Probe, Anchor, and Amend: Active Test-Time Adaptation of Vision-Language Models

Jiaqi Lin[1,2], Chaoqi Chen[3], Xiasi Wang[4], Mingfu Yan[5], Jiancheng Huang[6], Jianzhuang Liu[7], Wenming Yang[1,2†], and Qingmin Liao[1]

[1]Tsinghua University [2]Pengcheng Laboratory [3]Shenzhen University
[4]The Hong Kong University of Science and Technology [5]Southeast University
[6]ByteDance Inc. [7]Shenzhen Institutes of Advanced Technology

Abstract. Vision-language models (VLMs) excel at zero-shot recognition, yet their performance degrades under distribution shifts during inference. Unsupervised test-time adaptation (TTA) often relies on noisy pseudo-labels or heuristics, offering limited guidance for where decision boundaries should move; existing active TTA (ATTA) methods typically treat annotations as local fixes rather than structured updates to the shared semantics of VLMs. We introduce PAA (Probe, Anchor, and Amend), an ATTA framework that treats visual and textual prototypes as an evolving state and updates this state through a three-step cycle. In each cycle, the Probe step performs prototype-guided querying under a class-balanced budget, selecting both central samples to preserve source knowledge and boundary samples to track distribution shifts. Using only binary verifications, the Anchor step applies a prototype-regularized update that couples cross-modal alignment with intra-class compactness and inter-class separation, yielding a decision space that is stable yet plastic. As a slow controller, the Amend step revisits past hard cases stored in memory and converts temporal discrepancies into memory-driven self-refinement. Across VLM test-time benchmarks, PAA delivers consistent gains over strong TTA and ATTA baselines under comparable annotation budgets and compute, turning sparse, delayed feedback into structured evolution of the prototype-anchored decision space.

Keywords: Active Test-Time Adaptation · Vision-Language Models

1 Introduction

Vision-language models (VLMs) [5, 14, 16, 29, 37] revolutionize multimodal understanding through their ability to align visual and textual representations with large-scale pretraining. Although these models achieve impressive zero-shot generalization, their deployment in real-world scenarios is hindered by a critical challenge: adapting to dynamic data distributions during inference. In practice, the decision boundaries of the target domain evolve over time, requiring models to stay stable against noise yet plastic to genuine shifts. A natural solution

† Corresponding author

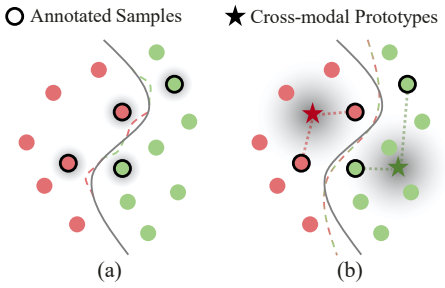

Fig. 1: Comparison of existing ATTA methods with PAA. (a) Existing ATTA methods apply local corrections, which often fail to consistently reshape the shared decision boundary under evolving shifts, especially for VLMs whose visual-textual semantics drift jointly. (b) Our PAA treats class-wise visual and textual prototypes as an evolving state and converts sparse supervision into stable yet plastic prototype updates that move the boundary in the right direction.

is to adapt VLMs to the target stream as it drifts. Efficient fine-tuning approaches such as prompt tuning [2, 18, 31, 45, 46, 48] or adapter-based methods [9, 20, 39, 41, 42] typically require labeled target data, which is impractical under abrupt shifts. Test-time adaptation (TTA) [15, 33] updates models online using unlabeled streams; recent VLM-specific extensions [6,8,17,22,24,27,40,43,44,47] show promising robustness gains. However, they often depend on pseudo-labels or heuristic strategies and lack guidance on where class boundaries should move under complex evolving shifts.

Active test-time adaptation (ATTA) [10,34] emerges as a promising paradigm to overcome these limitations by incorporating active learning into the adaptation process and querying limited human annotations to guide model adaptation. However, the supervision often operates as local corrections, with limited mechanisms that consolidate these signals into a structured update of the shared decision space, which is particularly limiting for VLMs whose multimodal semantics evolve across both visual and textual spaces; see Fig. 1(a). This reveals a gap: *we need a mechanism that turns sparse, delayed feedback into stable, structured updates of the evolving cross-modal distribution.*

To this end, we propose PAA (Probe, Anchor, and Amend), an ATTA framework for VLMs that treats visual and textual prototypes, namely class-level cached visual features and adapted text embeddings, as the evolving state of the target domain and updates this state through a simple three-step cycle over the test stream. For each incoming mini-batch, we first view the current prototypes as anchors that define the decision space and obtain predictions with a single forward pass. The cycle then executes three tightly coupled steps: the *Probe* step actively queries a few informative samples from the current mini-batch under a small annotation budget; the *Anchor* step uses lightweight binary verification to perform a fast, structure-preserving update of the shared prototype state; and the *Amend* step acts as a slow controller that revisits past hard cases across cycles and distills temporal discrepancies back into the prototypes.

By operating directly on prototypes, PAA converts sparse, delayed supervision into prototype-level adjustments while remaining inexpensive to deploy at test time; see Fig. 1(b).

For the *Probe* step, we use prototypes to both mitigate forgetting and track moving decision boundaries under a class-balanced budget. Concretely, we score samples against the current visual prototypes to form two complementary query sets: central samples with high prototype similarity, which reinforce core semantics and prevent the anchored state from eroding; and boundary samples with large nearest-prototype distance, which expose shifting distributions and indicate how prototypes and boundaries should be adjusted. To avoid over-focusing on frequent classes or transient bursts, the per-class query budget is adaptively split by combining a global distribution (statistics accumulated over the stream) with the current mini-batch distribution. This balanced probing policy ensures that a handful of binary verifications can both consolidate stable anchors and illuminate emerging shifts.

Given the selected samples, we perform two complementary updates to the prototype state. *First (Anchor)*, we use the binary verification obtained from the Probe step to calibrate confidence and to regularize the prototype-anchored decision space with three coordinated forces: (i) cross-modal alignment that keeps visual and textual prototypes consistent; (ii) an intra-class compactness term that pulls verified positives toward their visual prototypes; and (iii) an inter-class separation term that constrains textual residuals to move along discriminative directions rather than collapsing across classes. Together these forces yield a decision space that is stable yet plastic. *Second (Amend)*, we maintain a memory of error cases and periodically revisit them across cycles; when a previously wrong prediction flips to a different label with lower entropy, we treat the temporal discrepancy as a delayed but trustworthy rectification signal and distill it back into the prototype representation.

Extensive experiments across diverse datasets validate the effectiveness of PAA, which consistently outperforms VLM-specific TTA baselines and prior ATTA approaches under comparable annotation budgets and compute. Our contributions are summarized as follows:

- We present a simple yet effective PAA framework for active test-time adaptation in vision-language models, turning sparse, delayed feedback into structured evolution of the prototype-anchored decision space.
- We design a Probe strategy that leverages class prototypes to guide querying, effectively mitigating forgetting while enabling class-balanced exploration.
- We devise two complementary updates: an Anchor step that preserves cross-modal alignment and enhances class discrimination, and an Amend step that leverages temporal discrepancies for continual self-supervised refinement.

2 Related Work

Adaptation of Vision-Language Models. VLMs [5, 14, 16, 29, 37] excel at zero-shot tasks but struggle to adapt to new domains or distribution shifts. To

address this, prompt learning approaches [2, 18, 31, 45, 46, 48] optimize learnable text prompts with few-shot supervision, while adapter-based methods [9, 20, 39, 41, 42] refine features via lightweight modules without full fine-tuning. These methods require labeled target-domain data, limiting their practicality in dynamic, real-world scenarios. Test-time adaptation (TTA) methods adjust VLMs on unlabeled test streams. Test-time prompt tuning (TPT) [24] and its variants [8, 38] adapt prompts at test time. TDA [17] and BoostAdapter [43] leverage lightweight visual caches to store and retrieve information from test samples. Similarly, DMN [44] utilizes a dynamic memory to gather information from historical test data, while BATCLIP [22] and DPE [40] perform prototype-level cross-modal adaptation. ZERO [6] aggregates confident predictions over augmented views using a zero-temperature softmax, requiring only forward passes and no optimization. WATT [27] leverages weight averaging across various text templates. BCA [47] introduces a Bayesian framework to update both class embeddings and adaptive priors through posterior estimation. Different from these techniques, PAA leverages sparse online supervision in the form of binary verifications to drive a cyclic update during streaming VLM deployment.

Active Test-Time Adaptation. Test-time adaptation (TTA) [15, 21, 33] mitigates domain shifts by adapting pre-trained models with unlabeled test data, but its performance is often limited by the absence of ground-truth labels. To overcome this constraint, active test-time adaptation (ATTA) [10] integrates active learning into TTA, querying limited annotations during inference to guide model adaptation and mitigate catastrophic forgetting. It formulates adaptation as an optimization problem balancing supervised and unsupervised objectives with theoretical guarantees on error bounds. EATTA [34] advances this direction by introducing an effortless labeling paradigm that annotates at most one sample per batch. It identifies samples at the source-target domain boundary through feature perturbation sensitivity, which maximizes adaptation efficacy with minimal supervision. Different from these vision-only ATTA methods, our PAA operates on multimodal prototypes and couples active querying with prototype-regularized and memory-driven updates, turning sparse binary feedback into structured evolution of the VLM decision space.

3 Method

3.1 Preliminaries

In this paper, we propose a framework to perform ATTA on VLMs. We begin by introducing zero-shot inference with CLIP. Contrastive Language-Image Pretraining [29] is a VLM with two encoders: a visual encoder $\mathcal{E}_v(\cdot)$, which maps an image $\mathbf{x}$ to a visual feature $\mathcal{E}_v(\mathbf{x}) = \mathbf{z}^v \in \mathbb{R}^D$; and a text encoder $\mathcal{E}_t(\cdot)$, which converts a text prompt $\mathbf{t}$ into a text feature $\mathcal{E}_t(\mathbf{t}) = \mathbf{z}^t \in \mathbb{R}^D$. They are pretrained to align matched image-text pairs in a shared embedding space.

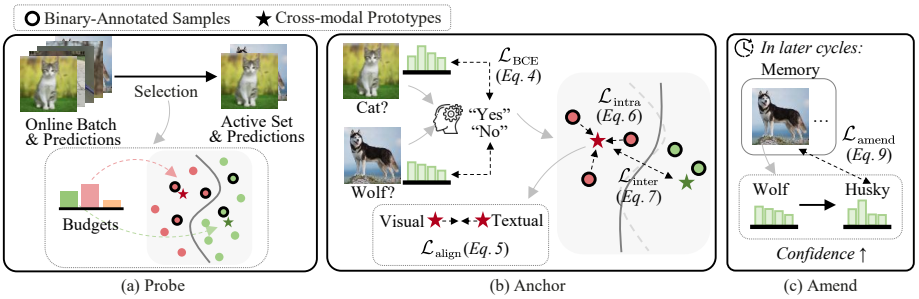

Fig. 2: Pipeline of PAA. (a) The *Probe* step scores the online batch against prototypes and, under a class-balanced budget, queries a few central and boundary samples. (b) The *Anchor* step turns binary verification into a prototype-regularized update that couples cross-modal prototype alignment with intra-class compactness and inter-class separation, adjusting the decision boundary while remaining stable. (c) The *Amend* step stores hard cases in a memory and, in later cycles, distills corrected lower-entropy predictions back into the prototypes for slow, robust refinement.

For zero-shot classification over C classes, each class c is represented by text prompts (*e.g.*, "a photo of a {class c}"). The text encoder transforms these prompts into class-specific embeddings $\{\mathbf{z}_c^t\}_{c=1}^C$. Given an image $\mathbf{x}$, the visual encoder outputs $\mathbf{z}^v$. The probability of assigning $\mathbf{x}$ to class c is computed as a softmax over the cosine similarities between $\mathbf{z}^v$ and each $\mathbf{z}_c^t$:

$$P(y=c \mid \mathbf{x}) = \frac{\exp\left(\cos(\mathbf{z}^v, \mathbf{z}_c^t)/\tau\right)}{\sum_{j=1}^{C} \exp\left(\cos(\mathbf{z}^v, \mathbf{z}_j^t)/\tau\right)}, \tag{1}$$

where τ is a temperature parameter scaling the logits, and $\cos(\mathbf{z}, \mathbf{z}') = \frac{\mathbf{z}^\top \mathbf{z}'}{\|\mathbf{z}\|_2 \cdot \|\mathbf{z}'\|_2}$ denotes cosine similarity.

3.2 Streaming Setup and Evolving State

At test time, we consider an unlabeled target stream arriving as mini-batches $\{\mathcal{X}^{(i)}\}$. For each mini-batch, we run a forward pass and construct an augmented batch $\mathcal{B}^{(i)} = \{(\mathbf{x}, \hat{y}) \mid \mathbf{x} \in \mathcal{X}^{(i)}\}$, where $\hat{y}$ is the predicted class. Starting from source-pretrained CLIP $(\mathcal{E}_v, \mathcal{E}_t)$, we adapt the model online by interleaving inference and updates.

Parameterization. To keep textual knowledge stable while allowing class prototypes to move toward the target domain, we freeze $\mathcal{E}_t$ and add a per-class learnable textual residual $\mathbf{r}_c^t$ to each base textual embedding $\mathbf{z}_c^t$, yielding adapted textual embeddings $\mathbf{p}_c^t = \text{norm}(\mathbf{z}_c^t + \mathbf{r}_c^t)$, where $\text{norm}(\cdot)$ denotes L2 normalization. The decision probability follows Eq. 1 with $\mathbf{p}_c^t$ substituting $\mathbf{z}_c^t$. On the visual side, the backbone is frozen and only LayerNorm parameters θ_{LN}^v are updated. The trainable set is thus $\Theta = \{\theta_{\text{LN}}^v, \mathbf{r}_1^t, \ldots, \mathbf{r}_C^t\}$.

Prototypes as an Evolving State. We treat class-wise visual and textual prototypes as the evolving state of the target distribution. For each class c, we keep a visual prototype cache $\mathcal{C}_c$ storing discriminative features together with prediction uncertainty and evolving during inference following TDA [17]. Given a test sample $\mathbf{x}$ with prediction $\hat{y}$ and entropy $\mathcal{H} = -\sum_{i=1}^{C} p_i \log p_i$, we insert $(\mathcal{E}_v(\mathbf{x}), \mathcal{H})$ into $\mathcal{C}_{\hat{y}}$; if the cache exceeds capacity H, we retain the H entries with the lowest prediction entropy. The visual prototype is the normalized average of cached features, denoted $\mathbf{p}_c^v$, and the textual prototype $\mathbf{p}_c^t$ is given in the parameterization above. This prototype state is the object maintained and updated by the three steps detailed next: **Probe** (Sec. 3.3), **Anchor** (Sec. 3.4), and **Amend** (Sec. 3.5).

3.3 Probe: Prototype-Guided Querying

Our Probe step identifies informative samples near class prototypes and decision boundaries for annotation, as illustrated in Fig. 2(a).

Distribution Estimation. To enable class-balanced and adaptive sample selection, we first estimate a global class distribution $\mathbf{G} = [g_1, g_2, \cdots, g_C] \in \mathbb{R}^C$ and a local batch-level distribution $\mathbf{L} = [l_1, l_2, \cdots, l_C] \in \mathbb{R}^C$. The global term $g_c = N_c^{\text{global}} / \sum_{c'=1}^{C} N_{c'}^{\text{global}}$ tracks class statistics over all processed test samples and aligns with the target domain, where N_c^{global} is the number of test samples pseudo-labeled as class c so far. Meanwhile, the local term $l_c = N_c^{\text{batch}} / \sum_{c'=1}^{C} N_{c'}^{\text{batch}}$ captures the proportion within the current batch $\mathcal{B}$, where N_c^{batch} counts pseudo-labeled samples of class c in $\mathcal{B}$. Both $\mathbf{G}$ and $\mathbf{L}$ are updated online and do not require any prior knowledge of the target stream. These two distributions jointly provide a principled basis for budget allocation.

Budget Allocation. To dynamically distribute the limited annotation budget, we maintain a cumulative budget K, initialized to 0, which tracks the total annotation budget over all cycles. At the beginning of each cycle, K is incremented by the per-cycle budget κ as $K \leftarrow K + \kappa$. We also maintain a historical selection counter $\mathbf{S} = [s_1, s_2, \cdots, s_C] \in \mathbb{N}^C$, initialized as $\mathbf{0}$, where s_c tracks the cumulative number of samples pseudo-labeled as class c and selected for annotation. The per-class budget k_c is then computed as $k_c = \lfloor \max\{K \cdot g_c - s_c, 0\} + \kappa \cdot l_c \rfloor$, where the global term $K \cdot g_c - s_c$ promotes long-term class balance aligned with $\mathbf{G}$, and the local term $\kappa \cdot l_c$ adjusts for short-term fluctuations captured by $\mathbf{L}$. This design balances exploration of globally under-represented classes with responsiveness to transient changes in class distribution.

Sample Selection. Leveraging the evolving prototype state described above, we use class-wise visual prototypes $\{\mathbf{p}_c^v\}$ to guide which samples to query. These prototypes summarize the current target geometry and serve as anchors for identifying both central and boundary samples within the batch $\mathcal{B}$, reinforcing well-established concepts while exposing ambiguous regions. For central samples, we

select the top-k_c instances ranked by descending cosine similarity $\cos(\mathcal{E}_v(\mathbf{x}), \mathbf{p}_c^v)$, prioritizing prototypical examples that reinforce current class semantics:

$$\mathcal{S}_c^{\text{center}} = \underset{(\mathbf{x},c)\in\mathcal{B}}{\text{argtopk}_{k_c}} \cos(\mathcal{E}_v(\mathbf{x}), \mathbf{p}_c^v), c \in \{1, ..., C\}. \tag{2}$$

For boundary samples, we compute the nearest-prototype distance over all classes $d(\mathbf{x}) = \min_{c'\in\{1,...,C\}}(1 - \cos(\mathcal{E}_v(\mathbf{x}), \mathbf{p}_{c'}^v))$, and select the top-$k_c$ instances ranked by descending $d(\mathbf{x})$, capturing challenging cases near decision boundaries:

$$\mathcal{S}_c^{\text{boundary}} = \underset{(\mathbf{x},c)\in\mathcal{B}}{\text{argtopk}_{k_c}} d(\mathbf{x}), c \in \{1, ..., C\}. \tag{3}$$

The final active query set for annotation is $\mathcal{S} = \bigcup_{c=1}^{C} (\mathcal{S}_c^{\text{center}} \cup \mathcal{S}_c^{\text{boundary}})$, and we update the historical selection counter as $s_c \leftarrow s_c + \min(k_c, |\{\mathbf{x}|(\mathbf{x}, c) \in \mathcal{B}\}|)$.

3.4 Anchor: Prototype-Regularized Update

Our Anchor step applies a prototype-regularized update to the decision space carried over from the previous cycle, using only binary verification to keep the annotation cost low and mitigate interaction bottlenecks. As shown in Fig. 2(b), for each sample-prediction pair $(\mathbf{x}, \hat{y}) \in \mathcal{S}$ with predicted probability $p_{\hat{y}}$, an oracle $\mathcal{O}$ provides a binary label $v_{\mathbf{x}} = \mathcal{O}(\mathbf{x}, \hat{y}) \in \{0, 1\}$, where $v_{\mathbf{x}} = 1$ indicates correctness and $v_{\mathbf{x}} = 0$ otherwise. We use a binary cross-entropy loss to calibrate model confidence with this binary supervision:

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{|\mathcal{S}|} \sum_{(\mathbf{x},\hat{y})\in\mathcal{S}} \left[v_{\mathbf{x}} \log p_{\hat{y}} + (1 - v_{\mathbf{x}}) \log(1 - p_{\hat{y}}) \right]. \tag{4}$$

Prototype Alignment. As established earlier, each class c is characterized by a visual prototype $\mathbf{p}_c^v$ and a textual prototype $\mathbf{p}_c^t$. To maintain cross-modal consistency, we employ an Info-NCE loss [26] that encourages alignment of matched visual-textual prototype pairs while discouraging mismatched ones:

$$\mathcal{L}_{\text{align}} = \frac{1}{C} \sum_{c=1}^{C} \left(-\log \frac{e^{\mathbf{p}_c^{t\top} \mathbf{p}_c^v}}{\sum_{c'} e^{\mathbf{p}_c^{t\top} \mathbf{p}_{c'}^v}} - \log \frac{e^{\mathbf{p}_c^{t\top} \mathbf{p}_c^v}}{\sum_{c'} e^{\mathbf{p}_{c'}^{t\top} \mathbf{p}_c^v}} \right). \tag{5}$$

Intra-Class Compactness. We further stabilize class representations with an intra-class compactness loss that pulls $\mathcal{E}_v(\mathbf{x})$ closer to its predicted class prototype when $v_{\mathbf{x}} = 1$:

$$\mathcal{L}_{\text{intra}} = -\frac{1}{|\mathcal{S}|} \sum_{(\mathbf{x},\hat{y})\in\mathcal{S}} v_{\mathbf{x}} \cos(\mathcal{E}_v(\mathbf{x}), \mathbf{p}_{\hat{y}}^v). \tag{6}$$

Inter-Class Separation. As previously defined, the textual prototype of class c in the target domain is given by $\mathbf{p}_c^t = \text{norm}(\mathbf{z}_c^t + \mathbf{r}_c^t)$, where $\mathbf{r}_c^t$ explicitly encodes the domain shift direction to adapt prototypes to the target distribution. To ensure prototypes remain discriminative under shift, we penalize alignment between each residual shift vector $\mathbf{r}_c^t$ and inter-class discrepancy vectors $(\mathbf{z}_{c'}^t - \mathbf{z}_c^t)$ derived from source-domain prototypes, through an inter-class separation loss:

$$\mathcal{L}_{\text{inter}} = \frac{1}{C(C-1)} \sum_{c=1}^{C} \sum_{c' \neq c} \cos(\mathbf{r}_c^t, \mathbf{z}_{c'}^t - \mathbf{z}_c^t). \tag{7}$$

The overall objective of the Anchor step combines the above components as a weighted sum:

$$\mathcal{L}_{\text{anchor}} = \mathcal{L}_{\text{BCE}} + \lambda_{\text{anchor}}(\mathcal{L}_{\text{align}} + \mathcal{L}_{\text{intra}} + \mathcal{L}_{\text{inter}}), \tag{8}$$

where λ_{anchor} is a balancing hyper-parameter.

3.5 Amend: Memory-Driven Self-Refinement

Our Amend step leverages historical errors as a slow controller that provides delayed, stabilizing corrections to the evolving prototypes, as shown in Fig. 2(c). Specifically, we record test samples from the active query set $\mathcal{S}$ whose predictions are verified as incorrect by the oracle. These samples are stored in a memory bank $\mathcal{M}$, where each entry $\mathcal{M}_i = (\mathbf{x}_i, \hat{y}_i, \mathcal{H}_i, \ell_i)$ contains a test sample $\mathbf{x}_i$, its initial pseudo-label $\hat{y}_i$, associated self-entropy $\mathcal{H}_i$, and a lifetime counter ℓ_i indicating its remaining validity.

In subsequent adaptation cycles, we assess whether previously incorrect predictions have been corrected. For each entry $(\mathbf{x}, \hat{y}, \mathcal{H}, \ell)$, the updated model generates a new prediction $\hat{y}'$, probability $p'_{\hat{y}'}$, and self-entropy $\mathcal{H}'$. If the new prediction $\hat{y}' \neq \hat{y}$ and the confidence improves (*i.e.*, $\mathcal{H}' < \mathcal{H}$), we treat the change as an error correction. A memory-driven self-refinement loss is then computed:

$$\mathcal{L}_{\text{amend}} = -\frac{1}{|\Delta|} \sum_{(\mathbf{x}, \hat{y}') \in \Delta} \log p'_{\hat{y}'}, \tag{9}$$

where $\Delta = \{(\mathbf{x}, \hat{y}') \mid (\mathbf{x}, \hat{y}, \mathcal{H}, \ell) \in \mathcal{M} \wedge \hat{y}' \neq \hat{y} \wedge \mathcal{H}' < \mathcal{H}\}$ collects corrected samples. The overall adaptation objective integrates the immediate Anchor loss with the delayed Amend loss:

$$\mathcal{L} = \mathcal{L}_{\text{anchor}} + \lambda_{\text{amend}} \mathcal{L}_{\text{amend}}, \tag{10}$$

where λ_{amend} controls the memory influence strength.

At the end of each adaptation cycle, each memory entry $(\mathbf{x}, \hat{y}, \mathcal{H}, \ell)$ is updated according to the following rule:

$$\begin{cases} (\mathbf{x}, \hat{y}, \mathcal{H}, \ell - 1), & \text{if } (\ell > 1) \wedge \big(\hat{y}' = \hat{y} \vee \mathcal{H}' \geq \mathcal{H}\big) \\ \text{removed}, & \text{otherwise} \end{cases}, \tag{11}$$

Table 1: Results on the Natural Distribution Shifts benchmark.

	Method	ImageNet	ImageNet-A	ImageNet-V2	ImageNet-R	ImageNet-S	Average	OOD Average
Non-Active	CLIP-ResNet-50	58.16	21.83	51.41	56.15	33.37	44.18	40.69
	Ensemble [24]	59.81	23.24	52.91	60.72	35.48	46.43	43.09
	TPT [24]	60.74	26.67	54.70	59.11	35.09	47.26	43.89
	DiffTPT [8]	60.80	31.06	55.80	58.80	37.10	48.71	45.69
	TDA [17]	61.35	30.29	55.54	62.58	38.12	49.58	46.63
	WATT [27]	59.86	27.53	53.70	59.42	36.55	47.41	44.30
	ZERO [6]	60.41	30.32	54.59	57.83	34.93	47.62	44.42
	DPE [40]	63.41	30.15	56.72	63.72	40.03	50.81	47.66
	BoostAdapter [43]	62.14	35.47	56.31	62.61	38.80	51.07	48.30
	BCA [47]	61.35	30.29	55.54	62.58	38.12	49.94	46.98
	BATCLIP [22]	62.73	32.68	56.02	63.11	39.47	50.80	47.82
Active	SimATTA [10]	60.31	35.15	56.34	62.63	39.47	50.78	48.40
	EATTA [34]	60.99	35.68	56.29	62.71	39.02	50.94	48.43
	PAA	**63.42**	**37.27**	**56.77**	**67.37**	**41.03**	**53.15**	**50.61**
Non-Active	CLIP-ViT-B/16	66.73	47.87	60.86	73.98	46.09	59.11	57.20
	Ensemble [24]	68.34	49.89	61.88	77.65	48.24	61.20	59.42
	TPT [24]	68.98	54.77	63.45	77.06	47.94	62.44	60.81
	DiffTPT [8]	70.30	55.68	65.10	75.00	46.80	62.28	60.52
	TDA [17]	69.51	60.11	64.67	80.24	50.54	65.01	63.89
	WATT [27]	69.10	55.91	63.43	77.60	48.74	62.96	61.42
	ZERO [6]	71.17	62.75	65.23	80.75	50.59	66.10	64.83
	DPE [40]	71.91	59.63	65.44	80.40	52.26	65.93	64.43
	BoostAdapter [43]	69.37	64.53	**65.51**	80.95	51.28	66.33	65.57
	BCA [47]	70.22	61.14	64.90	80.72	50.87	65.37	64.16
	BATCLIP [22]	70.60	62.03	65.18	80.66	51.49	65.99	64.84
Active	SimATTA [10]	67.07	63.63	64.42	79.30	50.12	64.91	64.37
	EATTA [34]	67.10	64.85	64.52	79.59	49.79	65.17	64.69
	PAA	**72.03**	**66.59**	**65.51**	**82.93**	**53.89**	**68.19**	**67.23**

where test samples with persistent errors have their lifetimes decremented, while corrected or expired entries are removed. Newly mispredicted test samples create new entries in $\mathcal{M}$ with initial lifetime ℓ_0 for future correction.

4 Experiments

4.1 Experimental Setups

Benchmarks. Following prior work on test-time adaptation for VLMs, we evaluate our approach on two benchmarks: the Natural Distribution Shifts benchmark and the Cross-Domain benchmark. The former evaluates the model's robustness on ImageNet [4] and its four variants, including ImageNet-A [13], ImageNet-V2 [30], ImageNet-R [12], and ImageNet-Sketch [35]. The latter assesses the model's transferring performance with 10 datasets, including Aircraft [23], Caltech101 [7], Cars [19], DTD [3], EuroSAT [11], Flower102 [25], Food101 [1], Pets [28], SUN397 [36], and UCF101 [32]. We follow the dataset split in CoOp [46] as per the common protocol and report the top-1 accuracy as our evaluation metric.

Implementation Details. We adopt CLIP with pre-trained ResNet50 and ViT-B/16 backbones as visual encoders. In our active test-time adaptation ex-

Table 2: Results on the Cross-Domain benchmark.

	Method	Aircraft	Caltech101	Cars	DTD	EuroSAT	Flower102	Food101	Pets	SUN397	UCF101	Average
Non-Active	CLIP-ResNet-50	15.66	85.88	55.70	40.37	23.69	61.75	73.97	83.57	58.80	58.84	55.82
	Ensemble [24]	16.11	87.26	55.89	40.37	25.79	62.77	74.82	82.97	60.85	59.48	56.63
	TPT [24]	17.58	87.02	58.46	40.84	28.33	62.69	74.88	84.49	61.46	60.82	57.66
	DiffTPT [8]	17.60	86.89	60.71	40.72	41.04	63.53	79.21	83.40	62.72	62.67	59.85
	TDA [17]	17.61	89.70	57.78	43.74	42.11	68.74	77.75	86.18	62.53	64.18	61.03
	WATT [27]	17.49	85.80	56.40	42.08	37.22	62.73	76.13	84.71	60.62	61.43	58.46
	ZERO [6]	17.73	85.31	58.36	40.96	15.64	60.33	75.73	84.22	60.92	60.72	55.99
	DPE [40]	19.80	90.83	59.26	**50.18**	41.67	67.60	77.83	85.97	64.23	61.98	61.93
	BoostAdapter [43]	18.87	88.36	59.82	43.91	40.12	67.72	78.79	85.96	62.61	64.45	61.06
	BCA [47]	19.89	89.70	58.13	48.58	42.12	66.30	77.19	85.58	63.38	63.51	61.44
	BATCLIP [22]	19.56	89.94	59.31	47.16	41.85	67.19	77.64	85.83	63.47	63.10	61.51
Active	SimATTA [10]	19.08	88.36	58.70	43.91	78.44	67.52	78.25	86.05	62.92	65.03	64.83
	EATTA [34]	19.74	88.36	59.64	44.09	79.84	67.64	78.43	85.50	62.86	64.71	65.08
	PAA	**19.98**	**90.87**	**61.62**	47.87	**83.43**	**71.09**	**80.05**	**87.65**	**64.62**	**68.65**	**67.58**
Non-Active	CLIP-ViT-B/16	23.67	93.35	65.48	44.27	42.01	67.44	83.65	88.25	62.59	65.13	63.58
	Ensemble [24]	23.22	93.55	66.11	45.04	50.42	66.99	82.86	86.92	65.63	65.16	64.59
	TPT [24]	24.78	94.16	66.87	47.75	42.44	68.98	84.67	87.79	65.50	68.04	65.10
	DiffTPT [8]	25.60	92.49	67.01	47.00	43.13	70.10	87.23	88.22	65.74	62.67	65.47
	TDA [17]	23.91	94.24	67.28	47.40	58.00	71.42	86.14	88.63	67.62	70.66	67.53
	WATT [27]	24.15	93.35	66.48	46.16	56.32	68.13	85.87	89.04	66.46	68.65	66.46
	ZERO [6]	25.21	93.66	68.04	46.12	34.33	67.68	86.53	87.75	65.03	67.77	64.21
	DPE [40]	**28.95**	94.81	67.31	**54.20**	55.79	75.07	86.17	91.14	70.07	70.44	69.40
	BoostAdapter [43]	27.45	94.77	69.30	45.69	61.22	71.66	87.17	89.51	68.09	71.93	68.68
	BCA [47]	28.59	94.69	66.86	53.49	56.63	73.12	85.97	90.43	68.41	67.59	68.59
	BATCLIP [22]	28.74	94.77	67.94	51.95	57.09	73.89	86.02	90.22	69.37	69.13	68.91
Active	SimATTA [10]	27.45	94.32	68.76	46.69	87.74	72.84	86.81	90.21	66.48	72.11	71.34
	EATTA [34]	27.84	94.85	68.64	47.93	88.21	71.34	86.89	89.29	67.22	72.64	71.49
	PAA	28.50	**95.70**	**70.46**	49.05	**90.93**	**75.56**	**88.24**	**92.18**	**70.29**	**75.05**	**73.60**

periments, the mini-batch size is set to 128, the per-cycle annotation budget to $\kappa = 16$ for each of central and boundary querying, and the initial memory lifetime to $\ell_0 = 3$. The balancing hyper-parameters are set to $\lambda_{\text{anchor}} = 0.1$ and $\lambda_{\text{amend}} = 0.1$. For data augmentation, we follow TPT [24] to generate 63 augmented views per test sample during inference, which are aggregated to improve adaptation stability. Following TDA [17], the prototype cache capacity is set to 3. All experiments are conducted on a single NVIDIA GeForce RTX 3090 GPU.

Compared Methods. We compare PAA with two groups of baselines: (1) Non-active methods, which do not query annotations at test time. This group includes zero-shot CLIP [29], its prompt-ensemble variant Ensemble [24], and recent test-time adaptation methods for VLMs: TPT [24], DiffTPT [8], TDA [17], WATT [27], ZERO [6], DPE [40], BoostAdapter [43], BCA [47], and BATCLIP [22]. (2) ATTA methods, which query limited annotations during adaptation. We adopt SimATTA [10] and EATTA [34], originally proposed for vision-only backbones, and extend them to our VLM setting. Their active query budgets are adjusted to match ours for a fair comparison.

4.2 Results and Discussions

Robustness on Natural Distribution Shifts. As shown in Tab. 1, PAA consistently outperforms both non-active and active baselines across all five datasets.

Table 3: Efficiency comparison on ImageNet-A. We report the testing time, achieved accuracy, and performance gains compared to zero-shot CLIP.

Method	Testing Time	Accuracy (%)	Gain
CLIP-ViT-B/16	**1min**	47.87	-
TPT [24]	> 2 h	54.77	+6.90
DiffTPT [8]	> 4 h	55.68	+7.81
TDA [17]	11 min	60.11	+12.24
DPE [40]	38 min	59.63	+11.76
BoostAdapter [43]	17 min	64.53	+16.66
SimATTA [10]	18 min	63.63	+15.76
PAA	17 min	**66.59**	**+18.72**

With the ResNet-50 backbone, it achieves an average accuracy of 53.15%, surpassing all active and non-active baselines. Compared with the strongest non-active TTA baselines (DPE, BoostAdapter, BCA, and BATCLIP), PAA yields average accuracy gains between 2.08% and 3.21%, and it exceeds two recent ATTA baselines, SimATTA and EATTA, by 2.37% and 2.21%. With the ViT-B/16 backbone, PAA achieves an average accuracy of 68.19%, outperforming the strongest non-active TTA baseline by 1.86% and the best ATTA baseline by 3.02%. The gains are particularly pronounced under stronger distribution shifts such as those in ImageNet-A and ImageNet-R, indicating that our framework is especially beneficial when the target distribution departs substantially from the pre-training domain. These results demonstrate that turning sparse, delayed feedback into structured prototype updates leads to consistently stronger robustness under natural distribution shifts.

Cross-Domain Generalization. In Tab. 2, we further evaluate the cross-domain generalization of PAA on ten fine-grained datasets. Our method consistently surpasses all baselines, achieving average top-1 accuracy gains from 2.50% to 11.76% with ResNet-50 and from 2.11% to 10.02% with ViT-B/16 backbone. The gains are particularly pronounced on EuroSAT and UCF101, where the target domains differ substantially from the pre-training distribution, indicating that prototype-anchored updates and class-balanced querying help PAA establish stable anchors even under large semantic and appearance shifts.

Efficiency Comparison. Tab. 3 compares the computational efficiency of PAA with baselines on ImageNet-A using a single RTX 3090 GPU. In our framework, the main computational cost stems from updating the LayerNorm parameters in the visual encoder, together with the class-wise textual residual vectors. Since PAA performs a single prototype-regularized update per mini-batch on a small set of parameters, it effectively mitigates redundant computation. Empirically, our method yields lower adaptation latency than TPT, DiffTPT, and DPE, while consistently outperforming them in accuracy. Compared to TDA, BoostAdapter, and SimATTA, PAA maintains comparable processing speed yet achieves superior performance, demonstrating a favorable trade-off between adaptation quality

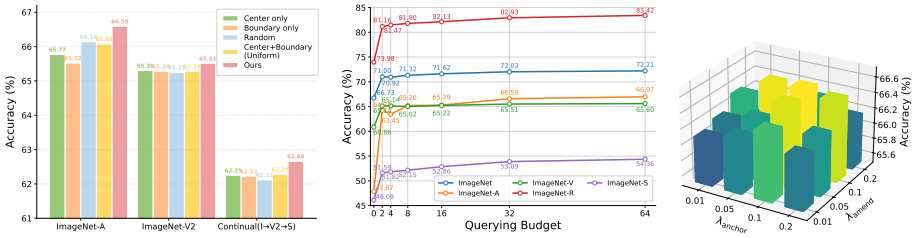

Fig. 3: Ablations of querying strategies, querying budgets and hyper-parameters.

and computational cost. These results validate the practicality of our method for real-world deployment.

4.3 Ablation Studies

Different Querying Strategies. Fig. 3 (*left*) compares five querying strategies under a fixed annotation budget: center-only, boundary-only, random, center-boundary querying with uniform budget, and our class-balanced method that integrates both central and boundary samples. Evaluation is conducted on both stationary domains (ImageNet-A and ImageNet-V2) and sequential shifts (ImageNet → ImageNet-A → ImageNet-S). Our strategy consistently outperforms all baselines by jointly reinforcing learned class semantics and encouraging exploration near decision boundaries. Compared to center-only, boundary-only, random, and uniform center-boundary querying, it achieves average improvements of 0.48%, 0.58%, 0.42%, and 0.37%, respectively, across the three scenarios. These results highlight the advantage of our prototype-guided, class-balanced querying strategy for effective and adaptive model refinement.

Different Querying Budgets. We assess the impact of varying annotation budgets on overall performance in the Natural Domain Shift benchmark, where budget 0 corresponds to zero-shot CLIP. As shown in Fig. 3 (*middle*), increasing the budget progressively enhances adaptation performance as more informative supervision becomes available. However, larger budgets incur higher computational costs. Setting the budget to 32 offers a balanced trade-off between adaptation performance and efficiency.

Hyper-Parameter Robustness. There are two main hyper-parameters in PAA, *i.e.*, the balancing parameters λ_{anchor} in Eq. 8 and λ_{amend} in Eq. 10. We conduct experiments on ImageNet-A and ViT-B/16 backbone to investigate the robustness by varying both parameters among $\{0.01, 0.05, 0.1, 0.2\}$. As shown in Fig. 3 (*right*), the performance remains stable across a wide range of parameter values and only degrades slightly when both terms are overly amplified. The default setting $\lambda_{\text{anchor}} = \lambda_{\text{amend}} = 0.1$ lies in a broad plateau region and is therefore adopted in all other experiments.

Table 4: Ablation studies of adaptation objectives and learnable modules.

	ImageNet	-A	-V2	-R	-S	Average
w/o $\mathcal{L}_{\text{align}}$	71.61	66.11	65.08	82.65	53.53	67.80
w/o $\mathcal{L}_{\text{intra}}$	71.82	66.16	65.31	82.63	53.69	67.92
w/o $\mathcal{L}_{\text{inter}}$	71.75	66.48	65.37	82.84	53.70	68.03
w/o $\mathcal{L}_{\text{amend}}$	71.79	66.27	65.50	82.68	53.67	67.98
Freeze θ^v_{LN}	71.55	65.24	65.14	81.85	52.61	67.28
Freeze $\{\mathbf{r}^t_c\}$	71.27	65.08	65.12	82.18	53.26	67.38
$\{\mathbf{r}^t_c\} \rightarrow \theta^t_{\text{LN}}$	71.76	65.33	65.22	82.47	53.28	67.61
Full Model	**72.03**	**66.59**	**65.51**	**82.93**	**53.89**	**68.19**

Table 5: Effects of visual cache sizes.

H (Cache Size)	1	2	**3**	4	5	6	7
Accuracy (%)	65.22	66.12	**66.59**	66.09	65.87	65.50	65.52

Effects of Adaptation Objectives. Tab. 4 (*upper block*) reports the impact of removing individual adaptation objectives. All components contribute positively to the overall effectiveness of our method. This supports our design of combining a fast, verification-driven Anchor update with a delayed, memory-based Amend signal: the former reacts immediately to newly queried labels, while the latter accumulates temporal corrections from historical hard cases and feeds them back to the prototype state in a more conservative manner. Notably, excluding the alignment loss $\mathcal{L}_{\text{align}}$ causes the most significant accuracy drop (0.39%), highlighting the importance of maintaining cross-modal consistency between visual and textual prototypes as the target distribution shifts.

Effects of Learnable Modules. We further evaluate the contributions of the LayerNorm parameters in visual encoder θ^v_{LN} and the textual residuals $\{\mathbf{r}^t_c\}$. As shown in Tab. 4 (*lower block*), freezing either module reduces flexibility for cross-modal alignment, resulting in an average accuracy loss of 0.91% for θ^v_{LN} and 0.81% for $\{\mathbf{r}^t_c\}$. Jointly fine-tuning both modules yields the best performance by allowing dynamic adaptation of visual and textual prototypes to evolving target domains. Moreover, the variant $\{\mathbf{r}^t_c\} \rightarrow \theta^t_{\text{LN}}$ that replaces the residual vectors $\{\mathbf{r}^t_c\}$ with trainable text-side LayerNorm parameters θ^t_{LN} also lags behind our full model. This shows that the textual residuals are more effective at steering class-wise prototypes while preserving the pretrained text semantics and incurring only a modest number of additional parameters.

Different Cache Sizes. We investigate the impact of prototype cache size H using ImageNet-A dataset and ViT-B/16 backbone. As shown in Tab. 5, increasing H from 1 to 3 yields a performance gain of 1.37%, while further enlarging the cache size results in a gradual performance drop. We attribute this observation to a trade-off between feature diversity and noise: a small cache may

Table 6: Effects of different lifetimes.

Lifetime	1	2	3	4	5
Accuracy (%)	66.32	66.42	66.59	66.60	66.63
Testing Time	16min	16min	17min	18min	19min

Table 7: Performance under different annotation levels.

Annotation	ImageNet	-A	-V2	-R	-S	Average
Zero-Shot	66.73	47.87	60.86	73.98	46.09	59.11
Binary Verification	72.03	66.59	65.51	82.93	53.89	68.19
Full-Class Label	72.27	67.57	65.68	83.25	55.45	68.84

fail to capture sufficient feature variation, while an excessively large cache tends to include lower-confidence or noisy samples.

Different Memory Lifetimes. Tab. 6 illustrates the effects of different initial memory lifetimes on ImageNet-A with ViT-B/16 backbone. While increasing the lifetime provides more opportunities for correcting persistent errors, it introduces a decrease in adaptation efficiency due to revisiting outdated entries. We observe that our method is robust across a range of ℓ_0 values, with average performance consistently surpassing the no-memory baseline in Tab. 4 (w/o $\mathcal{L}_{\text{amend}}$), and we adopt $\ell_0 = 3$ as a practical default balancing refinement and efficiency.

Binary Verification versus Full Label. To assess the effectiveness of our adaptation method under binary feedback, we compare it against a stronger variant that leverages full-class labels with standard cross-entropy loss, serving as an upper-bound baseline. Tab. 7 reports results for both variants on the Natural Distribution Shifts benchmark using a ViT-B/16 backbone. Remarkably, our method achieves comparable accuracy to the full-supervision counterpart, indicating that our PAA can effectively exploit sparse binary feedback to achieve competitive performance.

5 Conclusions and Limitations

We present PAA, an ATTA framework for VLMs that treats multimodal prototypes as an evolving state and updates this state through a Probe-Anchor-Amend cycle. By combining prototype-guided, class-balanced querying, a prototype-regularized update that preserves decision-space structure, and memory-driven self-refinement that distills temporal discrepancies into delayed corrections, PAA turns sparse binary feedback into stable evolution of the prototype-anchored decision space, yielding label-efficient robustness under diverse distribution shifts.

While our method significantly improves VLM robustness, prototype-based fine-tuning introduces additional latency; future work will explore lighter adapter designs to improve efficiency and deployment feasibility.

References

1. Bossard, L., Guillaumin, M., Van Gool, L.: Food-101 – mining discriminative components with random forests. In: ECCV (2014)
2. Cho, E., Kim, J., Kim, H.J.: Distribution-aware prompt tuning for vision-language models. In: ICCV (2023)
3. Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., Vedaldi, A.: Describing textures in the wild. In: CVPR (2014)
4. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)
5. Desai, K., Johnson, J.: VirTex: Learning Visual Representations from Textual Annotations. In: CVPR (2021)
6. Farina, M., Franchi, G., Iacca, G., Mancini, M., Ricci, E.: Frustratingly easy test-time adaptation of vision-language models. NeurIPS (2024)
7. Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In: CVPR Workshops (2004)
8. Feng, C.M., Yu, K., Liu, Y., Khan, S., Zuo, W.: Diverse data augmentation with diffusions for effective test-time prompt tuning. In: ICCV (2023)
9. Gao, P., Geng, S., Zhang, R., Ma, T., Fang, R., Zhang, Y., Li, H., Qiao, Y.: Clip-adapter: Better vision-language models with feature adapters. IJCV (2024)
10. Gui, S., Li, X., Ji, S.: Active test-time adaptation: Theoretical analyses and an algorithm. In: ICLR (2024)
11. Helber, P., Bischke, B., Dengel, A., Borth, D.: Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. IEEE J. Sel. Top. Appl. Earth Obs. Remote. Sens. (2019)
12. Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., Song, D., Steinhardt, J., Gilmer, J.: The many faces of robustness: A critical analysis of out-of-distribution generalization. In: ICCV (2021)
13. Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., Song, D.: Natural adversarial examples. CVPR (2021)
14. Hu, X., Gan, Z., Wang, J., Yang, Z., Liu, Z., Lu, Y., Wang, L.: Scaling up vision-language pre-training for image captioning. In: CVPR (2022)
15. Iwasawa, Y., Matsuo, Y.: Test-time classifier adjustment module for model-agnostic domain generalization. In: NeurIPS (2021)
16. Jia, C., Yang, Y., Xia, Y., Chen, Y.T., Parekh, Z., Pham, H., Le, Q., Sung, Y.H., Li, Z., Duerig, T.: Scaling up visual and vision-language representation learning with noisy text supervision. In: ICML (2021)
17. Karmanov, A., Guan, D., Lu, S., El Saddik, A., Xing, E.: Efficient test-time adaptation of vision-language models. In: CVPR (2024)
18. Khattak, M.U., Rasheed, H., Maaz, M., Khan, S., Khan, F.S.: Maple: Multi-modal prompt learning. In: CVPR (2023)
19. Krause, J., Stark, M., Deng, J., Fei-Fei, L.: 3d object representations for fine-grained categorization. In: ICCV Workshops (2013)
20. Li, X., Lian, D., Lu, Z., Bai, J., Chen, Z., Wang, X.: Graphadapter: Tuning vision-language models with dual knowledge graph. NeurIPS (2024)
21. Liang, J., He, R., Tan, T.: A comprehensive survey on test-time adaptation under distribution shifts. IJCV (2025)

22. Maharana, S., Zhang, B., Karlinsky, L., Feris, R., Guo, Y.: Batclip: Bimodal online test-time adaptation for clip. In: ICCV (2025)
23. Maji, S., Kannala, J., Rahtu, E., Blaschko, M., Vedaldi, A.: Fine-grained visual classification of aircraft. arXiv preprint arXiv:1306.5151 (2013)
24. Manli, S., Weili, N., De-An, H., Zhiding, Y., Tom, G., Anima, A., Chaowei, X.: Test-time prompt tuning for zero-shot generalization in vision-language models. In: NeurIPS (2022)
25. Nilsback, M.E., Zisserman, A.: Automated flower classification over a large number of classes. In: Indian Conference on Computer Vision, Graphics and Image Processing (2008)
26. Oord, A.v.d., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 (2018)
27. Osowiechi, D., Noori, M., Hakim, G.A.V., Yazdanpanah, M., Bahri, A., Cheraghalikhani, M., Dastani, S., Beizaee, F., Ayed, I.B., Desrosiers, C.: Watt: Weight average test-time adaption of clip. In: NeurIPS (2024)
28. Parkhi, O.M., Vedaldi, A., Zisserman, A., Jawahar, C.V.: Cats and dogs. In: CVPR (2012)
29. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: ICML (2021)
30. Recht, B., Roelofs, R., Schmidt, L., Shankar, V.: Do imagenet classifiers generalize to imagenet? In: ICML (2019)
31. Shen, S., Yang, S., Zhang, T., Zhai, B., Gonzalez, J.E., Keutzer, K., Darrell, T.: Multitask vision-language prompt tuning. In: WACV (2024)
32. Soomro, K., Zamir, A.R., Shah, M.: UCF101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402 (2012)
33. Wang, D., Shelhamer, E., Liu, S., Olshausen, B., Darrell, T.: Tent: Fully test-time adaptation by entropy minimization. In: ICLR (2021)
34. Wang, G., Ding, C.: Effortless active labeling for long-term test-time adaptation. In: CVPR (2025)
35. Wang, H., Ge, S., Lipton, Z., Xing, E.P.: Learning robust global representations by penalizing local predictive power. In: NeurIPS (2019)
36. Xiao, J., Hays, J., Ehinger, K.A., Oliva, A., Torralba, A.: Sun database: Large-scale scene recognition from abbey to zoo. In: CVPR (2010)
37. Yang, J., Duan, J., Tran, S., Xu, Y., Chanda, S., Chen, L., Zeng, B., Chilimbi, T., Huang, J.: Vision-language pre-training with triple contrastive learning. In: CVPR (2022)
38. Yoon, H.S., Yoon, E., Tee, J.T.J., Hasegawa-Johnson, M.A., Li, Y., Yoo, C.D.: C-TPT: Calibrated test-time prompt tuning for vision-language models via text feature dispersion. In: ICLR (2024)
39. Yu, T., Lu, Z., Jin, X., Chen, Z., Wang, X.: Task residual for tuning vision-language models. In: CVPR (2023)
40. Zhang, C., Stepputtis, S., Sycara, K., Xie, Y.: Dual prototype evolving for test-time generalization of vision-language models. In: NeurIPS (2024)
41. Zhang, R., Hu, X., Li, B., Huang, S., Deng, H., Qiao, Y., Gao, P., Li, H.: Prompt, generate, then cache: Cascade of foundation models makes strong few-shot learners. In: CVPR (2023)
42. Zhang, R., Zhang, W., Fang, R., Gao, P., Li, K., Dai, J., Qiao, Y., Li, H.: Tip-adapter: Training-free adaption of clip for few-shot classification. In: ECCV (2022)

43. Zhang, T., Wang, J., Guo, H., Dai, T., Chen, B., Xia, S.T.: Boostadapter: Improving vision-language test-time adaptation via regional bootstrapping. In: NeurIPS (2024)
44. Zhang, Y., Zhu, W., Tang, H., Ma, Z., Zhou, K., Zhang, L.: Dual memory networks: A versatile adaptation approach for vision-language models. In: CVPR (2024)
45. Zhou, K., Yang, J., Loy, C.C., Liu, Z.: Conditional prompt learning for vision-language models. In: CVPR (2022)
46. Zhou, K., Yang, J., Loy, C.C., Liu, Z.: Learning to prompt for vision-language models. IJCV (2022)
47. Zhou, L., Ye, M., Li, S., Li, N., Zhu, X., Deng, L., Liu, H., Lei, Z.: Bayesian test-time adaptation for vision-language models. In: CVPR (2025)
48. Zhu, B., Niu, Y., Han, Y., Wu, Y., Zhang, H.: Prompt-aligned gradient for prompt tuning. In: ICCV (2023)