SHiNe: Semantic Hierarchy Nexus for Open-vocabulary Object Detection

Supplementary Material

Appendix

In this appendix, we begin by detailing the detection datasets in App. A. Then, App. B delves into the process of synthetic semantic hierarchy generation using LLMs, providing the LLM prompts and a thorough summary of the generated hierarchies' statistics. We provide in App. C additional implementation specifics of SHiNe. We present in App. D an extended analysis of SHiNe's components on the FSOD dataset. App. F and App. G extend the main detection experiments, offering comprehensive summary statistics. App. I includes additional investigation of SHiNe's performance on the COCO and LVIS datasets. Finally, App. J showcases some qualitative detection results. Our code is publicly available at https://github.com/naver/shine.

A. Dataset Details

Table 7. Summary of datasets. FSOD [10] and iNatLoc [6] offer three and six levels of label hierarchies, respectively, with varying semantic granularity. This feature allows evaluating models on these datasets at different granularity level in our experiments. To evaluate, for FSOD, we use its official test split, and for iNatLoc, we use the combined test and validation splits.

	FSOD (test split)			iNatLoc (test + val splits)					
# of Levels	3			6					
# of Classes	L3	L2	L1	L6	L5	L4	L3	L2	L1
per Level	200	46	15	500	317	184	64	18	5
# of Images	14152			25000					
# of BBoxes	35102			25000					

In Tab. 7, we present a comprehensive summary of the two detection datasets used in our evaluation, FSOD [10] and iNatLoc [6], under the cross-dataset transfer open-vocabulary evaluation protocol. Given that the original FSOD dataset [10] provides only a two-level hierarchy, we manually constructed the L1 level of the label hierarchy (the most abstract one) for a more comprehensive evaluation, under which we grouped the categories of level L2 (which corresponds to the upper level category in the original label space). The L1 level consists of the following 15 label categories:

```
{"liquid", "instrument", "food", "art",
"plant", "component", "animal", "body",
"wearable item", "infrastructure",
"vehicle", "furnishing", "fungi",
"equipment", "beauty product"}
```

For FSOD and iNatLoc, we use their official test splits in our experiments, respectively. Notably, FSOD and iNatLoc offer three and six levels of label hierarchies, respectively, each characterized by distinct semantic granularities. Specifically, for the same set of evaluation images and their associated box-label annotations, the actual *label* used for evaluation can be mapped to different linked labels at each granularity level of the hierarchy. For example, in FSOD, a box region labeled as `"watermelon"` at the L3-level could be mapped to label `"fruit"` at the L2-level or `"food"` at the L1-level. This hierarchical approach to labeling facilitates the evaluation of these datasets at various granularity levels. See the annotation files in our codebase.

Evaluation level. During the evaluation, we consider only one hierarchy level at a time, where the label class vocabulary corresponding to the evaluation level of granularity serves as the target test (user-defined) vocabulary for both the methods being compared and our proposed SHiNe. This means that model has to assign labels solely from the given hierarchy level.

B. Semantic Hierarchy Generation

Being a hierarchy-based method, validating SHiNe's effectiveness with various hierarchy sources is crucial. In real-world applications, an ideal semantic hierarchy for the target data might not always be available. Therefore, our study focuses on evaluating SHiNe using not only the dataset-specific class taxonomies [6, 7, 10] (the ground-truth hierarchies provided by the datasets as described in Sec. A) but also hierarchies synthesized for the target test vocabulary via large language models (LLMs). Our key idea is that encyclopedic textual information about semantic categories is readily available on the Internet. Contemporary LLMs like ChatGPT [39], trained on vast internet-scale corpora, inherently encode the necessary semantic class taxonomic information in their weights. Similar to the approach used in CHiLS [38], we employ an LLM to automatically generate simple three-level semantic hierarchies for the target vocabularies. We use the ChatGPT [39] gpt-3.5-turbo model as our LLM via its public API to generate the synthetic semantic hierarchy with a temperature parameter of 0.7. See our codebase for the generated hierarchies.

In the following subsections, we first detail the process of prompting LLMs to generate hierarchies (Sec. B.1) and then summarize the statistics of the hierarchies we generated (Sec. B.2).

B.1. Prompting LLMs

In scenarios where a ground-truth hierarchy is unavailable, and given a label vocabulary $\mathcal{C}^{\text{test}}$ representing the target Classes of Interest (CoIs) at a specific granularity level of the evaluation dataset, the true super-/sub-categories for each CoI are *unknown*. To generate a simple 3-level hierarchy for $\mathcal{C}^{\text{test}}$, we first use ChatGPT [39] to generate a list of super-categories for each CoI $c \in \mathcal{C}^{\text{test}}$ using the following **super-category prompt**:

```
Generate a list of p super-categories that
the following [context] object belongs to
and output the list separated by '&':  c
```

where $p = 3$. Subsequently, following Novack *et al.* [38], for each CoI $c \in \mathcal{C}^{\text{test}}$, we query ChatGPT [39] to generate a list of sub-categories using the following **sub-category prompt**:

Table 8. Summary statistics of synthetic hierarchies generated by the LLM for our experiments. We present the number of label classes in the target vocabulary, the total number of generated super-categories and sub-categories, and the average number of generated super-categories and sub-categories per target class (CoI) for each dataset at each label vocabulary level. Additionally, links are provided to the experiments using these LLM-generated hierarchies. Note: N/A indicates that only one level of label vocabulary exists in the dataset. †: At the most abstract (coarsest) level L1 of iNatLoc, all target classes belong to the single super-category Kingdom `"Animalia"`.

Dataset	Corresponding Experiments	Label Vocabulary Level	Number of Label Classes	Number of Super-categories		Number of Sub-categories	
				Total	Avg. per class	Total	Avg. per class
iNatLoc [6]	Tab. 3, 4, 9, 10, 11	L6	500	317	0.6	10909	21.8
		L5	317	184	0.6	6675	21.1
		L4	184	64	0.3	3102	16.9
		L3	64	18	0.3	1018	16.7
		L2	18	5	0.3	273	15.2
		L1	5	1†	0.2	63	12.6
	Fig. 4(a)	Mis-spe. L6	1966	7585	3.9	35151	17.9
FSOD [10]	Tab. 3, 4, 9, 10, 11	L3	200	1298	6.5	4140	20.7
		L2	46	295	6.4	702	15.2
		L1	15	92	6.1	239	15.9
	Fig. 4(b)	Mis-spe. L3	1570	8069	5.1	26684	17.0
COCO [30]	Tab. 12	N/A	65	395	6.1	1303	20.1
LVIS [19]	Tab. 12	N/A	1203	6016	5.0	19975	16.6
ImageNet-1k [7]	Tab. 5	N/A	1000	6361	6.4	19741	19.7

`Generate a list of` **`q`** `types of the following` **`[context]`** `and output the list separated by '&': c`

where $q = 10$. The **`[context]`** prompt is consistently set to **`object`** across all datasets, except for iNatLoc [6], where context-specific prompts like **`species`** or **`genus`** are used, aligning with its biological tree of life structure. The `'&'` symbol serves as a separator prompt, facilitating the formatting of ChatGPT's responses for easier post-parsing of category names. Moreover, the final lists of super-categories and sub-categories are the *union* of results from $t = 3$ LLM queries. To be more specific, we employ the same super-/sub-category prompts for querying the LLM $t = 3$ times for each target CoI, and then amalgamate these LLM responses to form the final results.

In order to generate hierarchies for all datasets, we fix $p = 3$, $q = 10$, and $t = 3$. It is important to note that we did not perform any extensive hyperparameter tuning for p, q, and t, as our goal is to construct hierarchies automatically and validate SHiNe's effectiveness with open and noisy hierarchies. Apart from parsing category names from ChatGPT's responses, we do not perform any additional cleaning or organizing of the query results, ensuring an unbiased evaluation of our method's inherent efficacy. The hierarchies generated for the evaluation datasets are directly employed as LLM-generated hierarchies by SHiNe in our experiments to assess its performance.

Discussion: Differences between our hierarchy generation process and the one from CHiLS [38]. For any given target vocabulary, CHiLS uses GPT-3 to generate only sub-categories, forming a two-level hierarchy. In our work, we adopt the **sub-category prompt** from CHiLS for generating sub-categories. However, our hierarchy generation strategy significantly differs from CHiLS in three key respects: *i)* We generate both super-categories and sub-categories, creating a more comprehensive three-level hierarchy. *ii)* We query our LLM three times ($t = 3$) and use the union of the outcomes of these queries as the final set, aiming to enrich and diversify the category sets with varied categorization principles. *iii)* As a result of merging and de-duplicating the generated category names from three LLM queries, we do not have a predetermined (fixed) number of super-/sub-categories for each target CoI (class). Thus, our generated hierarchies are more varied and imbalanced, aligning more closely with real-world scenarios.

Discussion: The rationale behind generating $p = 3$ super-categories instead of just one. In real-world contexts, there is no single "optimal" hierarchy for any given vocabulary set. A single vocabulary can have multiple, equally valid hierarchical arrangements, depending on the categorization principles applied. For example, `"Vegetable salad"` might be classified under various super-categories—such as `"Appetizer"`, `"Cold dish"`, `"Side dish"`, or simply `"Vegetable"`—based on cultural or contextual differences. Therefore, **a truly robust and effective hierarchy-based method should function with hierarchies open to diverse categorization principles.** In such open hierarchies, categories are open to multiple categorization principles (*i.e.*, one class may link to several super-category nodes). Thus, we choose to generate $p = 3$ super-categories per target CoI (category) in $\mathcal{C}^{\text{test}}$ at each single LLM query. In our 3-level synthetic hierarchies, each target CoI falls under multiple super-categories generated from three times of LLM queries, reflecting various and diverse categorization principles. This approach allows us to rigorously evaluate the efficacy of our proposed SHiNe in realistic, diverse yet noisy categorization scenarios.

B.2. Summary statistics of the LLM hierarchies

In Tab. 8, we present comprehensive summary statistics for the synthetic hierarchies generated by the LLM across each dataset at every label vocabulary level. All synthetic hierarchies are created using $p = 3$ and $q = 10$, with the final super-/sub-categories a de-

duplicated union of results from $t = 3$ LLM queries. As shown in Tab. 8, the hierarchies synthesized are both highly open (each CoI is linked to multiple super-categories) and noisy (sub-categories might not be present in the dataset). Despite these challenges, as shown in Tab. 3, Tab. 4, Tab. 5, Tab. 12, and Fig. 4, SHiNe performs effectively using such open and noisy synthetic hierarchies, consistently improving the baseline results. This underlines the adaptability and robustness of SHiNe in using open and noisy semantic hierarchies when the ground-truth hierarchies are not available.

C. Further Implementation Details of SHiNe

C.1. Hierarchy-aware Sentences Integration

This section provides a further explanation of SHiNe's process for integrating hierarchy-aware sentences with different hierarchical structures, as illustrated in Fig. 5.

Single super-category path hierarchy case (ground-truth hierarchy structures). Fig. 5(a) illustrates the case where the target Class of Interests (CoI) is linked to a unique super-category at each higher hierarchical level and multiple sub-categories at each lower level. In this case, SHiNe employs the **Is-A** connector to form hierarchy-aware sentences by integrating the lowest linked sub-category, the target CoI, and the highest super-category, following their hierarchical relationships in a bottom-up manner. As a result, the total number of constructed sentences in this case equals the number of the lowest linked sub-categories.

Multiple super-category path hierarchy case (LLM-generated hierarchy structures). Fig. 5(b) displays the case where the target CoI is linked to multiple super-categories at the upper level and several sub-categories at the lower level. Here, SHiNe builds hierarchy-aware sentences by iterating through all combinations of the linked sub-categories, super-categories, and the target CoI. The **Is-A** connector is used to connect these categories in a specific-to-abstract order. The resulting number of constructed sentences in this case equals the product of the counts of the lowest linked sub-categories and the linked super-categories.

C.2. Pseudo-code of SHiNe

We show the pseudocode for the core implementation of SHiNe in Alg. 1, tailored for a three-level hierarchy.

C.3. Time Complexity Analysis of SHiNe

Let c be the number of Classes of Interest (CoIs) in a given vocabulary, and let p and q represent the average number of related super-categories and sub-categories per CoI, respectively, in a hierarchy. Our proposed method, SHiNe, aggregates hierarchy-aware information from both super-categories and sub-categories into c *nexus*-based embeddings (offline). Consequently, at inference, both memory and time complexity of SHiNe scale linearly as $\mathcal{O}(c)$. It is important to note that this scalability at inference is unaffected by the number of related super-/sub-categories, because they are only used offline to generate $\mathbf{n}_c$. The offline pipeline to construct SHiNe OvOD classifier needs to run only once.

In contrast, the time and memory complexities for CHiLS [38] scale at inference as $\mathcal{O}(c(1 + q))$, because image-text similarity scores are computed for vocabulary nodes *and* all their chil-

Algorithm 1: Pseudocode for constructing SHiNe classifier *offline* for OvOD detectors in a PyTorch-like style.

```
# target_vocabulary: input class vocabulary for
# inference
# shine_classifier: output SHiNe classifier for
# OvOD detectors
# hrchy: a semantic hierarchy for the target
# vocabulary
# aggregator: computes mean vector or principal
# eigenvector of the given embeddings
# tokenizer: tokenizes given text
# text_encoder: VLM text encoder

# container for SHiNe
shine_classifier = []

# the proposed Is-A connector
isa_connector = "which is a"

# build SHiNe classifier weight vector for each
# class in the vocabulary
for class_name in target_vocabulary:
    # retrieve super-category names
    super_names = hrchy.get_parents(class_name)
    # retrieve sub-category names
    sub_names = hrchy.get_children(class_name)

    # form specific-to-abstract branches combining
    # super-/sub-categories, and the target class
    # name
    branches = [
        [sub_name, class_name, super_name]
        for super_name in super_names
        for sub_name in child_names
    ]

    # construct hierarch-aware sentences in natural
    # language using the Is-A connector
    sentences = [
        f"a {branch[0]}"
        + "".join([
        f", {isa_connector} {name}"
        for name in branch[1:]
        ])
        for branch in branches
    ]

    # tokenize the sentences
    text_tokens = tokenizer(sentences)
    # extract textual feature representations
    text_embeddings = text_encoder(text_tokens)

    # fuse the embeddings into a single nexus-based
    # classifier vector
    nexus_vector = aggregator(text_embeddings)

    # append the single classifier vector to the
    # classifier container
    shine_classifier.extend(nexus_vector)

# stack all the constructed classifier vectors as
# the SHiNe classifier
shine_classifier = torch.stack(shine_classifier)

# l2-normalize the classifier vectors
shine_classifier = l2_normalize(shine_classifier,
                                dim=1)

# the shine_classifier is output and applied
# directly to the OvOD detector
```

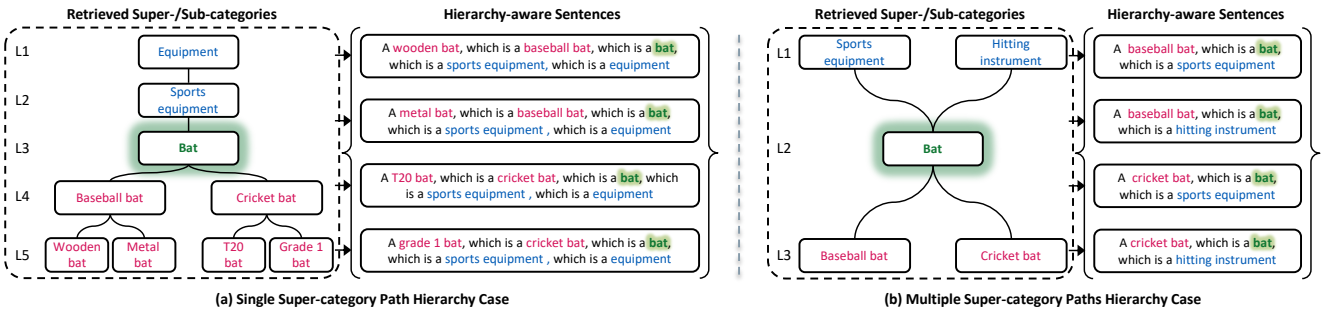

Figure 5. Examples of integrating hierarchy-aware sentences with different hierarchy structures. We use **"Bat"** as an example of the target Class of Interest (CoI) for example. The retrieved super-/sub-categories and the target CoI are color-coded in blue and red, and green, respectively. **(a)** The target CoI is linked to a unique super-category at each higher hierarchy level and multiple sub-categories at each lower level, akin to the ground-truth hierarchy structure of the datasets. **(b)** The target CoI is associated with multiple super-categories at the upper hierarchy level and multiple sub-categories at the lower level, akin to the simple three-level LLM-generated hierarchy structures.

dren. H-CLIP [14], on the other hand, involves a search procedure *online* across $p \cdot (c+1)$ prompt combinations for the top k (*e.g.*, $k = 5$) predicted CoIs, resulting in a time complexity of $\mathcal{O}(c + p \cdot (q+1) \cdot k)$. Crucially, the operations for $p \cdot (q+1) \cdot k$ only commence after the prediction based on the first c standard prompts. Unlike SHiNe and CHiLS [38], for which the embeddings are precomputed and the class predictions can be fully parallelized, H-CLIP requires encoding the latter $p \cdot (q+1) \cdot k$ CLIP [42] text embeddings at test time on-the-fly. Furthermore, it employs a search-on-the-fly mechanism, resulting in significant computational overheads. This makes H-CLIP a sub-optimal candidate for many applications, particularly those like detection and segmentation tasks that require per-box, per-mask, or even per-pixel prediction.

Given the extensive number of super-/sub-categories in the hierarchy employed in our experiments, as detailed in Tab. 8, the substantial computational overheads imposed by CHiLS and H-CLIP become evident.

C.4. Implementation Details of Aggregators

Mean-aggregator. During the semantic hierarchy *nexus* classifier construction phase, as illustrated in Fig. 2(3), SHiNe, by default, uses Eq. 1 where the "Aggregator" is the mean operation, as

$$\mathbf{n}_c = \frac{1}{K} \sum_{k=1}^{K} \mathcal{E}_{\text{txt}}\left(e_k^c\right) \quad , \tag{3}$$

where $\mathcal{E}_{\text{txt}}$ is the frozen CLIP [42] text encoder, and $\{e_k^c\}_{k=1}^K$ represents the K hierarchy-aware sentences, which are built by integrating all super-/sub-categories related to the target class (CoI) c using our proposed **Is-A** connector. This aggregator, which we call the **mean-aggregator**, calculates the mean of the encoded sentences' embeddings to form the final *nexus*-based classifier weight vector for c. This mean vector is the centroid represented within CLIP's embedding space, summarizing the general characteristics of the hierarchy-aware embeddings related to the target CoI. At inference, the classification decision for a region is based on the cosine similarity between the visual embedding of the region and the hierarchy-aware representation defined by the mean vector $\mathbf{n}$, which we call *nexus*. This approach renders the decision-making process less sensitive to variations in the semantic granularity of the name c. Note that all the embeddings are $l2$-normalized.

Principal Eigenvector Aggregator Drawing inspiration from text classification techniques in Natural Language Processing (NLP) [15, 28, 50], we introduce an alternative aggregation approach, called the **principal eigenvector aggregator**. This method uses the principal eigenvector of the sentence embeddings matrix as the classifier weight vector $\mathbf{n}_c$. Specifically, for a set of hierarchy-aware sentences $\{e_k^c\}_{k=1}^K$, we first apply a Singular Vector Decomposition (SVD) operation on their embedding matrix as:

$$\mathbf{U}\mathbf{S}\mathbf{V}^T = \text{SVD}\left(\text{concat}_{k=1}^{K}\left\{\mathcal{E}_{\text{txt}}\left(e_k^c\right)\right\}\right) \quad , \tag{4}$$

where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices representing the left and right singular vectors, respectively, and $\mathbf{S}$ is a diagonal matrix with singular values in descending order. Subsequently, we can derive the principal eigenvector, corresponding to the largest singular value in the sentence embedding matrix, by selecting the first column of matrix $\mathbf{V}$ as:

$$\mathbf{n}_c = \mathbf{V}[:, 0] \quad , \tag{5}$$

where $\mathbf{n}_c$ serves as the *nexus*-based classifier vector for the target class c. In contrast to the mean-aggregator, the **principal eigenvector aggregator** captures the dominant trend in the sentence embeddings (as known as their "theme", to maintain NLP terminology), to effectively represent the CoIs. Note that all the embeddings are $l2$-normalized.

Next, we explain the rationale behind this aggregator design. In high-dimensional semantic spaces like the 512-dimensional vision-language aligned embedding space of CLIP ViT-B/32, the principal eigenvector is able to capture the most significant semantic patterns or trends within the embeddings. This approach stems from the understanding that the direction of greatest variance in the space contains the most informative representation of semantic embeddings. Projecting the high-dimensional hierarchy-aware sentence embeddings of a target class (CoI) onto this principal eigenvector yields a condensed yet information-rich representation, preserving the essence of the original hierarchy-aware sentences. Consequently, during inference, classification decisions for a region are based on the cosine similarity between the region's

embedding and the semantic pattern or trend depicted by the principal eigenvector. This differs from the representation centroid approach used by the mean-aggregator.

We compare the **mean-aggregator** and the **principal eigenvector aggregator** in Sec. 4.1 of the main paper. While the principal eigenvector aggregator shows slightly lower performance compared to the mean-aggregator in general, its potential application in VLM tasks might be interesting for future research. In general, given the intimate connection between computer vision and NLP in open-vocabulary models, we believe in the importance of enabling more connections between the two fields—in this case, drawing from the NLP field of topic modeling.

D. Extended Analysis of SHiNe on FSOD

In Fig. 6, we present an expanded study of the core components of SHiNe, examining their effectiveness across various levels of label granularity on both the iNatLoc and FSOD datasets. The results from FSOD align with those observed in the iNatLoc-only study shown in Fig. 3 of the main paper. Next, we provide further analysis of SHiNe's core components.

Extended discussion: the Is-A connector effectively integrates hierarchy knowledge in natural sentences. The effectiveness of the proposed **Is-A** connector is studied in Fig. 6(a). Excluding the top (abstract) levels where all methods, including **Ens**, **Concat**, and **Is-A**, revert to the plain baseline due to the absence of further parent nodes, the methods leveraging super-category information consistently outperform the baseline across nearly all levels of granularity. This improvement is attributed to directing the model's focus towards more general concepts via super-category-inclusive classifiers. An exception occurs at the second level of FSOD (Fig. 6(a-FSOD-L2)), where no method exceeds the baseline. We speculate that at this level, target categories like `"Fruit"` are already highly abstract, rendering the addition of more abstract parent categories like `"Food"` redundant in clarifying ambiguities. Nevertheless, this challenge is alleviated when sub-categories are also included in the aggregation step. In comparative terms, the **Is-A** and **Concat** connectors yield greater gains than **Ens**, highlighting the advantage of capturing internal semantic relationships for distinguishing between classes. Notably, our **Is-A** connector surpasses **Concat** at all levels of granularity in both datasets, improving the baseline mAP50 by up to **+39.4** points on iNatLoc (Fig. 6(a-iNat-L5)) and **+2.5** points on FSOD (Fig. 6(a-FSOD-L3)). This indicates the superior effectiveness of **Is-A**'s explicit modeling of category relationships compared to the mere sequential ordering of class names from specific to abstract by **Concat**. Overall, the integration of more abstract concepts proves beneficial in object detection across diverse label granularities, with our **Is-A** connector particularly excelling due to its effective incorporation of hierarchical knowledge into natural language sentences, achieved by explicitly modeling internal category relationships.

Extended discussion: A simple mean-aggregator is sufficient for hierarchy-aware sentences fusion. The impact of the aggregation step is analyzed in Fig. 6(b), focusing on both the mean-aggregator (M-Agg) and the principal eigenvector aggregator (PE-Agg). These aggregators consistently outperform the baseline across various models and levels of label granularity in all datasets. Notably, their advantage becomes more pronounced with increasingly abstract target vocabularies, surpassing the benchmarks set by the **Is-A** method. This is especially evident in cases involving highly abstract label vocabularies, where these aggregation methods significantly improve baseline performance, achieving gains of up to **+9.8** points in iNatLoc (Fig. 6(b-iNat-L1)) and **+20.5** points in FSOD (Fig. 6(b-FSOD-L1)).

These results underscore the effectiveness of the aggregation step in fusing hierarchy-aware sentences into semantic *nexus*-based classifiers. This fusion allows the *nexus*-based classifier to use both specific knowledge from sub-categories and abstract knowledge from super-categories, thereby improving the baseline detector's ability to discriminate visual object robustly.

E. Comparison with Additional Baselines

To further validate the effectiveness of the proposed Is-A prompting method, we further compare SHiNe with two additional baselines: *i)* **Root-Stmt** prompting, which explicitly states the root (target) class and its super/sub-classes using the template like `"A bat, which is a sports equipment and can be instantiated in a wooden baseball bat or a baseball bat"`; *ii)* **80-Prompts**, where we embed the target class name into the 80 hand-crafted prompts from CLIP [42] and average the scores. As shown in Tab. 9, methods leveraging a hierarchy consistently surpass the 80-prompt ensemble baseline, demonstrating the benefits of leveraging hierarchy knowledge. Moreover, SHiNe's superior performance to the Root-Stmt baseline suggests that Is-A prompting and nexus aggregation is more effective for combining hierarchy information.

Table 9. Comparison with additional baseline methods on iNatLoc and FSOD datasets. Detic with Swin-B backbone trained with LVIS and IN-21k is used as baseline. mAP50 is reported.

Hrchy	Set	iNatLoc @mAP50							FSOD @mAP50			
	Level	L6	L5	L4	L3	L2	L1	$\overline{\Delta}$	L3	L2	L1	$\overline{\Delta}$
N/A	Baseline	58.6	54.9	73.1	63.8	65.3	65.4	-	66.0	38.4	24.7	-
N/A	80-Prompts	59.3	55.9	73.4	66.9	66.4	65.6	+1.1	66.1	38.7	26.0	+0.6
GT	Root-Stmt	86.3	83.1	83.9	82.6	72.1	66.6	+15.6	66.7	46.7	31.6	+5.3
GT	SHiNe	**86.3**	**86.8**	**87.7**	**86.9**	**78.1**	**70.3**	**+19.2**	**66.7**	**51.4**	**42.2**	**+10.4**

F. Extended Main Experimental Results

In Tab. 10, we present additional experimental results from applying our proposed SHiNe to Detic [72] with a Swin-B [32] backbone, trained using only LVIS and LVIS combined with IN-L [7] as auxiliary weak supervisory signals. This observation is consistent with those in Tab. 3 from the main paper, demonstrating that SHiNe consistently and substantially improves the performance of the baseline OvOD detector on both iNatLoc and FSOD datasets. This improvement spans across various label vocabulary granularities and is evident with both the ground-truth hierarchy (GT-H) and a synthetic hierarchy generated by LLM (LLM-H).

G. Summary of the Main Experiments

Beyond the per-level comparison in Tab. 3 of the main paper, Fig. 7 offers an extended comparison (calculated from Tab. 3) us-

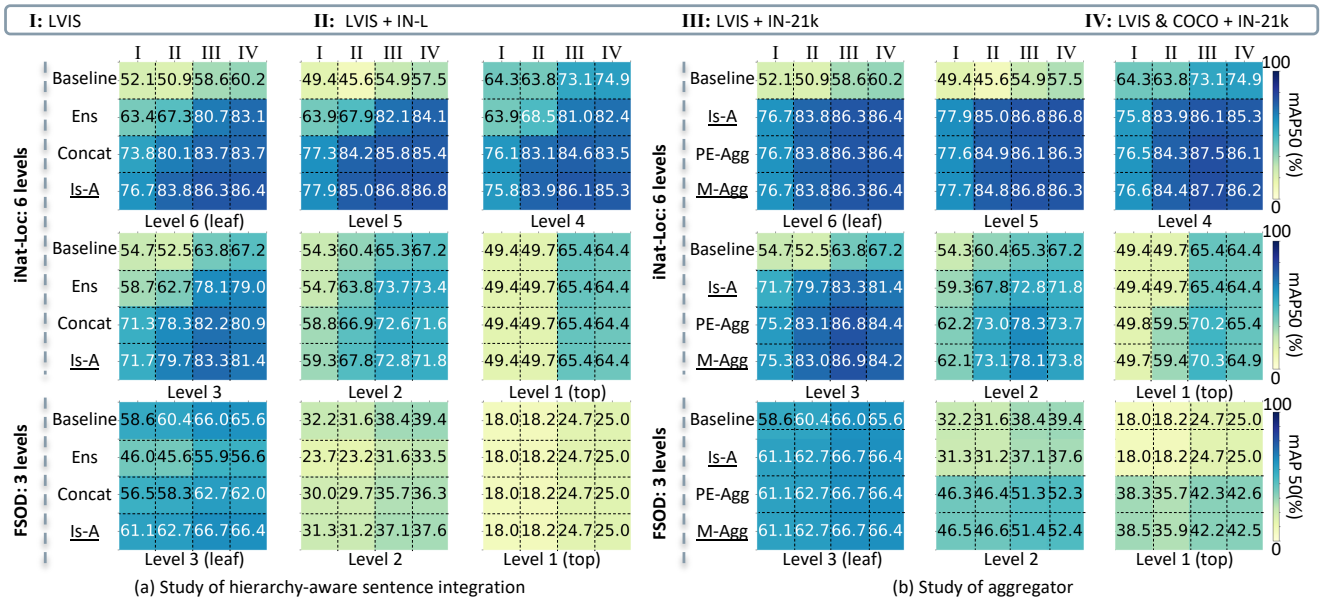

Figure 6. Further study of hierarchy-aware sentence integration methods **(left)** and aggregators **(right)** across various label granularity levels on both iNatLoc and FSOD datasets. Darker color indicates higher mAP50. Components used by default in SHiNe are underlined. Detic [72] with Swin-B backbone, trained using various combinations of supervisory signals described in Tab. 2, serves as the baseline open-vocabulary detector for all methods evaluated. To evaluate the effectiveness of hierarchy-based components, we use the ground-truth hierarchy for all methods that rely on hierarchies.

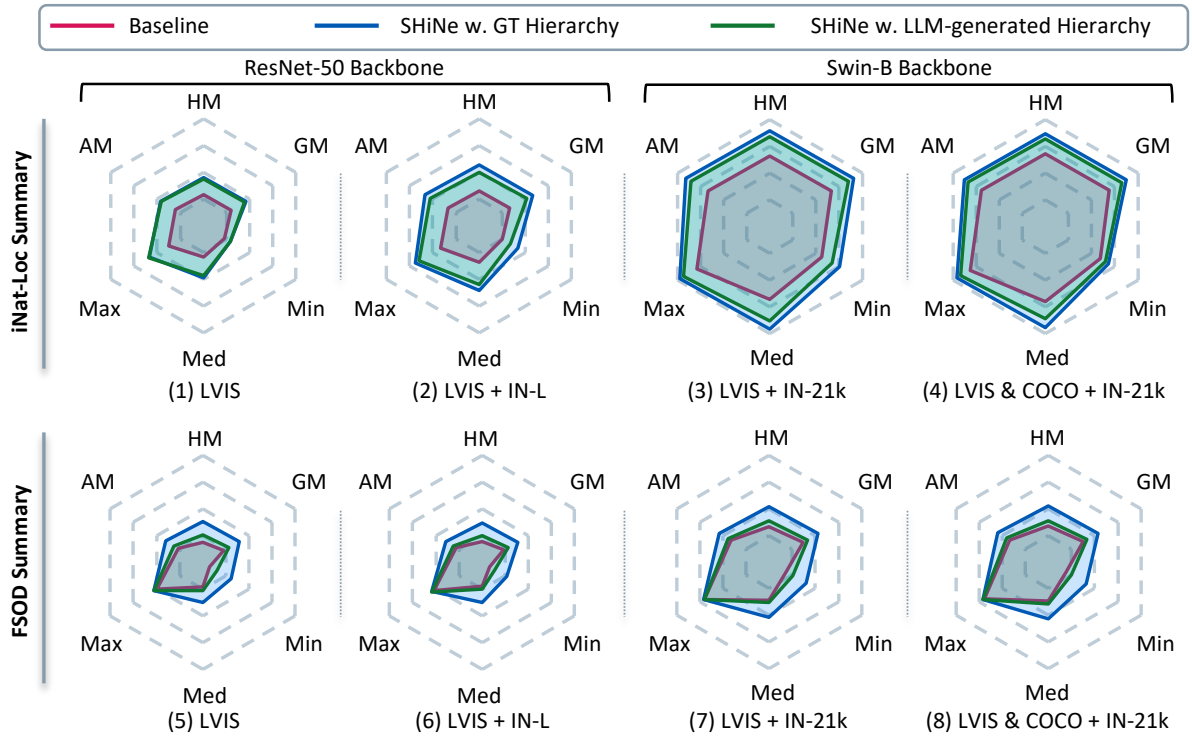

Figure 7. Additional summary statistics across all levels for the main experimental results in Tab. 3 for iNatLoc **(upper)** and FSOD **(lower)**, respectively. This summary includes various measures for mAP50, such as arithmetic mean (AM), harmonic mean (HM), geometric mean (GM), minimum value (Min), median (Med), and maximum value (Max), calculated across all granularity levels within each dataset. A larger area indicates better performance across various metrics. Gray dashed gridlines are scaled from 10 (innermost) to 100 (outermost).

Table 10. Additional results are provided for Detic [72] with a Swin-B backbone, trained using both **I**-LVIS and **II**-LVIS+IN-L supervisory signal combinations. Detection performance across varying label granularity levels on iNatLoc **(upper)** and FSOD **(lower)** datasets are reported. SHiNe is directly applied to the baseline detector (BL) [72] with ground-truth (GT-H) and LLM-generated (LLM-H) hierarchies. mAP50 (%) is reported.

Set	Level	Swin-B Backbone					
		I - LVIS			**II** - LVIS + IN-L		
		BL	SHiNe (GT-H)	SHiNe (LLM-H)	BL	SHiNe (GT-H)	SHiNe (LLM-H)
iNatLoc	L6	52.1	**76.7(+24.6)**	74.0(+21.9)	50.9	**83.8(+32.9)**	78.8(+27.9)
	L5	49.4	**77.7(+28.3)**	68.4(+19.0)	45.6	**84.8(+39.2)**	69.0(+23.4)
	L4	64.3	**76.6(+12.3)**	73.7(+9.4)	63.8	**84.4(+20.6)**	79.3(+15.5)
	L3	54.7	**75.3(+20.6)**	73.2(+18.5)	52.5	**83.0(+30.5)**	78.7(+26.2)
	L2	54.3	62.1(+7.8)	**62.8(+8.5)**	60.4	73.1(+12.7)	**75.1(+14.7)**
	L1	49.4	49.7(+0.3)	**50.7(+1.3)**	49.7	**59.4(+9.7)**	49.8(+0.1)
FSOD	L3	58.6	**61.1(+2.5)**	61.0(+2.4)	60.4	**62.7(+2.3)**	62.1(+1.7)
	L2	32.2	**46.5(+14.3)**	35.6(+3.4)	31.6	**46.6(+15.0)**	33.5(+1.9)
	L1†	18.0	**38.5(+20.5)**	23.7(+5.7)	18.2	**35.9(+17.7)**	22.3(+4.1)

ing various summary statistical metrics. As shown in Fig. 7, our proposed SHiNe consistently and markedly enhances the baseline OvOD detector's performance across a range of summary metrics, including arithmetic mean (AM), harmonic mean (HM), geometric mean(GM), on both datasets. The harmonic and geometric means are employed to present the evaluation results from diverse perspectives, particularly in contexts where extreme values might skew the interpretation. These means are less influenced by extreme values, such as exceptionally high or low mAP50 scores at specific granularity levels. The enhancement from SHiNe is apparent when employing both the ground-truth hierarchy and a synthetic hierarchy generated by the LLM. Notably, SHiNe most significantly improves the baseline's weakest performance (minimum mAP50), suggesting a notable improvement in performance consistency by improving the minimum achieved performance across granularity levels. These results demonstrate that SHiNe not only boosts overall performance but also enhances consistency across different vocabulary granularities, a crucial aspect for real-world applications.

H. Experiments with other OvOD Detectors

We further assess SHiNe's performance on top of an additional OvOD detector, CORA [61], and present the results alongside VLDet [29] with ResNet-50 [21] in Tab. 11. These results and improvements are consistent with those using Detic [72], CoDet [33], and VLDet [29], further validating SHiNe's effectiveness.

I. Further Experiments on COCO/LVIS

This section extends the evaluation of SHiNe to COCO [30] and LVIS [19], following the open-vocabulary evaluation (OVE) protocol as described in [73]. According to the OVE protocol, datasets are divided into base and novel classes; models are trained on base classes with bounding box annotations and then evaluated on novel classes and their union. The base classes are disjoint from the

Table 11. Comparison with CORA [61] and VLDet (VLD) [29] on iNatLoc and FSOD. SHiNe is applied to the baseline methods, respectively. All methods employ ResNet-50 [21] as backbone. Note that CORA uses only box-annotated COCO [30] base split for training, while VLDet uses box-annotated LVIS [19] and image-caption-annotated CC3M [49] as supervisory signals. mAP50 (%) is reported.

Set	Level	CORA	SHiNe (GT-H)	SHiNe (LLM-H)	VLD	SHiNe (GT-H)	SHiNe (LLM-H)
iNatLoc	L6	31.2	54.2(+23.0)	**54.8(+23.6)**	48.9	62.4(+13.5)	**64.8(+15.9)**
	L5	22.6	**51.9(+29.3)**	35.7(+13.1)	44.3	**60.6(+16.3)**	52.6(+8.3)
	L4	21.7	**50.7(+29.0)**	36.2(+14.5)	42.9	**58.7(+15.8)**	53.6(**+10.7**)
	L3	26.0	**50.5(+24.5)**	43.4(+17.4)	43.8	**63.5(+19.7)**	58.7(+14.9)
	L2	20.0	**33.2(+13.2)**	24.8(+4.8)	34.3	**54.4(+20.1)**	46.9(**+12.6**)
	L1	18.3	16.2(-2.1)	13.0(-5.3)	37.0	**43.2(+6.2)**	38.6(+1.6)
FSOD	L3	49.3	**51.4(+2.1)**	51.1(+1.8)	48.8	**53.8(+5.0)**	53.6(+4.8)
	L2	21.9	**33.6(+11.7)**	23.5(+1.6)	23.5	**39.3(+15.8)**	29.8(+6.3)
	L1	11.6	**26.2(+14.6)**	14.1(+2.5)	12.8	**31.0(+18.2)**	17.3(+4.5)

Table 12. Comparison of detection performance on COCO and LVIS benchmarks using the OVE protocol. We use Detic [72] with a ResNet-50 backbone as the baseline detector (BL). SHiNe is applied to the baseline using hierarchies generated by LLM. All models receive strong supervision on the base class partitions of both datasets, with box-class annotations. A comparison of different weak supervisory signals is also included. mAP50_{novel} and mAP_{novel} denote performance evaluated on the novel class partitions (17 classes for COCO and 337 classes for LVIS), while mAP50_{all} and mAP_{all} represent evaluations on both base and novel classes (65 classes for COCO and 1203 classes for LVIS).

	$\mathcal{D}^{\text{weak}}$	mAP50_{novel}		mAP50_{all}	
		BL	SHiNe	BL	SHiNe
COCO	N/A	1.3	**3.2(+1.9)**	39.3	**39.8(+0.5)**
	COCO Captions	24.0	**24.3(+0.3)**	44.8	**44.9(+0.1)**
	$\mathcal{D}^{\text{weak}}$	mAP_{novel}		mAP_{all}	
		BL	SHiNe	BL	SHiNe
LVIS	N/A	17.6	**20.9(+3.3)**	33.3	**33.6(+0.3)**
	IN-L	**26.7**	25.5(-1.2)	**35.8**	35.3(-0.5)
	Conceptual Captions	19.3	**21.5(+2.2)**	33.4	**33.5(+0.1)**

novel classes. We follow the base/novel class partitions for COCO and LVIS as used in [72]. Both datasets have a single, flat class vocabulary: COCO with 65 classes (48 base, 17 novel) and LVIS with 1203 classes (866 base, 337 novel). We use Detic [72] with a ResNet-50 [21] backbone, trained on the box-class annotated base classes with various weak supervisory signals, as the baseline OvOD detector in this experiment. Specifically, the baseline is trained on COCO-base with 48 classes or LVIS-base with 866 classes. We explore three types of weak supervisory signals as proposed in [72]: *i)* **N/A**, using only strong supervisory signals; *ii)* **IN-L**, a 997-class subset of ImageNet-21k [7] intersecting with the LVIS vocabulary; *iii)* Conceptual Captions [49] dataset; and *iv)* COCO Captions [72] dataset. For Conceptual Captions and COCO Captions, nouns are parsed from the captions, and both image labels and captions are used for weak supervision [72]. We report mAP50 for COCO and the official mask mAP metric for LVIS as suggested in [19].

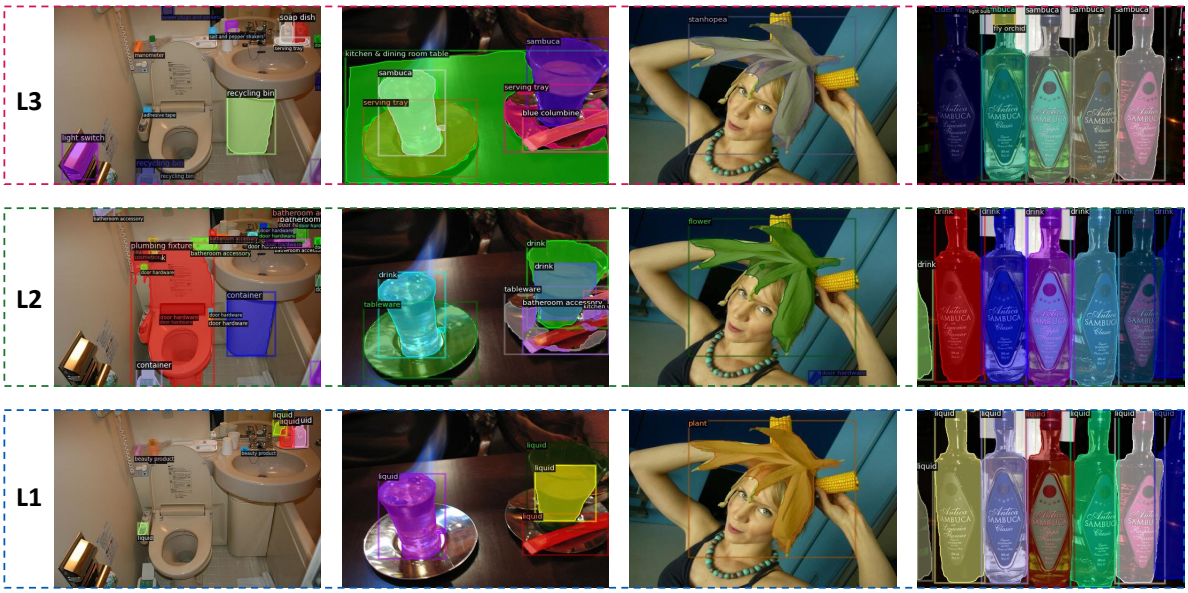

Figure 8. Qualitative detection results of SHiNe applied to Detic [72] with Swin-B [32], evaluated on the FSOD [10] dataset across three different label granularity levels. All models are trained using the LVIS + IN-L dataset as strong and weak supervisory signals, respectively. It is advisable to zoom in for a clearer view.

We evaluate and compare SHiNe with the baseline under the OVE protocol. In the absence of available ground-truth hierarchy information, we use the LLM to generate simple 3-level synthetic hierarchies for the target vocabularies of COCO and LVIS, as described in Tab. 8. Consequently, SHiNe is constructed using these generated hierarchies. As shown in the OVE evaluation results in Tab. 12, SHiNe notably improves the performance of the baseline detector on both COCO and LVIS benchmarks under the OVE protocol. Interestingly, SHiNe yields a greater performance gain on the novel class partitions. However, this advantage becomes less pronounced when assessing combined base and novel classes. This is attributed to the model overfitting on the base classes to the text classifier based on the standard `"a {Class Name}"` prompts during strongly supervised training. Replacing this overfit classifier with the SHiNe classifier leads to significant gains on novel class partitions, but slightly reduces performance on base class partition test data. Nevertheless, the consistent improvements achieved by SHiNe across most cases in Tab. 12 underscore its effectiveness on the COCO and LVIS benchmarks.

J. Qualitative Analysis of SHiNe

In Fig. 8 and Fig. 9, we showcase the qualitative detection results of SHiNe when applied to Detic [72] across various label granularity levels on the FSOD and iNatLoc datasets. For each granularity level, the same confidence threshold is consistently applied.

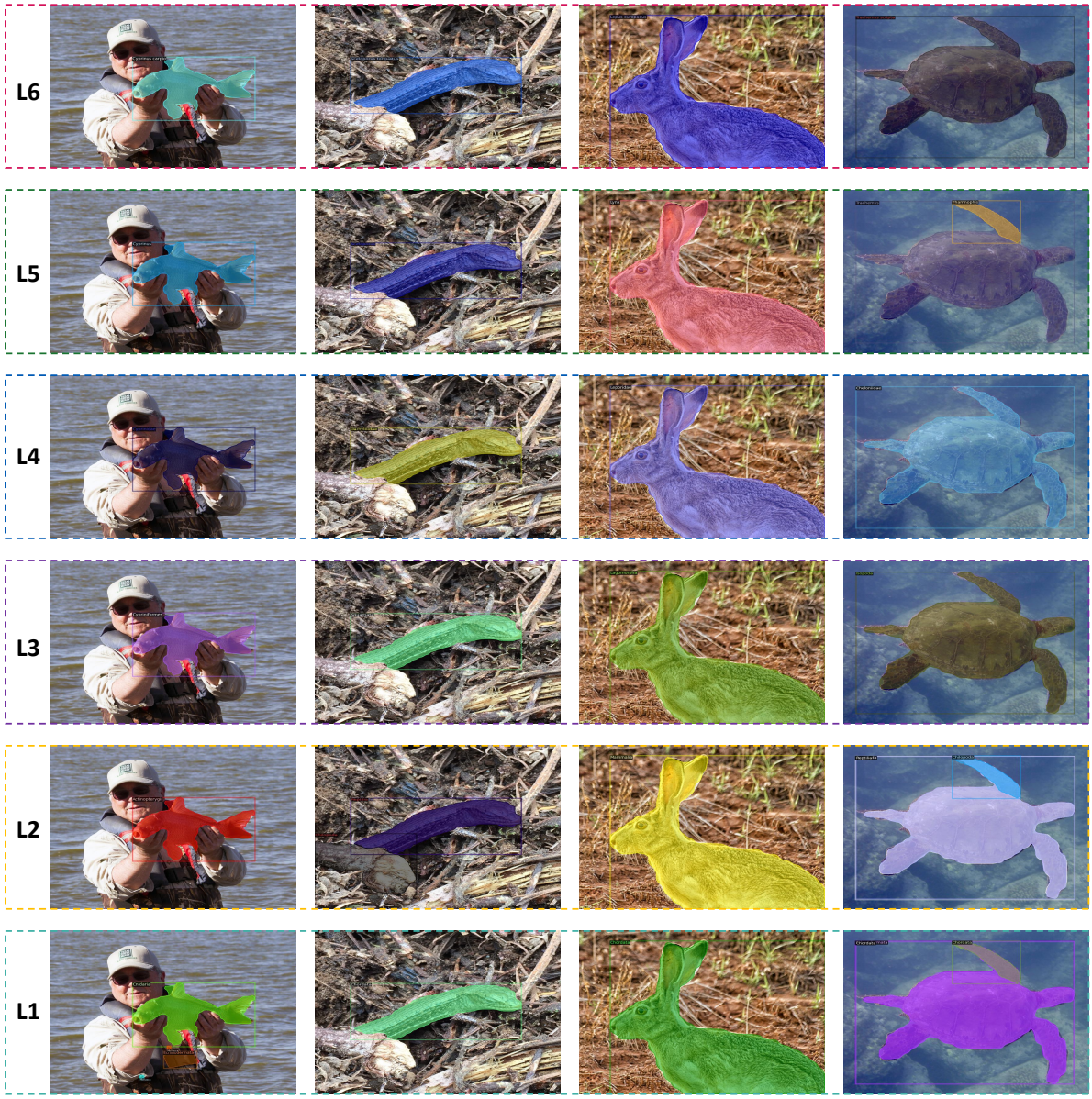

Figure 9. Qualitative detection results of SHiNe applied to Detic [72] with Swin-B [32], evaluated on the iNatLoc [6] dataset across six different label granularity levels. All models are trained using the LVIS + IN-L dataset as strong and weak supervisory signals, respectively. It is advisable to zoom in for a clearer view.