Unlabeled Data Improves Fine-Grained Image Zero-shot Classification with Multimodal LLMs

Yunqi Hong[1], **Sohyun An**[1], **Andrew Bai**[1], **Neil Y. C. Lin**[2], **Cho-Jui Hsieh**[1]
[1]Computer Science Department, University of California, Los Angeles
[2]Mechanical and Aerospace Engineering Department, University of California, Los Angeles
{yunqihong, sohyun0423, andrewbai, chohsieh}@cs.ucla.edu, neillin@g.ucla.edu

Abstract

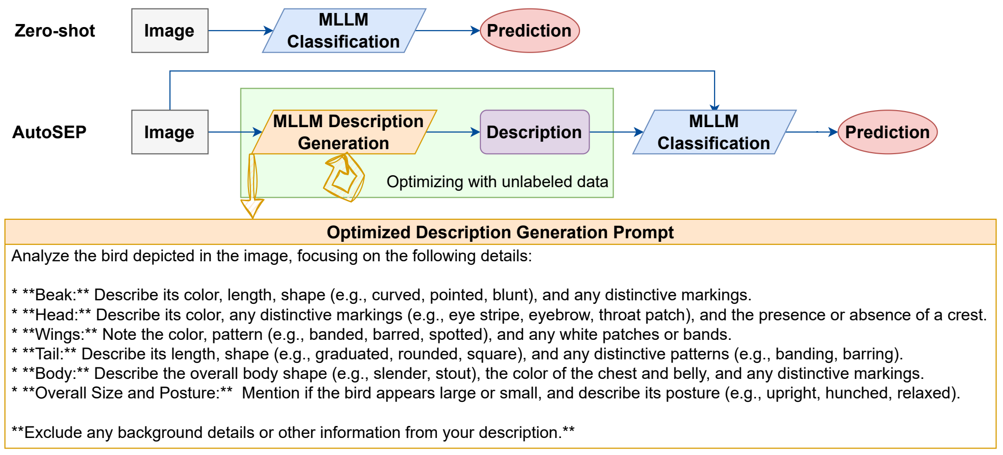
Despite Multimodal Large Language Models (MLLMs) showing promising results on general zero-shot image classification tasks, fine-grained image classification remains challenging. It demands precise attention to subtle visual details to distinguish between visually similar subcategories—details that MLLMs may easily overlook without explicit guidance. To address this, we introduce AutoSEP, an iterative self-supervised prompt learning framework designed to enhance MLLM fine-grained classification capabilities in a fully unsupervised manner. Our core idea is to leverage unlabeled data to learn a description prompt that guides MLLMs in identifying crucial discriminative features within an image, and boosts classification accuracy. We developed an automatic self-enhancing prompt learning framework called AutoSEP to iteratively improve the description prompt using unlabeled data, based on instance-level classification scoring function. AutoSEP only requires black-box access to MLLMs, eliminating the need for any training or fine-tuning. We evaluate our approach on multiple fine-grained classification datasets. It consistently outperforms other unsupervised baselines, demonstrating the effectiveness of our self-supervised optimization framework. Notably, AutoSEP in average improves 13% over standard zero-shot classification and 3% over the best-performing baselines. Code is available at https://github.com/yq-hong/AutoSEP.

1 Introduction

While Multimodal Large Language Models (MLLMs) exhibit impressive zero-shot image classification capabilities on general image datasets [1, 2, 8, 20, 4, 15, 40], their performance often declines on fine-grained classification tasks. These tasks, which involve distinguishing between visually similar subcategories like specific bird, animal, or plant species [12, 10, 21], pose a challenge for MLLMs. The difficulty stems from the need to distinguish subtle visual differences; without explicit guidance, MLLMs tend to overlook crucial details or misinterpret key characteristics [35, 18, 22].

Previous efforts to enhance MLLMs' fine-grained classification abilities have largely relied on training the model with large-scale labeled datasets [12, 38], which are highly resource-intensive, demanding both significant computational resources and vast amounts of high-quality labeled data. This is especially challenging for fine-grained classification tasks, where collecting and annotating data is expensive and time-consuming due to the subtle visual distinctions between classes. In contrast, while unlabeled images are readily available at test time, there has been limited exploration into how such data could be leveraged to enhance classification performance. This motivates us to explore whether MLLMs can self-improve their fine-grained classification ability **using only unlabeled data and with only black-box access to the MLLM**.

But how can unlabeled data be leveraged to improve MLLMs' zero-shot prediction capabilities? While self-supervised learning literature [3, 13, 31] has demonstrated that unlabeled data can be used to train encoders to extract better discriminative features, it is nontrivial to achieve this in a zero-shot and black-box setting. To address this challenge, we develop Automatic Self-Enhancing Prompt Learning (AutoSEP), a novel framework for improving fine-grained visual classification MLLM with unlabeled data (see Figure 1). AutoSEP introduces an intermediate image description step, in which the MLLM is prompted to generate a textual description of an image's visual features. The MLLM then makes the final classification based on the original image and the additional textual description. This image description step can be viewed as mapping the raw image to text that highlights discriminative and relevant features for the task, analogous to the role of encoders in contrastive learning. To evaluate the quality of descriptions using unlabeled data, we design an instance-level classification scoring function that measures the MLLM's ability to retrieve the correct description for each image relative to those of other images. A prompt optimization algorithm is designed to iteratively improve the description-generation prompt based on this scoring function, without requiring white-box access to the MLLMs. By focusing on subtle yet critical visual differences, the MLLM iteratively improves its fine-grained classification performance without requiring any labeled training data.


Zero-shot
Image
MLLM Classification
Prediction
AutoSEP
Image
MLLM Description Generation
Description
MLLM Classification
Prediction
Optimizing with unlabeled data
Optimized Description Generation Prompt

Analyze the bird depicted in the image, focusing on the following details:

* **Beak:** Describe its color, length, shape (e.g., curved, pointed, blunt), and any distinctive markings.
* **Head:** Describe its color, any distinctive markings (e.g., eye stripe, eyebrow, throat patch), and the presence or absence of a crest.
* **Wings:** Note the color, pattern (e.g., banded, barred, spotted), and any white patches or bands.
* **Tail:** Describe its length, shape (e.g., graduated, rounded, square), and any distinctive patterns (e.g., banding, barring).
* **Body:** Describe the overall body shape (e.g., slender, stout), the color of the chest and belly, and any distinctive markings.
* **Overall Size and Posture:** Mention if the bird appears large or small, and describe its posture (e.g., upright, hunched, relaxed).

Exclude any background details or other information from your description.

Figure 1: **Top:** Typical zero-shot pipeline for MLLM classification. **Middle:** Illustration of AutoSEP pipeline. **Bottom:** An example of an AutoSEP-optimized, description-generation prompt.

To summarize, our key contributions are as follows:

- We introduce a new self-supervised, black-box framework, AutoSEP, based on instance-level description retrieval to improve MLLMs fine-grained image classification capabilities.
- We show that AutoSEP significantly enhances MLLMs' fine-grained image classification capabilities (13% average improvement over zero-shot and 3% over the best-performing baselines) on 8 fine-grained tasks.
- To the best of our knowledge, this is the first work studying how to use unlabeled data to improve zero-shot image classification performance in the black-box setting for MLLMs.

2 Related Work

Fine-grained image classification with MLLMs. MLLMs have demonstrated remarkable performance across a broad spectrum of visual tasks, primarily due to pretraining on large-scale datasets [1, 20, 19, 40, 27, 14, 30]. However, since these models are predominantly trained on web-scraped corpora, they tend to underperform on fine-grained classification tasks [12, 10, 21]. To mitigate this limitation, prior works [12, 38] have employed extensive training using large-scale datasets. While effective, these approaches are computationally expensive and heavily reliant on high-quality labeled data, rendering them impractical in scenarios where fine-grained distinctions are critical. This motivates us to explore an alternative approach, which leverages readily available unlabeled images to improve fine-grained classification in MLLMs through a lightweight framework that requires neither labeled data nor white-box access to the model.

Prompt optimization. Automatic prompt optimization techniques have seen substantial progress in recent years, demonstrating their effectiveness in black-box, computationally constrained settings with a wide range of strategies for searching for effective prompts [26, 7, 37, 39, 24]. Specifically, several approaches [39, 24, 32] utilize LLMs to identify and summarize erroneous cases, which are then iteratively used to refine the prompts. Others [11, 9] adopt evolutionary strategies, generating diverse candidates through mutation and selection. Yang et al. [34] treats LLMs as optimizers, leveraging historical prompt trajectories and associated performance signals to guide prompt generation, while Yang et al. [36], Juneja et al. [17] decompose prompts into semantic segments for finer-grained control. Although these methods have shown strong performance, they largely depend on labeled data to supervise the optimization process. To mitigate this dependency, recent work such as SPO [33] formulates prompt optimization in a self-supervised manner by generating candidate prompts with LLMs and evaluating them through LLM-based pairwise preference comparisons of the resulting model outputs. In contrast, we proposed a higher-level framework for fine-grained visual classification that involves leveraging unlabeled images and explicit prompting for textual descriptions, whereas SPO is a lower-level prompt optimization technique. Furthermore, we find that SPO does not perform well even when incorporated into our framework due to its choice of scoring function (see Section 4).

3 Method

The key insight of AutoSEP is leveraging instance-level classification signal, which does not require label information, to inform the optimization of the description generation prompt for improving fine-grained classification. In Section 3.2, we introduce the technique of generating textual descriptions and combining it with image inputs to improve visual classification. In Section 3.3, we detail the procedures for learning the description-generation prompt in a self-supervised manner from instance-level classification scoring.

3.1 Problem Setting

Let M denote the MLLM and q denote the classification prompt. The vanilla zero-shot prediction is

$$\hat{y} = M(x, q), \tag{1}$$

where x is the given image. While MLLMs often achieve strong performance on commonly seen basic-level classes, they tend to underperform on fine-grained classification tasks. We assume there exists a small unlabeled dataset $X = \{x_1, x_2, ..., x_n\}$ (more detailed experiments on different dataset sizes can be found in Section 4.3.) Our goal is to utilize X to improve the zero-shot prediction accuracy on fine-grained categories. Also, we consider only **black-box access** to the MLLM, which means fine-tuning is prohibited.

3.2 Image Description Generation for Fine-grained Classification

Traditional self-supervised learning (SSL) leverages unlabeled data to contrastively learn better representations [3, 13, 31]. Inspired by SSL, we introduce an additional description generation step, where a "description generation prompt" p is used to generate a text description t for an image x. Both the description and the raw image are then jointly passed to the MLLM for predicting the label (see Figure 1):

$$\hat{y} = M(x, [q, t]), \quad \text{where } t = M(x, p). \tag{2}$$

Intuitively, $M(\cdot, p)$ functions as an encoder, translating the input x into a text description t. This description then helps the MLLM conduct the final prediction.

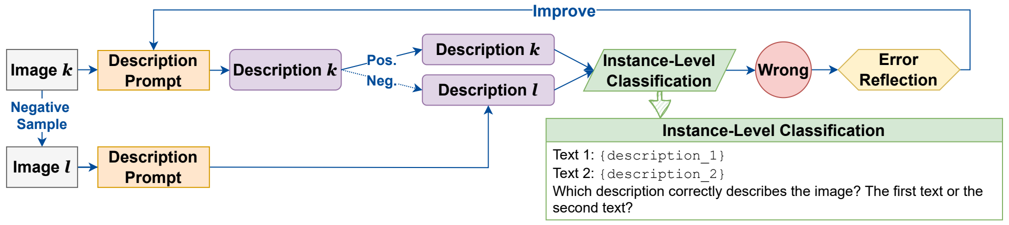
However, a naive description generation prompt yields non-informative descriptions, and even worse, many descriptions might be misleading. Therefore, our method utilize unlabeled data X to iteratively refine how MLLMs analyze and describe images, guiding them away from generic or incorrect visual features towards distinguishing and precise attributes.

3.3 Automatic Self-enhancing Prompt Learning

To learn a better description-generation prompt, we formalize an optimization objective based on a scoring function $\Psi(X, p)$ that evaluates the effectiveness of each candidate prompt p:

$$p^* = \arg\max_{p \in \mathcal{L}} \Psi(X, p), \tag{3}$$

where $\mathcal{L}$ represents the space of coherent natural language prompts.



Text 1: {description_1}
Text 2: {description_2}
Which description correctly describes the image? The first text or the second text?

Figure 2: An illustration of automatic self-enhancing prompt learning with instance-level classification.

3.3.1 Instance-level Classification for Unsupervised Scoring

Inspired by traditional self-supervised learning methods [3, 13, 31], which learn the encoders to pull positive pairs closer and push negative pairs further apart contrastively, we propose to optimize the prompt so that the MLLM generates highly distinctive descriptions for each individual unlabeled image. By doing so, our method maximizes the inter-image distinctiveness of the generated descriptions.

We frame the optimization as an instance-level classification task, where the MLLM learns to correctly associate each image with its own generated description while distinguishing it from the descriptions of other images (See Figure 2). To quantify how well the MLLM can distinguish between different image descriptions, we define a sample matching correctness indicator $V(x_i, x_j)$. Given an image x_i and two candidate descriptions t_i and t_j, which are generated from images x_i and x_j, respectively, we present the MLLM with a binary-choice prompt $G(t_a, t_b)$. The prompt asks "*Text 1:* $\{t_a\}$. *Text 2:* $\{t_b\}$. *Which description correctly describes the image? The first or the second?*", where (t_a, t_b) is either (t_i, t_j) or (t_j, t_i), randomly shuffled with equal probability. The correctness indicator function is then formulated as:

$$V(x_i, x_j) := \begin{cases} \mathbb{1}[M(x_i, G(t_i, t_j)) = \text{“\textit{first}”}], & Z = 0 \\ \mathbb{1}[M(x_i, G(t_j, t_i)) = \text{“\textit{second}”}], & Z = 1, \end{cases} \tag{4}$$

where $Z \sim \text{Bernoulli}(0.5)$ determines the random ordering of descriptions to prevent positional bias. The function $V(x_i, x_j)$ returns 1 if the MLLM correctly selects the description corresponding to x_i, and 0 otherwise.

We compute the unsupervised scoring function that measures the model's ability to correctly match descriptions to their respective images. Formally, the scoring function is defined as:

$$\Psi(X, p) := \sum_{x_i \in X} \sum_{x_j \in X \setminus \{x_i\}} V(x_i, x_j). \tag{5}$$

To avoid iterating over all descriptions, $\Psi(X, p)$ can be approximated with random negative samples:

$$\hat{\Psi}(X, p, k) := \frac{1}{k \cdot |X|} \sum_{x_i \in X} \sum_{x_j \in \mathcal{O}_i} V(x_i, x_j), \tag{6}$$

where $\mathcal{O}_i \sim \text{RandomSample}(X \setminus \{x_i\}, k)$ represents a set of k randomly sampled images from X, excluding x_i, acting as negative samples. $k \cdot |X|$ normalizes the score by the total number of evaluations conducted.

3.3.2 Description-generation Prompt Optimization

To begin the optimization process, we start with a simple yet effective initial generation prompt that instructs the MLLM to describe the object. The prompt can be found in Appendix A.

Feedback and modification. To iteratively refine the prompt, we adopt a feedback-driven approach that identifies and addresses the weaknesses in the current generation prompt. An illustration of AutoSEP workflow and prompts can be found in Appendix B. First, we collect error pairs

$J_{\text{error}} = \{(x_i, x_j) \mid x_i, x_j \in X, x_i \neq x_j, V(x_i, x_j) = 0\}$, where the generated descriptions failed to correctly distinguish between images. Then we sample a subset of J_{error} and construct a diagnostic prompt to analyze potential shortcomings of the current prompt. Specifically, we present the MLLM with the current prompt together with examples of mismatched descriptions, and ask it to explain why the current prompt may have led to these errors. This process, referred to as the *Reflect* operator, $\text{Reflect}(\cdot)$, yields a set of potential reasons explaining the observed errors. Based on MLLM's feedback, we then prompt it to revise the current prompt by explicitly addressing the identified shortcomings. This refinement step is denoted as the *Modify* operator, $\text{Modify}(\cdot)$, which outputs an improved version of the generation prompt. Each instance of feedback produces a candidate revised prompt, resulting in multiple refined versions aimed at enhancing the distinctiveness of the generated descriptions. It has been shown in the literature that this reflect-and-modify approach is effective in automatic prompt engineering [24].

Iterative optimization. Our AutoSEP optimization process follows an iterative framework where the generation prompt is progressively refined to generate highly distinctive descriptions (Algorithm 1). Starting with an initial prompt, we evaluate its effectiveness by generating descriptions for a batch of images and forming an instance-level classification set by sampling negative images. Errors in the instance-level classification are identified and used to expand the prompt by adjusting it to better capture distinctive attributes. At each iteration, a set of candidate prompts is evaluated using a scoring function defined in Section 3.3.1, and only the top-performing prompts are retained for the next round. This iterative adaptation ensures that the MLLM increasingly focuses on fine-grained details, improving its ability to generate more informative descriptions. The process terminates after a predefined number of iterations N, and the best-performing prompt is selected as the final optimized prompt.

By maximizing $\hat{\Psi}(X, p, k)$, we iteratively refine the prompt p to improve the distinctiveness of the generated descriptions, ultimately enhancing the MLLM's ability to perform fine-grained classification in a self-supervised manner.

Algorithm 1 AutoSEP: Automatic Self-Enhancing Prompt Learning

```
Require: $p_0$: initial prompt, $N$: iterations, $X$: unlabeled dataset, $k$: negative samples per image, $b$: top prompts
    retained per iteration, $l$: number of reflections, $\hat{\Psi}(\cdot)$: scoring function
 1: $P_0 \leftarrow \{p_0\}$                                              ▷ Initialize candidate prompt set
 2: $\Gamma \leftarrow \{\gamma_0\}$, where $\gamma_0 = \{M(x_i, p_0) \mid x_i \in X\}$             ▷ Generate descriptions
 3: for $t = 1$ to $N$ do
 4:     $X_t \sim \mathcal{U}(X)$                                          ▷ Random minibatch from $X$
 5:     $P_c \leftarrow P_{t-1}$
 6:     for $p \in P_{t-1}$ do
 7:         $J = \{(x_i, x_j) \mid x_i, x_j \in X_t, i \neq j\}$, where for each $x_i$, $k$ negative samples $x_j$ are randomly
    selected                                                 ▷ Construct instance-level classification set
 8:         $J_{\text{error}} = \{(x_i, x_j) \mid (x_i, x_j) \in J, V(x_i, x_j) = 0\}$                        ▷ Collect errors
 9:         $J^i_{\text{error}} \subset J_{\text{error}}$ is a sampled subset for each $i = 1, \dots, l$
10:         $G = \{g_1, g_2, ...g_l\} = \bigcup_{i=1,...l} \text{Reflect}(p, J^i_{\text{error}})$          ▷ Reflect on the errors (Sec. 3.3.2)
11:         $H = \{h_1, h_2, ...h_l\} = \bigcup_{i=1,...l} \text{Modify}(p, g_i, J^i_{\text{error}})$        ▷ Modify prompts (Sec. 3.3.2)
12:         $P_c \leftarrow P_c \cup H$
13:         $\Gamma \leftarrow \Gamma \cup \{\{M(x_i, h) \mid x_i \in X\} \mid h \in H\}$          ▷ Generate descriptions for new prompts
14:     end for
15:     $S_c = \{\hat{\Psi}(X, p, k) \mid p \in P_c\}$                                      ▷ Evaluate prompts
16:     $P_t \leftarrow \{p \in P_c \mid \hat{\Psi}(X, p, k) \geq \tau\}$, where $\tau$ is the $b^{\text{th}}$ highest score in $S_c$
17: end for
18: Return $p^* \leftarrow \arg\max_{p \in P_N} \hat{\Psi}(X, p, k)$
```

4 Experiments

4.1 Experiment Settings

Datasets. We conduct experiments on four fine-grained image classification datasets, including CUB-200-2011 [29] (bird classification), iNaturalist 2021 [28] (various wild species), Stanford Dogs [6], and VegFru [25]. For each dataset, we construct fine-grained classification tasks on subsets of categories that are particularly challenging for MLLMs. We construct three bird subsets with high confusion rates: ***CUB_cuckoo*** (Black-billed, Mangrove, and Yellow-billed cuckoo), ***CUB_oriole*** (Hooded, Orchard , and Scott's oriole), and ***CUB_vireo*** (Philadelphia, Red-eyed, and Warbling vireo). For iNaturalist, we select two subsets comprising visually similar species: ***iNat_butterfly*** (Symbrenthia lilaea, Claudina crescent, Elada checkerspot) and ***iNat_lupine*** (Arctic lupine, Silvery lupine, Arizona lupine). For Stanford Dogs, we focus on three closely related terriers: ***StanfordDogs_terrier*** (Lakeland terrier, Norwich terrier, Cairn terrier). For the VegFru dataset, we also select two subsets consisting of visually similar species: ***VegFru_greens*** (Dandelion, Shepherd's purse, Prickly lettuce) and ***VegFru_allium*** (Leek, Green Chinese onion, Bunching onion). All the results reported are evaluated on datasets that are balanced in class distribution.

Baselines. We compare our method with both optimization-free and optimization-based approaches. For optimization-free methods, we consider: (1) **Vanilla zero-shot**, where the MLLM is directly prompted without additional context; (2) **Zero-shot with descriptions**, using image descriptions generated from an initial human-crafted prompt; (3) **Zero-shot with majority vote**, where we prompt the MLLM to generate multiple responses with high temperature and take the most frequent prediction as the final answer; (4) **Few-shot with random labels**, where the MLLM is shown m images with randomly assigned labels and then asked to classify the $(m+1)^{\text{th}}$ image; (5) **Few-shot with MLLM labels**, where the MLLM is shown m images with its own predicted labels and then asked to classify the $(m+1)^{\text{th}}$ image; (6) **Multiple images display**, where the MLLM is directly shown m unlabeled images before predicting the $(m+1)^{\text{th}}$ image, as proposed by Jiang et al. [16]; (7) **K-means clustering**, where we cluster image features extracted by a pretrained image encoder, and then within each cluster, use the most frequent zero-shot prediction as the final label for all images in that cluster.

For prompt optimization-based methods, we consider three approaches to adapt them to the unsupervised setting: (1) **Optimization with random labels**, where we randomly assign labels to the images and use the classification accuracy under these random labels as the optimization objective; (2) **Optimization with majority vote**, where we iteratively assign pseudo-labels to images based on majority voting over multiple high-temperature outputs from the MLLM, and use the classification accuracy with respect to these pseudo-labels as the optimization objective; (3) **Self-Supervised Prompt Optimization (SPO)** [33], originally designed for text-only tasks, which perform pairwise comparisons with outputs generated by different prompts to infer their relative quality as signals to guide the optimization. We adopt SPO to our multimodal setting by asking the MLLM to compare the quality of image descriptions generated by different prompts. Following the original setup, we use GPT-4o [14] to conduct these comparisons. Prompts used for these baseline methods can be found in Appendix E.

Models. We conduct experiments with three state-of-the-art MLLMs: Gemini 1.5 Flash [27], GPT-4o [14], and Qwen2-VL-72B-Instruct [30], covering both proprietary and open-source models. We performed the experiments on local servers with 64 CPU cores and 4 Nvidia A6000 GPUs.

4.2 Experiment Results

Experiment results in Table 1 demonstrate that our method consistently outperforms all baseline approaches across all tested MLLMs. Notably, it achieves an average improvement of 13% over standard zero-shot classification and 3% over the best-performing baselines. These consistent gains highlight the robustness and generalizability of our approach across different model architectures and capabilities. Furthermore, our method operates without requiring any labeled data, making it particularly well-suited for low-resource or open-world scenarios.

Description-generation prompt benefits from optimization. We also observe that in some cases, the initial description generation prompt can slightly improve classification performance. However, these improvements are generally marginal. And in some cases, generating descriptions even degrades the performance. This suggests that unguided or naive prompting alone for description generation is insufficient. More principled guidance and optimization are necessary for the model to generate informative and task-relevant descriptions that more effectively support classification.

Table 1: Main results (accuracy %). AutoSEP outperforms baselines consistently across all tasks and MLLMs. Source of variability stems from samples in the evaluation dataset. Gemini refers to Gemini 1.5 Flash.

Model	Gemini	GPT-4o	Qwen2-VL	Gemini	GPT-4o	Qwen2-VL
	CUB_cuckoo			*CUB_oriole*		
Optimization-free						
Vanilla zero-shot	51.68±1.79	61.46±1.24	58.54±4.69	52.13±2.90	74.44±0.93	43.60±2.96
Zero-shot with descriptions	52.20±2.49	68.29±1.00	51.46±2.60	46.74±2.31	73.71±1.52	50.79±3.92
Zero-shot with majority vote	51.22±2.04	61.95±1.95	59.02±1.98	54.38±0.55	75.28±2.01	46.74±4.95
Few-shot with random labels	38.05±6.14	43.90±7.48	46.83±9.74	40.05±12.1	56.85±12.3	36.63±6.50
Few-shot with MLLM labels	53.41±5.31	61.22±2.82	52.93±8.57	57.75±4.42	64.94±7.96	36.85±4.79
Multiple images display [16]	48.05±6.84	52.68±9.27	**65.12**±3.05	**59.78**±6.60	71.46±8.75	46.74±3.80
K-means clustering	53.17±15.2	59.51±15.0	56.59±2.26	51.46±5.83	67.19±8.11	44.94±7.07
Optimization-based						
With random labels	59.15±1.83	65.85±0.77	52.20±3.65	51.69±2.36	72.58±1.83	51.01±1.83
With majority vote	44.88±4.69	67.32±2.10	53.41±2.36	57.90±1.56	73.03±1.74	38.65±1.74
SPO	55.98±1.34	66.59±2.63	56.10±6.19	49.49±2.71	**76.12**±2.16	46.52±3.37
AutoSEP (Ours)	**61.22**±2.30	**75.12**±1.83	63.41±2.77	58.20±0.88	75.96±0.90	**55.13**±3.58
	CUB_vireo			*iNat_butterfly*		
Optimization-free						
Vanilla zero-shot	49.21±3.36	75.73±1.83	55.28±3.05	61.14±2.11	68.00±1.46	40.25±4.65
Zero-shot with descriptions	51.46±4.11	86.07±0.90	50.79±3.72	56.57±3.45	71.14±3.88	48.25±2.48
Zero-shot with majority vote	49.21±1.80	76.69±2.01	55.06±3.01	62.00±2.14	66.07±1.86	46.56±2.59
Few-shot with random labels	45.62±8.03	57.08±18.2	43.82±7.78	45.43±13.8	64.00±10.7	32.13±8.47
Few-shot with MLLM labels	44.94±8.56	72.36±10.5	46.97±7.63	51.43±3.26	82.57±14.3	42.13±12.2
Multiple images display [16]	56.18±7.95	68.54±13.6	63.82±5.34	55.71±1.56	72.29±12.2	32.25±3.44
K-means clustering	40.22±9.25	57.08±11.2	57.53±15.0	58.57±5.89	59.14±5.39	38.38±6.92
Optimization-based						
With random labels	41.24±6.85	72.81±3.13	48.31±4.44	53.24±1.72	79.86±4.43	45.13±1.71
With majority vote	53.71±5.41	87.19±1.83	51.24±2.72	62.29±1.63	61.07±1.89	51.50±3.36
SPO	52.81±9.59	70.34±3.73	51.69±1.59	53.14±2.24	79.71±9.14	45.26±2.47
AutoSEP (Ours)	**59.18**±0.53	**87.42**±1.10	**65.45**±1.24	**66.57**±1.42	**82.71**±2.43	**55.38**±1.00
	iNat_lupine			*StanfordDogs_terrier*		
Optimization-free						
Vanilla zero-shot	62.00±2.72	70.00±3.59	44.22±3.74	60.80±2.13	91.39±0.48	73.20±2.40
Zero-shot with descriptions	56.67±1.92	73.00±1.25	44.00±2.49	66.22±2.41	91.42±1.17	76.00±2.53
Zero-shot with majority vote	61.78±1.24	72.33±1.70	47.33±2.06	62.67±0.73	90.89±1.09	76.27±2.13
Few-shot with random labels	47.56±7.31	48.33±5.48	42.00±7.38	60.27±10.3	66.89±24.7	52.13±8.90
Few-shot with MLLM labels	60.22±11.4	69.33±3.89	46.44±6.75	66.67±10.3	87.56±3.47	64.67±6.84
Multiple images display [16]	63.47±7.71	74.00±5.54	50.44±2.86	65.87±4.04	79.78±9.04	77.73±4.43
K-means clustering	47.11±6.50	50.00±2.11	30.00±1.99	39.60±7.50	68.67±1.09	64.13±5.66
Optimization-based						
With random labels	58.44±4.80	73.67±1.94	46.00±3.62	53.33±1.94	90.00±1.11	63.73±1.87
With majority vote	59.33±1.81	74.17±1.83	47.33±4.37	62.27±1.82	91.11±1.41	77.74±1.07
SPO	56.44±3.17	64.88±6.12	46.56±3.44	65.16±6.10	81.47±2.12	64.27±2.00
AutoSEP (Ours)	**66.15**±1.47	**76.60**±1.40	**54.67**±3.33	**69.28**±0.97	**92.18**±1.09	**81.70**±1.20
	VegFru_greens			*VegFru_allium*		
Optimization-free						
Vanilla zero-shot	86.00±0.56	78.82±0.54	68.89±3.06	72.67±3.18	60.44±1.66	54.22±3.40
Zero-shot with descriptions	77.11±3.19	77.33±1.94	63.61±1.82	74.44±1.98	64.89±1.51	59.17±1.98
Zero-shot with majority vote	85.83±0.48	78.00±0.44	71.39±2.53	75.56±1.82	62.44±2.57	56.89±2.37
Few-shot with random labels	74.44±6.30	70.22±8.47	33.33±0.00	67.33±3.18	61.33±10.6	33.33±0.00
Few-shot with MLLM labels	84.44±7.24	78.89±7.13	68.44±5.14	75.11±7.39	**73.11**±9.23	59.11±15.1
Multiple images display [16]	82.44±4.30	72.89±5.14	70.22±10.6	**79.56**±2.69	66.00±7.29	59.56±4.25
K-means clustering	67.78±5.24	67.56±4.94	64.00±5.19	51.33±14.1	50.89±9.33	42.67±7.72
Optimization-based						
With random labels	80.56±4.33	79.78±0.44	48.52±2.77	71.56±2.06	64.22±1.47	55.19±4.48
With majority vote	81.19±4.27	74.00±0.89	55.56±0.91	74.22±0.44	64.67±0.83	56.94±1.21
SPO	84.37±5.26	78.40±0.87	57.78±3.14	73.78±1.81	65.56±2.90	60.00±0.91
AutoSEP (Ours)	**87.78**±0.70	**81.11**±0.70	**72.22**±0.91	78.89±0.99	69.11±0.83	**62.44**±2.44

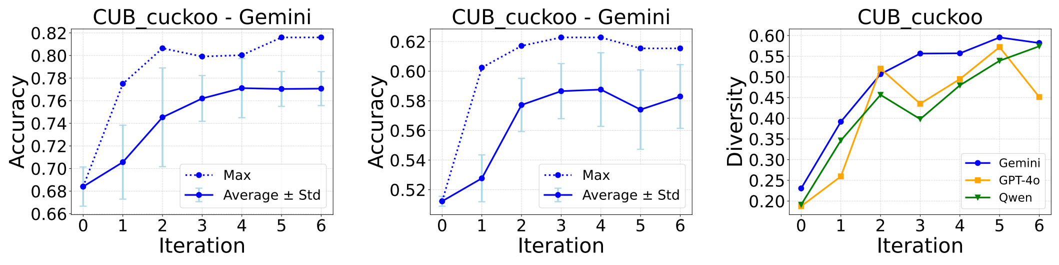
Instance-level classification is an effective unsupervised scoring signal, outperforming alternative unsupervised scoring strategies. A key component of our method is the use of instance-level classification accuracy as the unsupervised scoring signal during optimization. To better understand its effectiveness, we compare it against three alternative unsupervised scoring methods for optimization: (1) Classification accuracy with randomly assigned labels; (2) Classification accuracy with majority vote pseudo-labels, where pseudo-labels are obtained by aggregating high-temperature predictions from the MLLM; (3) Pairwise output evaluation in SPO [33], which relies on pairwise comparisons of model outputs to estimate relative prompt quality.

While these alternatives sometimes yield performance gains over optimization-free baselines, they exhibit limited reliability and overall lower performance compared to our instance-level classification approach. Majority vote pseudo-labeling tends to perform better than using random labels, but both methods fall short in stability and effectiveness. Although SPO has demonstrated strong performance on several text-only reasoning and question answering tasks, it does not transfer well to fine-grained image classification. The results suggest that while LLMs can effectively compare the quality of reasoning paths in purely textual contexts, MLLMs struggle to consistently evaluate the informativeness of image descriptions for fine-grained classification. As a result, such evaluations offer weak and often noisy feedback signals for optimization. These results highlight the importance of instance-level classification as a robust and effective unsupervised signal.

4.3 Discussion and Analysis

Learning dynamics. We study how the classification performance evolves throughout the optimization process and how many iterations are usually required for the algorithm to take effect. Full details of experimental settings are mentioned in Appendix C.1. The average results and their variances reported in Figure 3a and 3b are computed by averaging these classification performances across all three runs. The Max results are computed by taking the best performance among the four prompts at each iteration, then averaging across the three independent runs.

Both the instance-level (Figure 3a) and class-wise (Figure 3b) classification performance consistently improves with more optimization iterations. More demonstrations on other datasets can be found in Appendix C.1. The correlational evidence suggests that instance-level classification serves as an effective unsupervised optimization signal to guide the model toward better class-wise discrimination. These results validate our design of leveraging instance-level feedback to drive prompt optimization for fine-grained classification in the absence of labeled data.



(a) Instance-level Classification (b) Class-wise Classification (c) Prompt word diversity

Figure 3: Evolution of metrics with increasing optimization iterations.

Correlation between instance-level classification and class-wise classification. To further support the effectiveness of instance-level classification as an optimization signal, we also compute the statistical Pearson correlation between instance-level and class-wise classification performance. Table 2 shows strong positive Pearson correlation across different settings, suggesting that improvements in instance-level accuracy are closely associated with better class-wise classification performance. This empirical evidence reinforces our hypothesis that instance-level supervision can serve as a powerful proxy to guide the description optimization for fine-grained image classification in the absence of labeled data.

Diversity of generation prompts. To analyze how much new information is introduced during the optimization process, we evaluate the diversity of the generated prompts using a quantitative diversity

Table 2: Correlation between instance-level classification and class-wise classification.

Datasets	*CUB_cuckoo*	*CUB_oriole*	*CUB_vireo*	*iNat_butterfly*	*iNat_lupine*
Gemini 1.5 Flash	0.72±0.11	0.47±0.24	0.70±0.19	0.63±0.22	0.53±0.02
Qwen2-VL-72B-Instruct	0.63±0.13	0.38±0.15	0.64±0.10	0.84±0.11	0.70±0.17

score. Specifically, for each prompt, we count the number of semantic keywords (lemmas of nouns, verbs, and adjectives excluding stop words), as well as the number of unique words, defined as words that appear only in that specific prompt and not in any other prompt throughout the optimization. These counts are normalized to compute the diversity score for each prompt. Detailed definition can be found in Appendix F.

As shown in Figure 3c, the diversity score steadily increases over iterations, indicating that the optimization process encourages the generation of increasingly varied and semantically rich prompts. This trend suggests that the model is progressively exploring a broader and more informative prompt space, which may potentially contribute to improved classification performance.

Computational complexity. Let b denote the number of top prompts retained, l is the number of prompt reflections generated per retained prompt, k is the number of negative samples used per image in the instance-level classification, and n is the number of unlabeled images. Each iteration of our algorithm involves generating $b \cdot l$ reflections, and $b \cdot l$ modified candidate prompts accordingly. For each of these prompts, we generate descriptions for all n images, resulting in $b \cdot l \cdot n$ quiries to the MLLM for description generation. Additionally, instance-level classification requires comparing each image's description against k negative samples under each prompt, leading to $b \cdot l \cdot k \cdot n$ further MLLM queries. Therefore, the total number of MLLM queries per iteration is $(2 + n + kn) \cdot bl$. In our implementation, we typically set $b = 4, l \leq 5$, and $k = 2$, resulting in an approximate query complexity of $\mathcal{O}(60n)$ per iteration.

For optimization-free baselines, the computational cost is relatively low, since they do not require iterative prompt refinement. For optimization-based baselines, we used the same experimental setup as AutoSEP, resulting in comparable computational complexity. More specifically, optimization with random labels takes approximately $\mathcal{O}(2(1 + n) \cdot bl) = \mathcal{O}(40n)$ MLLM queries per iteration, as it involves only a single classification per image. Optimization with majority vote takes approximately $\mathcal{O}(2(1 + n) \cdot bl + cn) = \mathcal{O}(45n)$, with $c = 5$ denoting the number of majority votes collected for updates in each iteration. SPO [33] is more lightweight, requiring only $\mathcal{O}(2n)$ queries per iteration, since it only compares with the previous prompt and does not rely on several candidate prompts.

The actual runtime depends on the inference speed of the MLLMs. For example, one optimization iteration on 60 images with Gemini 1.5 Flash takes 12 minutes in average, demonstrating that AutoSEP is computationally feasible for practical use.

More samples, stronger instance-level signals, more stable optimization. We investigate how the number of unlabeled samples used during optimization impacts the performance of our method. Figure 4 shows a clear trend that the classification performance improves with increased number of samples. More demonstrations on other datasets can be found in Appendix C.2.

When the sample size is too small, instance-level classification is constrained by limited variation and coverage, making it difficult to reliably evaluate a prompt's ability to generate distinctive and meaningful descriptions. This leads to suboptimal performance and introduces more noise and instability into the optimization process. In contrast, with more samples, the instance-level signal becomes more robust, enabling more stable optimization and resulting in consistent gains in class-wise classification accuracy. Notably, our method begins to perform reliably with **as few as 60 unlabeled samples**, demonstrating its practicality and scalability even in limited-data settings.

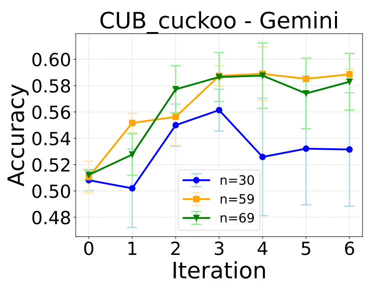


Figure 4: Classification accuracy of Gemini with various number of samples for optimization.

5 Conclusion

In this paper, we present a self-supervised prompt learning framework that improves the fine-grained image classification capabilities of MLLMs using only unlabeled data and black-box MLLM access. By optimizing the prompt for image description generation through instance-level classification feedback, our method enables MLLMs to focus on subtle, discriminative features critical for distinguishing between visually similar categories. Extensive experiments across multiple fine-grained classification tasks and MLLMs demonstrate the robustness of our approach.

Limitations. Our method relies on the implicit assumption that subtle, class-discriminative visual features can be accurately captured and described using natural language. This may not hold in cases where visual distinctions are extremely fine-grained or difficult to verbalize. Additionally, although our approach is fully unsupervised, it still requires a sufficient number of unlabeled samples to ensure stable and effective prompt optimization. The absence of labeled data inherently limits the performance compared to fully supervised methods that benefit from explicit class-level guidance.

Acknowledgments and Disclosure of Funding

This work is supported by NSF 2048280, 2325121, 2244760, 2331966 and ONR N00014- 23-1-2300:P00001.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning, 2022. URL https://arxiv.org/abs/2204.14198.

[2] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*, 1(2):3, 2023.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PmLR, 2020.

[4] Zhe Chen, Weiyun Wang, Hao Tian, Shenglong Ye, Zhangwei Gao, Erfei Cui, Wenwen Tong, Kongzhi Hu, Jiapeng Luo, Zheng Ma, et al. How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites. *Science China Information Sciences*, 67(12):220101, 2024.

[5] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

[6] E Dataset. Novel datasets for fine-grained image categorization. In *First workshop on fine grained visual categorization, CVPR. Citeseer. Citeseer. Citeseer*, volume 5, page 2. Citeseer, 2011.

[7] Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric P. Xing, and Zhiting Hu. Rlprompt: Optimizing discrete text prompts with reinforcement learning, 2022. URL https://arxiv.org/abs/2205.12548.

[8] Xiaoyi Dong, Pan Zhang, Yuhang Zang, Yuhang Cao, Bin Wang, Linke Ouyang, Xilin Wei, Songyang Zhang, Haodong Duan, Maosong Cao, et al. Internlm-xcomposer2: Mastering free-form text-image composition and comprehension in vision-language large model. *arXiv preprint arXiv:2401.16420*, 2024.

[9] Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution, 2023. URL https://arxiv.org/abs/2309.16797.

[10] Gregor Geigle, Radu Timofte, and Goran Glavaš. African or european swallow? benchmarking large vision-language models for fine-grained object classification. *arXiv preprint arXiv:2406.14496*, 2024.

[11] Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. Evoprompt: Connecting llms with evolutionary algorithms yields powerful prompt optimizers, 2025. URL https://arxiv.org/abs/2309.08532.

[12] Hulingxiao He, Geng Li, Zijun Geng, Jinglin Xu, and Yuxin Peng. Analyzing and boosting the power of fine-grained visual recognition for multi-modal large language models. *arXiv preprint arXiv:2501.15140*, 2025.

[13] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[14] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[15] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

[16] Yixing Jiang, Jeremy Irvin, Ji Hun Wang, Muhammad Ahmed Chaudhry, Jonathan H Chen, and Andrew Y Ng. Many-shot in-context learning in multimodal foundation models. *arXiv preprint arXiv:2405.09798*, 2024.

[17] Gurusha Juneja, Nagarajan Natarajan, Hua Li, Jian Jiao, and Amit Sharma. Task facet learning: A structured approach to prompt optimization, 2024. URL https://arxiv.org/abs/2406.10504.

[18] Saehyung Lee, Seunghyun Yoon, Trung Bui, Jing Shi, and Sungroh Yoon. Toward robust hyper-detailed image captioning: A multiagent approach and dual evaluation metrics for factuality and coverage. *arXiv preprint arXiv:2412.15484*, 2024.

[19] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models, 2023. URL https://arxiv.org/abs/2301.12597.

[20] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

[21] Mingxuan Liu, Subhankar Roy, Wenjing Li, Zhun Zhong, Nicu Sebe, and Elisa Ricci. Democratizing fine-grained visual recognition with large language models. *arXiv preprint arXiv:2401.13837*, 2024.

[22] Yexin Liu, Zhengyang Liang, Yueze Wang, Xianfeng Wu, Feilong Tang, Muyang He, Jian Li, Zheng Liu, Harry Yang, Sernam Lim, and Bo Zhao. Unveiling the ignorance of mllms: Seeing clearly, answering incorrectly, 2025. URL https://arxiv.org/abs/2406.10638.

[23] Rithesh Murthy, Ming Zhu, Liangwei Yang, Jielin Qiu, Juntao Tan, Shelby Heinecke, Caiming Xiong, Silvio Savarese, and Huan Wang. Promptomatix: An automatic prompt optimization framework for large language models. *arXiv preprint arXiv:2507.14241*, 2025.

[24] Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. Automatic prompt optimization with" gradient descent" and beam search. *arXiv preprint arXiv:2305.03495*, 2023.

[25] Yushan Feng Saihui Hou and Zilei Wang. Vegfru: A domain-specific dataset for fine-grained visual categorization. In *IEEE International Conference on Computer Vision*, 2017.

[26] Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts, 2020. URL https://arxiv.org/abs/2010.15980.

[27] Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

[28] Grant Van Horn, Elijah Cole, Sara Beery, Kimberly Wilber, Serge Belongie, and Oisin Mac Aodha. Benchmarking representation learning for natural world image collections. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12884–12893, 2021.

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

[30] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[31] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International conference on machine learning*, pages 9929–9939. PMLR, 2020.

[32] Xinyuan Wang, Chenxi Li, Zhen Wang, Fan Bai, Haotian Luo, Jiayou Zhang, Nebojsa Jojic, Eric P. Xing, and Zhiting Hu. Promptagent: Strategic planning with language models enables expert-level prompt optimization, 2023. URL https://arxiv.org/abs/2310.16427.

[33] Jinyu Xiang, Jiayi Zhang, Zhaoyang Yu, Fengwei Teng, Jinhao Tu, Xinbing Liang, Sirui Hong, Chenglin Wu, and Yuyu Luo. Self-supervised prompt optimization. *arXiv preprint arXiv:2502.06855*, 2025.

[34] Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers, 2024. URL https://arxiv.org/abs/2309.03409.

[35] Dingchen Yang, Bowen Cao, Guang Chen, and Changjun Jiang. Pensieve: Retrospect-then-compare mitigates visual hallucination. *arXiv preprint arXiv:2403.14401*, 2024.

[36] Muchen Yang, Moxin Li, Yongle Li, Zijun Chen, Chongming Gao, Junqi Zhang, Yangyang Li, and Fuli Feng. Dual-phase accelerated prompt optimization, 2024. URL https://arxiv.org/abs/2406.13443.

[37] Tianjun Zhang, Xuezhi Wang, Denny Zhou, Dale Schuurmans, and Joseph E. Gonzalez. Tempera: Test-time prompting via reinforcement learning, 2022. URL https://arxiv.org/abs/2211.11890.

[38] Yuhui Zhang, Alyssa Unell, Xiaohan Wang, Dhruba Ghosh, Yuchang Su, Ludwig Schmidt, and Serena Yeung-Levy. Why are visually-grounded language models bad at image classification? *arXiv preprint arXiv:2405.18415*, 2024.

[39] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers, 2023. URL https://arxiv.org/abs/2211.01910.

[40] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 5

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.

- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: [NA]

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.

- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4.1, Section 4.3 Computational Complexity, and Appendix E.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Table 1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Section 4.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.

- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: [Yes]

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: The paper introduces a technique for improving visual fine-grained classification, which is a net positive impact according to our understanding.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: No risk that we are aware of.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Sources are properly cited.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Code is anonymized and included in the supplementary materials.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: No crowdsourcing.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

Justification: No human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Examples of Descriptions

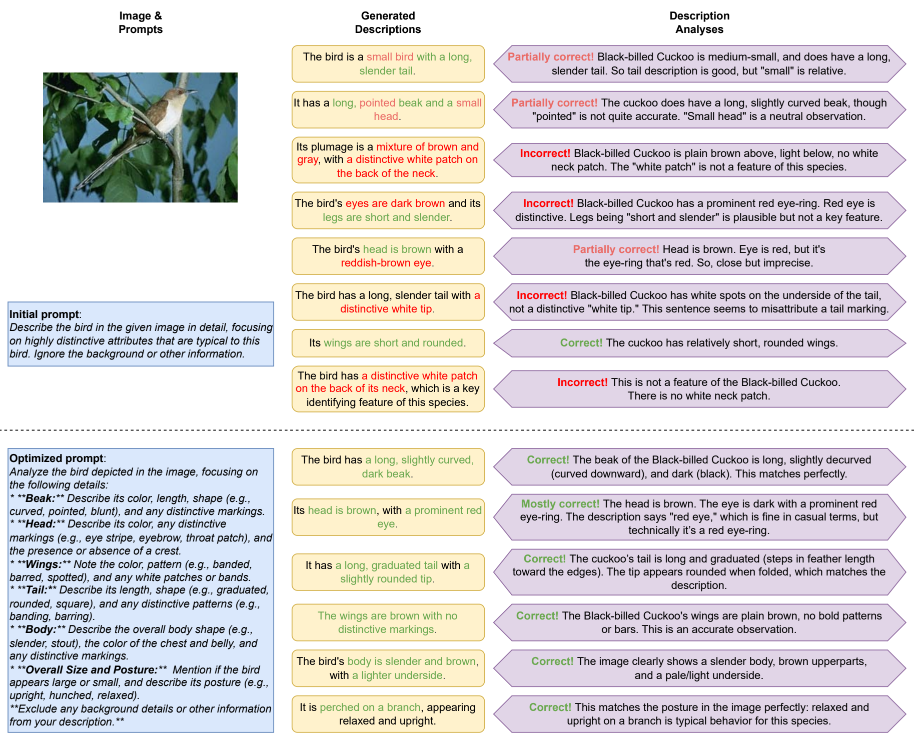


Image & Prompts	Generated Descriptions	Description Analyses
	The bird is a small bird with a long, slender tail.	Partially correct! Black-billed Cuckoo is medium-small, and does have a long, slender tail. So tail description is good, but "small" is relative.
	It has a long, pointed beak and a small head.	Partially correct! The cuckoo does have a long, slightly curved beak, though "pointed" is not quite accurate. "Small head" is a neutral observation.
	Its plumage is a mixture of brown and gray, with a distinctive white patch on the back of the neck.	Incorrect! Black-billed Cuckoo is plain brown above, light below, no white neck patch. The "white patch" is not a feature of this species.
	The bird's eyes are dark brown and its legs are short and slender.	Incorrect! Black-billed Cuckoo has a prominent red eye-ring. Red eye is distinctive. Legs being "short and slender" is plausible but not a key feature.
	The bird's head is brown with a reddish-brown eye.	Partially correct! Head is brown. Eye is red, but it's the eye-ring that's red. So, close but imprecise.
Initial prompt: *Describe the bird in the given image in detail, focusing on highly distinctive attributes that are typical to this bird. Ignore the background or other information.*	The bird has a long, slender tail with a distinctive white tip.	Incorrect! Black-billed Cuckoo has white spots on the underside of the tail, not a distinctive "white tip." This sentence seems to misattribute a tail marking.
	Its wings are short and rounded.	Correct! The cuckoo has relatively short, rounded wings.
	The bird has a distinctive white patch on the back of its neck, which is a key identifying feature of this species.	Incorrect! This is not a feature of the Black-billed Cuckoo. There is no white neck patch.

Optimized prompt:
Analyze the bird depicted in the image, focusing on the following details:
** **Beak:** Describe its color, length, shape (e.g., curved, pointed, blunt), and any distinctive markings.*
** **Head:** Describe its color, any distinctive markings (e.g., eye stripe, eyebrow, throat patch), and the presence or absence of a crest.*
** **Wings:** Note the color, pattern (e.g., banded, barred, spotted), and any white patches or bands.*
** **Tail:** Describe its length, shape (e.g., graduated, rounded, square), and any distinctive patterns (e.g., banding, barring).*
** **Body:** Describe the overall body shape (e.g., slender, stout), the color of the chest and belly, and any distinctive markings.*
** **Overall Size and Posture:** Mention if the bird appears large or small, and describe its posture (e.g., upright, hunched, relaxed).*
Exclude any background details or other information from your description.

Generated Descriptions	Description Analyses
The bird has a long, slightly curved, dark beak.	Correct! The beak of the Black-billed Cuckoo is long, slightly decurved (curved downward), and dark (black). This matches perfectly.
Its head is brown, with a prominent red eye.	Mostly correct! The head is brown. The eye is dark with a prominent red eye-ring. The description says "red eye," which is fine in casual terms, but technically it's a red eye-ring.
It has a long, graduated tail with a slightly rounded tip.	Correct! The cuckoo's tail is long and graduated (steps in feather length toward the edges). The tip appears rounded when folded, which matches the description.
The wings are brown with no distinctive markings.	Correct! The Black-billed Cuckoo's wings are plain brown, no bold patterns or bars. This is an accurate observation.
The bird's body is slender and brown, with a lighter underside.	Correct! The image clearly shows a slender body, brown upperparts, and a pale/light underside.
It is perched on a branch, appearing relaxed and upright.	Correct! This matches the posture in the image perfectly: relaxed and upright on a branch is typical behavior for this species.

(a) Black-billed Cuckoo

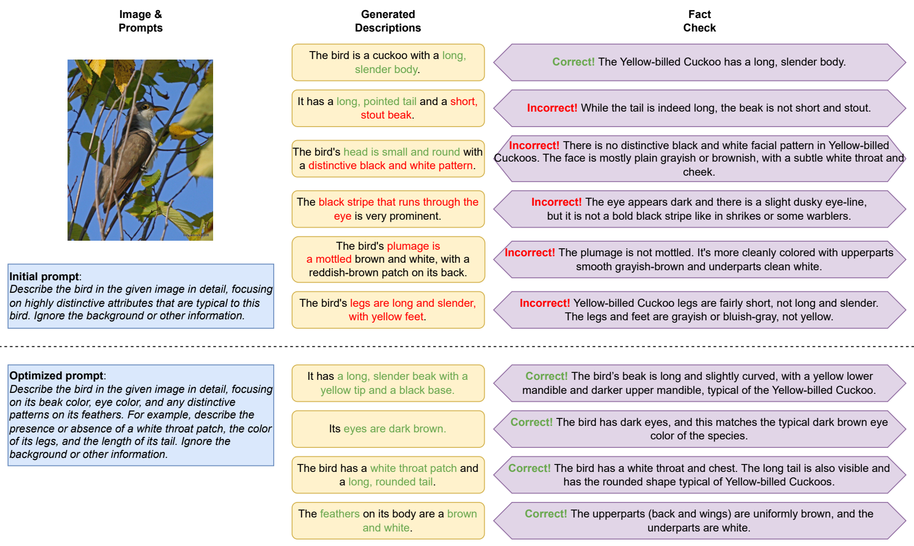


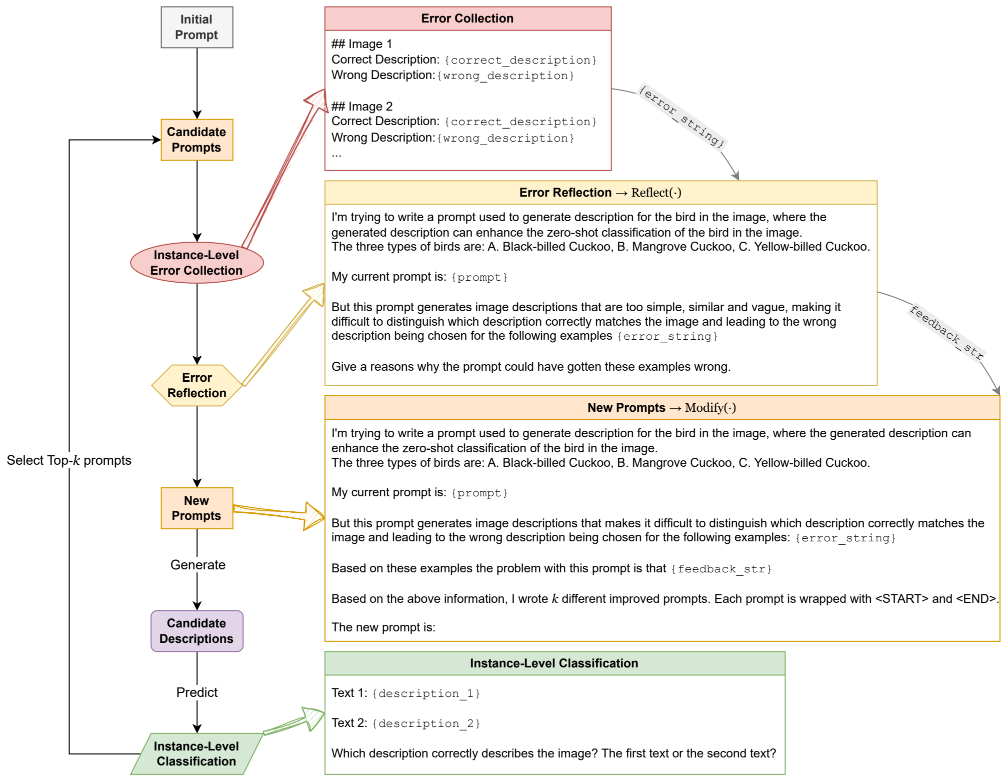
Image & Prompts	Generated Descriptions	Fact Check
	The bird is a cuckoo with a long, slender body.	Correct! The Yellow-billed Cuckoo has a long, slender body.
	It has a long, pointed tail and a short, stout beak.	Incorrect! While the tail is indeed long, the beak is not short and stout.
	The bird's head is small and round with a distinctive black and white pattern.	Incorrect! There is no distinctive black and white facial pattern in Yellow-billed Cuckoos. The face is mostly plain grayish or brownish, with a subtle white throat and cheek.
	The black stripe that runs through the eye is very prominent.	Incorrect! The eye appears dark and there is a slight dusky eye-line, but it is not a bold black stripe like in shrikes or some warblers.
	The bird's plumage is a mottled brown and white, with a reddish-brown patch on its back.	Incorrect! The plumage is not mottled. It's more cleanly colored with upperparts smooth grayish-brown and underparts clean white.
Initial prompt: *Describe the bird in the given image in detail, focusing on highly distinctive attributes that are typical to this bird. Ignore the background or other information.*	The bird's legs are long and slender, with yellow feet.	Incorrect! Yellow-billed Cuckoo legs are fairly short, not long and slender. The legs and feet are grayish or bluish-gray, not yellow.
Optimized prompt: *Describe the bird in the given image in detail, focusing on its beak color, eye color, and any distinctive patterns on its feathers. For example, describe the presence or absence of a white throat patch, the color of its legs, and the length of its tail. Ignore the background or other information.*	It has a long, slender beak with a yellow tip and a black base.	Correct! The bird's beak is long and slightly curved, with a yellow lower mandible and darker upper mandible, typical of the Yellow-billed Cuckoo.
	Its eyes are dark brown.	Correct! The bird has dark eyes, and this matches the typical dark brown eye color of the species.
	The bird has a white throat patch and a long, rounded tail.	Correct! The bird has a white throat and chest. The long tail is also visible and has the rounded shape typical of Yellow-billed Cuckoos.
	The feathers on its body are a brown and white.	Correct! The upperparts (back and wings) are uniformly brown, and the underparts are white.

(b) Yellow-billed Cuckoo

Figure 5: Examples showcasing descriptions generated from two different prompts using Gemini 1.5 Flash. Attributes highlighted in green indicate correct information, while those in red denote incorrect attributes.

Figure 5 shows two exampls of image descriptions generated by Gemini 1.5 Flash [27] using prompts before and after optimization. The initial descriptions tend to be vague or misleading, often misrepresenting key visual attributes. In contrast, the optimized prompts lead to more accurate and reliable descriptions that better reflect the distinctive features of the images.

B Detailed AutoSEP Workflow and Prompts



Error Collection

Image 1
Correct Description: `{correct_description}`
Wrong Description: `{wrong_description}`

Image 2
Correct Description: `{correct_description}`
Wrong Description: `{wrong_description}`
...

Error Reflection → Reflect(·)

I'm trying to write a prompt used to generate description for the bird in the image, where the generated description can enhance the zero-shot classification of the bird in the image.
The three types of birds are: A. Black-billed Cuckoo, B. Mangrove Cuckoo, C. Yellow-billed Cuckoo.

My current prompt is: `{prompt}`

But this prompt generates image descriptions that are too simple, similar and vague, making it difficult to distinguish which description correctly matches the image and leading to the wrong description being chosen for the following examples `{error_string}`

Give a reasons why the prompt could have gotten these examples wrong.

New Prompts → Modify(·)

I'm trying to write a prompt used to generate description for the bird in the image, where the generated description can enhance the zero-shot classification of the bird in the image.
The three types of birds are: A. Black-billed Cuckoo, B. Mangrove Cuckoo, C. Yellow-billed Cuckoo.

My current prompt is: `{prompt}`

But this prompt generates image descriptions that makes it difficult to distinguish which description correctly matches the image and leading to the wrong description being chosen for the following examples: `{error_string}`

Based on these examples the problem with this prompt is that `{feedback_str}`

Based on the above information, I wrote k different improved prompts. Each prompt is wrapped with <START> and <END>.

The new prompt is:

Instance-Level Classification

Text 1: `{description_1}`

Text 2: `{description_2}`

Which description correctly describes the image? The first text or the second text?

Figure 6: **Workflow** of AutoSEP with Optimization Prompts. Take *CUB_cuckoo* as an example.

Figure 6 shows the detailed workflow of AutoSEP and prompts used in the optimization. The prompt learning procedure is introduced in Section 3.3.2.

C Additional Ablation Studies

C.1 Learning Dynamics

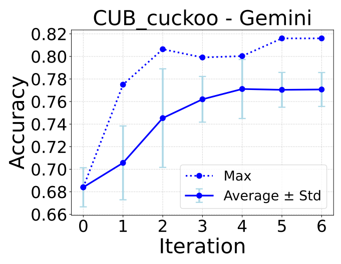
We study how the classification performance evolves throughout the optimization process and how many iterations are usually required for the algorithm to take effect. For each experimental setting, we perform three independent runs. In each run, we retain the top four candidate prompts at every iteration based on the scoring function. We use the average classification performance of these four prompts to represent the performance for that iteration. The average results and their variances reported in Figure 7 are computed by averaging these classification performances across all three runs. The Max results are computed by taking the best performance among the four prompts at each iteration, then averaging across the three independent runs.

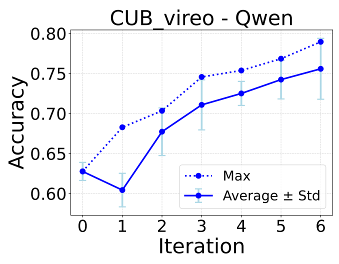
C.2 Number of Samples

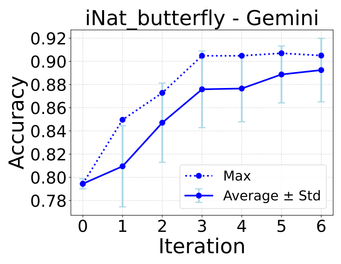
More experiments on different datasets are conducted to study the effect of the number of samples on our AutoSEP framework (See Figure 8). The results are quite consistent across different classification tasks. When the sample size is too small, the optimization process is unstable with large variance. In contrast, with more samples, the instance-level signal becomes more robust, resulting in more stable optimization and consistent gains in class-wise classification accuracy.

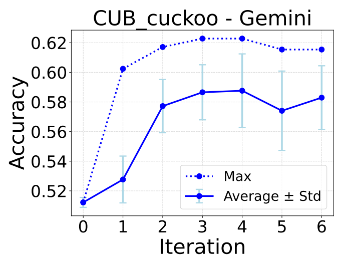
C.3 Different Sampling Methods

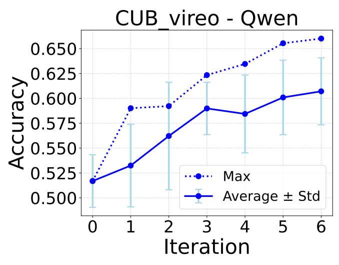
We conducted this ablation study to examine how the sampling strategy affects the final results. Specifically, we replaced the random image sampling in AutoSEP with two alternatives: sampling only same-class images


(a) Instance-level Classification


(b) Instance-level Classification


(c) Instance-level Classification


(d) Class-wise Classification


(e) Class-wise Classification

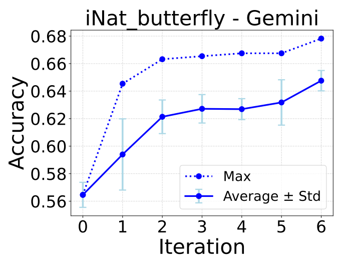
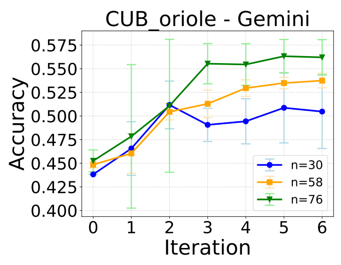
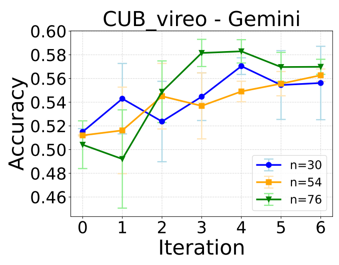

(f) Class-wise Classification

Figure 7: Classification accuracy as a function of iteration.


(a) Class-wise Classification


(b) Class-wise Classification

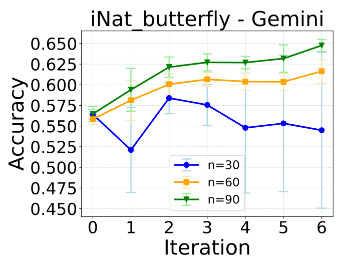

(c) Class-wise Classification

Figure 8: Classification accuracy of Gemini with various number of samples for optimization.

or only different-class images for instance-level classification. As shown in Table 3, neither same-class nor different-class sampling leads to a notable change in performance.

Table 3: Results (accuracy %) of different sampling processes.

Gemini 1.5 Flash	*CUB_cuckoo*
AutoSEP with different-class image sampling	60.98±3.59
AutoSEP with same-class image sampling	61.58±3.11
AutoSEP with random image sampling (ours)	61.22±2.30

The goal of our description generation process is to elicit detailed, fine-grained understanding of each image, instead of enforcing class-level discrimination. This image-specific descriptive richness is then used to guide the classification process in a separate classification prompt. In this context, encountering visually similar images–even from the same class–is not a flaw but an opportunity: it encourages the model to uncover subtle, discriminative visual cues that may otherwise be overlooked. In traditional self-supervised learning methods, similar images from the same class are often sampled together to learn meaningful feature representations through contrastively learning. Our framework is inspired by this idea and pushes the model to verbalize these subtle differences, which ultimately improves the downstream classification.

D Additional Experiment Results

D.1 Results with Gemini 2.0 Flash

We conducted additional experiments with Gemini 2.0 Flash [5], with the results presented in Table 4. While Gemini 2.0 Flash demonstrates a substantial improvement over Gemini 1.5 Flash, its performance does not surpass that of GPT-4o on our benchmark tasks. Nevertheless, our AutoSEP method consistently improves the performance of Gemini 2.0 Flash across both datasets, further confirming its effectiveness across different MLLMs.

Table 4: Results (accuracy %) on Gemini 2.0 Flash.

Gemini 2.0 Flash	*CUB_vireo*	*StanfordDogs_terrier*
Vanilla zero-shot	64.27±1.31	74.60±0.47
Zero-shot with descriptions	64.27±1.65	74.12±1.21
AutoSEP (ours)	**69.05**±1.78	**85.94**±1.05

D.2 Results with smaller MLLMs

We conducted additional experiments using Qwen2-VL-7B-Instruct [30]. Since AutoSEP relies on a strong agent for reflection and modification, we adopted a hybrid setup in which Gemini 1.5 Flash [27] served as the Reflect and Modify operators as well as for image description generation, while Qwen2-VL-7B-Instruct was used for instance-level classification and evaluation. This setup follows a common practice in prior prompt optimization studies [32, 33, 23]. As shown in Table 5, our method also yields substantial performance gains on Qwen2-VL-7B-Instruct, demonstrating its effectiveness on smaller models.

Table 5: Results (accuracy %) on Qwen2-VL-7B-Instruct.

Qwen2-VL-7B-Instruct	*CUB_cuckoo*	*CUB_oriole*
Vanilla zero-shot	40.24±3.43	39.78±7.41
Zero-shot with descriptions	39.27±2.68	50.45±1.56
AutoSEP (ours)	**61.79**±2.31	**54.02**±1.66

E Prompt Details

In this section, we provide the prompts used for baselines in our experiments. Blue boxes show those used directly for classification; orange box shows the description generation prompt before optimization; red box shows the prompt used for SPO [33].

Zero-shot (*CUB_cuckoo*)

```
# Task
Determine what kind of bird this image shows from the following options:

A. Black-billed Cuckoo
B. Mangrove Cuckoo
C. Yellow-billed Cuckoo

Answer the letter from A to C as prediction.

The answer is:
```

Zero-shot with image description (*CUB_cuckoo*)

```
# Task
Determine what kind of bird this image shows from the following options:

A. Black-billed Cuckoo
B. Mangrove Cuckoo
C. Yellow-billed Cuckoo

Answer the letter from A to C as prediction.

# Prediction
Text: The image shows the following features: {description}
The answer is:
```

Few-shot with random/MLLM labels (*CUB_cuckoo*)

```
Your task is to classify the image to three birds: A. Black-billed Cuckoo, B. Mangrove Cuckoo, C.
↪  Yellow-billed Cuckoo.

The classification of the 1 image is: C
The classification of the 2 image is: B

The classification of the 3 image is: C
...

The classification of the last image is: (Answer Letter A or B or C)
```

Multiple images display [16] (*CUB_cuckoo*)

```
Given the first k image examples, answer the following question:

What kind of bird does the last image show: A. Black-billed Cuckoo, B. Mangrove Cuckoo, C.
↪  Yellow-billed Cuckoo.

Directly Answer Letter A or B or C without any further information
```

Initial image description generation (*CUB_cuckoo*)

```
# Task
Describe the bird in the given image in detail, focusing on highly distinctive attributes that
↪  are typical to this bird. Ignore the background or other information.
```

SPO [33] description quality evaluation (*CUB_cuckoo*)

```
I'm trying to write a prompt used to generate description for the bird in the image, where the
↪  generated description can enhance the zero-shot classification of the bird in the image.
The three types of birds are: A. Black-billed Cuckoo, B. Mangrove Cuckoo, C. Yellow-billed Cuckoo.

There are two descriptions of the given images generated by two different prompts:

Text 1: {description_1}

Text 2: {description_2}

Which description better describes the image? The first text or the second text? Provide your
↪  analysis and the choice you believe is better, using XML tags to encapsulate your response.

<analyse>Some analysis</analyse>
<choose>First/Second (the better answer in your opinion)</choose>
```

F Diversity Score

Quantifying the amount of new semantic content introduced during optimization is essential for understanding the evolution of prompts. To this end, we design a diversity score that captures two aspects:

- Semantic richness, measured by counting the number of unique lemmas of content words (nouns, verbs, adjectives) after stop-word removal, denoted as($T(p)$);
- Uniqueness across prompts, defined as the number of words that appear exclusively in a given prompt and not in any others, denoted as($U(p)$).

To balance the contributions of these components, we normalize each by the maximum value observed across all prompts in the optimization:

$$\tilde{T}(p) = \frac{T(p)}{\max_i T(p_i)}, \qquad \tilde{U}(p) = \frac{U(p)}{\max_i U(p_i)}. \tag{7}$$

The final diversity score is then computed as the sum of these normalized terms:

$$D(p) = \tilde{T}(p) + \tilde{U}(p). \tag{8}$$

This metric captures inter-prompt uniqueness, which is central to our analysis of how prompts diversify throughout the optimization process.