VaMP: Variational Multi-Modal Prompt Learning for Vision-Language Models

Silin Cheng **Kai Han***
Visual AI Lab, The University of Hong Kong
hnslcheng@connect.hku.hk kaihanx@hku.hk

Abstract

Vision-language models (VLMs), such as CLIP, have shown strong generalization under zero-shot settings, yet adapting them to downstream tasks with limited supervision remains a significant challenge. Existing multi-modal prompt learning methods typically rely on fixed, shared prompts and deterministic parameters, which limits their ability to capture instance-level variation or model uncertainty across diverse tasks and domains. To tackle this issue, we propose a novel **Va**riational **M**ulti-Modal **P**rompt Learning (VaMP) framework that enables sample-specific, uncertainty-aware prompt tuning in multi-modal representation learning. VaMP generates instance-conditioned prompts by sampling from a learned posterior distribution, allowing the model to personalize its behavior based on input content. To further enhance the integration of local and global semantics, we introduce a class-aware prior derived from the instance representation and class prototype. Building upon these, we formulate prompt tuning as variational inference over latent prompt representations and train the entire framework end-to-end through reparameterized sampling. Experiments on few-shot and domain generalization benchmarks show that VaMP achieves state-of-the-art performance, highlighting the benefits of modeling both uncertainty and task structure in our method. Project page: https://visual-ai.github.io/vamp

1 Introduction

Vision-Language Models (VLMs), such as CLIP [1], have achieved impressive performance across a wide range of visual recognition tasks through multi-modal representation learning. Their ability to align images and texts in a shared embedding space enables strong zero-shot transfer. However, their large-scale parameters and the scarcity of training data, particularly in few-shot settings, make full model fine-tuning computationally expensive and prone to overfitting on downstream tasks.

To address this, prompt learning has emerged as a parameter-efficient alternative, where a small number of learnable tokens are prepended to the input to steer the frozen model toward task-specific behavior [2, 3, 4, 5, 6, 7, 8]. While effective, existing multi-modal prompt tuning methods typically rely on fixed, shared prompts that are applied uniformly across all samples. Such methods are inherently deterministic and lack the flexibility to adapt to instance-level variations or model uncertainty, limiting their generalization to unseen tasks and domains [9, 10].

While recent work has explored uncertainty-aware prompt tuning, most existing approaches remain limited in scope. Bayesian Prompt Learning introduces uncertainty modeling in text-only prompts, and Any-Shift Prompting leverages variational inference to enhance robustness across distribution shifts. However, these methods suffer from key limitations. First, by restricting variational modeling to input-level prompts with global latent variables, they fail to capture hierarchical feature interactions

*Corresponding Author

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

or fine-grained, token-level semantic variations. Second, their text-only prompt tuning overlooks valuable visual information that could enhance cross-modal alignment. Finally, the standard Gaussian prior, shared across all classes, fails to capture inter-class variations, resulting in less discriminative prompt distributions. Consequently, these models are inadequate for capturing fine-grained, input-specific variations, particularly in vision-language tasks where precise alignment is critical.

To overcome these limitations, we introduce a novel **Va**riational **M**ulti-Modal **P**rompt Learning (VaMP) framework that enables sample-specific, uncertainty-aware prompt tuning for vision-language models. We make three key contributions: First, we introduce token-wise variational modeling across multiple intermediate network layers. This approach treats individual prompt tokens as latent variables, enabling the model to capture fine-grained semantic relationships at multiple abstraction levels and improve generalization in low-data and out-of-distribution scenarios. Second, our multi-modal design incorporates both visual and textual signals when inferring posterior distributions, creating more aligned cross-modal representations. Third, by employing class-aware priors instead of standard Gaussian distributions, VaMP generates more discriminative prompts that better capture category-specific features and decision boundaries.

We evaluate VaMP on three challenging adaptation settings: base-to-new generalization, domain generalization and cross-dataset generalization. Our method consistently outperforms strong multi-modal prompt baselines while maintaining high parameter efficiency. Ablation studies further confirm the effectiveness of each component, including the variational modeling and task-aware prior.

2 Related Work

Pre-trained Vision-Language Models. Pre-trained vision-language models (VLMs) [1, 11, 12, 13] have gained significant attention for their strong performance across diverse vision-language tasks. These models typically follow four training paradigms: 1) masked language modeling [14, 15]; 2) masked region prediction [16, 17]; 3) image-text matching [16, 14]; and 4) contrastive learning [1, 11, 18, 19]. While VLMs provide robust, generalized representations, adapting them to downstream tasks remains challenging. Recent studies show that tailored approaches significantly improve performance in specific domains, such as few-shot image recognition [20, 21], object detection [22, 23, 24, 25, 26, 27], semantic segmentation [28, 29, 30, 31, 32] and visual grounding [33, 34]. In this work, we focus on adapting vision-language models for few-shot and zero-shot visual recognition tasks.

Prompt Tuning. Instructions provided to language models as text prompts enable task-specific understanding and performance in VLMs. These prompts, either manually designed or automatically optimized through "Prompt Learning" (originally from NLP [35, 36, 37]), have been adapted for computer vision in three primary forms: textual prompt leanring [2, 3, 38, 39, 40, 41, 42, 43, 44, 45, 46] that fine-tune CLIP's [1] by optimizing continuous prompt vectors in its language branch; visual prompt learning [4, 47, 48, 49, 50] that optimize task-specific learnable inputs in the visual input space while keeping pre-trained backbones frozen; and multi-modal prompt learning [5, 51, 52, 53, 54, 55, 6] that enhance alignment by applying prompts to both vision and language branches. Our work advances this research by introducing a variational framework for multi-modal prompt tuning, enabling sample-specific, uncertainty-aware prompt tuning with structured guidance from both visual inputs and class-level semantics.

Variational Inference. Variational inference has been widely applied to computer vision tasks, such as image generation [56, 57, 58, 59], action recognition [60], instance segmentation [61], anomaly detection [62], depth estimation [63], few-shot learning [64, 65, 66, 67], and domain generalization [68, 69]. Recently, variational inference has been applied to prompt learning to mitigate overfitting in low-shot settings and improve generalization. For example, Bayesian Prompt Learning [9] captures uncertainty by sampling prompts from learned distributions, while Any-Shift Prompting [10] uses a hierarchical probabilistic framework to model distribution shifts and generate adaptive prompts without test-time optimization. However, both methods are limited to the text modality and rely on globally shared prompts, which restrict their ability to capture fine-grained, input-specific variations. To overcome these limitations, we propose a probabilistic framework that combines multi-modal prompt tuning with variational modeling and sample-specific adaptation.

3 Preliminary

3.1 Revisiting CLIP

Our work builds upon the pre-trained vision-language model, CLIP [1], which comprises both a text encoder and a vision encoder. Following previous prompt-learning methods [2, 3, 5, 8], we adopt a ViT-based CLIP model that encodes both images $I \in \mathbb{R}^{H\times W\times 3}$ and text descriptions.

Encoding Image. The image encoder V consists of K transformer layers, denoted as $\{V_i\}_{i=0}^{K-1}$. It first divides the input image I into B non-overlapping patches. These patches are then projected into embeddings $e_0 \in \mathbb{R}^{B\times d_v}$, where d_v represents the embedding dimension. Subsequently, patch embeddings, along with a class token c_i, are sequentially processed through the transformer blocks:

$$[c_{i+1}, e_{i+1}] = V_i([c_i, e_i]) \quad i = 0, 1, \cdots, K-1. \tag{1}$$

The final image representation x is obtained by applying a linear projection to the last layer's class token, mapping it into the shared vision-language embedding space:

$$f_x = F_{\text{img}}(c_K) \quad f_x \in \mathbb{R}^{d_{vl}}. \tag{2}$$

Encoding Text. The text encoder converts tokenized words into embeddings $w_0 = [w_0^1, w_0^2, \cdots, w_0^N] \in \mathbb{R}^{N\times d_l}$, which are processed through K transformer layers:

$$[w_{i+1}] = L_i(w_i) \quad i = 0, 1, \cdots, K-1. \tag{3}$$

Similarly, the text representation t is obtained through a linear projection of the final embedding of the last token:

$$t = F_{\text{txt}}(w_K^N) \quad t \in \mathbb{R}^{d_{vl}}. \tag{4}$$

Zero-shot Classification. For classification, hand-crafted text prompts (*e.g.*, "`a photo of a <category>`") with class labels $y \in \{1, 2, \dots C\}$ are used. The prediction $\hat{y}$ for image I is determined by the highest cosine similarity:

$$\hat{y} = \arg\max_y \frac{\exp(\text{sim}(f_x, t_y)/\tau)}{\sum_{i=1}^{C} \exp(\text{sim}(f_x, t_i)/\tau)}, \tag{5}$$

where τ is the temperature coefficient.

3.2 Multi-Modal Prompt Learning

Multi-modal prompt learning methods [5, 70, 8] extend text-only prompt learning approaches [2, 3] by jointly tuning the text and image prompts to achieve improved alignment for downstream tasks. These methods typically specify H consecutive transformer layers, beginning from the J-th layer, for prompt tuning, while leaving the remaining transformer layers fixed (where $J \geq 0$ and $H < K - J + 1$). For example, MaPLe [5] focuses on prompt tuning the shallow layers ($J = 0, H = 9$), whereas MMRL [8] applies prompt tuning to deeper layers ($J = 5, H = 7$).

Deep Language Prompting. In the text branch, we introduce learnable prompts $z_i \in \mathbb{R}^{M\times d_l}$, consisting of M tokens, into the i-th transformer layer for prompt tuning. For each layer i from J to $J + H - 1$, these tokens are concatenated with the original token embeddings and fed into the i-th transformer layer to generate inputs for the next layer:

$$[\_, w_{i+1}] = L_{i+1}\big([z_i, w_i]\big), \tag{6}$$

while layers outside this range (from 0 to $J - 1$ and from $J + H$ to $K - 1$) remain unchanged:

$$[w_{i+1}] = L_{i+1}\big([w_i]\big). \tag{7}$$

Finally, the text representation is derived via Eq. 4.

$$[\_, w_{i+1}] = L_{i+1}\big([z_i, w_i]\big) \tag{8}$$

Meanwhile, layers outside this range (from 0 to $J - 1$ and from $J + H$ to $K - 1$) remain unchanged:

$$[w_{i+1}] = L_{i+1}\big([w_i]\big). \tag{9}$$

The final text representation is obtained through Eq. 4.

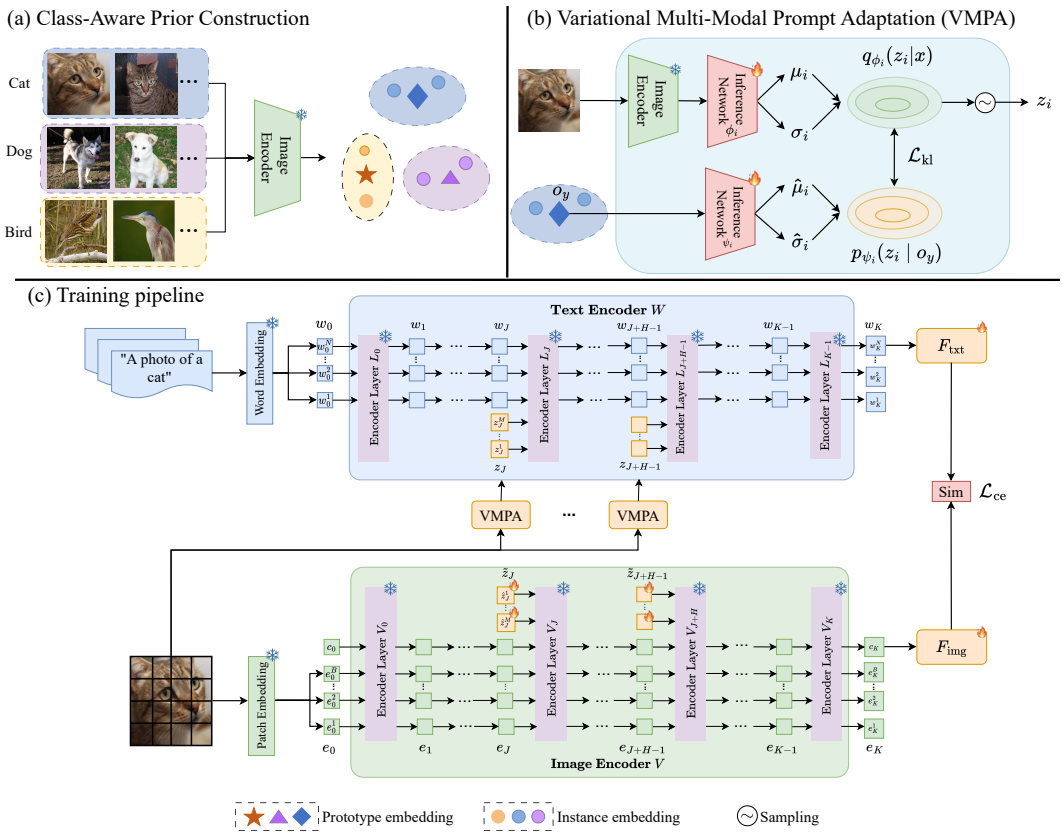


Figure 1: Overview of the VaMP framework. (a) Class-Aware Prior Construction: Utilizing CLIP's frozen image encoder to process training samples, generating offline class prototypes for subsequent adaptation. **(b) Variational Multi-Modal Prompt Adaptation (VMPA):** Variational modeling mechanism where image-conditioned posterior $q_\phi(z_i \mid x)$ and class prototype-based prior $p_\psi(z_i \mid c_y)$ are aligned through KL divergence regularization of latent prompt distributions. **(c) Training Pipeline:** Full architecture of our proposed VaMP framework.

Deep Vision Prompting. In the visual branch, M learnable tokens $\tilde{z}_i \in \mathbb{R}^{M \times d_v}$ are inserted into the i-th transformer layer for prompt tuning. The exact computation of these visual tokens depends on the specific multi-modal prompt learning method being used. For instance, in MaPLe [5], these visual tokens are generated from the language prompts via a linear transformation implemented with an MLP. In contrast, MMRL [8] obtains visual tokens from a shared latent space—a set of learnable tokens initialized by sampling from a Gaussian distribution—and then uses separate linear projection functions (also implemented with MLPs) to generate modality-specific prompts.

For each layer i from J to $J + H - 1$, these generated tokens are concatenated with the original patch token embeddings and fed into the i-th transformer layer to produce inputs for the next layer.

$$[c_{i+1}, e_{i+1}, \_] = V_{i+1}([c_i, e_i, \tilde{z}_i]). \tag{10}$$

For the remaining layers, the original operation without prompts is preserved:

$$[c_{i+1}, e_{i+1}] = V_{i+1}([c_i, e_i]). \tag{11}$$

The final image representation follows the same projection process outlined in Eq. 2. During inference, predictions follow standard classification procedures based on similarity.

4 Method

We propose VaMP—a variational multi-modal prompt learning framework that enables sample-specific, uncertainty-aware, and structured tuning within vision-language models. Our method

consists of three key components: (i) *sample-specific multi-modal prompt generation*, where image-conditioned prompts are injected across multiple transformer layers; (ii) *variational prompt adaptation for multi-modal representation learning*, which models the text-side prompts as latent variables to capture instance-level uncertainty; and (iii) *class-aware prior construction*, which regularizes the latent space using semantic information from both the input instance and its class prototype. An overview of the full framework is illustrated in Figure 1, which highlights the generation of text prompts from image features, the variational posterior, and the class-conditioned prior used for regularization.

4.1 Sample-specific Multi-Modal Prompt Generation

In multi-modal prompt learning, prior work typically learns a fixed set of prompt tokens shared across all input samples [5, 8]. While effective, such fixed prompts cannot adapt to instance-specific variations, which are crucial for robust downstream performance under distribution shifts.

To overcome this limitation, we propose *sample-specific prompt generation*, where prompts in the text encoder are dynamically generated based on the input image. Unlike previous methods that insert fixed prompts at a single layer, we generate and inject prompts across multiple transformer layers, enabling hierarchical and fine-grained modulation of the language stream.

Specifically, given an input image x, we first extract its global visual representation f_x using the frozen CLIP image encoder. To generate text-side prompts, we define a set of H layer-specific prompt generators $\{\Phi_i\}_{i=J}^{J+H-1}$, where each Φ_i is a lightweight MLP that maps the image feature to a set of M prompt tokens for the i-th transformer layer:

$$z_i = \Phi_i(f_x) \in \mathbb{R}^{M \times d}, \quad i = J, \ldots, J+H-1, \tag{12}$$

where d is the token embedding dimension. Although all Φ_i share the same architecture, they have independent parameters to allow for layer-specific adaptation.

The resulting prompts z_i are concatenated with the frozen text token embeddings W_i at each layer i of the CLIP text encoder, as formalized in Eq. 8, following the multi-layer prompt insertion strategy used in MaPLe [5] and MMRL [8].

In parallel, we also adopt deep vision prompting in the CLIP image encoder, where a set of learnable vision-side prompt tokens $\tilde{z}_{i-1} \in \mathbb{R}^{M \times d}$ is inserted at each selected transformer layer i. These vision prompts are shared across samples and optimized independently from the input x. They are not generated dynamically, nor modeled as latent variables.

In summary, our method introduces dynamic, sample-specific text prompts z_i conditioned on image features and static, shared vision prompts $\tilde{z}_i$, aligning both modalities through a structured and hierarchical prompting design. Only the text-side prompts z_i are modeled probabilistically in our variational framework.

4.2 Variational Multi-Modal Prompt Adaptation

While our sample-specific prompts z_i increase flexibility, they remain deterministic and lack the ability to capture uncertainty–a key factor in few-shot and distribution-shift settings. To address this, we formulate prompt adaptation as a probabilistic latent variable model, replacing deterministic prompt tokens with latent prompt tokens learned via variational inference.

For each input image x and fixed text template t, we define $z = \{z_i\}_{i=J}^{J+H-1}$, where each $z_i \in \mathbb{R}^{M \times d}$ is a latent variable representing the prompt tokens inserted at layer i of the text encoder. To model the posterior distribution over these latent variables, we introduce a set of layer-specific MLPs $\{\phi_i\}_{i=J}^{J+H-1}$, where each ϕ_i predicts the parameters of a Gaussian distribution:

$$[\mu_i, \sigma_i] = \phi_i(\overline{f}_x), \tag{13}$$

where $\overline{f}_x$ is the frozen CLIP image embedding. Using these predicted parameters, the posterior distribution is formulated as a product of layer-wise Gaussians, conditioned solely on the image x:

$$q_\phi(z_i \mid x) = \mathcal{N}\left(\mu_i, \mathrm{diag}(\sigma_i^2)\right) \tag{14}$$

Given label y, we aim to maximize the marginal likelihood:

$$\log p(y \mid x, t) = \log \int p(y \mid x, t, z)\, p(z \mid x)\, dz. \tag{15}$$

As the integral is intractable, we maximize the variational evidence lower bound (ELBO):

$$\mathcal{L}_{\text{ELBO}} = \mathbb{E}_{q_\phi(z|x)} \left[\log p(y \mid x, t, z)\right] - \text{KL}\left(q_\phi(z \mid x) \,\|\, p(z)\right). \tag{16}$$

Here, $p(z)$ represents the prior distribution over the latent prompts, which we initially set to a standard Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{I})$. In implementation, we apply the reparameterization trick [71] to enable gradient-based optimization:

$$z_i = \mu_i + \sigma_i \odot \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{17}$$

Each sampled z_i replaces the deterministic z_i, and is concatenated with the frozen tokens w_i to form the layer input, as formalized in Eq. 8. This variational formulation introduces a structured, uncertainty-aware distribution over prompts and allows the model to adaptively control prompt behavior per input.

4.3 Class-Aware Prior Construction

In variational inference, the prior distribution $p(z)$ serves as a crucial regularizer for the learned posterior $q_\phi(z \mid x)$. While a standard choice $p(z) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ provides a generic reference, it lacks semantic structure and offers no task-specific guidance. To incorporate class-level semantics into the latent prompt space, we introduce a *class-aware prior* that conditions on a prototype representation for each class.

During training, we assume access to the class label y for each sample. We compute a class prototype o_y by averaging the posterior means of training samples in class y:

$$o_y = \frac{1}{|D_y|} \sum_{x_i \in D_y} \overline{f}_{x_i}, \tag{18}$$

where D_y is the set of training instances labeled y.

To model layer-wise latent prompts $z = \{z_i\}_{i=J}^{J+H-1}$, we introduce a set of layer-specific prior networks $\{\psi_i\}_{i=J}^{J+H-1}$. Each prior network maps the class prototype c_y to the parameters of a Gaussian prior at layer i:

$$[\hat{\mu}_i, \hat{\sigma}_i] = \psi_i(c_y), \quad p_\psi(z_i \mid o_y) = \mathcal{N}(\hat{\mu}_i, \text{diag}((\hat{\sigma}_i)^2)). \tag{19}$$

The resulting ELBO objective now sums across layers:

$$\mathcal{L}_{\text{ELBO}} = \sum_{i=J}^{J+H-1} \left(\mathbb{E}_{q_\phi(z_i|x)}[\log p(y \mid x, t, z_i)] - \text{KL}\left(q_\phi(z_i \mid x) \,\|\, p_\psi(z_i \mid o_y)\right)\right). \tag{20}$$

This class-aware prior construction provides global semantic anchoring for each layer's prompt distribution. It encourages prompts from the same class to lie in nearby regions of the latent space, improving intra-class consistency and few-shot generalization. At test time, when y is unavailable, we revert to the standard prior $p(z_i) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ for all layers.

4.4 Inference Procedure

During inference, our method follows a single forward pass through the VaMP framework. Given an input image x and a fixed text template t (*e.g.*, "`A photo of a [CLASS]`"), we first extract the frozen CLIP image feature $\overline{f}_x$.

For probabilistic inference, we leverage Monte Carlo sampling over the latent prompt distribution. Specifically, given input image x, we draw S samples $\{z_{i,s}\}_{s=1}^{S}$ from the learned posterior $q_\phi(z_i \mid x)$ for each layer i:

$$[\mu_i, \sigma_i] = \phi_i(\overline{f}_x), \quad z_{i,s} = \mu_i + \sigma_i \odot \epsilon_{i,s}, \quad \epsilon_{i,s} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{21}$$

Each sampled latent prompt $z_{i,s}$ is concatenated with the frozen text tokens w_i at each transformer layer. Simultaneously, a set of shared vision-side prompts $\{\tilde{z}_i\}_{i=J}^{J+H-1}$ is injected into the corresponding layers of the CLIP image encoder. The model then computes the image and text features using the frozen CLIP encoders with inserted prompts, and generates a prediction $p_s(y \mid x, t, \{z_{i,s}\}_{i=J}^{J+H-1})$ based on the similarity between the projected image and text representations.

Finally, the predictions are averaged across samples to form the final output:

$$p(y \mid x, t) = \frac{1}{S} \sum_{s=1}^{S} p_s(y \mid x, t, \{z_{i,s}\}_{i=J}^{J+H-1}). \tag{22}$$

In our experiments, we use $S = 10$ samples for all evaluations. This inference-time ensembling preserves the expressiveness of the variational model while improving robustness and stability.

5 Experiments

5.1 Experiments Setup

We evaluate the performance of VaMP under three different settings: base-to-novel generalization, cross-dataset evaluation, domain generalization, and few-shot learning. All conducted under a 16-shot setting, where each category has only 16 training examples.

Base-to-Novel Generalization. In this setting, dataset classes are split into base and novel classes. The model is trained exclusively on base classes and tested on both base and novel classes, enabling an assessment of its transfer learning performance on base classes and its ability to preserve the inherent generalization and zero-shot capabilities of pre-trained VLMs for novel classes. This evaluation is conducted across 11 diverse classification datasets: ImageNet [72], Caltech101 [73], OxfordPets [74], StanfordCars [75], Flowers102 [76], Food101 [77], FGVCAircraft [78], SUN397 [79], UCF101 [80], DTD [81], and EuroSAT [82].

Cross-Dataset Evaluation. This evaluation examines the model's ability to generalize to unseen datasets. Following CoCoOp [3], the model is trained on all 1000 ImageNet classes in a few-shot setting and directly tested, without further fine-tuning, on the same datasets used for base-to-novel generalization, allowing us to assess cross-dataset transferability.

Domain Generalization. To evaluate the model's robustness to domain shifts and its generalization to out-of-distribution data, we train it on ImageNet and test it on four domain-variant datasets: ImageNetV2 [83], ImageNet-Sketch [84], ImageNet-A [85], and ImageNet-R [86], each introducing distinct types of domain variation.

Implementation Details. We follow prior studies [5, 8] and adopt a 16-shot training setting for all experiments unless otherwise noted. We build on the ViT-B/16 variant of CLIP [89] as the visual backbone and apply multi-layer prompt tuning on the text and vision encoders starting from the J-th transformer layer. For MMRL-style settings, we set $J = 5$, $H = 7$ and insert $M = 5$ learnable representation tokens per layer. For MaPLe-style configurations, we adopt $J = 0$, $H = 9$ with prompt length $M = 2$ for both modalities. All experiments are conducted on a single NVIDIA V100 GPU.

5.2 Main Results

Base-to-Novel Generalization. We evaluate VaMP against recent prompt tuning methods across 11 diverse datasets under the base-to-new generalization protocol. As shown in Table 1, VaMP achieves competitive performance on base classes while consistently outperforming all baselines on novel classes. In particular, VaMP attains the highest average novel accuracy of 78.67%, outperforming the best previous method (MMRL) by 1.51% and demonstrating strong generalization to unseen categories. The advantage is especially pronounced on challenging datasets with large domain shifts. For example, on DTD—a dataset characterized by fine-grained textures rather than semantic categories—VaMP achieves a novel accuracy of 75.50%, surpassing MMRL by 2.67%. These results highlight the strength of our structured, variational modeling in adapting to unfamiliar domains without sacrificing base class performance.

Domain Generalization. We further evaluate VaMP in a domain generalization setting,

Table 1: Comparison of VaMP with previous state-of-the-art methods on base-to-novel generalization across 11 datasets. Bold values indicate the best results. VaMP consistently enhances base class performance without compromising generalization.

Method	Average			ImageNet			Caltech101			OxfordPets		
	Base	Novel	H	Base	Novel	H	Base	Novel	H	Base	Novel	H
CLIP [1]	69.34	74.22	71.70	72.43	68.14	70.22	96.84	94.00	95.40	91.17	97.26	94.12
CoOp [2]	82.69	63.22	71.66	76.47	67.88	71.92	98.00	89.81	93.73	93.67	95.29	94.47
CoOpOp [3]	80.47	71.69	75.83	75.98	70.43	73.10	97.96	93.81	95.84	95.20	97.69	96.43
ProDA [38]	81.56	72.30	76.65	75.40	70.23	72.72	98.27	93.23	95.68	95.43	97.83	96.62
KgCoOp [41]	80.73	73.60	77.00	75.83	69.96	72.78	97.72	94.39	96.03	94.65	97.76	96.18
MaPLe [5]	82.28	75.14	78.55	76.66	70.54	73.47	97.74	94.36	96.02	95.43	97.76	96.58
PromptSRC [6]	84.26	76.10	79.97	77.60	70.73	74.01	98.10	94.03	96.02	95.33	97.30	96.30
TCP [42]	84.13	75.36	79.51	77.27	69.87	73.38	98.23	94.67	96.42	94.67	97.20	95.92
MMA [70]	83.20	76.80	79.87	77.31	71.00	74.02	98.40	94.00	96.15	95.40	**98.07**	96.72
2SFS [87]	85.55	75.48	80.20	77.71	70.99	74.20	98.71	94.43	96.52	95.32	97.82	96.55
SkipT [88]	85.04	77.53	81.11	77.73	70.40	73.89	98.50	95.33	96.89	95.70	97.87	**96.77**
MMRL [8]	85.68	77.16	81.20	77.90	71.30	74.45	**98.97**	94.50	96.68	95.90	97.60	96.74
VaMP	**86.45**	**78.67**	**82.37**	**78.98**	**73.45**	**76.11**	98.95	**95.96**	**97.43**	**96.95**	95.24	96.08

Method	StanfordCars			Flowers102			Food101			FGVCAircraft		
	Base	Novel	H	Base	Novel	H	Base	Novel	H	Base	Novel	H
CLIP [1]	63.37	74.89	68.65	72.08	77.80	74.83	90.10	91.22	90.66	27.19	36.29	31.09
CoOp [2]	78.12	60.40	68.13	97.60	59.67	74.06	88.33	82.26	85.19	40.44	22.30	28.75
CoOpOp [3]	70.49	73.59	72.01	94.87	71.75	81.71	90.70	91.29	90.99	33.41	23.71	27.74
ProDA [38]	74.70	71.20	72.91	97.70	68.68	80.66	90.30	88.57	89.43	36.90	34.13	35.46
KgCoOp [41]	71.76	75.04	73.36	95.00	74.73	83.65	90.50	91.70	91.09	36.21	33.55	34.83
MaPLe [5]	72.94	74.00	73.47	95.92	72.46	82.56	90.71	92.05	91.38	37.44	35.61	36.50
PromptSRC [6]	78.27	74.97	76.58	98.07	76.50	85.95	90.67	91.53	91.10	42.73	37.87	40.15
TCP [42]	80.80	74.13	77.32	97.73	75.57	85.23	90.57	91.37	90.97	41.97	34.43	37.83
MMA [70]	78.50	73.10	75.70	97.77	75.93	85.48	90.13	91.30	90.71	40.57	36.33	38.33
2SFS [87]	82.50	74.80	78.46	98.29	76.17	85.83	89.11	91.34	90.21	47.48	35.51	40.63
SkipT [88]	82.93	72.50	77.37	98.57	75.80	85.70	90.67	92.03	91.34	45.37	37.13	40.84
MMRL [8]	81.30	75.07	78.06	**98.97**	77.27	86.78	90.57	91.50	91.03	46.30	37.03	41.15
VaMP	**83.78**	**80.14**	**81.91**	98.96	**83.97**	**90.85**	**92.77**	**93.16**	**92.96**	**46.77**	**41.13**	**43.76**

Method	SUN397			DTD			EuroSAT			UCF101		
	Base	Novel	H	Base	Novel	H	Base	Novel	H	Base	Novel	H
CLIP [1]	69.36	75.35	72.23	53.24	59.90	56.37	56.48	64.05	60.03	70.53	77.50	73.85
CoOp [2]	80.60	65.89	72.51	79.44	41.18	54.24	92.19	54.74	68.69	84.69	56.05	67.46
CoOpOp [3]	79.74	76.86	78.27	77.01	56.00	64.85	87.49	60.04	71.21	82.33	73.45	77.64
ProDA [38]	78.67	76.93	77.79	80.67	56.48	66.44	83.90	66.00	73.88	85.23	71.97	78.04
KgCoOp [41]	80.29	76.53	78.36	77.55	54.99	64.35	85.64	64.34	73.48	82.89	76.67	79.65
MaPLe [5]	80.82	78.70	79.75	80.36	59.18	68.16	94.07	73.23	82.35	83.00	78.66	80.77
PromptSRC [6]	82.67	78.47	80.52	83.37	62.97	71.75	92.90	73.90	82.32	87.10	78.80	82.74
TCP [42]	82.63	78.20	80.35	82.77	58.07	68.25	91.63	74.73	82.32	87.13	80.77	83.83
MMA [70]	82.27	78.57	80.38	83.20	65.63	73.38	85.46	**82.34**	83.87	86.23	80.03	82.20
2SFS [87]	82.59	78.91	80.70	84.60	65.01	73.52	96.91	67.09	79.29	87.85	78.19	82.74
SkipT [88]	82.40	79.03	80.68	83.77	67.23	74.59	92.47	83.00	87.48	87.30	**82.47**	**84.81**
MMRL [8]	83.20	79.30	**81.20**	85.67	65.00	73.82	95.60	80.17	**87.21**	88.10	80.07	83.89
VaMP	**83.37**	**78.95**	81.09	**86.14**	**67.20**	**75.50**	**95.78**	77.21	85.49	**88.52**	78.99	83.48

where prompts are optimized on ImageNet and directly applied to its variants (-V2, -S, -A, -R) without any target supervision. As shown in Table 2, VaMP achieves the highest accuracy across all four target domains, with an average accuracy of 61.73%, outperforming the best baseline (MMRL) by 1.20%. The gain is particularly significant on Sketch, a challenging domain due to its abstract and textureless nature. On this subset, VaMP achieves 49.69%, improving over MMRL by 0.52%. These results validate the effectiveness of our variational prompt modeling and class-aware regularization in enhancing out-of-distribution generalization, even in low-texture, cross-modality scenarios.

Table 2: Comparison of VaMP with previous state-of-the-art methods on domain generalization across 4 datasets.

	Source	Target			
	ImageNet	-V2	-S	-A	-R
CLIP [1]	66.73	60.83	46.15	47.77	73.96
CoOp [2]	71.51	64.20	47.99	49.71	75.21
CoOpOp [3]	71.02	64.07	48.75	50.63	76.18
MaPLe [5]	70.72	64.07	49.15	50.90	76.98
PromptSRC [6]	71.27	64.35	49.55	50.90	77.80
MMA [70]	71.00	64.33	49.13	51.12	77.32
MMRL [8]	72.03	64.47	49.17	51.20	77.53
VaMP	**72.83**	**64.96**	**49.69**	**51.97**	**78.01**

Cross-Dataset Generalization. To assess robustness under domain shift, we evaluate VaMP on cross-dataset transfer, where prompts are tuned on one source dataset (ImageNet) and evaluated directly on unseen target datasets. As shown in Table 3, VaMP achieves the best average performance across 10 diverse target datasets, with an average accuracy of 67.74%, outperforming the strongest baseline (MMRL) by 0.49%. The improvement is particularly notable on challenging datasets with large domain gaps. For example, on EuroSAT—a remote sensing dataset with significant visual discrepancy from natural images—VaMP achieves a target accuracy of 53.82%, improving over

Table 3: Comparison of VaMP with previous state-of-the-art methods on cross-dataset evaluation across 10 datasets.

	Source	Target										
	ImageNet	Average	Caltech101	OxfordPets	StanfordCars	Flowers101	Food101	FGVCAircraft	SUN397	DTD	EuroSAT	UCF101
CoOp [2]	71.51	63.88	93.70	89.14	64.51	68.71	85.30	18.47	64.15	41.92	46.39	66.55
CoOpOp [3]	71.02	65.74	94.43	90.14	65.32	71.88	86.06	22.94	67.36	45.73	45.37	68.21
MaPLe [5]	70.72	66.30	93.53	90.49	65.57	72.23	86.20	24.74	67.01	46.49	48.06	68.69
PromptSRC [6]	71.27	65.81	93.60	90.25	65.70	70.25	86.15	23.90	67.10	**46.87**	45.50	68.75
TCP [42]	71.40	66.29	93.97	91.25	64.69	71.21	**86.69**	23.45	67.15	44.35	51.45	68.73
MMA [70]	71.00	66.61	93.80	90.30	**66.13**	72.07	86.12	25.33	**68.17**	46.57	49.24	68.32
MMRL [8]	72.03	67.25	94.67	91.43	66.10	72.77	86.40	26.30	67.57	45.90	53.10	68.27
VaMP	**72.83**	**67.74**	**94.96**	**91.79**	66.10	**73.18**	**86.97**	**26.76**	68.04	46.82	**53.82**	**68.93**

Table 4: Ablation studies: impact of sample-specific multi-modal prompt generation, variational prompt adaptation and class-aware prior on base-to-new generalization performance, averaged across 11 datasets.

(a) Effects of sample-specific multi-modal prompt generation

Method	Prompt Type	Base	New	H
MaPLe [5]	task-specific	82.28	75.14	78.55
	sample-specific	82.95	76.95	79.83
MMRL [8]	task-specific	85.68	77.16	81.20
	sample-specific	**85.93**	**78.13**	**81.84**

(b) Effects of variational multi-modal prompt generation

Method	Prompt Type	Base	New	H
MaPLe [5]	Deterministic prompt	82.95	76.95	79.83
	Variational prompt	84.77	77.32	80.87
MMRL [8]	Deterministic prompt	85.93	78.13	81.84
	Variational prompt	**86.11**	**78.45**	**82.10**

(c) Effects of class-aware prior on base-to-new generalization

Method	Prompt Type	Base	New	H
MaPLe [5]	Normal gaussian prior	84.77	77.32	80.87
	Class-aware prior	85.13	78.07	81.45
MMRL [8]	Normal gaussian prior	86.11	78.45	82.10
	Class-aware prior	**86.45**	**78.67**	**82.37**

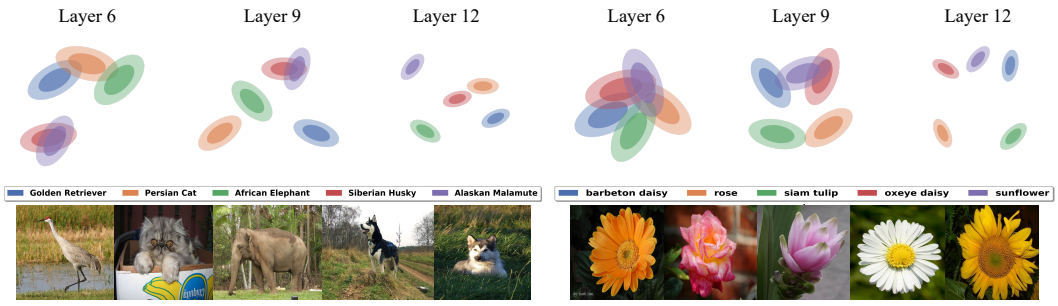
MMRL by 0.72%. These results demonstrate that our structured and uncertainty-aware adaptation generalizes well across domains, even when the target distribution differs substantially from the source.

5.3 Ablation Study

Effects of Sample-specific Multi-Modal Prompt Generation. We begin by assessing the effect of sample-specific multi-modal prompt generation. As shown in Table 4a, previous methods such as MaPLe and MMRL use task-specific prompts shared across all inputs. In contrast, our sample-specific design generates prompts conditioned on each input instance and injected across multiple layers. This structured, instance-aware formulation consistently improves generalization to novel categories, demonstrating the advantage of personalized multi-modal adaptation over fixed prompt configurations.

Effects of Variational Prompt Adaptation. Next, we evaluate the effect of variational modeling on the prompt tokens. While prior work adopts deterministic prompt embeddings, our approach treats the text-side prompts as latent variables inferred per instance. This formulation introduces uncertainty modeling into the prompt space, enabling better adaptation to ambiguous or out-of-distribution inputs. As shown in Table 4b, applying variational prompt learning improves generalization across both MaPLe and MMRL backbones, validating the benefit of latent, sample-specific modeling.

Effects of Class-Aware Prior. We further assess the role of structured priors in regularizing the latent prompt space. Unlike standard variational methods that use an isotropic Gaussian prior, our method constructs a class-aware prior from prototype representations computed over training data. This provides global semantic guidance for latent prompt inference. As shown in Table 4c, replacing the normal prior with a class-aware one consistently improves performance, highlighting the importance of semantic regularization for class-consistent prompt adaptation.



(a) ImageNet (b) Flowers102

Figure 2: Qualitative analysis. Layer-wise visualization of aggregated posterior mean distributions $q_\phi(\mathbf{z}_i|x)$ for sample images from (a) ImageNet and (b) Flowers102.

5.4 Qualitative Analysis

To gain intuitive insights into the learned latent prompt space, we visualize the posterior distributions across text encoder layers. Figure 2 illustrates this process for representative samples from ImageNet and Oxford Flowers datasets. For each input image x, we extract the image-conditioned posterior $q_\phi(z_i|x)$ at layers 6, 9, and 12. Since VaMP generates M prompt tokens per layer, we aggregate their means and covariances as $\mu_{\text{agg},i}(x) = \frac{1}{M}\sum_{j=1}^{M}\mu_i^j(x)$ and $\Sigma_{\text{agg},i}(x) = \frac{1}{M}\sum_{j=1}^{M}\Sigma_i^j(x)$, then project these high-dimensional vectors to 2D using PCA. Contours represent uncertainty regions at 1.5σ and 2.5σ standard deviations.

These visualizations provide insights into how VaMP achieves uncertainty-aware prompt learning through its variational design. First, the varying distributional characteristics across layers and samples—reflected in the spatial positions, orientations, and covariance structures of the posterior distributions—demonstrate that VaMP successfully models input-dependent uncertainty through its variational framework, moving beyond deterministic point estimates. Second, the layer-wise evolution reveals hierarchical uncertainty refinement: early layers (Layer 6) exhibit broader posterior distributions with larger variance to accommodate diverse prompt adaptations, while deeper layers (Layer 12) manifest tighter, more concentrated distributions with reduced uncertainty reflecting task-specific specialization. This progressive variance reduction enables adaptive representation learning at different semantic abstraction levels. Third, the consistent inter-class separation of posterior distributions across all depths demonstrates the effectiveness of the class-aware prior $p_\theta(z|c)$, which regularizes the variational posterior toward discriminative regions of the latent space. This class-conditional guidance ensures that prompts encode category-specific information from early feature extraction stages. Finally, VaMP's stochastic sampling mechanism, manifested through the distributional spread and overlap patterns, provides inherent robustness against overfitting to singular prompt configurations. By parameterizing prompts as probability distributions rather than fixed embeddings, VaMP balances representation flexibility with discriminative capacity throughout the hierarchical architecture.

6 Conclusion

We presented VaMP, a variational framework for prompt adaptation in multi-modal representation learning. Our approach addresses the limitations of fixed, shared prompts by introducing a structured and uncertainty-aware mechanism that adapts to individual input instances. VaMP comprises three key components: (*i*) sample-specific multi-modal prompt generation, where visual features condition prompt tokens across multiple transformer layers; (*ii*) variational modeling of text-side prompts as latent variables, enabling instance-specific and probabilistic adaptation; and (*iii*) a class-aware prior constructed from semantic prototypes, which regularizes the latent space with global class-level information. Through extensive experiments on few-shot and domain generalization benchmarks, we demonstrate that VaMP achieves state-of-the-art performance while maintaining high parameter efficiency. Our findings highlight the importance of modeling both instance-level variability and task structure in prompt-based adaptation for vision-language models.

Acknowledgements

This work is supported by the Hong Kong Research Grants Council - General Research Fund (Grant No.: 17211024).

References

[1] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, and Jack Clark. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[2] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *IJCV*, 2022.

[3] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *CVPR*, 2022.

[4] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *ECCV*, 2022.

[5] Muhammad Uzair Khattak, Hanoona Rasheed, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Maple: Multi-modal prompt learning. In *CVPR*, 2023.

[6] Muhammad Uzair Khattak, Syed Talal Wasim, Muzammal Naseer, Salman Khan, Ming-Hsuan Yang, and Fahad Shahbaz Khan. Self-regulating prompts: Foundational model adaptation without forgetting. In *ICCV*, 2023.

[7] Yingjun Du, Gaowen Liu, Yuzhang Shang, Yuguang Yao, Ramana Kompella, and Cees GM Snoek. Prompt diffusion robustifies any-modality prompt learning. *arXiv preprint arXiv:2410.20164*, 2024.

[8] Yuncheng Guo and Xiaodong Gu. Mmrl: Multi-modal representation learning for vision-language models. In *CVPR*, 2025.

[9] Mohammad Mahdi Derakhshani, Enrique Sanchez, Adrian Bulat, Victor G Turrisi da Costa, Cees GM Snoek, Georgios Tzimiropoulos, and Brais Martinez. Bayesian prompt learning for image-language model generalization. In *ICCV*, 2023.

[10] Zehao Xiao, Jiayi Shen, Mohammad Mahdi Derakhshani, Shengcai Liao, and Cees GM Snoek. Any-shift prompting for generalization over distributions. In *CVPR*, 2024.

[11] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*, 2021.

[12] Lewei Yao, Runhui Huang, Lu Hou, Guansong Lu, Minzhe Niu, Hang Xu, Xiaodan Liang, Zhenguo Li, Xin Jiang, and Chunjing Xu. Filip: Fine-grained interactive language-image pre-training. In *ICLR*, 2022.

[13] Juncheng Li, Xin He, Longhui Wei, Long Qian, Linchao Zhu, Lingxi Xie, Yueting Zhuang, Qi Tian, and Siliang Tang. Fine-grained semantically aligned vision-language pre-training. In *NeurIPS*, 2022.

[14] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In *ICML*, 2021.

[15] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*, 2019.

[16] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. In *EMNLP-IJCNLP*, 2019.

[17] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. Vl-bert: Pre-training of generic visual-linguistic representations. In *ICLR*, 2019.

[18] Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. Align before fuse: Vision and language representation learning with momentum distillation. In *NeurIPS*, 2021.

[19] Yuqi Huo, Manli Zhang, Guangzhen Liu, Haoyu Lu, Yizhao Gao, Guoxing Yang, Jingyuan Wen, Heng Zhang, Baogui Xu, and Weihao Zheng. Wenlan: Bridging vision and language by large-scale multi-modal pre-training. *arXiv preprint arXiv:2103.06561*, 2021.

[20] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. Clip-adapter: Better vision-language models with feature adapters. *IJCV*, 2024.

[21] Renrui Zhang, Rongyao Fang, Peng Gao, Wei Zhang, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-adapter: Training-free clip-adapter for better vision-language modeling. In *ECCV*, 2022.

[22] Jiaming Li, Jiacheng Zhang, Jichang Li, Ge Li, Si Liu, Liang Lin, and Guanbin Li. Learning background prompts to discover implicit knowledge for open vocabulary object detection. In *CVPR*, 2024.

[23] Muhammad Maaz, Hanoona Rasheed, Salman Khan, Fahad Shahbaz Khan, Rao Muhammad Anwer, and Ming-Hsuan Yang. Class-agnostic object detection with multi-modal transformer. In *ECCV*, 2022.

[24] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *ECCV*, 2022.

[25] Xiuye Gu, Tsung-Yi Lin, Weicheng Kuo, and Yin Cui. Open-vocabulary object detection via vision and language knowledge distillation. In *ICLR*, 2022.

[26] Yuhang Zang, Wei Li, Kaiyang Zhou, Chen Huang, and Chen Change Loy. Open-vocabulary detr with conditional matching. In *ECCV*, 2022.

[27] Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. Yolo-world: Real-time open-vocabulary object detection. In *CVPR*, 2024.

[28] Xiangtai Li, Haobo Yuan, Wei Li, Henghui Ding, Size Wu, Wenwei Zhang, Yining Li, Kai Chen, and Chen Change Loy. Omg-seg: Is one model good enough for all segmentation? In *CVPR*, 2024.

[29] Yongming Rao, Wenliang Zhao, Guangyi Chen, Yansong Tang, Zheng Zhu, Guan Huang, Jie Zhou, and Jiwen Lu. Denseclip: Language-guided dense prediction with context-aware prompting. In *CVPR*, 2022.

[30] Xiangtai Li, Henghui Ding, Haobo Yuan, Wenwei Zhang, Jiangmiao Pang, Guangliang Cheng, Kai Chen, Ziwei Liu, and Chen Change Loy. Transformer-based visual segmentation: A survey. *IEEE TPAMI*, 2024.

[31] Timo Lüddecke and Alexander Ecker. Image segmentation using text and image prompts. In *CVPR*, 2022.

[32] Wenfang Sun, Yingjun Du, Gaowen Liu, Ramana Kompella, and Cees GM Snoek. Training-free semantic segmentation via llm-supervision. *arXiv preprint arXiv:2404.00701*, 2024.

[33] Zeyu Han, Fangrui Zhu, Qianru Lao, and Huaizu Jiang. Zero-shot referring expression comprehension via structural similarity between images and captions. In *CVPR*, 2024.

[34] Silin Cheng, Yang Liu, Xinwei He, Sebastien Ourselin, Lei Tan, and Gen Luo. Weakmcn: Multi-task collaborative network for weakly supervised referring expression comprehension and segmentation. In *CVPR*, 2025.

[35] Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *EMNLP*, 2020.

[36] Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. How can we know what language models know? *TACL*, 2020.

[37] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 2023.

[38] Yuning Lu, Jianzhuang Liu, Yonggang Zhang, Yajing Liu, and Xinmei Tian. Prompt distribution learning. In *CVPR*, 2022.

[39] Beier Zhu, Yulei Niu, Yucheng Han, Yue Wu, and Hanwang Zhang. Prompt-aligned gradient for prompt tuning. In *ICCV*, 2023.

[40] Xinyang Liu, Dongsheng Wang, Miaoge Li, Zhibin Duan, Yishi Xu, Bo Chen, and Mingyuan Zhou. Patch-token aligned bayesian prompt learning for vision-language models. *arXiv preprint arXiv:2303.09100*, 2023.

[41] Hantao Yao, Rui Zhang, and Changsheng Xu. Visual-language prompt tuning with knowledge-guided context optimization. In *CVPR*, 2023.

[42] Hantao Yao, Rui Zhang, and Changsheng Xu. Tcp: Textual-based class-aware prompt tuning for visual-language model. In *CVPR*, 2024.

[43] Jinyoung Park, Juyeon Ko, and Hyunwoo J Kim. Prompt learning via meta-regularization. In *CVPR*, 2024.

[44] Xinyu Tian, Shu Zou, Zhaoyuan Yang, and Jing Zhang. Argue: Attribute-guided prompt tuning for vision-language models. In *CVPR*, 2024.

[45] Adrian Bulat and Georgios Tzimiropoulos. Lasp: Text-to-text optimization for language-aware soft prompting of vision & language models. In *CVPR*, 2023.

[46] Yingjun Du, Wenfang Sun, and Cees Snoek. Ipo: Interpretable prompt optimization for vision-language models. In *NeurIPS*, 2024.

[47] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *CVPR*, 2022.

[48] Hyojin Bahng, Ali Jahanian, Swami Sankaranarayanan, and Phillip Isola. Exploring visual prompts for adapting large-scale models. *arXiv preprint arXiv:2203.17274*, 2022.

[49] Feng Li, Qing Jiang, Hao Zhang, Tianhe Ren, Shilong Liu, Xueyan Zou, Huaizhe Xu, Hongyang Li, Jianwei Yang, Chunyuan Li, et al. Visual in-context prompting. In *CVPR*, 2024.

[50] Lingfeng Yang, Yueze Wang, Xiang Li, Xinlong Wang, and Jian Yang. Fine-grained visual prompting. In *NeurIPS*, 2024.

[51] Yi-Lun Lee, Yi-Hsuan Tsai, Wei-Chen Chiu, and Chen-Yu Lee. Multimodal prompting with missing modalities for visual recognition. In *CVPR*, 2023.

[52] Yaowei Li, Ruijie Quan, Linchao Zhu, and Yi Yang. Efficient multimodal fusion via interactive prompting. In *CVPR*, 2023.

[53] Shuvendu Roy and Ali Etemad. Consistency-guided prompt learning for vision-language models. In *ICLR*, 2024.

[54] Zheng Li, Xiang Li, Xinyi Fu, Xin Zhang, Weiqiang Wang, Shuo Chen, and Jian Yang. Promptkd: Unsupervised prompt distillation for vision-language models. In *CVPR*, 2024.

[55] Haoyang Li, Liang Wang, Chao Wang, Jing Jiang, Yan Peng, and Guodong Long. Dpc: Dual-prompt collaboration for tuning vision-language models. In *CVPR*, 2025.

[56] Ming Ding, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, Hongxia Yang, et al. Cogview: Mastering text-to-image generation via transformers. In *NeurIPS*, 2021.

[57] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *ICML*, 2021.

[58] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. In *NeurIPS*, 2023.

[59] Peihao Wang, Dejia Xu, Zhiwen Fan, Dilin Wang, Sreyas Mohan, Forrest Iandola, Rakesh Ranjan, Yilei Li, Qiang Liu, Zhangyang Wang, et al. Taming mode collapse in score distillation for text-to-3d generation. In *CVPR*, 2024.

[60] Hyung-gun Chi, Myoung Hoon Ha, Seunggeun Chi, Sang Wan Lee, Qixing Huang, and Karthik Ramani. Infogcn: Representation learning for human skeleton-based action recognition. In *CVPR*, 2022.

[61] Namdar Homayounfar, Yuwen Xiong, Justin Liang, Wei-Chiu Ma, and Raquel Urtasun. Levelset r-cnn: A deep variational method for instance segmentation. In *ECCV*, 2020.

[62] Mingyu Lee and Jongwon Choi. Text-guided variational image generation for industrial anomaly detection and segmentation. In *CVPR*, 2024.

[63] Ziyao Zeng, Daniel Wang, Fengyu Yang, Hyoungseob Park, Stefano Soatto, Dong Lao, and Alex Wong. Wordepth: Variational language prior for monocular depth estimation. In *CVPR*, 2024.

[64] Jian Zhang, Chenglong Zhao, Bingbing Ni, Minghao Xu, and Xiaokang Yang. Variational few-shot learning. In *CVPR*, 2019.

[65] Jingyi Xu and Hieu Le. Generating representative samples for few-shot classification. In *CVPR*, 2022.

[66] Wenfang Sun, Yingjun Du, Xiantong Zhen, Fan Wang, Ling Wang, and Cees GM Snoek. Metamodulation: Learning variational feature hierarchies for few-shot learning with fewer tasks. In *ICML*, 2023.

[67] Jie Liu, Yingjun Du, Zehao Xiao, Cees GM Snoek, Jan-Jakob Sonke, and Efstratios Gavves. Memory-augmented variational adaptation for online few-shot segmentation. In *CVPR*, 2023.

[68] Yingjun Du, Jun Xu, Huan Xiong, Qiang Qiu, Xiantong Zhen, Cees GM Snoek, and Ling Shao. Learning to learn with variational information bottleneck for domain generalization. In *ECCV*, 2020.

[69] Pan Li, Da Li, Wei Li, Shaogang Gong, Yanwei Fu, and Timothy M Hospedales. A simple feature augmentation for domain generalization. In *ICCV*, 2021.

[70] Lingxiao Yang, Ru-Yuan Zhang, Yanchen Wang, and Xiaohua Xie. Mma: Multi-modal adapter for vision-language models. In *CVPR*, 2024.

[71] Diederik P Kingma, Max Welling, et al. Auto-encoding variational bayes. In *ICLR*, 2014.

[72] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[73] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *CVPR workshop*, 2004.

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

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

[76] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *ICVGIP*, 2008.

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

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

[79] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010.

[80] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[81] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *CVPR*, 2014.

[82] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2019.

[83] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *ICML*, 2019.

[84] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *NeurIPS*, 2019.

[85] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *CVPR*, 2021.

[86] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, and Mike Guo. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *ICCV*, 2021.

[87] Matteo Farina, Massimiliano Mancini, Giovanni Iacca, and Elisa Ricci. Rethinking few-shot adaptation of vision-language models in two stages. In *CVPR*, 2025.

[88] Shihan Wu, Ji Zhang, Pengpeng Zeng, Lianli Gao, Jingkuan Song, and Heng Tao Shen. Skip tuning: Pre-trained vision-language models are effective and efficient adapters themselves. In *CVPR*, 2025.

[89] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

[90] Seokju Cho, Heeseong Shin, Sunghwan Hong, Anurag Arnab, Paul Hongsuck Seo, and Seungryong Kim. Cat-seg: Cost aggregation for open-vocabulary semantic segmentation. In *CVPR*, 2024.

[91] Xiaohu Huang, Hao Zhou, Kun Yao, and Kai Han. Froster: Frozen clip is a strong teacher for open-vocabulary action recognition. In *ICLR*, 2024.

[92] Jian Ding, Nan Xue, Gui-Song Xia, and Dengxin Dai. Decoupling zero-shot semantic segmentation. In *CVPR*, 2022.

[93] Feng Liang, Bichen Wu, Xiaoliang Dai, Kunpeng Li, Yinan Zhao, Hang Zhang, Peizhao Zhang, Peter Vajda, and Diana Marculescu. Open-vocabulary semantic segmentation with mask-adapted clip. In *CVPR*, 2023.

[94] Mengde Xu, Zheng Zhang, Fangyun Wei, Han Hu, and Xiang Bai. Side adapter network for open-vocabulary semantic segmentation. In *CVPR*, 2023.

[95] Hanoona Rasheed, Muhammad Uzair Khattak, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Fine-tuned clip models are efficient video learners. In *CVPR*, 2023.

[96] Zejia Weng, Xitong Yang, Ang Li, Zuxuan Wu, and Yu-Gang Jiang. Open-vclip: Transforming clip to an open-vocabulary video model via interpolated weight optimization. In *ICML*, 2023.

[97] Quan Sun, Yuxin Fang, Ledell Wu, Xinlong Wang, and Yue Cao. Eva-clip: Improved training techniques for clip at scale. *arXiv preprint arXiv:2303.15389*, 2023.

[98] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *ICCV*, 2023.

[99] Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, et al. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

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: We clearly state our contributions in Abstract and also Section 1. These contributions are well validated by our experimental results in Section 5.

   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: We provide a "limitation" subsection in the supplementary material.

   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: [Yes]

Justification: We have provided the theoretical proof in the appendix.

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: We have specified all the training details in Section 5.1 and supplemental materials.

   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: (1) Data: All the datasets we used in this paper are publicly available online, and all the readers are free to download them. We list the statistics of all the used datasets in the supplementary material. (2) Code: Code will be available after paper got accepted.

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:

   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: Following the standard experimental setup, we repeat each experiment over 3 random seeds and report the mean of the results.

   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: We provide the computing resources in experiments 5.

   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: We reviewed and followed the NeurIPS Code of Ethics.

   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: [Yes]

    Justification: We provide the potential broader impacts in supplemental materials.

    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: The data and models pose no such risks.

    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: We cite the original papers that produced the code package and datasets.

    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: Details of the datasets/model are provided in the supplemental materials. The code will be released after the paper get accepted.

    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: This paper does not involve crowdsourcing nor research with human subjects.

    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: This paper does not involve crowdsourcing nor research with 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.

Appendix

A Dataset Details

Details of 14 datasets are shown in Table A1.

Table A1: Summary of the 14 datasets.

Dataset	Classes	Train	Val	Test	Description
ImageNet [72]	1,000	1.28M	∼	50,000	Recognition of generic objects
Caltech101 [73]	100	4,128	1,649	2,465	Recognition of generic objects
OxfordPets [74]	37	2,944	736	3,669	Fine-grained classification of pets
StanfordCars [75]	196	6,509	1,635	8,041	Fine-grained classification of cars
Flowers102 [76]	102	4,093	1,633	2463	Fine-grained classification of flowers
Food101 [77]	101	50,500	20,200	30,300	Fine-grained classification of foods
FGVCAircraft [78]	100	3,334	3,333	3,333	Fine-grained classification of aircrafts
SUN397 [79]	397	15,880	3,970	19,850	Scene classification
DTD [81]	47	2,820	1,128	1,692	Texture classification
EuroSAT [82]	10	13,500	5,400	8,100	Land use & cover classification with satellite images
UCF101 [80]	101	7,639	1,898	3,783	Action recognition
ImageNetV2 [83]	1,000	∼	∼	10,000	New test data for ImageNet
ImageNet-Sketch [84]	1,000	∼	∼	50,889	Sketch-style images of ImageNet classes
ImageNet-A [85]	200	∼	∼	7,500	Natural adversarial examples of 200 ImageNet classes
ImageNet-R [86]	200	∼	∼	30,000	Renditions of 200 ImageNet classes

B More Implementation Details

All models are trained using the AdamW optimizer with a learning rate of 0.001 and a weight decay of 0.01. The batch size is set to 32 for ImageNet and 4 for all other datasets. We apply automatic mixed-precision training throughout to improve efficiency. For base-to-novel generalization on ImageNet, we train for 5 epochs; for other datasets we train for 10 epochs. For cross-dataset and domain generalization tasks, we train on ImageNet for a single epoch. Few-shot learning tasks use 5 training epochs on ImageNet and 50 epochs on target datasets. All reported results are averaged over three independent runs. All prompts and representation tokens are initialized from a zero-mean Gaussian distribution with a standard deviation of 0.02. For EuroSAT, we follow MMRL [8] and set the representation token dimension $d_r = 2048$; for all other datasets, we use $d_r = 512$. The fusion parameter α in MMRL-style classifiers is fixed to 0.7. The average accuracy is reported over three independent runs. For variational modeling, we use a two-layer MLP with GELU activation to parameterize both the posterior network ϕ and the prior network ψ, outputting mean and log-variance vectors per layer. The latent variables z are sampled using the reparameterization trick, and we perform $S = 10$ Monte Carlo samples at inference time. Class prototypes o_y are computed offline at the start of training.

C Efficiency Analysis

We analyze the efficiency of our variational multi-modal prompt framework, focusing on the trade-off between inference cost and generalization ability. As detailed in Table A2, VaMP introduces only a modest overhead compared to the strong MMRL baseline. For instance, with $S = 10$, our method improves the Harmonic Mean (HM) from 81.20 to 82.37 with a minimal latency increase of just 0.8ms per image on an NVIDIA V100 GPU. The performance gains saturate quickly as S increases, demonstrating that VaMP achieves an effective balance between accuracy and efficiency even with limited sampling. Furthermore, our approach is parameter-efficient, increasing the number of learnable parameters by less than 6% when integrated into MMRL (from 4.992M to 5.132M). This balance demonstrates that VaMP can be deployed with low overhead while still leveraging uncertainty modeling for improved generalization.

Table A2: Analysis of performance vs. efficiency trade-off.

Method	S	Inference Time (ms/img)	Base	Novel	HM
MMRL	-	5.3	85.68	77.16	81.20
MMRL+VaMP	1	5.5	86.10	77.35	81.44
MMRL+VaMP	5	5.8	86.37	78.26	82.03
MMRL+VaMP	10	6.1	**86.45**	**78.67**	**82.37**
MMRL+VaMP	20	6.5	**86.47**	78.37	82.22
MMRL+VaMP	50	9.3	86.43	78.15	82.01

D More Ablation Studies

We conduct ablation studies by integrating our variational prompt adaptation into the MMRL [8] framework, analyzing the impact of key hyperparameters such as the prompt depth, and prompt length. All results are reported on the base-to-novel generalization benchmark (11 datasets), using average accuracy across base and novel splits.

Effect of Prompt Insertion Depth J and H. We vary the transformer layer index J where prompt tuning begins, and the number of consecutive layers H to which prompts are added. As shown in Table A3, inserting prompts deeper into the encoder (e.g., starting from layer $J = 5$) and extending them to more layers (e.g., $H = 7$) leads to better performance on both base and novel classes. This confirms the benefit of hierarchical prompt injection for deeper vision-language alignment.

Table A3: Effect of prompt insertion depth (J, H).

J	H	Base	Novel	H
4	3	85.57	77.14	81.13
6	5	86.28	78.02	82.01
6	7	**86.45**	**78.67**	**82.37**

Effect of Prompt Token Length M. We analyze the sensitivity to the number of prompt tokens M injected per layer. Table A4 shows that increasing M from 1 to 5 improves accuracy, as the model benefits from higher representational capacity. However, further increasing to $M = 8$ slightly reduces generalization, likely due to redundancy and overfitting. Hence, we set $M = 5$ as the default in our main experiments.

Table A4: Effect of prompt token length M per layer.

M	Base	Novel	H
1	85.76	77.21	81.30
3	86.18	78.05	82.02
5	**86.45**	**78.67**	**82.37**
8	86.19	78.12	82.00

E Detailed Derivation of the Variational Lower Bound (ELBO)

We derive the evidence lower bound (ELBO) used in our variational prompt learning framework. Our goal is to estimate the conditional likelihood of the class label y given an image x and a text prompt template t, where the latent variables z represent the sample-specific text prompt tokens injected across multiple layers of the language encoder.

The marginal likelihood is obtained by integrating out the latent variables z:

$$\log p(y \mid x, t) = \log \int p(y \mid x, t, z)\, p(z \mid x)\, dz. \tag{23}$$

Since this integral is generally intractable, we introduce a variational posterior $q_\phi(z \mid x)$ and apply Jensen's inequality:

$$\log p(y \mid x, t) = \log \int q_\phi(z \mid x) \cdot \frac{p(y \mid x, t, z)\, p(z \mid x)}{q_\phi(z \mid x)}\, dz \tag{24}$$

$$\geq \mathbb{E}_{q_\phi(z|x)} \left[\log \frac{p(y \mid x, t, z)\, p(z \mid x)}{q_\phi(z \mid x)} \right] \tag{25}$$

$$= \underbrace{\mathbb{E}_{q_\phi(z|x)}[\log p(y \mid x, t, z)]}_{\text{expected log-likelihood}} - \underbrace{\mathrm{KL}(q_\phi(z \mid x) \,\|\, p(z \mid x))}_{\text{KL divergence}}. \tag{26}$$

We model z as a collection of layer-wise latent prompt embeddings $\{z_i\}_{i=J}^{J+H-1}$, one for each of the H transformer layers in the text encoder. We assume the posterior and prior factorize across layers:

$$q_\phi(z \mid x) = \prod_{i=J}^{J+H-1} q_\phi(z_i \mid x), \tag{27}$$

$$p(z \mid x) = \prod_{i=J}^{J+H-1} p(z_i \mid x). \tag{28}$$

This leads to the following form of the ELBO:

$$\mathcal{L}_{\text{ELBO}} = \sum_{i=J}^{J+H-1} \left[\mathbb{E}_{q_\phi(z_i|x)} \log p(y \mid x, t, z_i) - \mathrm{KL}(q_\phi(z_i \mid x) \,\|\, p(z_i \mid x)) \right], \tag{29}$$

where z_i is injected at layer i of the frozen text encoder, modulating the token representations via concatenation.

During training, we replace $p(z_i \mid x)$ with a class-aware prior:

$$p_\psi(z_i \mid o_y) = \mathcal{N}(\hat{\mu}_i, \mathrm{diag}((\hat{\sigma}_i)^2)) \quad [\hat{\mu}_i, \hat{\sigma}_i] = \psi_i(c_y). \tag{30}$$

where the class prototype o_y is the mean of posterior means $\hat{\mu}_i$ over all training samples in class y.

The final training objective maximizes the ELBO in Eq. 29.

This variational formulation enables our model to learn expressive, uncertainty-aware, sample-specific prompts while regularizing the latent space with class-level semantic structure.

F Extension to Other Tasks

To assess the generalizability of VaMP beyond standard image classification, we extended our evaluation to the more complex domains of open-vocabulary segmentation and action recognition. Our approach was integrated into two state-of-the-art frameworks: CAT-Seg [90] for segmentation and FROSTER [91] for action recognition, both of which utilize a CLIP ViT-B/16 backbone. This compatibility allowed for a seamless and direct evaluation of VaMP's effectiveness in these diverse tasks.

For the open-vocabulary segmentation task, we adhered to the established CAT-Seg protocol. The model was trained on the COCO-Stuff dataset (118K images, 171 categories) and subsequently evaluated on several challenging benchmarks, including ADE20K, PASCAL-Context, and PASCAL VOC, which feature a wide range of category scales (59-847 classes). The results are detailed in Table A5.

For open-vocabulary action recognition, we adopted the base-to-novel evaluation protocol from FROSTER. This setup involves training the model exclusively on the base classes from two widely used video benchmarks, Kinetics-400 and UCF-101. The primary metric is the model's ability to generalize to novel, unseen action categories during testing. Table A6 summarizes these results.

The consistent performance gains observed across both segmentation and action recognition tasks underscore the scalability and robust generalization capabilities of our proposed framework.

Table A5: Performance for open-vocabulary segmentation.

Method	Prompt Tuning	ADE-847	PC-459	ADE-150	PC-59	VOC-20
ZegFormer [92]	-	5.6	10.4	18.0	45.5	89.5
OVSeg [93]	-	7.1	11.0	24.8	53.3	92.6
SAN [94]	-	10.1	12.6	27.5	53.8	94.0
CAT-Seg [90]	-	12.0	19.0	31.8	57.5	94.6
CAT-Seg	MMRL	12.8	18.7	32.4	57.9	94.3
CAT-Seg	MMRL+VaMP	**13.9** (+1.1)	**20.3** (+1.6)	**33.3** (+0.9)	**58.6** (+0.7)	**95.2** (+0.9)

Table A6: Performance for open-vocabulary action recognition. B: Base classes, N: Novel classes, HM: Harmonic mean.

Method	Prompt Tuning	K400(B)	K400(N)	K400(HM)	UCF(B)	UCF(N)	UCF(HM)
ViFi-CLIP [95]	-	76.4	61.1	67.9	92.9	67.7	78.3
Open-VCLIP [96]	-	76.5	62.6	68.9	94.8	77.5	85.3
FROSTER [91]	-	77.8	64.3	70.4	95.3	80.0	87.0
FROSTER	MMRL	78.3	64.1	70.5	95.5	80.2	87.2
FROSTER	MMRL+VaMP	**78.8** (+0.5)	**64.8** (+0.7)	**71.1** (+0.6)	**96.1** (+0.6)	**81.0** (+0.8)	**87.9** (+0.7)

G Scalability to Other VLM Architectures

To demonstrate the architecture-agnostic nature of our method, we validated VaMP on several state-of-the-art VLMs that represent the latest advances in CLIP-style architectures. Our method can be readily integrated into more complex VLMs, as it only requires access to intermediate transformer layers and the ability to inject prompt tokens. The variational adaptation module is lightweight (MLPs per layer) and does not assume any specific backbone structure. We therefore integrated VaMP into several prominent models, including EVA-CLIP [97], SigLIP [98], and SigLIP 2 [99]. These models share similar structural designs with CLIP, which enabled their rapid adaptation for our experiments. As shown in Table A7, VaMP obtains clear and consistent performance improvements across all VLMs, demonstrating its strong generalization capability to unseen categories—a critical challenge in prompt learning. The consistent improvements across different VLMs further substantiate the effectiveness and generalizability of our approach to different architectures.

Table A7: Performance comparison of different VLMs on ViT-B/16 backbone under base-to-novel generalization setting.

VLM	Method	Base	Novel	H
CLIP [1]	MMRL	85.68	77.16	81.20
CLIP [1]	MMRL+VaMP	**86.45** (+0.77)	**78.67** (+1.51)	**82.37** (+1.17)
EVA-CLIP [97]	MMRL	85.97	77.69	81.62
EVA-CLIP [97]	MMRL+VaMP	**86.59** (+0.62)	**79.18** (+1.49)	**82.71** (+1.09)
SigLIP [98]	MMRL	86.12	78.21	81.97
SigLIP [98]	MMRL+VaMP	**86.88** (+0.76)	**79.45** (+1.24)	**83.00** (+1.03)
SigLIP 2 [99]	MMRL	86.64	78.97	82.62
SigLIP 2 [99]	MMRL+VaMP	**87.09** (+0.45)	**80.02** (+1.05)	**83.40** (+0.78)

H Broader Impact and Limitations

Our work presents a variational framework for sample-specific, uncertainty-aware prompt adaptation in vision-language models, aiming to improve robustness under distribution shifts and limited supervision. The proposed method has the potential to benefit a wide range of downstream applications where multi-modal understanding and generalization are essential, such as assistive AI systems, open-world recognition, or low-resource domain transfer. The probabilistic modeling component can further inspire future efforts in calibrated and interpretable multi-modal adaptation. At present, we have not identified any ethical concerns associated with the real-world applications of this technology. However, we strongly recommend continuous monitoring and evaluation to ensure its responsible

and ethical deployment. Our approach also has certain limitations. First, our class-aware prior construction relies on access to class prototypes computed from training data, which may not be available in zero-label scenarios. Extending our method to work under fully unsupervised or few-label conditions remains an open direction. Second, our experiments focus on classification tasks; the extension to generative or structured prediction settings (*e.g.*, image captioning, VQA) requires further investigation and architectural adaptation. Despite these challenges, we believe our framework takes an important step toward principled and efficient multi-modal prompt learning, and hope it provides useful insights for future research in uncertainty-aware adaptation, vision-language alignment, and lightweight tuning strategies.