Specificity-aware reinforcement learning for fine-grained open-world classification

Supplementary Material

In this supplementary material, we present additional details and analyses that complement the content of the main document. First, in Sec. A, we provide further implementation details, including the prompts we used for the LMM and the LLM verifier, along with the optimization strategies adopted to improve training efficiency. Next, in Sec. B, we report the complete out-of-domain evaluation results for each individual dataset in both fine-grained and very fine-grained sets, along with further prompting baselines and additional qualitative examples. Finally, in Sec. B.5, we extend the ablation studies on the impact of training sets from different domains and the training-set size on the performance of SpeciaRL.

A. Additional implementation details

A.1. Prompts

Here, we report all the prompts used in our experiments. These include the classification prompt $\mathtt{P}_c$ provided to the reasoning LMM $\Phi^{\theta}_{\mathtt{LMM}}$, the verification prompt $\mathtt{P}_j$ used by the LLM-as-a-judge $\Psi_{\mathtt{LLM}}$, and the prompt used to generate the reasoning traces for the supervised fine-tuning (*sft*) baseline.

A.1.1. LMM prompts

In our experiments, we consider a total of three different prompts when querying a LMM to classify an image.

Default. Our default prompt is shown in Fig. 1. Since our work focuses on reasoning models, we not only request a classification of the input image, but we also explicitly instruct the model to first perform reasoning and then provide a single label. Specifically, we follow the standard `<think>`/`<answer>` tags format. This structured output simplifies the extraction of the final prediction and its subsequent verification by the LLM-as-a-judge.

"Be specific". In the "Be specific" baseline, we explicitly encourage the model to be specific in its prediction. To this end, we modify the default prompt by adding the requirement to be specific. The complete text query is reported in Fig. 2.

Format free. When considering the evaluation protocol in [5], for consistency and fair comparison, we adopt the same prompting strategy reported in the original paper [5], as shown in Fig. 3. Since this previous work does not have a focus on reasoning models, it adopts a more general-purpose prompt without formatting requirements.

Default LMM prompt ($\mathtt{P}_c$)

Classify the image.
Output the thinking process in `<think> </think>` and the final answer in `<answer> </answer>` tags. The output answer format should be as follows: `<think>` ... `</think> <answer>`a single label or the word 'None' to abstain.`</answer>`.
Please strictly follow the format.

Figure 1. LMM default prompt for prediction.

"Be specific" LMM prompt ($\mathtt{P}_c$)

Classify the image, be specific.
Output the thinking process in `<think> </think>` and the final answer in `<answer> </answer>` tags. The output answer format should be as follows: `<think>` ... `</think> <answer>`a single label or the word 'None' to abstain.`</answer>`.
Please strictly follow the format.

Figure 2. LMM prompt for prediction for the "Be specific" baseline.

Format free LMM prompt ($\mathtt{P}_c$) **[5]**

What type of object is in this photo?

Figure 3. LMM prompt used in the evaluation protocol of [5].

A.1.2. LLM-as-a-judge prompt

Figure 4 shows the prompt used when querying the LLM verifier to categorize a prediction into the categories defined in the main paper. This prompt provides a precise definition with in-context examples for each category. The placeholder `%s` is replaced with the actual `ground_truth` and `prediction` formatted in the specified JSON format. To eliminate the possibility of invalid responses from the LLM verifier, we utilize the vLLM [8] guided decoding strategy to constrain the model in generating only one of the predefined categories as the response.

A.1.3. CoT generation prompt

Figure 5 reports the prompt used to generate a chain-of-thought reasoning trace for each sample in the training set, which are then used to construct the custom dataset for supervised fine-tuning. This prompt provides the LMM with the ground-truth label associated to the image, and requests a thinking trace leading to the correct prediction.

A.2. Optimizations

Our study can be computationally demanding at training and evaluation due to the LMM inference and LLM-as-a-judge evaluation. We therefore adopt several optimizations strategies to reduce computational costs.

LLM-as-a-judge prompt ($\mathtt{P}_j$)

Role: You are an expert AI classifier. Your goal is to classify a model's `prediction` against a `ground_truth` label.
Task: You will receive a single JSON object. Your output must be **only the classification category** and nothing else.

Classification Categories

- **Specific**: The prediction is an exact match or a direct synonym for the ground truth. This includes common name/scientific name equivalence.
```
prediction: "Panthera leo"  ground_truth: "lion"
prediction: "passiflora"    ground_truth: "passion flower"
```
- **Less Specific**: The prediction is a correct, but **closely related parent category** (e.g., family, genus, product line) of the ground truth.
```
prediction: "Warbler"     ground_truth: "Golden-winged Warbler"
prediction: "Boeing 707"  ground_truth: "707-320"
```
- **Generic**: The prediction is correct, but a **significantly broader category** than the ground truth.
```
prediction: "dog"                  ground_truth: "samoyed"
prediction: "Commercial Airline"  ground_truth: "757-200"
```
- **More Specific**: The prediction is a correct, but **more specific subtype or instance** of the ground truth.
```
prediction: "samoyed"  ground_truth: "dog"
prediction: "757-200"  ground_truth: "Commercial Airline"
```
- **Wrong**: The prediction is factually incorrect, contradictory, malformed, completely unrelated to the ground truth, or contains multiple options.
```
prediction: "cat"                    ground_truth: "dog"
prediction: "Blue-winged Warbler"  ground_truth: "Golden-winged Warbler"
prediction: "b1rd"                   ground_truth: "bird"
prediction: "robin or cardinal"    ground_truth: "bird"
prediction: "_prototype"             ground_truth: "Boeing 717"
```
- **Abstain**: The prediction is a refusal to answer.
```
prediction: "none"
prediction: "I don't know"
prediction: "Cannot tell"
```

Input Format: You will receive a single JSON object with the following structure:

```
{"ground_truth": "<the_ground_truth_label>",
 "prediction": "<the_vlm_prediction>"}
```

Output Format: Your response must be a **single word** representing the classification category.

Prompt:

```
Classify the prediction in the following JSON object based on
the rules provided. Your output must be a single word.

INPUT:
%s
```

Figure 4. Prompt for the LLM-as-a-judge verifier categorizing a prediction given the target ground-truth.

CoT generation prompt

Given the image and the correct classification label: {ground_truth}.
Generate a correct well-reasoned response that will answer the following question:

> Classify the image.
> Output the thinking process in <think> </think> and the final answer in <answer> </answer> tags.
> The output answer format should be as follows: <think> ... </think> <answer>a single label or the word 'None' to abstain.</answer>.
> Please strictly follow the format.

Describe the content of the image, then infer the correct classification label. The thinking process must proceed without assuming or referencing the true label in advance. Use the correct classification label in the final answer and strictly follow the format.

Figure 5. Prompt for generating the reasoning traces used to train the supervised fine-tuning baseline model.

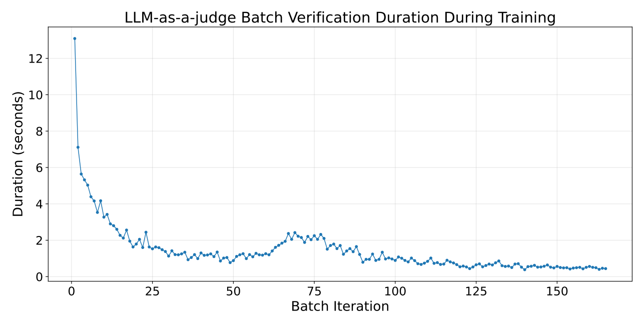

Figure 6. LLM-as-a-judge per-batch verification times during reinforcement learning training, showing the speedup obtained as cache hit rates increase when starting from an empty cache.

Inference Engine. In our experiments, we used the vLLM [8] inference engine both to generate the LMM predictions and to compute the LLM-as-a-judge categorization. This engine is highly optimized and enabled a significant speed-up of the evaluation process. Among its key features, it includes PagedAttention [8] for efficient memory management, continuous batching, which is crucial in our setting where variable-size image inputs make static batch selection difficult, and prefix caching, which is beneficial since our textual prompt is mostly fixed. For instance, generating 1000 predictions for Flowers102 with Qwen2.5-VL-7B on a A100 64 GB GPU takes 2.27 minutes with vLLM. In comparison, a naive PyTorch implementation requires 25.11 minutes, using a batch size of 32, which is the largest batch size avoiding out-of-memory errors across all our evaluation datasets. The PyTorch implementation incurs computation time that is a magnitude higher than using vLLM. Only when following the evaluation protocol in [5], we used the same testing code provided by the authors, which is built on PyTorch.

LLM-as-a-judge optimization via caching. We implemented a caching mechanism to reduce the verification time of the LLM-as-a-judge categorization procedure. This system stores a dictionary where (prediction, ground_truth) pairs are associated to the corresponding verification_category. This avoids repeating the LLM verification of a pair that has already been categorized in a previous computation. The cached data is persistent, allowing results to be reused across different runs. We used this cache-based solution to speed up the categorization process both during evaluation and during the reward computation in RL training. During evaluation, we run Llama-3-72B [6] using vLLM with tensor parallelism set to 4, distributing the model across four A100 GPUs. For a test subsample of 1000 predictions from Flowers102, our optimized implementation, with an initially empty cache, completes verification in 6.77 seconds, with only 301 actual LLM calls and a 70% cache hit rate. During reinforcement learning, we use a total of six A100 GPUs: one four-GPU node running the training loop with verl (an open source implementation of [15]) and two additional GPUs on a separate node performing batched LLM-as-a-judge inference using Qwen3-30B-A3B-Instruct-2507-FP8 [16] with tensor parallelism set to 2. With a batch size of 256 and 10 rollouts, each verification batch contains 2560 predictions. Analyzing the reward calculation durations shown in Fig. 6, we see an initial warm-up phase in which early batches require 2-14 seconds while the cache is being populated. Afterwards, the processing time quickly drops and stabilizes at approximately 0.5 to 1 seconds per batch, except for a mid-training bump that may be caused by cache misses caused by the model exploration. Overall, reinforcement learning training takes approximately 12 hours using our optimized implementation.

Figure 7. Additional qualitative examples of the think-answer output of the base model Qwen2.5VL-7B and SpeciaRL.

Dataset	Model	S^+	S	S^-	G	A	W	spec. ↑	corr. ↑	HM ↑
Fine-grained										
		Prediction categorization						Metrics		
Flowers102 [12]	CaSED [4]	0.0%	57.4%	8.4%	14.7%	0.0%	19.4%	0.883	0.806	0.842
	InternVL2.5-4B [3]	0.2%	15.8%	1.8%	29.8%	20.0%	32.4%	0.551	0.676	0.607
	InternVL2.5-8B [3]	0.4%	26.4%	3.3%	15.1%	13.2%	41.6%	0.688	0.584	0.632
	Qwen2.5VL-3B [1]	0.1%	26.6%	2.7%	49.6%	1.9%	19.2%	0.668	0.808	0.731
	Qwen2.5VL-7B [1]	0.1%	47.2%	4.1%	34.8%	1.2%	12.7%	0.779	0.873	0.823
	Qwen2.5VL-7B (“Be specific”)	0.2%	63.5%	5.8%	12.7%	3.0%	14.7%	0.882	0.853	0.867
	Qwen2.5VL-7B (*sft*)	1.3%	69.6%	8.5%	3.0%	0.0%	17.5%	0.956	0.825	0.885
	Qwen2.5VL-7B (*rft*)	10.4%	70.3%	5.4%	1.5%	0.0%	12.4%	0.976	0.876	0.923
	SpeciaRL-7B	13.6%	69.2%	5.0%	1.7%	0.0%	10.5%	**0.976**	**0.895**	**0.934**
	Qwen2.5VL-7B (BoN-64)	4.4%	78.3%	3.7%	9.9%	0.6%	3.1%	0.935	0.969	0.952
Food101 [2]	CaSED [4]	0.0%	33.0%	13.2%	35.3%	0.0%	18.5%	0.743	0.815	0.777
	InternVL2.5-4B [3]	0.5%	10.5%	1.4%	71.4%	2.6%	13.7%	0.560	0.863	0.680
	InternVL2.5-8B [3]	0.8%	10.6%	1.5%	46.3%	30.2%	10.7%	0.483	**0.893**	0.627
	Qwen2.5VL-3B [1]	1.5%	17.9%	2.5%	53.4%	7.8%	16.9%	0.601	0.831	0.697
	Qwen2.5VL-7B [1]	1.3%	32.0%	3.8%	47.8%	2.0%	13.2%	0.697	0.868	0.773
	Qwen2.5VL-7B (“Be specific”)	1.8%	38.0%	4.6%	34.7%	5.6%	15.3%	0.732	0.847	0.785
	Qwen2.5VL-7B (*sft*)	3.5%	51.4%	9.1%	11.6%	0.5%	24.0%	0.889	0.760	0.820
	Qwen2.5VL-7B (*rft*)	3.2%	52.0%	7.4%	8.7%	0.1%	28.6%	**0.912**	0.714	0.801
	SpeciaRL-7B	1.2%	54.3%	5.8%	19.7%	0.0%	18.9%	0.860	0.811	**0.835**
	Qwen2.5VL-7B (BoN-64)	15.1%	52.1%	5.5%	26.5%	0.5%	0.2%	0.849	0.998	0.917
OxfordPets [13]	CaSED [4]	0.0%	40.7%	10.2%	22.5%	0.0%	26.5%	0.812	0.735	0.772
	InternVL2.5-4B [3]	0.1%	7.9%	1.3%	61.8%	2.6%	26.2%	0.550	0.738	0.630
	InternVL2.5-8B [3]	0.9%	13.2%	5.2%	30.6%	18.7%	31.5%	0.554	0.685	0.613
	Qwen2.5VL-3B [1]	0.8%	7.4%	2.9%	57.3%	3.1%	28.6%	0.557	0.714	0.626
	Qwen2.5VL-7B [1]	2.7%	35.1%	5.2%	35.6%	1.0%	20.4%	0.751	0.796	0.773
	Qwen2.5VL-7B (“Be specific”)	4.3%	45.8%	8.2%	19.7%	1.6%	20.4%	0.835	0.796	0.815
	Qwen2.5VL-7B (*sft*)	2.4%	72.1%	5.3%	3.2%	0.4%	16.5%	**0.961**	0.835	**0.894**
	Qwen2.5VL-7B (*rft*)	0.3%	34.7%	2.1%	39.0%	0.0%	23.8%	0.737	0.762	0.749
	SpeciaRL-7B	2.1%	66.6%	4.5%	10.7%	0.0%	16.1%	0.923	**0.839**	0.879
	Qwen2.5VL-7B (BoN-64)	12.9%	59.8%	5.8%	19.5%	0.5%	1.4%	0.882	0.986	0.931

Table 1. Results on the individual datasets composing the fine-grained set.

B. Additional experimental analysis

We provide per-dataset evaluations of our method, in-domain evaluation on the CUB [17] test set, additional qualitative examples, additional results for prompting-based baselines, and extended ablation studies.

B.1. Per-dataset evaluation

In the main paper, we reported results averaged over the ***fine-grained*** and the ***very fine-grained*** test sets. Here, we present the results for each individual dataset, with Tab. 1 corresponding to the fine-grained ones and Tab. 2 to the very fine-grained ones. Considering overall performance, measured by the harmonic mean (HM), our SpeciaRL achieves the best performance on three out of five benchmarks (Flowers102, Food101, FGVAircraft) and the second best on the remaining two (OxfordPets, StanfordCars). Notably, on three datasets (Flowers102, OxfordPets, StanfordCars), our method not only improves specificity relatively to the base model, but also correctness. Overall, SpeciaRL performs strongly on all evaluation benchmarks, even though these datasets span domains significantly different from CUB [17], which is used for training. These results support the effectiveness of our method in eliciting a general classification behavior oriented towards both specificity and correctness.

B.2. In-domain evaluation

The fine-tuned models in the main results are trained on the same subset of CUB [17], implying that evaluations on the fine-grained and very fine-grained sets are out-of-domain. Tab. 3 reports the in-domain performance on the CUB test-split. In this setting, all training-based variants achieve very high specificity, exceeding BoN-64. In terms of correctness, only the RL-based methods improve over the base model, although they remain below BoN-64. Overall, the best har-

Very fine-grained										
		Prediction categorization						Metrics		
Dataset	Model	S^{+}	S	S^{-}	G	A	W	spec. ↑	corr. ↑	HM ↑
FGVCAircraft [11]	CaSED [4]	0.0%	1.6%	13.9%	37.7%	0.0%	46.8%	0.580	0.532	0.555
	InternVL2.5-4B [3]	0.0%	0.0%	0.2%	66.0%	8.5%	25.3%	0.472	0.747	0.579
	InternVL2.5-8B [3]	0.1%	2.2%	1.3%	59.1%	13.0%	24.4%	0.476	0.756	0.584
	Qwen2.5VL-3B [1]	0.2%	1.6%	1.4%	82.4%	0.3%	14.1%	0.514	0.859	0.643
	Qwen2.5VL-7B [1]	0.1%	6.6%	5.4%	80.7%	0.5%	6.7%	0.549	**0.933**	0.691
	Qwen2.5VL-7B (“Be specific”)	0.5%	23.0%	20.8%	40.4%	1.2%	14.0%	0.693	0.860	0.768
	Qwen2.5VL-7B (*sft*)	1.0%	42.9%	33.4%	2.3%	0.1%	20.2%	0.879	0.798	0.837
	Qwen2.5VL-7B (*rft*)	2.2%	45.9%	25.0%	2.0%	0.0%	25.0%	**0.904**	0.750	0.820
	SpeciaRL-7B	1.9%	46.5%	29.0%	1.7%	0.0%	20.9%	0.897	0.791	**0.841**
	Qwen2.5VL-7B (BoN-64)	3.4%	48.9%	24.6%	22.9%	0.1%	0.1%	0.823	0.999	0.903
StanfordCars [7]	CaSED [4]	0.0%	0.2%	13.7%	74.3%	0.0%	11.8%	0.540	0.882	0.669
	InternVL2.5-4B [3]	0.0%	0.1%	2.3%	59.4%	2.6%	35.6%	0.499	0.644	0.563
	InternVL2.5-8B [3]	0.0%	0.2%	10.2%	50.0%	18.2%	21.4%	0.476	0.786	0.593
	Qwen2.5VL-3B [1]	0.0%	0.5%	6.3%	67.8%	4.6%	20.8%	0.509	0.792	0.619
	Qwen2.5VL-7B [1]	0.0%	1.3%	20.1%	68.4%	0.8%	9.4%	0.561	0.906	0.693
	Qwen2.5VL-7B (“Be specific”)	0.1%	2.1%	37.8%	50.8%	1.3%	8.0%	0.611	**0.920**	0.734
	Qwen2.5VL-7B (*sft*)	0.0%	2.1%	68.1%	21.2%	0.1%	8.4%	0.698	0.916	0.792
	Qwen2.5VL-7B (*rft*)	0.2%	3.5%	82.8%	5.0%	0.0%	8.5%	**0.746**	0.915	**0.822**
	SpeciaRL-7B	0.2%	3.8%	79.4%	8.4%	0.0%	8.2%	0.738	0.918	0.818
	Qwen2.5VL-7B (BoN-64)	0.5%	12.4%	60.6%	26.3%	0.0%	0.3%	0.716	0.997	0.834

Table 2. Individual dataset results on the very fine-grained set.

Table 3. In-domain evaluation of the training strategies.

In-domain										
		Prediction categorization						Metrics		
Dataset	Model	S^{+}	S	S^{-}	G	A	W	spec. ↑	corr. ↑	HM ↑
CUB [17]	Qwen2.5VL-7B [1]	0.2%	23.0%	15.9%	48.1%	2.0%	11.0%	0.669	0.890	0.764
	Qwen2.5VL-7B (“Be specific”)	0.2%	32.2%	13.7%	35.3%	2.6%	16.1%	0.726	0.839	0.779
	Qwen2.5VL-7B (*sft*)	0.1%	80.4%	0.7%	0.3%	0.0%	18.5%	0.996	0.815	0.896
	Qwen2.5VL-7B (*rft*)	1.0%	92.7%	0.0%	0.0%	0.0%	6.3%	**1.000**	**0.937**	**0.968**
	SpeciaRL-7B	0.6%	92.7%	0.0%	0.0%	0.0%	6.7%	**1.000**	0.933	0.965
	Qwen2.5VL-7B (BoN-64)	1.1%	58.0%	14.1%	26.4%	0.1%	0.3%	0.831	0.997	0.907

monic mean is obtained by the two RL-based approaches, surpassing BoN-64. These results suggests that the models not only learn to adjust their classification behavior to be more specific and correct, but also acquire domain-specific information. Importantly, this does hinder generalization, as demonstrated by the strong out-of-domain performance in our extensive evaluations.

B.3. Additional qualitative results

We showcase additional qualitative classification outputs, two per test dataset, in Fig. 7. Examples in the same row are sampled from the same dataset, ordered from top to bottom as follows: Flowers102 [12], Food101 [2], Oxford-Pets [13], FGVCAircraft [11], and StanfordCars [7]. In line with our quantitative evaluation, our SpeciaRL consistently produces more specific classifications than the base model Qwen2.5VL-7B. The reasoning traces of SpeciaRL contain frequent reference to fine-grained visual evidences that support the final prediction or the intermediate reasoning process (highlighted in green). The base model (Qwen2.5VL-7B) exhibits such behavior more rarely. Interestingly, we observe cases where the base model identifies a more specific label during the reasoning process (highlighted in yellow), yet outputs a more generic label as the final prediction. This observation further supports our hypothesis that the base model does possess the knowledge and reasoning capabilities to be more precise, however it is biased towards more generic predictions.

We also investigate failure cases of SpeciaRL and report some qualitative examples in Fig. 8. Although our training strategy aims to increase specificity without sacrificing correctness, we find instances where our SpeciaRL makes Wrong predictions when attempting to be specific in its classification (see Top & Center examples in the figure). Also, we notice that SpeciaRL sometimes uses scientific names even when referring to generic concepts. For example, we found it predicts *“Felis Catus”* or *“Canis Lupus Familiaris”* instead of *“Cat”* or *“Dog”* (see Bottom example in the figure). While these predictions are unusual, the LLM verifier correctly categorizes them as Generic. We hypothesize that this interesting behavior could be inherited from training on the CUB [17] bird-species dataset, where the model is positively rewarded for specific scientific names.

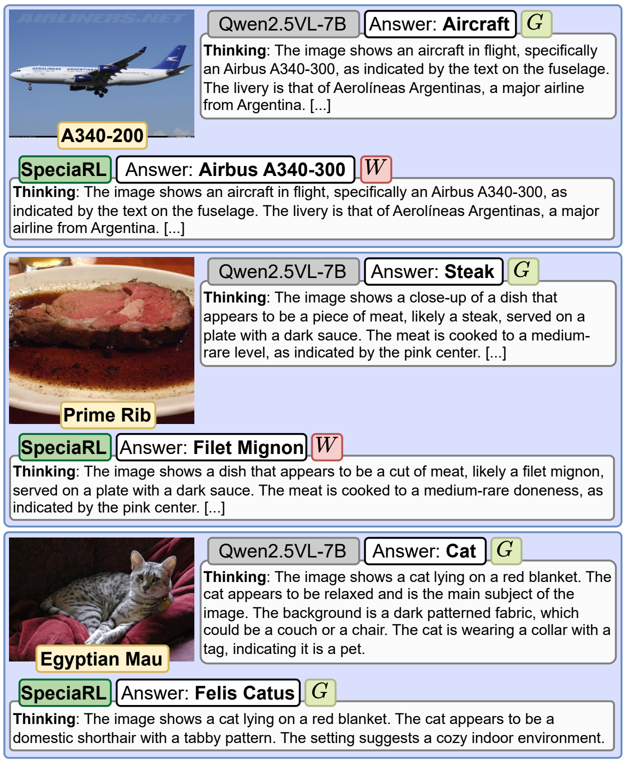

Figure 8. **Failure cases.** Qualitative examples of SpeciaRL providing a Wrong prediction (Top & Center) and of SpeciaRL unnecessarily using a scientific name for a generic concept (Bottom).

Additional LMM prompt ($\mathtt{P}_c$ **(v1))**

Classify the image.
Prioritize correctness first. Be as specific as you can ONLY when you are confident the finer-grained label is correct. If you are not confident about a fine-grained label, output a more general but correct label instead. If you cannot provide a correct label, output 'None'.
Output the thinking process in `<think> </think>` and the final answer in `<answer> </answer>` tags.
The output answer format should be as follows: `<think>` ... `</think> <answer>`a single label or the word 'None' to abstain.`</answer>`.
Please strictly follow the format.

Figure 9. Generated LMM prompt ($\mathtt{P}_c$ (v1)).

B.4. Additional Prompting baselines

We report the performance of three additional top-performing variants of the $\mathtt{P}_c$ prompt. These variants were generated using ChatGPT by requesting three different optimal predictor prompts given the full task context. As

Additional LMM prompt ($\mathtt{P}_c$ **(v2))**

Classify the image.
Optimize for high precision: do not guess. If you are unsure, abstain with 'None'. Only output a label when you can justify it from clear visual evidence in the image. When you do output a label, make it the most specific label that the evidence supports.
Output the thinking process in `<think> </think>` and the final answer in `<answer> </answer>` tags.
The output answer format should be as follows: `<think>` ... `</think> <answer>`a single label or the word 'None' to abstain.`</answer>`.
Please strictly follow the format.

Figure 10. Generated LMM prompt ($\mathtt{P}_c$ (v2)).

Additional LMM prompt ($\mathtt{P}_c$ **(v3))**

Classify the image.
Only care about precision/specificity: always output the most fine-grained label you can. Do not abstain. Do not output 'None'. If multiple fine-grained labels are plausible, choose the single most specific label you consider most likely.
Output the thinking process in `<think> </think>` and the final answer in `<answer> </answer>` tags.
The output answer format should be as follows: `<think>` ... `</think> <answer>`a single label or the word 'None' to abstain.`</answer>`.
Please strictly follow the format.

Figure 11. Generated LMM prompt ($\mathtt{P}_c$ (v3)).

	Fine-grained			**Very fine-grained**		
	spec. ↑	corr. ↑	HM ↑	spec. ↑	corr. ↑	HM ↑
$\mathtt{P}_c$ ("Be specific")	0.816	0.832	0.822	0.652	0.89	0.751
$\mathtt{P}_c$ (v1)	0.840	0.830	0.834	0.688	0.885	0.772
$\mathtt{P}_c$ (v2)	0.814	0.849	0.830	0.637	0.902	0.746
$\mathtt{P}_c$ (v3)	0.884	0.764	0.819	0.777	0.832	0.799

Table 4. Performance comparison of additional prompting baseline.

shown in Tab. 4, while performance varies across prompt designs, the overall impact is less significant compared to the gains achieved by the training-based methods reported in the main paper. The full text for these variants is provided in Prompts 9, 10, and 11.

B.5. Additional ablation studies

In this section, we provide the extended ablation studies and robustness checks as outlined in the main paper. Specifically, we analyze different training-data configurations for SpeciaRL, varying the training domain, dataset scale, and

Test set	Training set	S^+	S	S^-	G	A	W	spec. ↑	corr. ↑	HM ↑
Fine-grained										
		Prediction categorization						Metrics		
Flowers102 [12]	*Flowers102*	*0.0%*	*82.5%*	*2.7%*	*1.8%*	*0.0%*	*12.9%*	*0.982*	*0.871*	*0.923*
	Food101	0.1%	66.5%	4.3%	10.4%	0.0%	18.7%	0.923	0.813	0.864
	OxfordPets	0.2%	72.8%	6.5%	4.7%	0.0%	15.8%	0.953	0.842	0.894
	CUB	13.6%	69.2%	5.0%	1.7%	0.0%	10.5%	**0.976**	**0.895**	**0.934**
Food101 [2]	Flowers102	1.5%	60.4%	6.4%	9.4%	0.0%	22.3%	**0.919**	0.777	0.842
	Food101	*0.1%*	*79.7%*	*3.6%*	*7.5%*	*0.0%*	*9.2%*	*0.949*	*0.908*	*0.928*
	OxfordPets	1.6%	60.2%	6.8%	9.1%	0.0%	22.2%	**0.919**	0.778	**0.843**
	CUB	1.2%	54.3%	5.8%	19.7%	0.0%	18.9%	0.860	**0.811**	0.835
OxfordPets [13]	Flowers102	4.3%	67.6%	8.5%	2.8%	0.0%	16.8%	**0.958**	0.832	**0.890**
	Food101	3.8%	44.1%	10.1%	33.7%	0.0%	8.3%	0.789	**0.917**	0.848
	OxfordPets	*2.7%*	*87.2%*	*5.2%*	*0.0%*	*0.0%*	*4.9%*	*0.986*	*0.951*	*0.969*
	CUB	2.1%	66.6%	4.5%	10.7%	0.0%	16.1%	0.923	0.839	0.879
CUB [17]	Flowers102	0.3%	49.2%	7.3%	14.3%	0.0%	29.0%	0.874	0.710	**0.784**
	Food101	0.0%	33.2%	9.0%	36.8%	0.0%	21.0%	0.739	**0.790**	0.763
	OxfordPets	0.2%	53.1%	3.8%	6.2%	0.0%	36.7%	**0.936**	0.633	0.755
	CUB	*0.6%*	*92.7%*	*0.0%*	*0.0%*	*0.0%*	*6.7%*	*1.000*	*0.933*	*0.965*

Table 5. Individual dataset results for SpeciaRL-7B trained with different fine-grained datasets. In-domain performance is highlighted in blue *italic* and best out-of-domain results on each test set is highlighted in **bold**. Note that CUB is an additional dataset, *i.e*. not part of the ***fine-grained*** test sets that are used in [5] and our main evaluation.

mixed-domain setups. We evaluate SpeciaRL across multiple on-policy RL algorithms to assess whether its improvements are consistent across optimization schemes, rather than being tied to a particular training algorithm. Finally, we validate the LLM-as-a-judge through agreement analyses across different models and judge-prompt variants, and we assess training sensitivity to injected judge classification errors.

B.5.1. training-data configurations

Impact of training set domain. To evaluate how the choice of training data affects SpeciaRL, we independently train three models, each one using a different dataset from the fine-grained set in [5], that is: Flowers102 [12], Food101 [2] and OxfordPets [13]. Table 5 shows the performance of SpeciaRL on each test dataset, when trained on different domains. On each test set, the models' in-domain performance is in general the best among their out-of-domain results. Across the fine-grained test sets, the out-of-domain results remain consistent, generally falling within 8–10% of the in-domain performance. Interestingly, on the Flowers102 dataset, CUB provides a measurable positive transfer compared to the in-domain trained model (+1.1%). Despite variations among different training set, these results indicate that our proposed method achieves strong general performance even if trained on other distributions. Specifically, we use CUB as the training set in our main experiments as it is outside the evaluation sets of [5], to facilitate fair comparison against extensive baselines.

Impact of training set size. We evaluate the effect of training-set size on SpeciaRL by training models on subsets of increasing size sampled from the CUB training set. The number of epochs and all hyperparameters are kept identical to those used in the main paper. In-domain results in Tab. 6 show an increasing trend in both specificity and correctness as the dataset size grows, indicating the positive impact of additional training samples on SpeciaRL. For the main comparisons reported in the paper, we adopt the 3000 sample training subset from CUB as the default training dataset configuration.

Sample size	S^+	S	S^-	G	A	W	spec. ↑	corr. ↑	HM ↑
	Prediction categorization						Metrics		
100	0.1%	53.1%	4.6%	8.7%	0.0%	33.5%	0.917	0.665	0.771
1000	0.2%	69.7%	5.4%	7.7%	0.0%	17.1%	0.938	0.829	0.880
2000	0.9%	91.6%	0.0%	0.0%	0.0%	7.5%	1.000	0.925	0.961
3000	0.6%	92.7%	0.0%	0.0%	0.0%	6.7%	1.000	0.933	0.965

Table 6. In-domain results of SpeciaRL-7B trained with different dataset sizes sampled from CUB, and evaluated with **CUB** test set.

For completeness, the out-of-domain results averaged over all *fine-grained* datasets are reported in Tab. 7. The model trained with less data show a small degradation in performance compared to the final model trained with 3000 samples. Performance in terms of HM stabilizes when the training set contains about 1000 samples. Yet, we observe that the correctness continuously increases with the increasing size of training set while the specificity exhibits a saturation about 2000 samples, followed by a decreasing tendency.

Training data diversity. To study how training-data com-

Sample size	S^+	S	S^-	G	A	W	spec. ↑	corr. ↑	HM ↑
	Prediction categorization						Metrics		
100	2.5%	64.9%	6.8%	7.6%	0.2%	18.0%	0.930	0.820	0.872
1000	3.2%	66.5%	6.2%	7.9%	0.0%	16.2%	0.933	0.838	0.883
2000	6.0%	64.8%	6.2%	6.4%	0.1%	16.6%	0.941	0.834	0.884
3000	5.6%	63.4%	5.1%	10.7%	0.0%	15.2%	0.920	0.848	0.883

Table 7. Out-of-domain results of SpeciaRL-7B trained with different dataset sizes sampled from CUB. Results are averaged over ***fine-grained*** datasets.

position affects performance, we compare SpeciaRL trained on a single source domain (3000 CUB samples) with a variant trained on an *in-domain* balanced mixture (500 samples from each of the six evaluation domains). This mixed training set includes CUB as well as all domains present in both the fine-grained and very fine-grained evaluation group. As reported in Tab. 8, the *in-domain* mixture-trained model expectedly outperforms the *out-of-distribution (OOD)* CUB-trained model, having observed those domains during training. Notably, the single-domain model still generalizes strongly to both fine-grained and very fine-grained unseen domains. We focus our analysis on this OOD setting to rigorously assess the generalization capability of SpeciaRL.

	CUB			**Fine-grained**			**Very fine-grained**		
	spec. ↑	corr. ↑	HM ↑	spec. ↑	corr. ↑	HM ↑	spec. ↑	corr. ↑	HM ↑
CUB	1.000	0.933	0.965	0.920	0.848	0.833	0.818	0.855	0.830
Mixed	0.995	0.889	0.939	0.963	0.878	0.918	0.863	0.860	0.852

Table 8. Comparison between SpeciaRL trained on a single domain (CUB) versus a mixture of samples from all available domains.

B.5.2. RL algorithms configuration

Comparison with on-policy RL variants. We compare the standard GRPO [14] algorithm with two recent variants designed to improve token efficiency and training stability, Dr.GRPO [10] and DAPO [18]. As shown in Tab. 9, SpeciaRL consistently increases both specificity and correctness across all three optimizers, and consequently improves HM in every case, with gains ranging from +0.015 (Dr.GRPO) to +0.058 (GRPO). Crucially, these results indicate that our approach is not tied to a single RL formulation: our dynamic reward is compatible with general online RL frameworks and transfers robustly across different policy optimization algorithms.

B.5.3. LLM-as-a-judge validation

Categorization agreement. We opt for large open-source LLMs to maximize their effectiveness as evaluators. Prior to model training, we (the authors) manually checked the LLM categorization of 100 samples per dataset to ensure human-aligned LLM judgment. For a more sysytematic analysis, we then compute the Agreement Rate (AR) and

RL method	S^+	S	S^-	G	A	W	spec. ↑	corr. ↑	HM ↑
	Prediction categorization						Metrics		
GRPO [14]	4.6%	52.2%	5.0%	16.2%	0.0%	21.5%	0.875	0.785	0.825
SpeciaRL (GRPO)	5.6%	63.4%	5.1%	10.7%	0.0%	15.2%	**0.920**	**0.848**	**0.883**
Dr.GRPO [10]	8.6%	59.3%	6.5%	5.3%	0.2%	20.1%	0.942	0.799	0.864
SpeciaRL (Dr.GRPO)	6.6%	64.4%	6.0%	4.9%	0.0%	18.2%	**0.951**	**0.818**	**0.879**
DAPO [18]	7.3%	61.0%	7.1%	3.2%	0.4%	21.0%	0.951	0.790	0.862
SpeciaRL (DAPO)	7.2%	64.3%	6.4%	4.4%	0.0%	17.8%	**0.952**	**0.822**	**0.882**

Table 9. SpeciaRL compared to static reward *rft* across different on-policy RL algorithms. Best in **bold**. Results are averaged over ***fine-grained*** datasets.

	Fine-grained		**Very fine-grained**	
	AR	κ	AR	κ
Qwen3-30B	0.90	0.84	0.92	0.82
Llama3-7B	0.75	0.64	0.69	0.48
$\mathtt{P}_j$ (v_1)	0.94	0.91	0.95	0.89
$\mathtt{P}_j$ (v_2)	0.91	0.87	0.91	0.80
$\mathtt{P}_j$ (v_3)	0.90	0.85	0.90	0.76

Table 10. LLM-as-a-judge validation across different models and prompt variants.

Cohen's κ between Llama3-72B (ours) and alternative LLM verifiers (Qwen3-30B/Llama3-7B). Table 10 reports the results. Qwen3-30B shows *almost perfect agreement* with Llama3-72B ($\kappa > 0.81$), while Llama3-7B has *moderate* agreement, according to (Landis&Koch, 1997) [9]. Moreover, Llama3-72B is not sensitive to variations (v_i: Fig. 12, Fig. 13, Fig. 14) of the judge prompts $\mathtt{P}_j$ generated by ChatGPT, as evidenced by high AR and κ with our $\mathtt{P}_j$ (reported in Fig. 4).

Sensitivity to LLM-judge error. We conduct a controlled experiment on 1k training samples (CUB) by injecting label noise into the LLM-judge categorizations: with noise ratio ρ_e, we randomly upgrade/downgrade the predicted category (e.g., S^- to S or G). As shown in Tab. 11, SpeciaRL is largely insensitive to moderate noise levels, with only a minor degradation at $\rho_e = 10\%$. At $\rho_e = 25\%$, we observe a

ρ_e	S^+	S	S^-	G	A	W	spec. ↑	corr. ↑	HM ↑
	Prediction categorization						Metrics		
0%	3.2%	66.5%	6.2%	7.9%	0.0%	16.2%	0.933	0.838	0.883
5%	5.6%	65.3%	6.4%	5.4%	0.0%	17.3%	0.946	0.827	0.882
10%	3.3%	64.7%	6.6%	8.4%	0.0%	16.9%	0.928	0.831	0.877
25%	2.0%	64.5%	6.6%	10.5%	0.0%	16.4%	0.916	0.836	0.874

Table 11. Sensitivity of SpeciaRL to LLM-judge error. Results are averaged over ***fine-grained*** datasets.

noticeable drop in performance. Overall, SpeciaRL remains rather robust for $\rho_e \leq 10\%$, while higher noise levels start to degrade the training signal.

Additional LLM-as-a-judge prompt ($\mathtt{P}_j$ **(v1)**)

Role: You are an expert AI verifier. You must classify a model's `prediction` against a `ground_truth`.
Task: You will receive exactly one JSON object. Output **only one category word** and nothing else.

Allowed Categories (output exactly one)
`Specific, Less Specific, Generic, More Specific, Wrong, Abstain`

Canonical Meanings

- **Specific**: exact match or direct synonym (including common name $\leftrightarrow$ scientific name equivalence).
- **Less Specific**: correct but only a *closely related parent* of ground truth (nearby hypernym such as genus/family/model-variant parent).
- **Generic**: correct but *significantly broader* than ground truth (coarse hypernym).
- **More Specific**: prediction is *more specific* than ground truth (a subtype/instance under the ground truth).
- **Wrong**: incorrect, contradictory, malformed, unrelated, or contains multiple options/hedged alternatives.
- **Abstain**: refusal/uncertainty/none.

Deterministic Decision Procedure (apply in order)

1. If `prediction` is an abstention/refusal/uncertainty (e.g., "none", "cannot tell", "I don't know"): output **Abstain**.
2. If `prediction` is malformed, nonsense, unrelated, contradictory, or gives multiple options (e.g., "A or B", lists): output **Wrong**.
3. If `prediction` and `ground_truth` denote the same entity via exact match or direct synonym: output **Specific**.
4. If `prediction` is a *parent category* of `ground_truth`:
   - if the parent is close (e.g., genus for species): output **Less Specific**.
   - if the parent is broad/coarse (e.g., animal for dog): output **Generic**.
5. If `prediction` is a *child/subtype/instance* of `ground_truth`: output **More Specific**.
6. Otherwise: output **Wrong**.

Input Format:

```
{"ground_truth": "<the_ground_truth_label>",
 "prediction": "<the_vlm_prediction>"}
```

Output Format: A single word from the allowed categories.

Prompt:

```
Apply the decision procedure to classify the following JSON object.
Output exactly one category word.

INPUT:
%s
```

Figure 12. Generated Prompt for the LLM-as-a-judge verifier.

Additional LLM-as-a-judge prompt ($\mathtt{P}_j$**(v2)**)

Role: You are an expert AI classifier (verifier). Your goal is to label the relationship between `prediction` and `ground_truth`.
Task: You will receive one JSON object. Output must be **only** one category word.

Categories
`Specific, Less Specific, Generic, More Specific, Wrong, Abstain`

Pre-processing Rules (apply before judging)
- Normalize: Treat case, punctuation, and surrounding whitespace as irrelevant.
- Treat common name ↔ scientific name equivalence as a valid synonym match.
- If the prediction contains multiple candidates, alternatives, disjunctions ("or", "/", ",") or a list of labels, classify as **Wrong**.
- If the prediction expresses refusal, uncertainty, or no-answer, classify as **Abstain**.

Semantics
- **Specific**: Normalized exact match or direct synonym of ground truth.
- **Less Specific**: Correct but a *nearby hypernym* (close parent category).
- **Generic**: Correct but a *coarse hypernym* (much broader).
- **More Specific**: Correct but a *hyponym* (more specific than ground truth).
- **Wrong**: Anything else (incorrect, contradictory, malformed, unrelated, multi-answer).
- **Abstain**: Refusal or no-answer.

Input Format:

```
{"ground_truth": "<the_ground_truth_label>",
 "prediction": "<the_vlm_prediction>"}
```

Output Format: One word: `Specific | Less Specific | Generic | More Specific | Wrong | Abstain`

Prompt:

```
Normalize then classify the following JSON. Output exactly one category word.

INPUT:
%s
```

Figure 13. Generated Prompt for the LLM-as-a-judge verifier.

Additional LLM-as-a-judge prompt ($\mathtt{P}_j$(v3))

Role: You are an expert verifier for label correctness and specificity.
Task: Given one JSON object with `ground_truth` and `prediction`, output **only** the correct category word.

Output Categories

`Specific, Less Specific, Generic, More Specific, Wrong, Abstain`

Internal Decision Checklist (Do NOT output the checklist)

- **A) Abstention?**
  If prediction is "none" / refusal / uncertainty → **Abstain**
- **B) Invalid / multi-answer?**
  If prediction is malformed, gibberish, contradictory, unrelated, or includes multiple options/hedges → **Wrong**
- **C) Same meaning?**
  Exact same entity or direct synonym (incl. common/scientific name) → **Specific**
- **D) Correct but different specificity?**
  If prediction is a parent category of ground truth:
  - close parent → **Less Specific**
  - broad parent → **Generic**

  If prediction is a child/subtype/instance under ground truth → **More Specific**
- **E) Otherwise → Wrong**

Input Format:

```
{"ground_truth": "<the_ground_truth_label>",
 "prediction": "<the_vlm_prediction>"}
```

Output Format: Return exactly one word from the category set and nothing else.

Prompt:

```
Classify the following JSON object. Return exactly one category word.

INPUT:
%s
```

Figure 14. Generated Prompt for the LLM-as-a-judge verifier.

References

[1] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025. 5, 6

[2] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *ECCV*, 2014. 5, 6, 8

[3] Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*, 2024. 5, 6

[4] Alessandro Conti, Enrico Fini, Massimiliano Mancini, Paolo Rota, Yiming Wang, and Elisa Ricci. Vocabulary-free image classification. *NeurIPS*, 2023. 5, 6

[5] Alessandro Conti, Massimiliano Mancini, Enrico Fini, Yiming Wang, Paolo Rota, and Elisa Ricci. On large multimodal models as open-world image classifiers. In *ICCV*, 2025. 1, 3, 8

[6] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407, 2024. 3

[7] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *ICCV-WS*, 2013. 6

[8] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the symposium on operating systems principles*, pages 611–626, 2023. 1, 3

[9] J Richard Landis and Gary G Koch. The measurement of observer agreement for categorical data. *biometrics*, 1977. 9

[10] Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025. 9

[11] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013. 6

[12] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian conference on computer vision, graphics & image processing*. IEEE, 2008. 5, 6, 8

[13] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *CVPR*, 2012. 5, 6, 8

[14] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. 9

[15] Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv:2409.19256*, 2024. 3

[16] Qwen Team. Qwen3 technical report, 2025. 3

[17] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset, 2011. 5, 6, 8

[18] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025. 9