. 1.

A.3. Evaluated Method Details

Implementation Details. We conduct a comprehensive evaluation of existing Incremental Object Detection (IOD) methods. To ensure fairness and consistency, all methods are re-implemented on the Grounding DINO-T backbone, pre-trained on Objects365, GoldG, and Cap4M. All methods are trained without a memory bank.

- **CL-DETR** [16]. A full fine-tuning-based method. It adopts a from-scratch Deformable-DETR and generates pseudo-labels that do not overlap with ground truth. The pseudo-label generation strategy is architecture-agnostic and was directly reproduced on the Grounding DINO backbone. Following the original paper, we select the top 10 predictions to generate pseudo-labels, setting the overlap with ground truth to not exceed 0.7. The learning rate is set to $1e-4$. A learning rate decay of 0.1 is applied to the vision backbone, while the language backbone remains frozen.
- **GCD** [21]. A full fine-tuning-based method. It adopts a from-scratch Grounding-DINO and employs correspondence distillation to transfer the teacher model's responses and topological relationships. Following the official implementation, we set the pseudo threshold to 0.4. The coefficients of the correspondence distillation loss are set as $\gamma = 1, \lambda_1 = 3, \lambda_2 = 5$. The learning rate is set to $5e-5$, with a learning rate decay of 0.1 applied to the vision backbone. The language backbone is frozen.
- **MD-DETR** [1]. A PEFT-based method. It was originally initialized from a Deformable-DETR pre-trained on LVIS. It introduces vision prompts organized in a prompt pool, which are injected into the self-attention per decoder layer. For our re-implementation, we follow the official code, using 100 memory units ($N_m = 100$) with a length of 10 ($L_m = 10$) and a dimension of 256 ($D = 256$). We set $\lambda_Q = 0.01$ and $\delta_{bt} = 0.65$. Only

the prompt pool and query function are updated during training, with the learning rate set to $1e-2$.

- **Zira** [3]. A PEFT-based method. It is initialized from Grounding DINO pre-trained on Objects365, GoldG, and Cap4M. It integrates a re-parameterizable dual-branch module for task adaptation, inserted into both the language and vision backbone-to-enhancer connections within the neck. Following the official code, we set the coefficient for the Zil loss to $\lambda = 0.1$ and the learning rate decay for LLRB is $\eta = 0.2$. The learnable scale factor is initialized with $s = 0.1$. Only the RDB module is updated. The learning rate is initialized at $1e-3$ for the first epoch and then decays to $1e-4$.
- **MR-GDINO** [4]. A PEFT-based method. It is initialized from Grounding DINO pre-trained on Objects365, GoldG, and Cap4M. Task-specific adapters and prompts are trained to mitigate cross-task interference, and a prototype-based routing strategy is used during inference. Following the official code, LoRA are inserted into the fusion layer of the enhancer. The prompt length is set to 10 and LoRA rank to 16. The out-of-distribution threshold τ is set to 0.89. The learning rate is initialized at $1e-3$ for the first epoch and then decays to $1e-4$.
- **BCL** [22]. A PEFT-based Method from incremental classification, originally implemented in CLIP. It introduces adapters organized by a Mixture-of-Experts (MoE) for task adaptation and employs an activate-freeze strategy to alleviate catastrophic forgetting. For our re-implementation, we insert the MoE-Adapters into the FFN of the enhancer. Following the official code, we set the bottleneck for the adapter to $D = 64$ and use a top-2 gating strategy. For each task, we use 2 experts and 1 router, resulting in $N_E = 20$ experts and $N_R = 10$ routers in 10 phases setting. For routing, we use the mean of the image tokens instead of the [CLS] token. We set the learning rate $1e-3$ for adapter and router, $3e-3$ for domain predictor.

A.4. Related Benchmarks

Incremental Object Detection (IOD). Incremental Object Detection typically indicates Class-Incremental Object Detection. Conventional IOD benchmarks typically partition a general-domain dataset such as COCO into disjoint tasks defined by category labels. Each phase introduces new object classes under the assumption of a consistent data distribution. IOD can be seen as a special case of CDIOD where the domain remains fixed.

Domain Incremental Object Detection (DIOD). DIOD focuses on the challenge of a model continuously adapting to a sequence of shifting domains. While the domain changes incrementally, the set of object classes is typically assumed to remain fixed. Our CDIOD benchmark presents a more complex problem by combining the challenges of both DIOD (domain shift) and IOD (class-incremental learning), requiring a method to handle both novel domains and novel classes simultaneously.

Incremental Vision-Language Object Detection (IVLOD). IVLOD [3] focuses on incrementally adapting pre-trained vision-language models (VLMs) to a sequence of tasks from the ODinW-13 benchmark while preserving zero-shot generalization. IVLOD primarily addresses task-incremental scenarios, where the model's predictions are confined to a task-specific class space and the label space is allowed to overlap across tasks. In contrast, our work tackles a more complex and realistic class-incremental challenge. For training, CDIOD features a disjoint label space, 'person' labeled in task t will not be labeled in subsequent tasks $t^{'} > t$. For inference, all learned classes are evaluated together without knowing which task set the test image belongs to.

Dual Incremental Object Detection. The most recently related work, DuET [18], pioneered the exploration of co-existing domain and class IOD. However, DuET focuses primarily on style-level domain shifts (e.g., Daytime $\rightarrow$ Night, or Comic $\rightarrow$ Clipart $\rightarrow$ VOC), where new classes arrive alongside subtle stylistic variations. In contrast, CDIOD targets more challenging scenarios relevant to modern VLM-based detectors by jointly modeling intra-domain and cross-domain incremental learning under substantial semantic and imaging gaps (e.g., natural images vs. remote sensing), thereby introducing new stability–adaptivity challenges not covered by existing settings.

Open-Vocabulary Object Detection (OVD). Open-Vocabulary Object Detection focuses on building models capable of detecting any category without being explicitly trained on them beforehand. The primary goal of OVD is zero-shot generalization, the ability to identify unseen categories in downstream tasks without the need for additional supervision. Starting from an generalizable OVD model, CDIOD focues on the subsequent challenge of continual learning. CDIOD evaluates a model's ability to continuously learn from a sequence of supervised tasks, focusing on challenges of adaptation to new data and preservation of previously acquired knowledge.

Domain Adaptation Object Detection (DAOD). Domain Adaptation Object Detection typically focuses on a single-step adaptation process where a model trained on a labeled source domain is adapted to an unlabeled target domain. A key assumption in most DAOD methods is that the set of object classes across the source and target domains is identical. Unlike DAOD, our benchmark addresses a multi-phase, supervised continuous learning scenario where both the domain and the object classes change over time, and all new data is provided with labels.

B. Additional Method Details

In this section, we provide supplementary details of our method, including: (i) Pseudo code illustrating the training and inference pipelines. (ii) A detailed formulation of our method (iii) Hyperparameter configurations adopted in our experiments, and (iv) An intuitive explanation of the IGC.

B.1. Pseudo Code.

We present the pseudo code of our proposed method in Algorithm 1 and Algorithm 2, detailing both the training and inference procedures.

B.2. Detailed Dynamic Task Grouping.

The Dynamic Task Grouping (DTG) mechanism adaptively clusters tasks by estimating and comparing their feature distributions. This ensures that tasks with high similarity promote knowledge sharing, while dissimilar tasks are isolated. For a given task $\mathcal{D}_t$, we first extract image features $\mathcal{F}_t$ using the frozen vision backbone. Features can be taken either from the last-stage backbone features for faster inference or from a multi-level representation for higher accuracy (we default to the last-stage). We approximate the task's distribution in the feature space by a multivariate Gaussian $\mathcal{N}_t = \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$, where the mean vector $\boldsymbol{\mu}_t$ and the regularized covariance matrix $\boldsymbol{\Sigma}_t$ are computed as:

$$\boldsymbol{\mu}_t = \frac{1}{N_t} \sum_{i=1}^{N_t} f_i, \tag{1}$$

$$\boldsymbol{\Sigma}_t = \frac{1}{N_t - 1} \sum_{i=1}^{N_t} (f_i - \boldsymbol{\mu}_t)(f_i - \boldsymbol{\mu}_t)^\top + \epsilon I. \tag{2}$$

Here, N_t is the number of samples in the task, d is the feature dimension, and ϵI is a small diagonal regularizer for numerical stability. To compare two Gaussian task distributions, $\mathcal{N}_i$ and $\mathcal{N}_j$, we employ the symmetrized Kullback–Leibler divergence [8]:

$$D_{\text{sym}}(i, j) = \frac{1}{2} \left[D_{\text{KL}}(\mathcal{N}_i \| \mathcal{N}_j) + D_{\text{KL}}(\mathcal{N}_j \| \mathcal{N}_i) \right], \tag{3}$$

where $D_{\text{KL}}(\mathcal{N}_i | \mathcal{N}_j)$ is the KL divergence between two multivariate Gaussians. The similarity between a new task $\mathcal{D}_t$ and an existing group g (containing tasks $\{k \in g\}$) is defined by the minimum distance within that group:

$$\text{KL}(t, g) = \min_{k \in g} \left[D_{\text{sym}}(t, k) \right]. \tag{4}$$

The most similar group g^* is then identified by:

$$g^* = \arg\min_g \text{KL}(t, g). \tag{5}$$

The final decision follows the expansion threshold:

$$\phi(t) = \begin{cases} g^*, & \text{if KL}(t, g^*) < \tau_{\text{expand}}, \\ \text{Initialize new group}, & \text{otherwise}, \end{cases} \tag{6}$$

where τ_{expand} is the expansion threshold controlling the creation of new groups.

B.3. Detailed Group Routing for Inference.

At inference, a test image x is routed using a compact representation extracted from the frozen Swin-T backbone. We apply global average pooling to the last-stage features to obtain a resolution-invariant vector $\mathbf{z}$. The test sample is modeled as a Gaussian $\mathcal{N}(\mathbf{z}, \Sigma_t)$, sharing covariance with each candidate task distribution $\{\mathcal{N}(\mu_t, \Sigma_t)\}$ to avoid unreliable variance estimation from a single image. DTG selects the closest group g^* by minimizing KL divergence and activates the corresponding adapters $\mathcal{A}_{g^*}$ across all relevant layers for prediction.

B.4. Detailed knowledge distillation loss.

For a task t assigned to an existing group g, the corresponding IGA $\mathcal{A}_g$ is expanded with a new adapter α_g^{new}, which is initialized from the group's base adapter α_g^{base}. We denote the model output as $\mathcal{M}(x; \alpha_g)$ for an input sample x. During training, we compute: (i) the student output using α_g^{new} and (ii) the teacher output using the group's base adapter α_g^{base}. The knowledge distillation loss is defined as:

$$\mathcal{L}_{\text{kd}} = \mathcal{L}\left(\mathcal{M}(x; \alpha_g^{\text{new}}), ; \mathcal{M}(x; \alpha_g^{\text{base}})\right), \tag{7}$$

In practice, $\mathcal{L}$ serves as a soft constraint term, which we implement as topology distillation. Specifically, we define the object prototype for class c as:

$$p_c = \frac{1}{N_c} \sum_{i=1}^{N_c} \alpha_i q_i, \tag{8}$$

where q_i is the output query feature (from the last decoder layer), α_i is the confidence score derived from the predicted logits, and N_c is the number of instances in class c.We then compute the pairwise relation matrix over classes:

$$R_{ij} = \|p_i - p_j\|_2, \quad i, j \in C_{1:t-1}, \tag{9}$$

and define the topology loss as:

$$\mathcal{L}\left(\mathcal{M}(x; \alpha_g^{\text{new}}), ; \mathcal{M}(x; \alpha_g^{\text{base}})\right) = \|R^{\text{new}} - R^{\text{base}}\|_2, \tag{10}$$

where R^{base} is obtained by switching to the base adapter. Similarly, to maintain cross-modal structural consistency. Finally, the total distillation loss is expressed as:

$$\mathcal{L}_{\text{kd}} = \gamma_1 \mathcal{L}_{\text{topology\_image}} + \gamma_2 \mathcal{L}_{\text{topology\_text}}, \tag{11}$$

where γ_1 and γ_2 are balancing coefficients controlling the contributions of image and text topology preservation, respectively.

B.5. Hyperparameter setup.

Our training objective integrates five hyperparameters. The overall loss is formulated as:

$$\mathcal{L} = \mathcal{L}_{\text{align}} + \mathcal{L}_{\text{reg}} + \delta(t)\mathcal{L}_{\text{kd}} \quad (12)$$

$$= \lambda_{\text{focal}}\mathcal{L}_{\text{focal}} + \lambda_{\text{L1}}\mathcal{L}_{\text{L1}} + \lambda_{\text{GIoU}}\mathcal{L}_{\text{GIoU}} + \delta(t)\mathcal{L}_{\text{kd}}, \quad (13)$$

where $\delta(t) = 1$ indicates we expand existing group, and the knowledge distillation loss is applied. Following [15, 21], we set $\lambda_{\text{focal}} = 1$, $\lambda_{\text{L1}} = 5$, $\lambda_{\text{GIoU}} = 2$ and $\gamma_1 = 3$ and $\gamma_2 = 5$ for all experiments.

B.6. How Intra-Group Consolidation works.

To intuitively understand task-specific subspaces and group shared subspace, we provide visualizations in Fig. 7 and descriptions in Sec. D.1.

Group Initialization. For tasks within an existing group, we initialize a new adapter from the group's base one. This "warm-start" approach provides the model with a knowledgeable starting point that already contains information from old tasks, rather than beginning from a random state, which has been proven effective for alleviating catastrophic forgetting [9, 13]. This also aligns with the principle of linear mode connectivity (LMC) [17, 19], which states that a sharing initialization is crucial for keeping solutions of related tasks within a connected low error basin. By initializing from the base adapter, we start training already inside this optimal basin, making learning a more stable and efficient search for a nearby solution. This connectivity provides the theoretical foundation for why adapters in the same group can be linearly merged: since they exist in the same basin, their weighted average is also likely a high-performing solution.

Group Alignment. The assumption of LMC generally holds for tasks like PASCAL VOC that are well-aligned with the pre-training distribution. In such cases, the pre-trained model has already situated the parameters within a favorable low-error basin, requiring the adapter to perform only minimal exploration to reach its optimal solution. However, for tasks that are dissimilar to the pre-training distribution (e.g., DIOR sub-tasks), their optimal solutions in the parameter space can be far from the base adapter's. In this case, a direct linear merge becomes suboptimal, as the simple average of two distant points is likely to fall into a high-error region. To address this, we leverage group alignment as an implicit constraint.

C. Additional Results

This section presents additional comparison results including : (i) Detailed CDIOD results and method comparisons (ii) per-task performance analysis (iii) extended ablation studies on various design choices and hyperparameters, and (iiii) analysis of DTG and global routing robustness.

C.1. Additional comparison results

Detailed CDIOD results and Impact of training Order. To further assess the impact of training order of tasks. we provide a detailed per order performance. As shown in Tab. 1, most existing methods are sensitive to the sequence of tasks and exhibit significant performance fluctuations. This instability is a key limitation in real-world applications where the arrival of new data is unpredictable. In contrast, our method consistently delivers stable performance across all three orders. This robustness to variations in the training sequence is a critical property for a practical incremental learning algorithm, demonstrating our framework's reliability in non-stationary and unpredictable environments.

Detailed comparison and discussion. As shown in Tab. 2, we report the Average Performance (3 runs) and computational costs of each method under the 10-phase setting. And we provide a detailed discussion of each method.

- **CL-DETR [16].** It directly fine-tunes the base model using pseudo-labels, which results in high trainable parameters but introduces no additional overhead during inference. However, under cross-domain scenarios, old classes may be absent or suffer from modality gaps, which makes generating robust pseudo-labels impractical.
- **GCD [21].** It uses KD to transfer knowledge. This approach also results in a high number of trainable parameters but introduces no overhead at inference. While the KD loss is designed to force the student model to align with the teacher's output, this alignment can be problematic in cross-domain tasks where the teacher's responses are often noisy, which may lead to catastrophic forgetting.
- **MD-DETR [1].** It utilizes expandable prompt pools to incrementally learn new tasks and employs a retrieval function to compose prompts with weighted sum for inference. Although the prompts are memory-efficient (1.84M), the retrieval function introduces substantial overhead (207.4M), which originates from linear projection $(num\_queries \times emb\_dims, num\_queries)$ that maps decoder outputs to weights of proposals. The expressive power of prompts limits the model's adaptivity.
- **Zira [3].** It introduces a fixed dual-branch module attached to the base model. After learning each task, a high-learning-rate branch is merged into a low-learning-rate branch. While this design allows Zira to adapt to new tasks with a fixed parameter budget, it struggles to balance stability and adaptivity.
- **BCL [22].** It employs a MoE structure to combine adapters. It first uses a auto-encoder based domain discriminator to select a task-specific router, which then performs token-wise routing to activate the corresponding adapter. When applied to object detection tasks, this complex design faces two issues. First, it encounters performance bottleneck of task-ID inference. Furthermore, the complex detection scenes make its token-wise routing

Table 1. CDIOD performance ($AP\%$) under 0–5 and 0–10 settings across three different training orders. Performance is reported on all datasets following the completion of the final incremental phase. The extreme 'Shuffled' setting, where tasks from three domains are randomly ordered to assess robustness, reports the average performance over three independent runs.

Training Orders	Method	0-10 (5 phases)				0-5 (10 phases)			
		DIOR	PascalVOC	RUOD	Avg	DIOR	PascalVOC	RUOD	Avg
	Zero-shot	2.7	51.9	19.6	24.7	2.7	51.9	19.6	24.7
	Joint	69.5	72.0	64.3	68.6	69.5	72.0	64.3	68.6
Order 1	CL-DETR	35.3	57.7	**63.9**	52.3	23.0	47.6	**57.5**	42.7
	MD-DETR	34.9	61.4	49.9	48.7	28.6	59.4	38.1	42.0
	BCL	32.7	66.1	59.5	52.8	20.5	59.3	41.5	40.4
	GCD	43.2	60.0	62.8	55.3	37.0	52.0	58.3	49.1
	Zira	29.8	66.2	54.3	50.1	22.0	63.9	47.2	44.4
	MR-GDINO	44.5	67.3	57.2	56.3	33.6	63.4	48.7	48.6
	Ours	**64.7**	**68.2**	62.6	**65.2**	**58.5**	**65.5**	57.2	**60.4**
Order 2	CL-DETR	**65.8**	34.0	35.5	45.1	**66.1**	17.6	21.6	35.1
	MD-DETR	35.4	58.5	48.4	47.4	30.1	57.9	36.5	41.5
	BCL	33.7	63.9	55.7	51.1	20.8	59.2	36.2	38.7
	GCD	63.5	41.5	40.8	48.6	61.9	35.1	27.5	41.5
	Zira	40.4	59.3	41.1	46.9	30.8	59.3	36.1	42.1
	MR-GDINO	45.2	66.7	57.1	56.3	34.1	63.4	48.6	48.7
	Ours	63.6	**68.7**	**62.3**	**64.9**	58.9	**64.5**	**56.7**	**60.0**
Order 3	CL-DETR	**65.6**	47.2	23.2	45.3	**65.8**	26.4	20.3	37.5
	MD-DETR	35.2	59.0	47.1	47.1	29.7	58.2	35.8	41.2
	BCL	34.2	64.0	57.7	52.0	21.7	61.0	43.4	42.0
	GCD	63.1	53.5	30.9	49.2	58.2	39.0	23.9	40.4
	Zira	40.2	62.3	40.6	47.7	29.7	60.6	35.8	42.0
	MR-GDINO	44.3	66.8	56.7	55.9	35.4	63.4	48.9	49.2
	Ours	61.2	**68.4**	**62.5**	**64.0**	58.9	**65.8**	**56.2**	**60.3**
Shuffled	CL-DETR	61.0	56.3	42.5	53.3	52.6	54.5	38.5	48.5
	MD-DETR	36.8	61.2	43.6	47.2	32.5	58.6	34.4	41.8
	BCL	38.2	65.8	58.7	54.1	23.1	59.9	44.5	42.5
	GCD	61.6	56.5	44.1	54.1	53.1	54.9	37.0	48.3
	Zira	41.5	62.4	44.1	49.3	31.9	63.3	40.1	45.1
	MR-GDINO	44.6	67.2	57.3	56.4	33.2	**63.8**	47.6	48.2
	Ours	**62.3**	**68.2**	**62.0**	**64.2**	**56.8**	63.5	**56.7**	**59.0**

Table 2. Performance and Computation Costs under 0–5 (10 phases) settings. 'Avg' denotes the average performance across the three fixed training orders. Train and Test Params refer to parameters updated during training and activated at inference, respectively. Percentages denote ratio to base model parameters; ↑ indicates increased percentage. All methods are PEFT-based except GCD and CL-DETR.

Method	Technical	Activation strategy	Avg	Train Params	Test Params	FLOPS
CL-DETR	Pseudo-label	Base model only	38.4	64.2M(37.1%)	173.1M(↑0.0%)	464G
GCD	Knowledge Distillation	Base model only	43.7	64.2M(37.1%)	173.1M(↑0.0%)	464G
MD-DETR	Prompt pool	Retrieval function	41.6	209.2M(120.9%)	383.3M(↑120.9%)	502G
Zira	Rep Dual-branch	Fixed branch	42.8	4.38M(2.5%)	177.5M(↑ 2.5%)	467G
BCL	Mixture of Task-wise Adapters	Token-wise routing	40.4	9.69M(5.6%)	175.5M(↑1.47%)	479G
MR-GDINO	Task-wise LoRA	Task-wise routing	48.8	4.94M(2.8%)	173.6M(↑0.28%)	470G
T-LoRA	Task-wise LoRA	Task-wise routing	51.5	6.90M(4.0%)	173.8M(↑ 0.4%)	473G
Ours	Dynamic Group-wise LoRA	Group-wise routing	60.2	2.06M(1.2%)	173.8M(↑0.4%)	473G

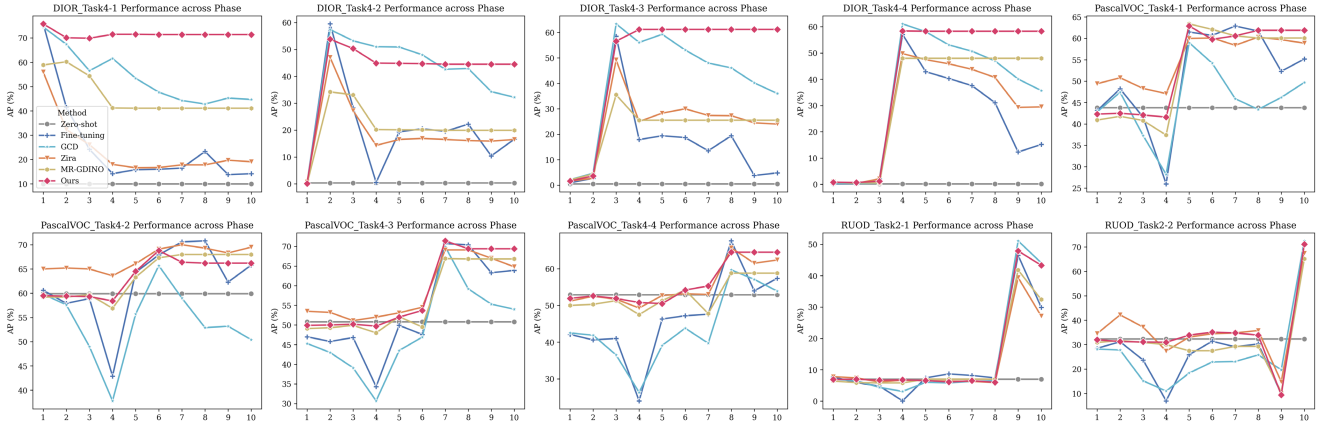


Figure 2. Performance across phases under the 0–5 (10 phases) setting with training order 1, where the x-axis denotes the training phase.

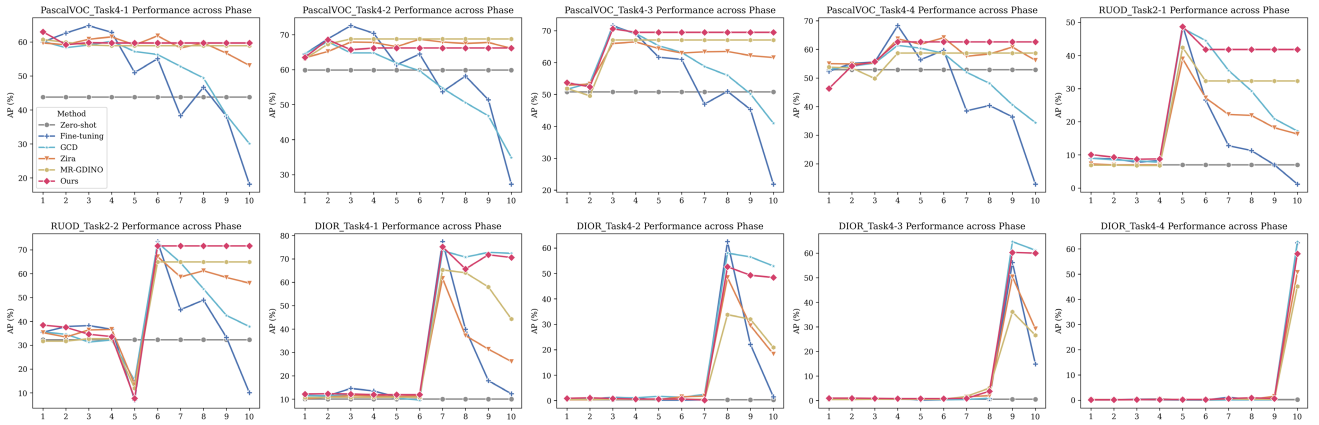


Figure 3. Performance across phases under the 0–5 (10 phases) setting with training order 2, where the x-axis denotes the training phase.

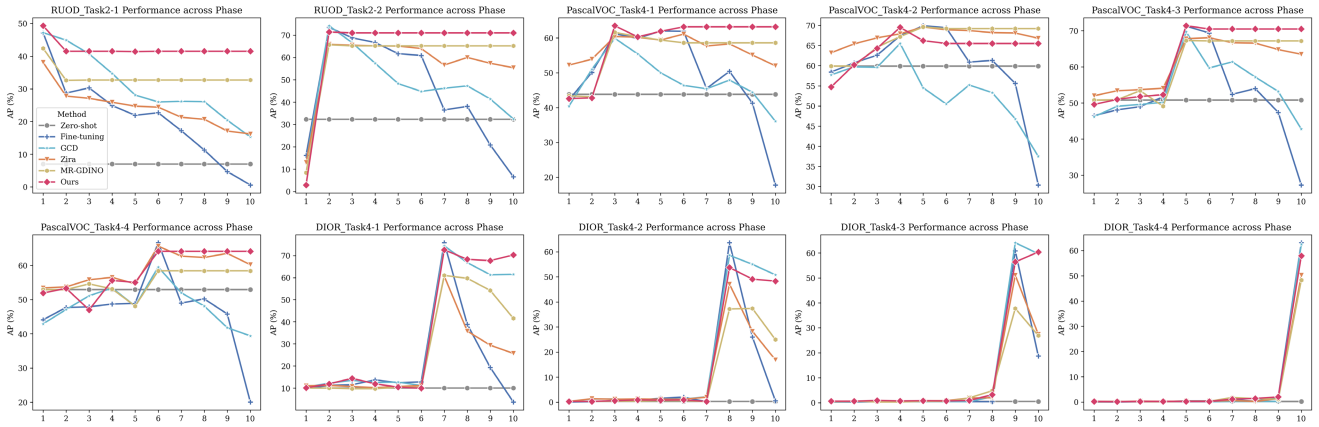


Figure 4. Performance across phases under the 0–5 (10 phases) setting with training order 3, where the x-axis denotes the training phase.

susceptible to misallocation.

- **MR-GDINO [4].** It trains task-specific LoRA modules and prompts. LoRA are directly inserted into the fusion layer. This design limits the model's adaptivity, as performance has not been explored for other insertion locations. While prototype-based retrieval mitigates some task-ID inference issues, performance can still degrade if retrieval is incorrect, making task routing a key bottleneck.

- **T-LoRA.** A naive baseline of ours. It trains task-specific LoRA modules and uses task-level routing to select the optimal ones via distribution-based routing for inference. This straightforward approach achieves good performance. However, it models each task in isolation, ignoring shared knowledge and leading to a linear increase in parameters. Its primary performance bottleneck remains the reliance on accurate task-ID inference.

To address these limitations, our method dynamically groups related tasks, enabling knowledge reuse while keeping parameter growth minimal. This design removes the reliance on precise task-ID inference, as routing at the group level is far more robust. The dynamic pipeline also supports creating new groups for novel tasks to improve adaptivity, while integrating similar tasks into existing groups to preserve stability. As shown in Table 1, our approach adds only 1.2% trainable parameters, activates just 0.4% extra parameters during inference, and increases FLOPs by only 9G over the base model.

Per-task performance across phases. In Fig. 2, we present the detailed performance of each sub-task under the 0-5 (10 phases) setting with training order1 (DIOR $\rightarrow$ Pascal VOC $\rightarrow$ RUOD). Given the zero-shot capability of pre-trained VLMs, we evaluate all sub-tasks after each training phase to track their performance evolution across 10 phases. This serves purely as an analytical experiment, since future tasks are unknown in practical CDIOD training. For example, with 50 total categories, DIOR_Task4-1 introduces the first 5 classes in phase 1, achieves peak performance, and gradually degrades over subsequent phases. From the results, our method demonstrates strong adaptivity, achieving immediate performance comparable to fine-tuning for new tasks, while maintaining stability by effectively preserving prior task performance in cross-domain scenarios. Furthermore, our approach also well sustains the model's zero-shot capability. For instance, when PascalVOC_Task4-3 is introduced in phase 7, its zero-shot performance from phases 1 to 6 remains largely intact. Same trend can be observed in Order 2 (Fig. 3) and Order 3 (Fig. 4).

C.2. Additional ablation results

Detailed ablation of each components. We provide a detailed ablation results of each component of our framework. Row 0 indicates Zero-shot performance. Row 1 represent sequentially fine-tuning LoRA, which leads to severe catastrophic forgetting. Row 2 represent fine-tune LoRA with group alignment (treating all tasks in same group and the LoRA is updated within accumulated label spaces), the result show that this practice still unable to overcome forgetting under cross-domain scenarios. Row 3 represent train task-specific expert module per task and combine them through task-wise routing mechanism, a design widely used in recent works [1, 22]. In this case, the LoRA parameters increase linearly with task numbers, which is still suboptimal due to performance bottleneck of task-id predictions. Row 4 represent we merge LoRA weights trained on each task through averaging merging, however knowledge of different domains varies significantly which leads to poor performance.

Dynamic Task Grouping (DTG) alone just construct task-to-group mapping which can't be ablated. In Row 5, instead, we leverage DTG and Group_Merge to train Group-wise LoRA. Knowledge from different domains are managed by groups, where we could merge relevant LoRA weights. This design transforms the task-wise routing to robust group-wise routing, which builds up a strong baseline for our framework. To enhance model compositionality, we introduce group initialization mechanism. Once we expand existing group, we initialize the new task-specific adapter from the base one. Row 7 and Row 8, we also ablate the merging process to focus on the impact of Group_Align. The results show that alignment is crucial for alleviating forgetting, particularly for remote sensing and underwater tasks, which are not well-aligned with the pre-trained model. Our full method (Row 9) combines all components and adopt dynamic training pipeline which automatically switch between direct adapter updates and updates with Intra-Group Consolidation (IGC), effectively balance adaptivity and stability.

Impact of Merge Factor λ_{merge}. We investigate the effect of the merge factor λ_{merge} on model performance. A smaller value encourages the model to absorb more task-specific knowledge, whereas a larger value emphasizes the preservation of base knowledge. The merging step essentially serves as a regularization mechanism on the parameter space. As shown in Tab. 4, omitting the merging step entirely leads the model to overfit to the new task, while an excessively large λ_{merge} overly constrains model updates, both of which result in suboptimal performance. A relatively small λ_{merge} achieves better results. Therefore, we set $\lambda_{\text{merge}} = 0.2$ in our experiments to strike an effective balance between knowledge retention and task-specific adaptation.

Out-of-distribution threshold. At inference time, **as an optional choice**, we introduce an out-of-distribution (OOD) threshold th_{ood} to handle inputs that are far from any known group distribution. If an input exceeds this threshold, no IGA is activated; instead, it is processed solely by the base model, leveraging the zero-shot capability of the pre-trained VLM for unseen samples. To evaluate this, we use **ZCOCO** (following IVLOD) by performing zero-shot testing on COCO after CDIOD training. As shown in Tab. 5, a low threshold risks misrouting known tasks to zero-shot prediction. We set **500 as the default**, as it effectively balances task-specific accuracy with the retention of general zero-shot capability.

Comparison of adapter variants. We evaluated differ-

Table 3. Impact of different components, reporting Extra Parameters Percentage (EPP, %) and Average performance ($AP\%$) under 0-5 (10 phases) setting. Row 0 indicates Zero-shot. LoRA is set rank 16 and inserted into enhancer's FFN for all ablation results in this table.

Row	LoRA	Task-wise	Raw Merge	DTG	Intra-Group Consolidation			EPP	DIOR	VOC	RUOD	Avg
					Group_Merge	Group_Init	Group_Align					
0								0.00%	2.7	51.9	19.6	24.7
1	✓							0.40%	3.2	42.5	42.9	29.5
2	✓						✓	0.40%	5.1	52.0	53.8	37.0
3	✓	✓						4.00%	41.0	63.7	49.7	51.5
4	✓	✓	✓					4.00%	29.7	61.5	40.1	43.8
5	✓			✓	✓			1.20%	48.7	65.2	49.1	54.3
6	✓			✓	✓	✓		1.20%	51.6	66.7	51.9	56.7
7	✓			✓			✓	1.20%	50.5	60.9	52.3	54.6
8	✓			✓		✓	✓	1.20%	55.7	61.3	55.4	57.5
9	✓			✓	✓	✓	✓	**1.20%**	**58.8**	65.3	**56.7**	**60.2**

Table 4. Impact of the merging factor λ_{merge} under 0-5 setting.

Threshold	DIOR	VOC	RUOD	Avg
$\lambda_{merge} = 0.0$	55.7	61.3	55.4	57.5
$\lambda_{merge} = 0.1$	58.4	65.2	56.8	60.1
$\lambda_{merge} = 0.2$	58.8	65.3	56.7	60.2
$\lambda_{merge} = 0.3$	55.0	64.5	57.0	58.8
$\lambda_{merge} = 0.4$	47.8	65.9	55.0	56.2
$\lambda_{merge} = 0.5$	39.3	64.8	49.4	51.2
$\lambda_{merge} = 0.6$	31.7	64.1	44.5	46.8
$\lambda_{merge} = 0.7$	23.6	63.3	35.6	40.8

Table 5. Impact of the OOD threshold th_{ood} under 0-5 setting. ZCOCO follows the setting in IVLOD, where zero-shot testing is conducted on COCO after training on CDIOD to evaluate the preservation of zero-shot capability.

Threshold	DIOR	VOC	RUOD	Avg	ZCOCO
$th_{ood} = 200$	54.2	64.2	53.4	57.3	0.470
$th_{ood} = 300$	57.9	65.3	55.8	59.7	0.465
$th_{ood} = 400$	58.3	65.5	56.4	60.1	0.461
$th_{ood} = 500$	58.8	65.3	56.7	60.2	0.458
$th_{ood} = 600$	58.5	65.5	56.7	60.2	0.457

Table 6. Comparison of adapter variants within our framework, reporting Extra Parameters Percentage (EPP, %) and Average performance ($AP\%$) under 0-5 (10 phases) CDIOD setting. Both are inserted into the enhancer's FFN.

Method	EPP	DIOR	VOC	RUOD	Avg
LoRA(r=4)	0.30%	53.2	64.1	52.9	56.7
LoRA(r=8)	0.60%	55.6	64.7	55.2	58.5
LoRA(r=16)	1.20%	58.8	65.3	56.7	60.2
LoRA(r=32)	2.40%	59.8	65.9	57.6	61.1
Adapter(d=16)	0.18%	48.7	63.8	49.7	54.1
Adapter(d=32)	0.35%	51.5	64.1	52.3	56.0
Adapter(d=64)	0.69%	53.9	64.5	53.5	57.3
Adapter(d=128)	1.38%	55.8	65.0	56.5	59.1
Adapter(d=256)	2.75%	57.7	65.3	57.0	60.0

Table 7. Method generality evaluation on GLIP under the CDIOD setting (Order1 0–10). All methods are evaluated after the full incremental process.

Method	Params (M)	DIOR	VOC	RUOD	Avg
Zero-shot	231.8	3.8	56.2	10.4	23.5
Fine-tuning	231.8	0.8	18.0	65.7	28.2
Ours(GLIP)	234.7	54.8	61.4	52.4	56.2

ent adapter modules for the base adapter in our framework, comparing LoRA [7] and Adapter [2]. In our implementation, LoRA with rank r is attached to the two linear layers of the feed-forward network (FFN), while the Adapter module with bottleneck dimension d is placed outside the FFN. As shown in Tab. 6, when the number of effective parameters is comparable (e.g., LoRA(r=16) and Adapter(d=128)), LoRA consistently achieves superior performance.

Method Compatibility with VLM Detectors. Our method is built upon LoRA-style parameter-efficient adaptation and is therefore directly applicable to detectors that contain linear layers. In addition, the proposed DTG routing mechanism only relies on frozen backbone features, which are readily available in modern vision-language model (VLM) based detectors, such as GLIP, YOLO-World, and OV-DINO. As a sanity check for method generality beyond the main architecture, we instantiate our framework on GLIP [12], a representative VLM detector. We adopt the publicly released GLIP model pre-trained on Objects365 and GoldG. During incremental training, the Swin-T backbone is fully frozen and used exclusively for DTG routing.

Table 8. Pseudo-label quality analysis under the CDIOD protocol (Order 1). We report precision (P), recall (R), and the number of false positives (FP), where true positives (TP) are defined by IoU > 0.5. Here σ denotes the confidence threshold for pseudo-labeling

Method	VOC (T_6–T_8, Avg)			RUOD (T_9)			RUOD (T_{10})			All Phases (Avg)		
	P ↑	R ↑	FP ↓	P	R	FP	P	R	FP	P	R	FP
GCD	0.40	0.86	3757	N/A	N/A	3276	0.55	0.78	4537	0.41	0.67	3261
$Ours(\sigma = 0.3)$	0.44	0.88	3835		N/A		0.52	0.81	5431	0.50	0.85	4137
$Ours(\sigma = 0.4)$	0.57	0.77	2081		N/A		0.65	0.72	2712	0.66	0.77	1894
$Ours(\sigma = 0.5)$	0.60	0.74	1763		N/A		0.69	0.69	2221	0.70	0.74	1519
$Ours(\sigma = 0.6)$	0.63	0.70	1491		N/A		0.73	0.65	1756	0.74	0.70	1200

Table 9. Routing accuracy under different input resolutions. We report average routing accuracy across all tasks and per-task routing accuracy from T_1 to T_{10}. The default input resolution used in our experiments (800×1333) is highlighted.

Inference Image Size	Average Accuracy.	Per-Task Routing Accuracy ($T_1 \rightarrow T_{10}$)
512×512	0.966	0.98, 1.00, 0.98, 1.00, 1.00, 1.00, 1.00, 1.00, 0.90, 0.80
640×640	0.988	1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 0.96, 0.92
800×1333	0.998	1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 0.98

Following the design of GLIP, we insert the proposed IGA modules into the Bi-attention blocks of the fusion layer (DyHead), while keeping all other components unchanged. All training schedules and hyper-parameters follow the same protocol as in the main experiments.

As shown in Tab. 7, our method achieves consistently strong performance across all datasets , demonstrating that the proposed framework is not tied to a specific detector architecture and can be effectively extended to modern VLM-based detectors.

Pseudo-Label Quality and Robustness. Pseudo-labeling is a commonly adopted component in IOD for leveraging unlabeled data, but it is not the core contribution of our method. Accordingly, we adopt a simple confidence-based strategy with a fixed threshold σ and report the quality of generated pseudo labels as shown in Tab. 8.

A key design of our framework is its dynamic pipeline. When the DTG detects a domain transition (e.g., from VOC to RUOD at T_9), the model initializes and directly updates a new base adapter without applying pseudo-labeling or knowledge distillation. This explicitly avoids introducing large amounts of noisy pseudo labels, a failure mode commonly observed in baseline methods such as GCD under severe domain shifts. For tasks routed to an existing group (e.g., T_{10}), pseudo-labeling is enabled, as the corresponding base adapter already encodes domain-specific knowledge, which effectively controls pseudo-label noise.

Overall, the results confirm that pseudo-labeling can be safely and effectively utilized when domain knowledge is already established, while explicitly disabling it at domain transitions is critical for robustness under distribution shifts.

Robustness Analysis of DTG Routing. We further analyze the robustness of the proposed DTG routing mechanism by examining the expansion signal, defined as the minimum KL divergence between the test task distribution and existing groups, under varying per-task data sizes. Figure 5 visualizes the evolution of this signal across tasks, where vertical dashed lines indicate domain transitions.

When a sufficient number of samples is available for distribution estimation (at least 260 images per task in our experiments), the expansion signal remains stable and yields consistent grouping results. Specifically, under the CDIOD protocol, DTG consistently recovers three groups corresponding to $\{T_{1\text{-}4}, T_{5\text{-}8}, T_{9\text{-}10}\}$, which align well with the underlying domain structure. A similar behavior is observed on the ACDC benchmark [20], where DTG correctly groups weather conditions into $\{\text{Fog/Rain}, \text{Night}, \text{Snow}\}$. These results indicate that the proposed routing mechanism can reliably identify domain shifts and reuse existing groups when adequate data is provided.

Under data-scarce conditions (fewer than 260 samples), the expansion signal becomes noisier, occasionally leading to unstable grouping decisions. This behavior is expected, as accurate Gaussian distribution estimation becomes unreliable with extremely limited samples. We explicitly acknowledge this limitation of explicit distribution modeling. As a potential alternative for such scenarios, implicit distribution modeling methods, such as autoencoder-based representations using reconstruction error as the expansion signal, could be explored to improve robustness under severe data scarcity.

Robustness of Group Routing to Input Resolution. At inference time, each test image is routed to the group corresponding to the closest task distribution based on the DTG criterion. Since routing relies on global average pooled features extracted from the last stage of the frozen backbone,

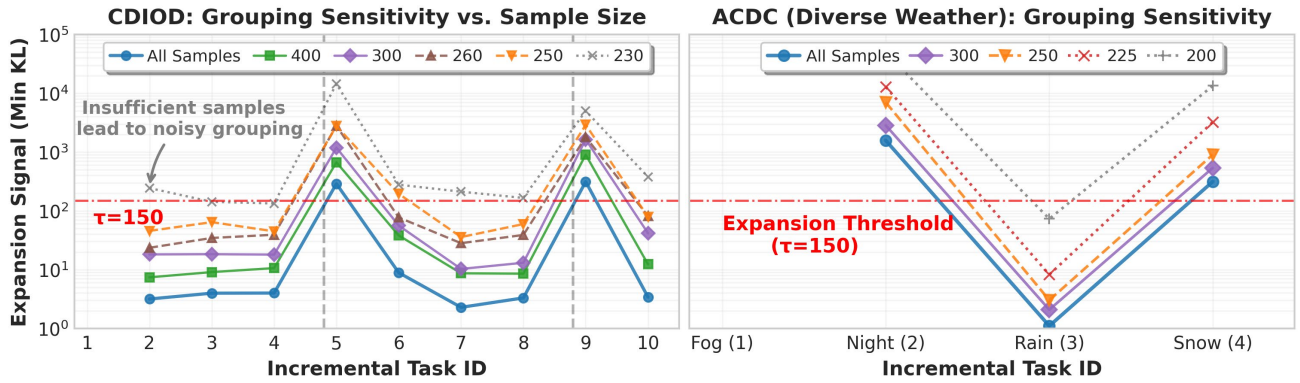


Figure 5. Analysis of DTG expansion signal under varying task data sizes. The minimum KL divergence is used as the expansion signal, with vertical dashed lines indicating domain transitions. DTG produces stable and consistent grouping when sufficient samples are available, while data-scarce settings lead to noisier signals. ACDC is a driving scene understanding benchmark covering diverse adverse weather conditions (fog, night, rain, and snow). We use it here to verify DTG's robustness in identifying and grouping distinct weather-driven domain shifts in real-world scenarios.

the resulting representation is resolution-invariant by design. To empirically verify this property, we evaluate routing accuracy under different input resolutions.

As shown in Table 9, routing accuracy remains consistently high across different input resolutions, with only minor degradation at the lowest resolution. This confirms that the proposed group routing mechanism is robust to input size variations and can be reliably applied under different inference settings without re-tuning.

D. Additional Visualization

D.1. Visualization of subspaces.

To intuitively understand how the proposed Dynamic Group Subspace operates, we visualize both the task-specific subspaces and the consolidated shared subspace in Fig. 7.

For **RUOD** (Row 1), the subspaces learned by individual tasks show strong task dependency: each adapter forms a feature space in which only its own classes are clearly separable, whereas classes from the other task become entangled and poorly structured. This reveals an inherent limitation of task-wise routing—misrouting a sample to the wrong task-specific subspace inevitably produces non-discriminative features, ultimately leading to degraded detection performance. Row 1(c) further illustrates that naively averaging two task-specific subspaces results in a collapsed representation where class boundaries almost disappear. In contrast, the consolidated subspace produced by IGC forms a coherent and well-organized space where classes from both tasks remain clearly separated. This demonstrates that IGC effectively aligns task features and constructs a shared subspace that maintains discriminability while seamlessly integrating knowledge across tasks. For **DIOR** (Row 2), we observe a similar trend: task-specific subspaces struggle to generalize beyond their own tasks, whereas the consolidated subspace consistently preserves class separability across different incremental steps. This further verifies that DGS builds stable group-wise subspaces.

D.2. Pre-trained vs. Learned Features.

To qualitatively assess the learned representations, we perform a t-SNE visualization on the output query features. Each data point in the plot is colored and labeled according to its predicted class. For this analysis, we sample 10 classes from each of the three datasets to generate the t-SNE plots. The top row of Fig. 6 displays the features extracted from the base model's zero-shot outputs, while the bottom row corresponds to the features of our method. As shown in Fig. 6, our method consistently produces features that are well-separated by class across all datasets, highlighting its effectiveness in learning distinct and robust representations.

D.3. Qualitative Results.

We provide a qualitative comparison of Zero-shot, GCD, Zira, and our method under the 0-5 (10-phase) setting, with all models evaluated after completing all training phases.

- **DIOR (Remote Sensing):** The pre-trained Grounding-DINO exhibits limited zero-shot generalization on DIOR, likely because remote sensing targets are treated as background in its pre-training datasets (e.g., Objects365). Under incremental learning, both GCD and Zira suffer from severe catastrophic forgetting on this domain. In contrast, our method maintains predictions closely aligned with the ground truth, demonstrating stronger retention of DIOR-specific knowledge.
- **Pascal VOC (Natural Scenes):** As Pascal VOC is closely aligned with the pre-training distribution, most PEFT-based methods perform well. However, KD-based

methods like GCD shows clear performance degradation in the incremental setting, indicating evident forgetting and a weak ability to retain previously learned knowledge under domain shifts.

- **RUOD (Underwater):** RUOD poses additional challenges due to its distinct underwater visual characteristics. Zira struggles to adapt effectively, which can be attributed to its limited adaptivity on out-of-distribution tasks.

References

[1] Gaurav Bhatt, James Ross, and Leonid Sigal. Preventing catastrophic forgetting through memory networks in continuous detection. In *European Conference on Computer Vision*, pages 442–458. Springer, 2024. 2, 5, 8

[2] Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. *Advances in Neural Information Processing Systems*, 35:16664–16678, 2022. 9

[3] Jieren Deng, Haojian Zhang, Kun Ding, Jianhua Hu, Xingxuan Zhang, and Yunkuan Wang. Zero-shot generalizable incremental learning for vision-language object detection. *Advances in Neural Information Processing Systems*, 37: 136679–136700, 2024. 3, 5

[4] Bowen Dong, Zitong Huang, Guanglei Yang, Lei Zhang, and Wangmeng Zuo. Mr-gdino: efficient open-world continual object detection. *arXiv preprint arXiv:2412.15979*, 2024. 3, 7

[5] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 1

[6] Chenping Fu, Risheng Liu, Xin Fan, Puyang Chen, Hao Fu, Wanqi Yuan, Ming Zhu, and Zhongxuan Luo. Rethinking general underwater object detection: Datasets, challenges, and solutions. *Neurocomputing*, 517:243–256, 2023. 1

[7] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022. 9

[8] Edwin T Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957. 4

[9] 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. 5

[10] Chunyuan Li, Haotian Liu, Liunian Li, Pengchuan Zhang, Jyoti Aneja, Jianwei Yang, Ping Jin, Houdong Hu, Zicheng Liu, Yong Jae Lee, et al. Elevater: A benchmark and toolkit for evaluating language-augmented visual models. *Advances in Neural Information Processing Systems*, 35:9287–9301, 2022. 1

[11] Ke Li, Gang Wan, Gong Cheng, Liqiu Meng, and Junwei Han. Object detection in optical remote sensing images: A survey and a new benchmark. *ISPRS journal of photogrammetry and remote sensing*, 159:296–307, 2020. 1

[12] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10965–10975, 2022. 1, 9

[13] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 5

[14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 1

[15] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European conference on computer vision*, pages 38–55. Springer, 2024. 1, 5

[16] Yaoyao Liu, Bernt Schiele, Andrea Vedaldi, and Christian Rupprecht. Continual detection transformer for incremental object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23799–23808, 2023. 2, 5

[17] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Dilan Gorur, Razvan Pascanu, and Hassan Ghasemzadeh. Linear mode connectivity in multitask and continual learning. *arXiv preprint arXiv:2010.04495*, 2020. 5

[18] Munish Monga, Vishal Chudasama, Pankaj Wasnik, and Biplab Banerjee. Duet: Dual incremental object detection via exemplar-free task arithmetic. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3121–3131, 2025. 3

[19] Yujia Qin, Cheng Qian, Jing Yi, Weize Chen, Yankai Lin, Xu Han, Zhiyuan Liu, Maosong Sun, and Jie Zhou. Exploring mode connectivity for pre-trained language models. *arXiv preprint arXiv:2210.14102*, 2022. 5

[20] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Acdc: The adverse conditions dataset with correspondences for semantic driving scene understanding. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10765–10775, 2021. 10

[21] Xu Wang, Zilei Wang, and Zihan Lin. Gcd: Advancing vision-language models for incremental object detection via global alignment and correspondence distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8015–8023, 2025. 2, 5

[22] Jiazuo Yu, Yunzhi Zhuge, Lu Zhang, Ping Hu, Dong Wang, Huchuan Lu, and You 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*, pages 23219–23230, 2024. 3, 5, 8

Algorithm 1 Training Procedure

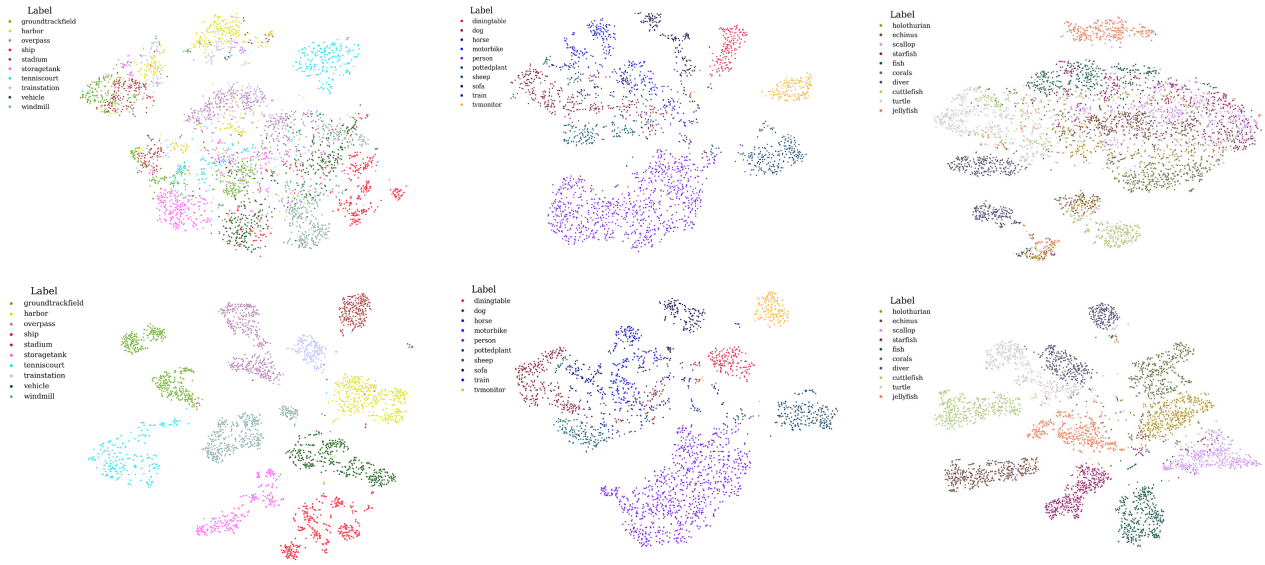
Input: Sequence of tasks $\{\mathcal{D}_1, \ldots, \mathcal{D}_T\}$; Model $\mathcal{M}$; Set of groups G; Incremental Group Adapters (IGA) $\{\mathcal{A}_g\}_{g \in G}$.
Output: A set of updated IGA $\{\mathcal{A}_g\}_{g \in G}$.

for task $\mathcal{D}_t \in \{\mathcal{D}_1, \ldots, \mathcal{D}_T\}$ **do**
  {**1. Dynamic Task Grouping**}
  Iterate $\mathcal{D}_t$ to estimate task distribution $\mathcal{N}_t$.
  Find closest group: $g^* \leftarrow \arg\min_{g \in G} \mathrm{KL}(t, g)$.
  if $G = \emptyset$ **or** $\mathrm{KL}(t, g^*) \geq \tau$ **then**
    $g^* \leftarrow g_{(|G|+1)}$ ▷ Create a new group
    $G \leftarrow G \cup g^*$
    $\alpha_{\text{active}} \leftarrow \alpha_{g^*}^{\text{base}}$ ▷ Initialize new $\mathcal{A}_{g^*}$
  else
    $\alpha_{\text{active}} \leftarrow \alpha_{g^*}^{\text{new}}$ ▷ Expand existing $\mathcal{A}_{g^*}$, init $\alpha_{g^*}^{\text{new}}$ from group base $\alpha_{g^*}^{\text{base}}$
  end if
  {**2. Dynamic Training Pipeline**}
  for each training epoch **do**
    for each batch $x \in \mathcal{D}_t$ **do**
      if task t is assigned to a newly created group **then**
        $\mathcal{L} \leftarrow \mathcal{L}_{\text{align}} + \mathcal{L}_{\text{reg}}$ ▷ Train without constraints for adaptivity
      else
        $\mathcal{L} \leftarrow \mathcal{L}_{\text{align}} + \mathcal{L}_{\text{reg}} + \mathcal{L}_{\text{kd}}$ ▷ Train with group alignment for stability
      end if
      Update parameters of α_{active} using loss $\mathcal{L}$.
    end for
  end for
  {**3. Adapter Merging**}
  if task t was assigned to an existing group g^* **then**
    $\alpha_{g^*}^{\text{base}} \leftarrow \lambda \alpha_{g^*}^{\text{base}} + (1 - \lambda)\alpha_{\text{active}}$ ▷ Merge into group's base adapter
    Discard α_{active}
  end if
end for

Algorithm 2 Inference Procedure

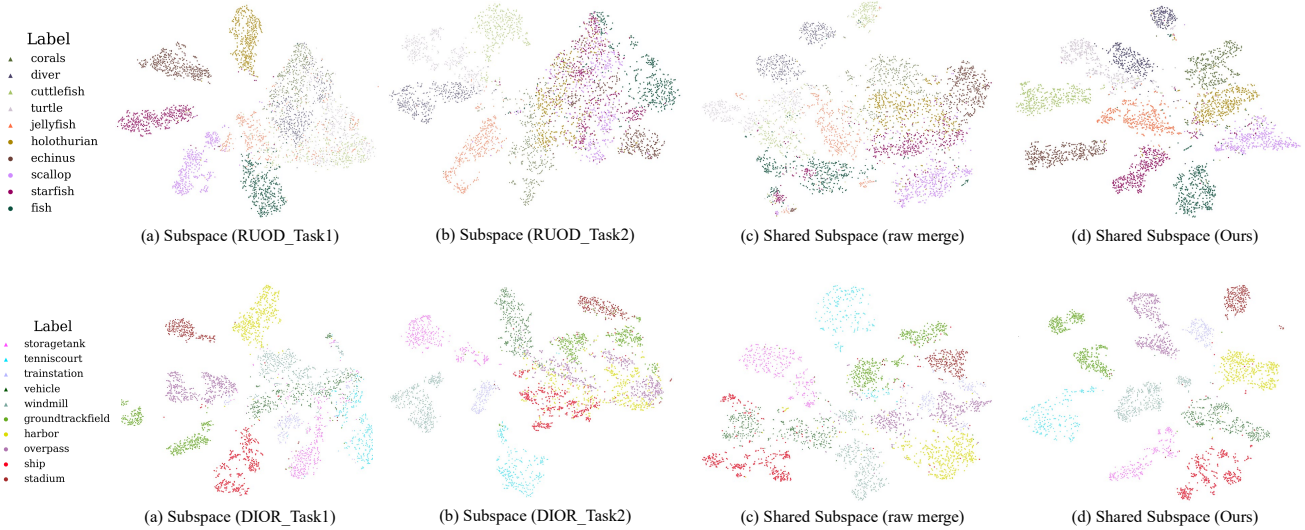
Input: Test image x; Model $\mathcal{M}$ with frozen backbone f_v; Group set G with distributions $\{\mathcal{N}_g\}_{g \in G}$ and adapters $\{\mathcal{A}_g\}_{g \in G}$; OOD threshold τ_{ood}.
Output: Prediction y.

{**1. Extract Sample Distribution**}
Compute $\mathcal{N}_x = \mathcal{N}(\mu_x, \Sigma_x)$ from x using f_v.
{**2. Group Routing**}
$g^* \leftarrow \arg\min_{g \in G} \mathrm{KL}(x, g)$.
{**3. Adapter Activation and Prediction**}
if $KL(x, g^*) < \tau_{\text{ood}}$ **then**
  $y \leftarrow \mathcal{M}(x; \mathcal{A}_{g^*})$ ▷ In-distribution: Activate matched IGA
else
  $y \leftarrow \mathcal{M}(x)$ ▷ OOD: Fallback to base model (zero-shot)
end if
return y



(a) DIOR (b) Pascal VOC (c) RUOD

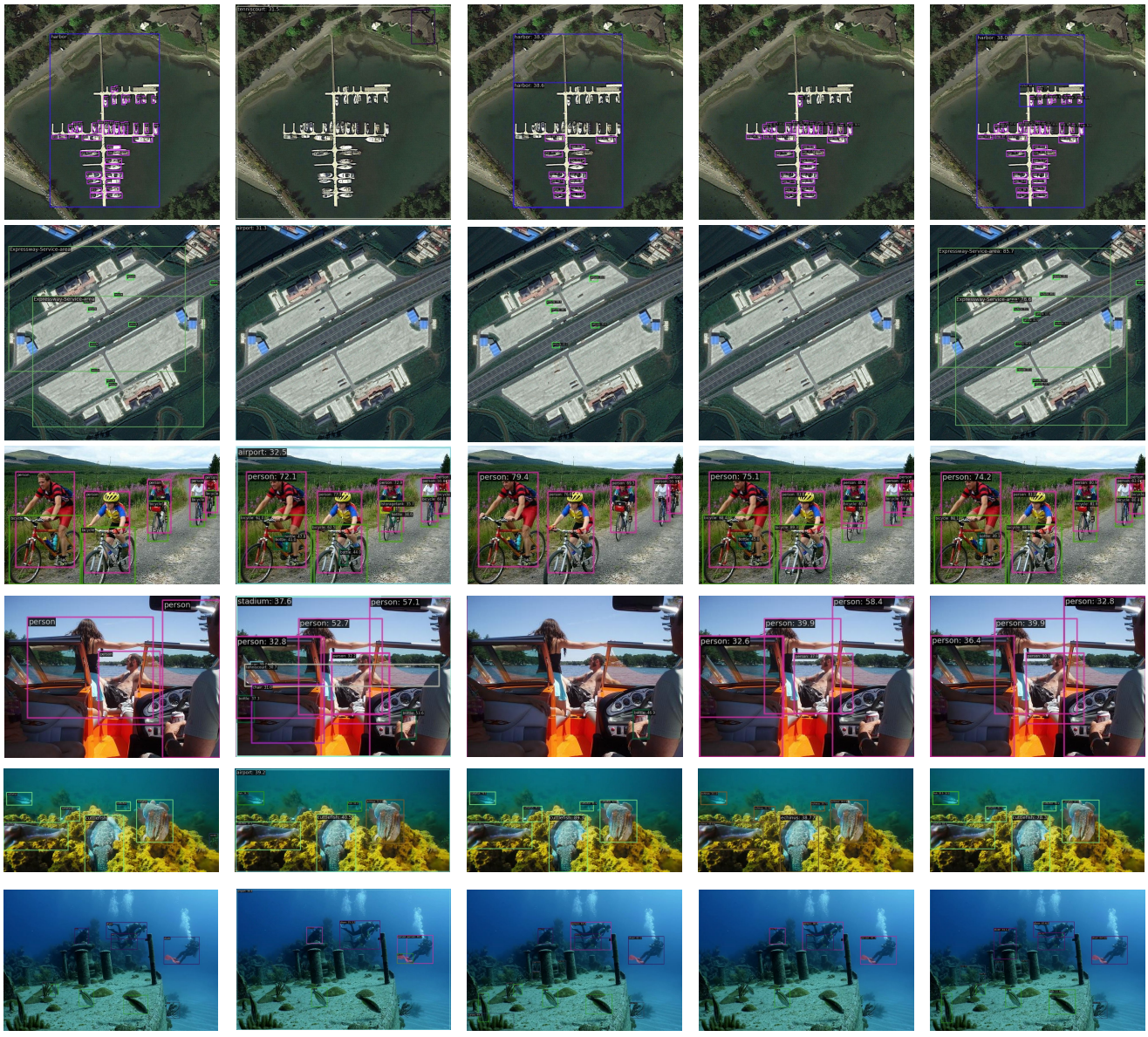
Figure 6. Class-level t-SNE of output query features, with each point labeled by its corresponding class. Row 1: Zero-shot; Row 2: Ours.



(a) Subspace (RUOD_Task1) (b) Subspace (RUOD_Task2) (c) Shared Subspace (raw merge) (d) Shared Subspace (Ours)

(a) Subspace (DIOR_Task1) (b) Subspace (DIOR_Task2) (c) Shared Subspace (raw merge) (d) Shared Subspace (Ours)

Figure 7. t-SNE visualizations of the subspace defined by different adapters, highlighting how each subspace discriminates its corresponding task. Row 1 (RUOD): Classes from RUOD_Task 1 are marked with ◯, and classes from RUOD_Task 2 with △. (a) Subspace trained only on RUOD_Task 1, where ◯ classes are well separated; (b) Subspace trained only on RUOD_Task 2, where △ classes are well separated; (c) Naively merging the two subspaces via weighted averaging leads to a poorly consolidated subspace; (d) Our intra-group consolidation produces a shared subspace that aligns both tasks. Row 2 (DIOR): t-SNE visualizations of subspaces learned from DIOR tasks.



(a) Ground Truth (b) Zero-shot (c) GCD (d) Zira (e) Ours

Figure 8. Qualitative results of Zero-shot, GCD, Zira, and our method under the 0–5 (10-phase) setting. Rows 1 to 2 show samples from DIOR, 3 to 4 from Pascal VOC, and 5 to 6 from RUOD.