HARE: HIERARCHY-AWARE ROUTING OF LORA EXPERTS FOR OPEN-VOCABULARY OBJECT DETECTION AT HIGHER SEMANTIC LEVELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Open-vocabulary object detectors perform strongly on concrete categories but often miss relevant instances when queried with higher-level concepts such as *animal*, *vehicle*, or *furniture*. We study *higher-level category detection*, which requires localizing instances from visually diverse concrete subclasses using their shared ancestor concept as the query. To support training and evaluation, we construct hierarchy-annotated versions of COCO, LVIS-mini, and Objects365 by associating each source category with a concrete-to-abstract semantic path and propagating its instance annotations to the concepts on that path without additional bounding-box annotation. We further propose Hierarchy-Aware Routing of Experts (HARE), a simple parameter-efficient adaptation framework that uses parent–child relations as explicit supervision for allocating adaptation parameters across category queries. Specifically, a ranking objective encourages parent concepts to receive higher abstraction scores than their children, guiding a text-conditioned router over lightweight LoRA experts. HARE is adapted only on Objects365 and evaluated on held-out Objects365 categories and the unseen COCO and LVIS-mini datasets. Under direct higher-level queries, HARE improves average box AP over Grounding DINO by 8.5, 5.1, and 4.2 points on COCO, LVIS-mini, and Objects365, respectively, while matching its concrete-category AP on COCO. Under identical subclass augmentation, HARE retains gains of 3.7, 2.3, and 1.1 points. These results demonstrate that explicit hierarchy supervision improves higher-level detection and transfers across datasets without relying on auxiliary subclass cues at inference.

1 INTRODUCTION

Open-vocabulary object detection (OVOD) has made substantial progress in detecting objects specified by textual category queries and generalizing to novel concrete categories beyond a fixed training vocabulary. However, existing detectors often struggle to detect the same instances when queried with higher-level categories. For example, a detector that successfully localizes a *dog* may miss the same instance when queried with its semantic ancestor *animal*. We study higher-level category detection in OVOD, which requires localizing instances across diverse concrete subclasses using their shared higher-level category as the query.

In real-world applications, detection requests often extend beyond individual concrete categories to higher-level semantic categories. A household robot may need to locate *furniture*, a traffic monitoring system may need to detect *vehicles*, and an ecological monitoring system may need to identify *animals* without prior knowledge of the species present. In such settings, enumerating all relevant concrete subclasses can be impractical, motivating direct detection using higher-level category queries. Yet, as illustrated in Figure 1, pretrained Grounding DINO (Liu et al., 2023) misses relevant instances under the queries *animal*, *furniture*, and *vehicle*, highlighting incomplete instance coverage under higher-level category queries.

Adapting pretrained detectors with higher-level category supervision offers a natural way to address this limitation. However, standard adaptation strategies, including full fine-tuning and single-LoRA adaptation, employ a shared set of trainable parameters across category queries without ex-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



Figure 1: Qualitative comparison for the higher-level categories *animal*, *furniture*, and *vehicle*. From left to right: pretrained Grounding DINO with higher-level category queries; HARE with the same queries and no prompt augmentation; HARE with subcategory prompt augmentation, where detections from concrete-category prompts are mapped to the corresponding higher-level category; and ground-truth boxes labeled with the corresponding higher-level category.

plicitly accounting for semantic granularity. Such shared adaptation must accommodate distinct alignment requirements: concrete-category queries require discrimination between closely related classes, whereas higher-level queries require associating visually diverse subclasses with a common ancestor concept. Updates driven by higher-level supervision may therefore disrupt the pretrained visual–semantic alignments that support concrete-category discrimination, potentially improving higher-level detection at the expense of concrete-category performance. This tension motivates a granularity-aware adaptation mechanism that accommodates higher-level semantic supervision while preserving concrete-category detection capabilities.

To this end, we propose **H**ierarchy-**A**ware **R**outing of **E**xperts (HARE), a simple hierarchy-supervised adaptation framework for higher-level category detection. Our central idea is to use the relative ordering of semantic abstraction encoded by parent–child relations as explicit supervision for allocating adaptation parameters across category queries. We implement this idea with a text-conditioned router over lightweight LoRA experts (Hu et al., 2022). A hierarchy-ranking objective encourages the router to assign higher abstraction scores to parent categories than to their children, and the resulting routing decisions determine which expert adapts each category query. To support training and evaluation at higher semantic levels, we construct hierarchical annotations for COCO (Lin et al., 2014), LVIS-mini (Gupta et al., 2019), and Objects365 (Shao et al., 2019) by associating each annotated instance with the ancestor concepts of its concrete category, without additional bounding-box annotation. We adapt HARE on Objects365 and evaluate it on held-out Objects365 categories and on COCO and LVIS-mini for cross-dataset transfer. Experiments demonstrate consistent improvements in higher-level detection across the three benchmarks while maintaining the pretrained detector's concrete-category performance on COCO.

Our main contributions are summarized as follows:

• **We study higher-level category detection in OVOD and construct hierarchical annotations for training and evaluation.** We associate existing instance annotations with the ancestor concepts of their concrete categories without additional bounding-box annotation, and establish evaluation protocols that distinguish categories seen and unseen during adaptation.

• **We propose a simple hierarchy-supervised approach to granularity-aware adaptation.** HARE uses the relative semantic ordering encoded by parent–child relations to supervise category-dependent routing over lightweight LoRA experts, guiding adaptation across semantic granularities.

• **We demonstrate improved higher-level detection across three benchmarks.** Experiments on COCO, LVIS-mini, and Objects365 show consistent gains in average box AP over Grounding DINO under both direct higher-level queries and matched subclass-augmented queries. HARE also maintains the pretrained detector's concrete-category AP on COCO.

2 Related Work

2.1 Open-Vocabulary Object Detection

Open-vocabulary object detection (OVOD) localizes objects beyond a fixed training vocabulary by transferring vision–language knowledge. Prior work expands detector vocabularies through image-level supervision (Zhou et al., 2022), detection-oriented prompt learning (Du et al., 2022), or region-level alignment (Zhong et al., 2022). Large-scale grounded detectors further connect detection with free-form text, including GLIP (Li et al., 2022), Grounding DINO (Liu et al., 2023), OWL-ViT (Minderer et al., 2022), and YOLO-World (Cheng et al., 2024). Despite strong novel-category and phrase grounding performance, these methods generally treat categories as a flat vocabulary and do not explicitly address changes in query granularity. We instead study whether the same instances can be reliably localized when a concrete query is replaced by a higher-level semantic category.

2.2 Semantic Hierarchies in Visual Recognition

Semantic hierarchies transfer information between concrete categories and their abstract ancestors. Prior work distills visual–language knowledge across representation levels (Ma et al., 2022), organizes open-world concepts in hyperbolic spaces (Kong et al., 2024), and, most directly, uses abstract and concrete category descriptions to construct hierarchy-aware classifiers for open-vocabulary detection (Liu et al., 2024). HARE shares the goal of improving detection across semantic granularities but uses the hierarchy differently: rather than only enriching category representations, it uses parent–child relations as relative supervision for allocating adaptation parameters. We further evaluate higher-level detection under different combinations of concrete and ancestor categories seen or unseen during adaptation.

2.3 Parameter-Efficient Adaptation and Mixture of Experts

Parameter-efficient adaptation updates a small parameter subset while keeping the pretrained backbone frozen; LoRA parameterizes such updates with low-rank matrices (Hu et al., 2022). Mixtures of adaptation modules, including AdaMix (Wang et al., 2022) and MOLE (Huang & Wei, 2024), increase conditional capacity by selecting or composing lightweight experts, while HMoRA combines task- and token-level routing in a hierarchical mixture of LoRA experts (Liao et al., 2025). These methods establish the effectiveness of expert-based adaptation, but their routing is designed for task, domain, or token specialization rather than ordered category granularity. HARE instead conditions routing on category phrases and directly supervises expert allocation with parent–child relations: a ranking objective imposes relative abstraction order without assigning fixed experts to semantic levels. An identity expert further provides an unmodified pretrained pathway for queries that require no update.

3 Task Formulation and Dataset Construction

3.1 Problem Formulation

Given an image I and textual category queries $\mathbf{T} = \{q_1, \ldots, q_M\}$, OVOD predicts bounding boxes, confidence scores, and query labels. We study *higher-level category detection*, in which a query denotes a higher-level semantic concept and the detector localizes instances from its concrete subclasses.

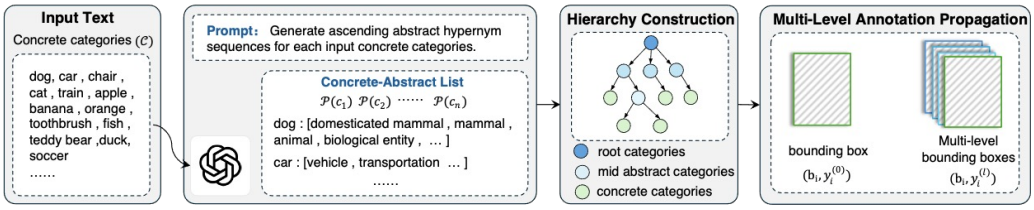


Figure 2: Hierarchy construction and annotation propagation. Each instance retains its original bounding box and receives the category labels along its fixed semantic path.

Let $\mathcal{C}$ denote the original concrete-category vocabulary. We associate each $c \in \mathcal{C}$ with a fixed semantic path

$$\mathcal{P}(c) = \left(c^{(0)}, c^{(1)}, \ldots, c^{(L_c)}\right), \tag{1}$$

where $c^{(0)} = c$, L_c is the number of upward steps, and each subsequent node is the immediate parent of the preceding node along the path. For example, *sheep* $\rightarrow$ *mammal* $\rightarrow$ *animal* $\rightarrow$ *living organism*. Our goal is to localize instances using ancestor concepts on their assigned paths as queries, while preserving concrete-category detection performance.

3.2 Hierarchy Construction and Annotation Propagation

The source-dataset categories define the finest granularity of the benchmark; we introduce only higher-level concepts above them. As shown in Figure 2, GPT-5 (OpenAI, 2025) generates one concrete-to-abstract path per category using a fixed prompt. The first cleaning stage normalizes concept names, merges synonyms, and removes relations that violate the taxonomic *is-a* criterion. We then merge the cleaned paths into a global hierarchy and resolve cycles in a second cleaning stage. The resulting hierarchy has two top-level groups: biological and non-biological entities. The finalized category paths are fixed before training. Prompts, cleaning rules, and cycle statistics are provided in Appendix A.

The adjacent parent–child relations in the finalized paths form

$$\mathcal{H} = \bigcup_{c \in \mathcal{C}} \left\{ \left(c^{(\ell+1)}, c^{(\ell)}\right) \;\middle|\; 0 \le \ell < L_c \right\}, \tag{2}$$

where each pair is ordered as (parent, child). Shared concepts may have multiple parents in the merged hierarchy, while annotation propagation remains restricted to each category's fixed path. We denote the concepts on this path by $\mathcal{A}_{\mathcal{P}}(c) = \{c^{(0)}, \ldots, c^{(L_c)}\}$. The training subset of $\mathcal{H}$ is used for hierarchy-supervised routing.

Given the original image annotations $\mathcal{D}_I = \{(\mathbf{b}_i, y_i)\}_{i=1}^{N_I}$, where $\mathbf{b}_i$ and $y_i \in \mathcal{C}$ denote the bounding box and concrete category of instance i, we construct

$$\widetilde{\mathcal{D}}_I = \bigcup_{i=1}^{N_I} \{(\mathbf{b}_i, a) \mid a \in \mathcal{A}_{\mathcal{P}}(y_i)\}. \tag{3}$$

Propagation augments semantic labels while preserving the original bounding boxes. Instances sharing an ancestor remain separate, providing multi-granularity annotations without additional bounding-box annotation.

3.3 Evaluation Protocols

We define three protocols according to whether the concrete category and its queried ancestor are seen during adaptation. Let $\mathcal{C}_{\mathrm{tr}}$ and $\mathcal{C}_{\mathrm{tr}}^{\mathrm{high}}$ denote the seen concrete and higher-level categories. Unseen categories are excluded from both router training and detector adaptation; thus, the term unseen is relative to adaptation, not base-detector pretraining. For the concrete–ancestor relations

$$\mathcal{R} = \{(c, h) \mid c \in \mathcal{C},\ h \in \mathcal{A}_{\mathcal{P}}(c) \setminus \{c\}\},$$

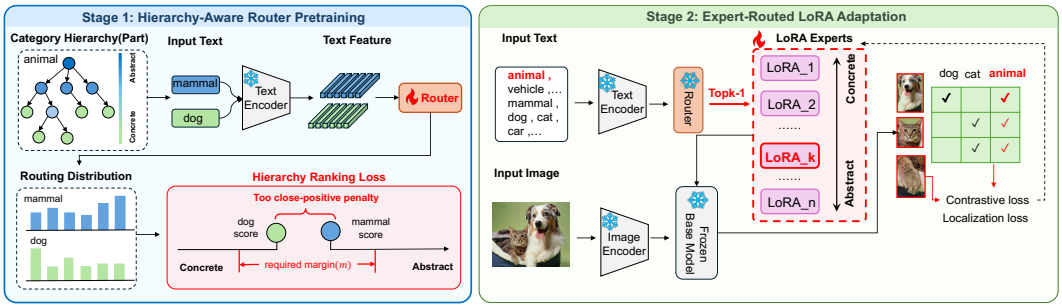


Figure 3: Overview of HARE. Parent–child relations supervise the relative abstraction scores of a text-conditioned router, guiding the selection of lightweight LoRA experts for category-dependent adaptation. The pretrained detector remains frozen.

we define

$$\begin{aligned} \mathcal{S}_{\mathrm{SS}} &= \{(c,h) \in \mathcal{R} \mid c \in \mathcal{C}_{\mathrm{tr}},\ h \in \mathcal{C}_{\mathrm{tr}}^{\mathrm{high}}\}, \\ \mathcal{S}_{\mathrm{US}} &= \{(c,h) \in \mathcal{R} \mid c \notin \mathcal{C}_{\mathrm{tr}},\ h \in \mathcal{C}_{\mathrm{tr}}^{\mathrm{high}}\}, \\ \mathcal{S}_{\mathrm{UU}} &= \{(c,h) \in \mathcal{R} \mid c \notin \mathcal{C}_{\mathrm{tr}},\ h \notin \mathcal{C}_{\mathrm{tr}}^{\mathrm{high}}\}. \end{aligned} \tag{4}$$

SS evaluates seen concrete categories under seen ancestors; US evaluates unseen concrete subclasses of seen ancestors; and UU evaluates cases in which both are unseen. In every protocol, the detector is queried and evaluated with h; c is used only to assign the evaluation setting.

4 Method

4.1 Overview

HARE uses category hierarchies to guide adaptation across semantic granularities. Specifically, parent–child relations provide supervision for allocating adaptation parameters across category queries. We implement this supervision through a text-conditioned router over lightweight LoRA experts in the text-enhancement pathway of Grounding DINO.

As illustrated in Figure 3, HARE is trained in two stages. We first train the router with a hierarchy-ranking objective that encourages higher abstraction scores for parent concepts than for their children. We then freeze the router and the pretrained detector and optimize the non-identity LoRA experts using detection supervision. During both detector adaptation and inference, each category query is assigned to the expert with the highest routing probability, which determines the update applied to its text features. The resulting features are used by the pretrained detector to predict bounding boxes and category scores.

4.2 Hierarchy-Supervised Adaptation

Parent–child relations provide relative supervision about semantic granularity: a parent concept is more abstract than its child. This supervision specifies relative abstraction order without requiring absolute granularity labels or predefined category-to-expert assignments. HARE uses these relations to guide a text-conditioned router over E experts, allowing category queries to select different adaptation parameters according to their semantic granularity.

Learning hierarchy-supervised routing. Let $\mathbf{h}_c$ denote the text representation of category query c. A lightweight MLP router produces a distribution over experts:

$$\mathbf{p}_c = \mathrm{softmax}\left(R_{\psi}(\mathbf{h}_c)\right), \tag{5}$$

where $p_{c,e}$ is the probability of assigning c to expert e. We assign ordered values to the expert indices and define the abstraction score as their expectation under the routing distribution:

$$a(c) = \sum_{e=0}^{E-1} (e+1) p_{c,e}. \tag{6}$$

Let $\mathcal{H}_{\mathcal{B}} \subseteq \mathcal{H}$ denote a mini-batch of distinct immediate parent–child relations drawn only from the fixed paths of the adaptation training split. We use only these adjacent edges for router supervision; non-adjacent ancestor–descendant pairs are not added as separate ranking constraints. If a shared concept has multiple parents in the merged DAG, each distinct immediate edge contributes one constraint, without choosing a canonical parent. We train the router using

$$\mathcal{L}_{\text{rank}} = \frac{1}{|\mathcal{H}_{\mathcal{B}}|} \sum_{(u,v)\in\mathcal{H}_{\mathcal{B}}} [m - a(u) + a(v)]_+ , \tag{7}$$

where u is the parent of v, $m > 0$ is the ranking margin, and $[x]_+ = \max(0, x)$. A relation incurs no loss once the parent score exceeds the child score by at least m. The objective therefore encourages relative ordering with a minimum separation across adjacent edges.

The expected score allows the ranking objective to be optimized through the routing distribution. During detection adaptation, we select the highest-probability expert from this distribution. The ranking objective thus supervises continuous abstraction scores, while Top-1 routing determines the discrete expert assignment; consistency between the resulting assignments and the hierarchy is evaluated empirically.

Detection adaptation with hierarchy-supervised routing. After hierarchy-supervised router training, we freeze the router and all pretrained detector parameters and optimize only the non-identity LoRA experts using the propagated annotations $\widetilde{\mathcal{D}}_I$ from the adaptation training split (Eq. 3). This keeps the hierarchy-trained routing parameters unchanged while the experts learn detection-specific updates from the concrete and ancestor labels on each instance category's fixed path.

We instantiate experts $1, \ldots, E-1$ as trainable low-rank updates and reserve expert 0 as an identity pathway. For a frozen projection $\mathbf{W}$, the expert updates are

$$\Delta\mathbf{W}_0 = \mathbf{0}, \qquad \Delta\mathbf{W}_e = \frac{\alpha}{r}\mathbf{B}_e\mathbf{A}_e, \quad e = 1, \ldots, E-1, \tag{8}$$

where r is the LoRA rank and α is the scaling factor. For query c, the selected expert and adapted output are

$$e^\star(c) = \arg\max_e p_{c,e}, \tag{9}$$

$$\mathbf{y}_c = \mathbf{W}\mathbf{x}_c + \Delta\mathbf{W}_{e^\star(c)}\mathbf{x}_c. \tag{10}$$

Here, $\mathbf{x}_c$ is the text feature associated with query c at the adapted projection. The same Top-1 selection rule is used during adaptation and inference. Gradients through each routed branch update only the selected trainable expert, so queries assigned to different experts use separate adaptation parameters at that projection. Selecting the identity pathway applies no additional update, yielding $\mathbf{y}_c = \mathbf{W}\mathbf{x}_c$.

4.3 Optional Subclass-Augmented Inference

HARE directly processes higher-level category queries without auxiliary subclass names ($K = 0$). We additionally evaluate an optional inference setting that supplements each higher-level query with up to K concrete subclasses. For each target category, GPT-5 (OpenAI, 2025) generates a fixed ordered list , guided by a prompt requesting common and visually recognizable subclasses. The higher-level query and selected subclass names are concatenated into a single text input as separately delimited category phrases, while routing is performed independently for each phrase.

Predictions from auxiliary subclass queries are mapped to their associated higher-level categories and combined with direct higher-level predictions using class-wise NMS (Neubeck & Van Gool, 2006). This procedure changes only the inference queries and post-processing, leaving hierarchy-supervised adaptation unchanged. We use $K = 0$ to evaluate direct higher-level detection and $K > 0$ to examine whether hierarchy-supervised adaptation remains beneficial with additional concrete-category cues. Under matched subclass augmentation, all compared methods use the same auxiliary lists and aggregation procedure.

Table 1: Comparison with existing methods on higher-level category detection under direct ($K = 0$) and subclass-augmented ($K = 6$) queries. Best results are bold and underlined; second-best results are bold.

		COCO				LVIS-mini				Objects365		
Setting	Method	SS	US	UU	**Avg.**	SS	US	UU	**Avg.**	US	UU	**Avg.**
$K = 0$	Grounding DINO	17.7	16.6	**13.1**	15.8	7.4	**12.2**	**10.2**	9.9	15.1	28.6	21.9
	Detic	11.2	17.7	6.5	11.8	6.0	9.2	6.3	7.2	10.4	16.0	13.2
	YOLO-World	**23.6**	**30.7**	13.0	**22.4**	**12.4**	**17.1**	7.2	**12.2**	**21.2**	**29.4**	**25.3**
	HARE	**31.3**	**26.8**	**14.9**	**24.3**	**15.4**	**17.1**	**12.5**	**15.0**	**20.6**	**31.6**	**26.1**
$K = 6$	Grounding DINO	32.0	32.3	34.4	32.9	**17.4**	**21.1**	21.9	**19.8**	**29.0**	**44.3**	**36.7**
	Detic+SHiNe	21.6	20.1	9.3	17.0	10.6	11.2	16.8	12.9	12.6	18.6	15.6
	YOLO-World	**34.8**	**34.7**	**35.2**	**34.9**	16.5	19.8	**22.6**	19.6	27.8	43.9	35.9
	HARE	**39.1**	**36.0**	**34.8**	**36.6**	**18.0**	**23.0**	**25.2**	**22.1**	**29.5**	**46.1**	**37.8**

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Datasets and evaluation protocols. We evaluate HARE on hierarchy-annotated versions of COCO (Lin et al., 2014), LVIS-mini (Gupta et al., 2019), and Objects365 (Shao et al., 2019), constructed following Section 3.2. Objects365 provides the adaptation training data and held-out categories for within-dataset evaluation, while COCO and LVIS-mini are used to assess cross-dataset transfer. Both adaptation stages use the same Objects365 training split: Stage 1 uses parent–child relations to supervise the router, and Stage 2 uses detection annotations to train the LoRA experts. We follow the SS, US, and UU protocols defined in Section 3.3, with category visibility determined across both stages. Dataset statistics and split construction are provided in Appendix A.

Evaluation metrics and inference settings. We report COCO-style box AP averaged over IoU thresholds from 0.50 to 0.95 for each available protocol. Objects365 UU includes ten higher-level evaluation categories. The reported Avg. is the arithmetic mean of AP across the available protocols for each dataset. We evaluate direct higher-level queries ($K = 0$) and subclass-augmented queries ($K > 0$). Comparisons under matched subclass augmentation use the same auxiliary subclass lists and prediction aggregation procedure.

Implementation details. HARE builds on Grounding DINO with a Swin-T image backbone (Liu et al., 2021) and a BERT-base text encoder (Devlin et al., 2019). We use $E = 10$ routing options, comprising nine trainable rank-16 LoRA experts and one identity pathway. The pretrained detector remains frozen throughout adaptation, and the router is frozen during Stage 2. Parameter counts and inference-time overhead are reported in Appendix B.1. Additional training and implementation details are provided in Appendix B.

5.2 MAIN RESULTS

We compare HARE with representative open-vocabulary detectors under direct higher-level queries ($K = 0$) and subclass-augmented queries ($K = 6$). At $K = 6$, we additionally compare with Detic+SHiNe (Liu et al., 2024), a hierarchy-aware detection baseline. Table 1 summarizes the results across the three datasets.

With higher-level queries alone ($K = 0$), HARE achieves the highest average AP among the compared methods on all three datasets, improving over Grounding DINO by 8.5, 5.1, and 4.2 AP points on COCO, LVIS-mini, and Objects365, respectively. It also achieves the highest UU AP on each dataset, supporting generalization to concrete categories and queried ancestors unseen during adaptation. Since HARE is adapted only on Objects365, the results on COCO and LVIS-mini further demonstrate cross-dataset transfer.

At $K = 6$, HARE retains the highest average AP on all three datasets, exceeding Grounding DINO by 3.7, 2.3, and 1.1 AP points, respectively. It also outperforms Detic+SHiNe in average AP across the three datasets, providing a comparison with an existing method that incorporates semantic hierar-

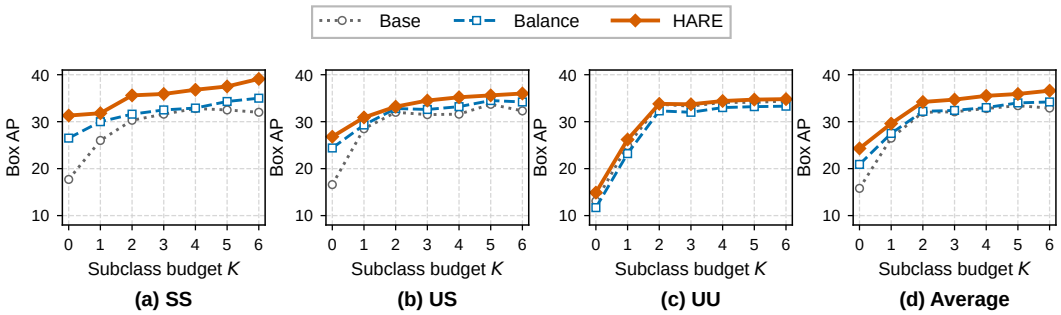


Figure 4: Effect of router supervision and subclass budget K on COCO. Hierarchy-supervised routing (HARE) consistently outperforms the frozen baseline (Base) and load-balancing routing (Balance) across SS, US, UU, and average AP.

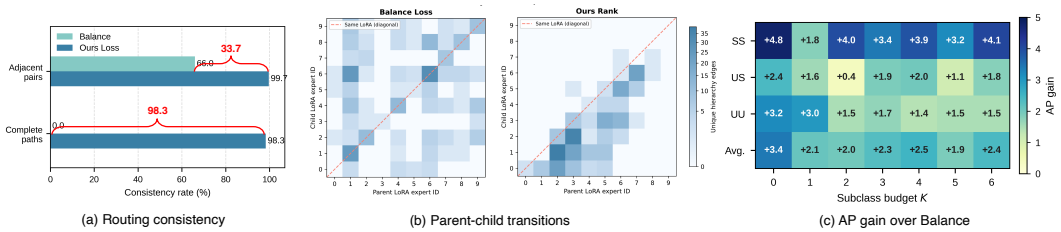


Figure 5: Analysis of hierarchy-supervised routing. (a) Adjacent parent–child and complete-path ordering consistency; red braces mark the improvements of HARE over Balance. (b) Parent–child expert transitions; assignments satisfying $e_{\text{parent}} > e_{\text{child}}$ are hierarchy-consistent. (c) AP gains of HARE over Balance across subclass budgets K.

chy information. These results show that HARE remains effective when auxiliary concrete-category cues are available.

5.3 Ablation Studies

Effect of hierarchy-supervised routing. To isolate the contribution of hierarchy supervision, we compare HARE with Balance, which replaces the hierarchy-ranking objective with conventional load-balancing supervision while retaining the same router, expert architecture, and adaptation procedure. At $K = 0$, HARE improves average AP from 20.9 to 24.3, with gains of 4.8, 2.4, and 3.2 AP points on SS, US, and UU, respectively (Figure 4). In contrast, Balance falls below the frozen baseline on UU. HARE remains ahead of Balance across all evaluated subclass budgets (Figure 5(c)), showing that the benefit of hierarchy supervision persists when auxiliary subclass cues are introduced. These controlled results demonstrate that parent–child supervision provides more effective routing guidance than load-balancing supervision under the same architecture and adaptation procedure.

We next examine whether the detection gain is accompanied by hierarchy-consistent routing. Adjacent-pair consistency measures whether each parent is assigned an expert index no smaller than that of its child, whereas the stricter complete-path consistency requires this ordering to hold at every edge along a semantic path. As shown in Figure 5(a), HARE raises adjacent-pair consistency from 66.0% to 99.7% relative to Balance, an absolute improvement of 33.7 percentage points. On complete paths, consistency increases more markedly from 0.0% to 98.3%. These near-perfect scores indicate that the hierarchy-ranking objective encourages discrete expert assignments that follow the semantic ordering of the hierarchy. The transition heatmaps in Figure 5(b) visualize the distribution of parent–child relations across expert pairs. Transitions satisfying $e_{\text{parent}} > e_{\text{child}}$ are hierarchy-consistent, so a larger proportion of such transitions indicates stronger ordering consistency. Balance assigns substantial mass to pairs that violate this order, whereas HARE concentrates transitions among hierarchy-consistent pairs. This pattern is consistent with the quantitative re-

Table 2: Comparison of adaptation strategies on COCO under direct ($K = 0$) and subclass-augmented ($K = 6$) queries. Among configurations that preserve pretrained knowledge, the best results are bold and underlined, and second-best results are bold. The light-red row marked by † denotes severe catastrophic forgetting; it is reported for completeness but excluded from the ranking.

		SS		US		UU	
Method	Concrete	$K = 0$	$K = 6$	$K = 0$	$K = 6$	$K = 0$	$K = 6$
Grounding DINO	**<u>56.9</u>**	17.7	32.0	16.6	32.3	**13.1**	**34.4**
Single LoRA	**51.6**	**26.9**	**38.7**	**25.3**	**<u>36.4</u>**	11.5	34.1
Full fine-tuning†	16.9	37.5	42.2	29.4	34.9	4.2	13.9
HARE	**<u>56.9</u>**	**<u>31.3</u>**	**<u>39.1</u>**	**<u>26.8</u>**	**36.0**	**<u>14.9</u>**	**<u>34.8</u>**

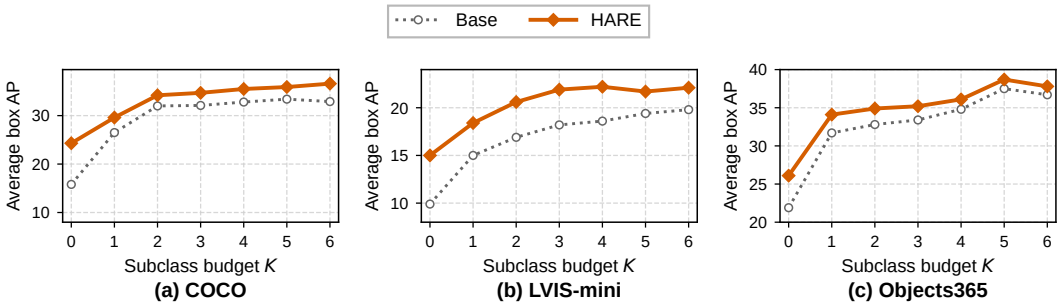


Figure 6: Average box AP across subclass budgets K on COCO, LVIS-mini, and Objects365. HARE consistently outperforms the frozen Grounding DINO baseline (Base) at every matched budget.

sults in Figure 5(a), showing that hierarchy supervision guides expert selection toward the expected parent–child order.

Comparison of adaptation strategies. Table 2 compares adaptation strategies on COCO. Full fine-tuning achieves the highest SS AP at both subclass budgets and the highest US AP at $K = 0$, but these gains come with substantial degradation in concrete-category and UU performance. Concrete-category AP decreases from 56.9 to 16.9, while UU AP drops from 13.1 to 4.2 at $K = 0$ and from 34.4 to 13.9 at $K = 6$. This imbalance shows that adaptation to seen higher-level categories does not necessarily preserve concrete-category performance or generalization to unseen categories. Single LoRA causes a smaller decline but still reduces concrete-category AP by 5.3. In contrast, HARE preserves the original concrete-category AP of 56.9 while outperforming Single LoRA by 4.4, 1.5, and 3.4 AP on SS, US, and UU at $K = 0$, respectively. HARE therefore provides a better balance between higher-level adaptation and concrete-category performance preservation on this benchmark.

Effect of the subclass budget K. Figure 6 evaluates average AP for $K \in \{0, \ldots, 6\}$ on all three benchmarks. At $K = 0$, HARE improves average AP over the frozen detector by 8.5, 5.1, and 4.2 points on COCO, LVIS-mini, and Objects365, respectively, confirming that its benefit does not depend on auxiliary subclass prompts. Increasing K generally improves both methods by supplying concrete visual cues, with most gains occurring at small budgets and then saturating. HARE remains better than the frozen detector for every dataset and matched K, showing that hierarchy-supervised adaptation and subclass augmentation are complementary rather than interchangeable.

6 Conclusion

We study open-vocabulary object detection across semantic granularities and identify a cross-granularity alignment gap for higher-level category queries. We propose HARE, which combines hierarchy-aware routing with multiple LoRA experts for granularity-aware adaptation. Experiments on COCO, LVIS-mini, and Objects365 demonstrate consistent improvements under both direct and subclass-augmented queries while preserving concrete-category detection.

AI USE STATEMENT

We used generative AI tools, including GPT-5 and related language-model assistants, during dataset construction, code development, literature review, and manuscript preparation. GPT-5 was used to generate and clean candidate concrete-to-abstract semantic paths, assist with resolving inconsistent taxonomic relations and cycles, and generate the fixed subclass lists used in the optional subclass-augmented inference setting. The corresponding prompts and validation procedures are documented in Appendix A.

All AI-assisted outputs were reviewed and validated by the authors. AI-generated code was reviewed and tested, literature-related content was checked against original sources where relevant, and the authors take responsibility for all AI-assisted content in this work.

ETHICS STATEMENT

This work studies open-vocabulary object detection using COCO, LVIS, and Objects365 and does not involve human subjects or the collection of new personally identifiable information. Our hierarchy-annotated benchmarks are derived from existing object-detection annotations: the original bounding boxes are preserved, while category labels are propagated to higher-level concepts along fixed semantic paths. Generative AI is used to assist the construction of these semantic hierarchies. Because automatically generated taxonomies may contain linguistic, cultural, or semantic biases, we apply explicit taxonomic constraints and consistency checks and document the construction prompts and cleaning procedure to make these choices inspectable. Nevertheless, the resulting hierarchies should be understood as one operational taxonomy for evaluating semantic-granularity generalization rather than as a universal or exhaustive ontology. As with general-purpose object-detection systems, the proposed method could potentially be deployed in applications involving monitoring or surveillance; such deployments may raise privacy, fairness, or misuse concerns beyond the scope of the benchmark experiments considered here.

REPRODUCIBILITY STATEMENT

We provide the information required to reproduce the principal components of this work throughout the main paper and appendix. The task definition, hierarchy construction, annotation-propagation procedure, and SS/US/UU evaluation protocols are described in Section 3. The HARE routing objective, abstraction score, Top-1 expert selection, and LoRA adaptation procedure are specified in Section 4.2. Dataset statistics, hierarchy-generation and cleaning prompts, cycle-resolution procedure, and evaluation split construction are provided in Appendix A. Model architecture, router representation, optimization settings, LoRA placement, training objectives, and other implementation details are reported in Appendix B. We additionally report computational overhead and controlled ablations over routing supervision, adaptation strategy, number of experts, and subclass budget to facilitate verification of the main empirical findings.

REFERENCES

Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. YOLO-World: Real-time open-vocabulary object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16901–16911, 2024.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4171–4186, 2019. doi: 10.18653/v1/N19-1423.

Yu Du, Fangyun Wei, Zihe Zhang, Miaojing Shi, Yue Gao, and Guoqi Li. Learning to prompt for open-vocabulary object detection with vision-language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14084–14093, 2022.

Agrim Gupta, Piotr Dollar, and Ross Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5356–5364, 2019.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Shaohan Huang and Furu Wei. Mixture of LoRA experts. In *International Conference on Learning Representations*, 2024.

Fanjie Kong, Yanbei Chen, Jiarui Cai, and Davide Modolo. Hyperbolic learning with synthetic captions for open-world detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16762–16771, 2024.

Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, Kai-Wei Chang, and Jianfeng Gao. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10965–10975, 2022.

Mengqi Liao, Wei Chen, Junfeng Shen, Shengnan Guo, and Huaiyu Wan. HMoRA: Making LLMs more effective with hierarchical mixture of LoRA experts. In *International Conference on Learning Representations*, 2025.

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 *European Conference on Computer Vision*, pp. 740–755, 2014.

Mingxuan Liu, Tyler L. Hayes, Elisa Ricci, Gabriela Csurka, and Riccardo Volpi. SHiNe: Semantic hierarchy nexus for open-vocabulary object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16634–16644, 2024.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022, 2021.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Zongyang Ma, Guan Luo, Jin Gao, Liang Li, Yuxin Chen, Shaoru Wang, Congxuan Zhang, and Weiming Hu. Open-vocabulary one-stage detection with hierarchical visual-language knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14074–14083, 2022.

Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, Xiao Wang, Xiaohua Zhai, Thomas Kipf, and Neil Houlsby. Simple open-vocabulary object detection with vision transformers. In *European Conference on Computer Vision*, 2022.

Alexander Neubeck and Luc Van Gool. Efficient non-maximum suppression. In *Proceedings of the 18th International Conference on Pattern Recognition*, volume 3, pp. 850–855, 2006. doi: 10.1109/ICPR.2006.479.

OpenAI. GPT-5 system card. Technical report, OpenAI, 2025. URL `https://cdn.openai.com/gpt-5-system-card.pdf`.

Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8430–8439, 2019.

Yaqing Wang, Sahaj Agarwal, Subhabrata Mukherjee, Xiaodong Liu, Jing Gao, Ahmed Hassan Awadallah, and Jianfeng Gao. AdaMix: Mixture-of-adaptations for parameter-efficient model tuning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 5744–5760, 2022.

Yiwu Zhong, Jianwei Yang, Pengchuan Zhang, Chunyuan Li, Noel Codella, Liunian Harold Li, Luowei Zhou, Xiyang Dai, Lu Yuan, Yin Li, and Jianfeng Gao. RegionCLIP: Region-based language-image pretraining. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16793–16803, 2022.

Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *European Conference on Computer Vision*, 2022.

A Dataset Details and Evaluation Splits

A.1 Datasets

We construct hierarchy-annotated benchmarks from COCO (Lin et al., 2014), LVIS-mini (Gupta et al., 2019), and Objects365 (Shao et al., 2019). Objects365 provides the training data used to adapt HARE, while COCO and LVIS-mini are used for cross-dataset transfer evaluation. We additionally evaluate held-out categories from Objects365 to measure generalization within the source dataset.

Table 3 summarizes the dataset statistics used in our experiments.

A.2 Hierarchy Construction

For each dataset, the original category vocabulary is provided to GPT-5 together with a fixed prompt requesting one complete concrete-to-abstract chain per category. A first GPT-5 cleaning stage normalizes names, merges synonymous nodes, filters relations that do not satisfy the taxonomic *is-a* criterion, and removes leaf re-entry. We then merge the cleaned paths and use strongly connected component analysis to detect global cycles. Each remaining cyclic component and its source paths are submitted to a second GPT-5 cleaning stage. The proposed edge revisions are applied and the complete graph is checked again until no repeated-node path or directed cycle remains.

A.2.1 Prompt Templates

The original API message formatting was not retained. For reproducibility, we report below the complete normalized prompt templates corresponding to the instructions used in the three construction steps. Bracketed fields denote the dataset-specific inputs. All category names and returned paths are in English.

Hierarchy-generation prompt.

> You are constructing a taxonomic hierarchy for an object-detection dataset. The input is the complete list of original dataset categories: `[CATEGORY LIST]`. For every input category, return one complete semantic path ordered from the original category to progressively more abstract categories. Treat every original category as a leaf and do not introduce a finer subclass below it. Every adjacent pair must express an intrinsic *is-a* relation: the preceding concept is a kind of the following concept. Use concise, singular, lower-case English noun phrases and reuse the same name for the same concept across categories. Exclude functional or affordance-based relations, attributes, materials, locations, and part–whole relations. For example, a horse is an animal; it is not a transportation category merely because it can be ridden. End each path in the appropriate broad biological or non-biological group. Return only a JSON object that maps each original category to an ordered list of its ancestors, from the nearest parent to the most abstract category. Include every input category exactly once and do not add dataset categories that are absent from the input list.

Semantic-cleaning prompt.

> You are cleaning category paths for a taxonomic object hierarchy. The original leaf vocabulary is `[ORIGINAL CATEGORY LIST]`, and the generated paths are `[GENERATED PATHS]`. Rewrite the paths using only intrinsic *is-a* relations. Normalize capitalization, plurality, spelling, and noun phrase form; map synonymous concepts to one canonical name; and reuse that name consistently in all paths. Remove functional, affordance-based,

Table 3: Statistics of the hierarchy-annotated COCO, LVIS-mini, and Objects365 benchmarks.

Dataset	Concrete categories	Higher-level categories	Maximum depth	Training images	Test images
COCO	80	91	9	–	4,952
LVIS-mini	804	319	9	–	4,483
Objects365	365	198	9	41,591	15,663

Table 4: Cycle statistics during construction of the Objects365 hierarchy. A cyclic SCC is an SCC containing at least two nodes (or a self-loop); cycle nodes and cycle edges are the unions of the nodes and directed edges occurring in at least one simple directed cycle. Counts are computed after deduplicating identical directed edges.

Construction stage	Cyclic SCCs	Simple cycles	Cycle nodes	Cycle edges
Raw GPT-5 paths	2	3	5	6
Stage-1 cleaned and merged graph	6	8	17	19
Final graph after Stage 2	0	0	0	0

attribute, material, contextual, and part–whole relations. Do not create a category finer than an original leaf. If an original leaf occurs later in its own ancestor path, remove that occurrence and the invalid segment that causes the leaf to be its own ancestor. Retain distinct valid parents when they express compatible taxonomic memberships. Preserve the original leaf keys and return only a JSON object mapping each leaf to its cleaned, concrete-to-abstract ancestor list. Every adjacent pair in the returned paths must be interpretable as "is a kind of."

Cycle-resolution prompt.

You are resolving a directed cycle in a merged taxonomic hierarchy. Edges are directed from a child to its immediate parent. The cyclic strongly connected component is `[SCC NODES]`, its directed relations are `[SCC EDGES]`, and the category paths that produced those relations are `[SOURCE PATHS]`. Determine which relations satisfy an intrinsic *is-a* test. Remove or revise the smallest set of invalid or directionally inconsistent relations needed to break every directed cycle. Reject functional, affordance-based, attribute, material, contextual, and part–whole relations. Merge synonymous node names when this is the source of the inconsistency, but do not merge concepts with different meanings. Do not make an original leaf its own ancestor, and preserve multiple parents when they are taxonomically valid and do not recreate a cycle. Return a JSON object with three fields: `remove_edges`, `add_edges`, and `canonical_renames`. Represent every edge as `[parent, child]`. Return only the JSON object.

Table 4 reports the cycle statistics for Objects365 at each construction stage. The raw GPT-5 output contained two cyclic strongly connected components (SCCs). After semantic normalization and merging all category-specific paths, six cyclic SCCs were exposed because relations from different paths interacted in the global graph. The second cleaning stage resolved all of them; the finalized Objects365 training graph contains neither repeated-node paths nor directed cycles.

We compute these values by representing every adjacent concept pair along a concrete-to-abstract path as a directed parent-to-child edge, enumerating SCCs and simple directed cycles, and repeating the check after every revision. The statistics concern cycle resolution only and do not imply exhaustive manual review or relabeling of all hierarchy edges.

The resulting hierarchies are finalized before model training and remain unchanged throughout training and evaluation. Access to category labels and parent–child relations during training is restricted according to the evaluation splits.

A.3 Evaluation Split Construction

We construct the evaluation splits according to the three hierarchical generalization protocols defined in Section 3.3. Category visibility is determined with respect to the Objects365 training vocabulary.

Table 5: Statistics of the SS, US, and UU evaluation splits.

Dataset	Protocol	Concrete categories	Higher-level categories	Images
COCO	SS	26	53	923
COCO	US	24	35	2,704
COCO	UU	30	25	2,422
LVIS-mini	SS	83	48	2,362
LVIS-mini	US	74	36	2,052
LVIS-mini	UU	263	44	2,853
Objects365	US	106	53	15,663
Objects365	UU	64	10	9,835

Table 6: Computational overhead of HARE relative to the frozen Grounding DINO baseline under the same inference configuration.

Metric	Base	HARE	Change
Total parameters (M)	172.978	175.961	+2.983 (1.72%)
Top-1 active parameters (M)	172.978	173.307	+0.329 (0.19%)
Latency (ms)	257.44	299.86	+42.42 (16.48%)
Throughput (FPS)	3.88	3.33	-14.15%
Peak memory (MiB)	1226.25	1238.29	+12.04 (0.98%)

Specifically, SS contains cases in which both the concrete category and its corresponding higher-level category are observed during training. US evaluates unseen concrete categories associated with higher-level categories observed during training, while UU evaluates cases in which both the concrete and higher-level categories are unseen.

Table 5 summarizes the resulting evaluation splits.

COCO and LVIS-mini are partitioned according to the same visibility criteria, providing cross-dataset evaluation under consistent generalization settings. For Objects365, held-out categories are used to evaluate generalization within the source dataset. To ensure that higher-level evaluation categories represent meaningful semantic groups rather than one-to-one relabelings of individual concrete categories, we retain only higher-level categories supported by multiple concrete subcategories.

B Additional Implementation and Efficiency Details

Router representation and architecture. For a category phrase c, we locate its WordPiece span in the BERT text sequence, discard special and delimiter tokens, and average the projected token features in that span to obtain $\mathbf{h}_c \in \mathbb{R}^{256}$. Multi-token category names are therefore represented by mean pooling over all of their constituent tokens rather than by the `[CLS]` token. The router is a two-layer MLP, $256 \to 128 \to E$, with a ReLU activation between the two linear layers, followed by a temperature-scaled softmax. With $E = 10$, it has 34,186 trainable parameters, including biases.

Stage-1 router training. Both stages use the same Objects365 adaptation split and vocabulary, and categories designated as unseen are excluded from both stages. Stage 1 freezes the detector and text encoder and trains only the router for 7 epochs using AdamW with a learning rate of 1×10^{-3}, weight decay 10^{-4}, and batch size 32. We set the ranking margin to $m = 1$ and optimize only the hierarchy-ranking loss in Eq. equation 7; no load-balancing loss is used.

Category names and relations are normalized before batching. For each training sample, adjacent parent–child edges are collected from the finalized paths, self-relations and edges whose endpoints are absent from the text input are removed, and repeated ordered pairs are deduplicated. All remaining distinct edges in the sample contribute to the mean ranking loss; we do not add non-adjacent ancestor pairs or apply a separate random edge sampler.

LoRA placement and routed queries. HARE uses $E = 10$ experts with rank $r = 16$, scaling factor $\alpha = 32$, and LoRA dropout 0.05. The adapters replace the callable linear projections under all

Table 7: Effect of the total number of routing experts E under direct ($K = 0$) and subclass-augmented ($K = 6$) queries. E includes one identity pathway; Base is the frozen detector without routing or LoRA adaptation. The best value in each column within each K block is bold.

	COCO				LVIS-mini			
Setting	SS	US	UU	Avg.	SS	US	UU	Avg.
				$K = 0$				
Base	17.7	16.6	13.1	15.8	7.4	12.2	10.2	9.9
$E = 6$	27.9	25.1	12.7	21.9	12.1	14.7	11.3	12.7
$E = 8$	27.4	25.8	**15.9**	23.0	11.5	14.2	**14.0**	13.2
$E = 10$	**31.3**	**26.8**	14.9	**24.3**	**15.4**	**17.1**	12.5	**15.0**
				$K = 6$				
Base	32.0	32.3	34.4	32.9	17.4	21.1	21.9	19.8
$E = 6$	38.0	**38.1**	**35.8**	**37.3**	17.3	22.5	**26.6**	**22.1**
$E = 8$	37.4	37.4	33.3	36.0	17.4	21.6	25.5	21.5
$E = 10$	**39.1**	36.0	34.8	36.6	**18.0**	**23.0**	25.2	**22.1**

six Grounding DINO text-enhancement encoder layers (`encoder.text_layers`). One router is shared by all adapted projections. In Stage 2, the pretrained detector and router are frozen and only the routed non-identity LoRA parameters are optimized.

For each image, all concrete and propagated ancestor labels are included as positive category phrases in one dot-delimited text sequence, together with up to 85 sampled negative categories, subject to a maximum length of 256 BERT tokens. Character spans are converted to token spans, and each instance box is linked through the positive map to its concrete label and all ancestor labels on its fixed path. These labels participate simultaneously in a single Grounding DINO detection loss; no hierarchy level is sampled separately.

Stage-2 detection objective. Stage 2 is trained for two epochs with AdamW (Loshchilov & Hutter, 2019), learning rate 1×10^{-5}, batch size 4, and weight decay 10^{-4}. The implementation calls the original Grounding DINO detection head and retains its matching, classification, box regression, IoU, auxiliary decoder/encoder, and denoising losses. No additional hierarchy loss is applied in Stage 2. All propagated labels participate simultaneously through the shared positive map, and all Grounding DINO detection-loss terms

and their original weights are retained unchanged.

B.1 Computational Overhead

Table 6 compares HARE with the frozen Grounding DINO baseline under the same inference configuration. HARE stores multiple low-rank experts, but Top-1 routing activates only one expert for each category phrase.

Relative to the baseline, HARE adds 1.72% total parameters but only 0.19% Top-1 active parameters and 0.98% peak memory. Latency increases by 16.48%, while direct higher-level COCO AP improves from 15.8 to 24.3 (Table 8).

C Additional Ablation Analysis

C.1 Effect of the Number of Routing Experts

Table 7 varies the total number of routing experts E, including one identity pathway and $E - 1$ trainable LoRA experts, while holding the training protocol and all remaining hyperparameters fixed. Results are reported for direct higher-level queries ($K = 0$) and subclass-augmented queries ($K = 6$), isolating whether additional adaptation capacity alone explains the gains of HARE.

At $K = 0$, $E = 10$ obtains the highest average AP on both datasets, although $E = 8$ performs best on UU. At $K = 6$, the dependence on expert count is non-monotonic: $E = 6$ gives the highest COCO average AP and ties $E = 10$ on LVIS-mini average AP. More experts therefore do not

Table 8: Effect of the subclass budget K on COCO. For each method and protocol, the best result across K is bold.

	SS		US		UU		Avg.	
K	Base	HARE	Base	HARE	Base	HARE	Base	HARE
0	17.7	31.3	16.6	26.8	13.1	14.9	15.8	24.3
1	26.0	31.8	28.5	30.9	24.9	26.2	26.5	29.6
2	30.3	35.6	32.0	33.2	33.7	33.8	32.0	34.2
3	31.6	35.9	31.5	34.5	33.3	33.7	32.1	34.7
4	**32.8**	36.8	31.6	35.2	34.1	34.4	32.8	35.5
5	32.5	37.5	**33.7**	35.6	34.1	34.7	**33.4**	35.9
6	32.0	**39.1**	32.3	**36.0**	**34.4**	**34.8**	32.9	**36.6**

uniformly improve performance, especially when subclass cues already provide additional semantic information.

This capacity control complements the Balance comparison in Section 5.3. Balance and HARE use the same router, expert architecture, and adaptation procedure, but only HARE uses parent–child hierarchy supervision for routing. HARE consistently outperforms Balance, whereas increasing E alone yields inconsistent gains. Together, these results show that HARE's improvement cannot be attributed solely to the number of LoRA experts. We use $E = 10$ in the main experiments because it provides the strongest average performance in the primary direct-query setting.

C.2 Effect of the Subclass Budget

We vary the number of auxiliary concrete subclasses from $K = 0$ to $K = 6$, using the same subclass lists and prediction aggregation rule for Base and HARE. Results on COCO, LVIS-mini and Objects365 are presented in Table 8, Table 9, and Table 10, respectively.

Table 9: Effect of the subclass budget K on LVIS-mini. For each method and protocol, the best result across K is bold.

	SS		US		UU		Avg.	
K	Base	HARE	Base	HARE	Base	HARE	Base	HARE
0	7.4	15.4	12.2	17.1	10.2	12.5	9.9	15.0
1	15.3	19.0	14.8	18.0	14.9	18.3	15.0	18.4
2	17.5	19.9	15.0	18.4	18.3	23.5	16.9	20.6
3	**18.7**	**20.4**	17.1	20.5	18.9	24.6	18.2	21.9
4	16.2	17.7	20.3	**23.5**	19.4	**25.3**	18.6	**22.2**
5	16.0	17.3	20.5	22.6	21.7	25.2	19.4	21.7
6	17.4	18.0	**21.1**	23.0	**21.9**	25.2	**19.8**	22.1

Table 10: Effect of the subclass budget K on Objects365. For each method and protocol, the best result across K is bold.

	US		UU		Avg.	
K	Base	HARE	Base	HARE	Base	HARE
0	15.1	20.6	28.6	31.6	21.9	26.1
1	24.0	26.9	39.3	41.3	31.7	34.1
2	24.7	26.0	40.9	43.7	32.8	34.9
3	26.1	27.2	41.0	43.3	33.4	35.2
4	27.4	27.9	42.2	44.3	34.8	36.1
5	**30.2**	**30.8**	**44.7**	**46.5**	**37.5**	**38.7**
6	29.0	29.5	44.3	46.1	36.7	37.8

HARE consistently outperforms Base across all datasets, evaluation protocols, and values of K, which verifies the efficacy of our hierarchy-aware routing supervision. The largest average performance gains appear at $K = 0$. Under this setting without any auxiliary subclass cues, improvements are purely attributed to hierarchical supervision, directly demonstrating its standalone effectiveness. As K increases, abundant semantic information brought by subclass candidates significantly boosts the baseline performance. Though the relative performance margin shrinks, HARE still maintains steady superiority over Base for all tested K.