Training-Free Active Test-Time Adaptation for Vision-Language Models

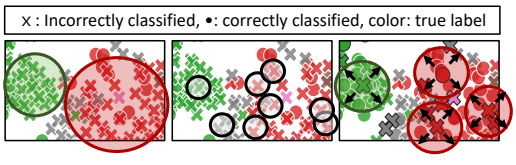
Jihwan Bang[1] **Sumyeong Ahn**[2] **Hwanjun Song**[1] **Jae-Gil Lee**[1*]
KAIST[1] KENTECH[2]
{jihwan.bang, songhwanjun, jaegil}@kaist.ac.kr, sumyeongahn@kentech.ac.kr

Abstract

Test-time adaptation (TTA) for vision-language models (VLMs) is usually studied in a fully unsupervised regime, where adaptation relies on pseudo-labels, confidence scores, or entropy computed by the model itself. This is a severe restriction on hard shifted samples: the model is asked to repair its own uncertain predictions without any external evidence. At the other extreme, supervised online adaptation assumes labels can directly train or correct the model. We study the intermediate deployment regime between these two extremes. In *Active Test-time Adaptation for VLMs* (**Active VLM-TTA**), a pretrained VLM may spend an explicit labeling budget on selected test samples, but the prediction for the current sample must be emitted before its label can be used; queried labels are therefore costed, delayed evidence for future samples rather than free correction for the present one. Under this future-only constraint, the central question is how to reuse sparse labels conservatively. We instantiate this setting with **ACT**ive VLM-TTA for **O**nline **R**etrieval (ACTOR), a training-free plug-in module that routes uncertain samples among base prediction, label query, and memory-based correction using a consistency-and-reliability check over previously queried labels. Across ten cross-domain benchmarks and a domain-transfer benchmark based on ImageNet, ACTOR consistently improves strong VLM-TTA baselines on ResNet and ViT backbones with negligible inference overhead. Our implementation is publicly available at https://github.com/kaist-dmlab/ACTOR.

1 Introduction

Vision-language models (VLMs) [1–6] such as CLIP provide strong zero-shot recognition by aligning images and text in a shared embedding space, but their predictions can still degrade under deployment-time distribution shift. Test-time adaptation (TTA) [7–10] addresses this problem without source data, and most VLM-TTA methods do so in a fully unsupervised way: they adapt from pseudo-labels, confidence scores, entropy, or caches produced by the same model being adapted. This label-free protocol is attractive, but it is also structurally limited on hard shifted samples. When the model is uncertain or miscalibrated, its self-generated signal can reinforce rather than repair an incorrect decision (See Figure 1).



(a) (b) (c)

Figure 1: **Why sparse labels can help VLM test-time adaptation. (a)** Under shift, a conventional VLM-TTA method can still make persistent errors on hard samples. **(b)** ACTOR selectively queries labels for a small number of uncertain samples and stores them as labeled anchors. **(c)** These anchors are then used to improve future predictions on related uncertain samples through retrieval-augmented correction.

[*] indicates the corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

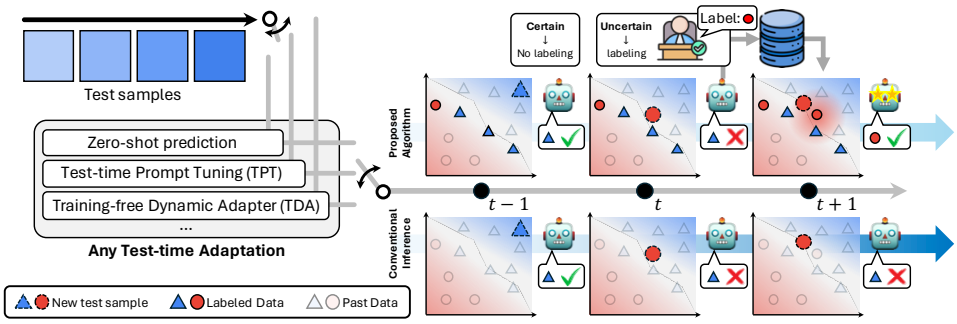


Figure 2: **Active test-time adaptation for vision-language model (Active VLM-TTA).** The model relies on a base VLM-TTA method for confident samples, while selectively querying labels for uncertain ones under a budget. Queried labels cannot revise the current prediction; instead, they are stored in labeled memory and used to improve future uncertain predictions. ACTOR instantiates this idea in a training-free and plug-and-play manner.

This paper studies a complementary regime between two extremes. Fully unsupervised VLM-TTA assumes no external supervision at deployment, while supervised online adaptation assumes labels can be used to update the model. Many practical systems lie between these extremes: a small amount of expert feedback may be available, but labels arrive after a prediction deadline and the deployed VLM must remain frozen. Existing active TTA methods such as simATTA [11] query labels for conventional classifiers and incorporate them through online parameter updates. This design typically operates at the batch/buffer level: predictions are first made for an incoming test batch, informative samples are selected for annotation, and the acquired labels are then used to update model parameters for subsequent batches. In contrast, our setting keeps the VLM frozen and treats queried labels as delayed, reusable evidence: once returned, they enter memory directly rather than being mediated by gradient-based adaptation.

We formalize this regime as *Active Test-time Adaptation for VLMs* (**Active VLM-TTA**). At time t, the model must first predict on the current sample x_t; it may then query the label under a fixed budget, but that label can only be used for samples arriving after time t (Figure 2). Thus, the role of human supervision is deliberately constrained: labels are costed, sparse, and future-only. This formulation is not a replacement for label-free TTA, nor does it assume the convenience of a pre-existing support set. Instead, it asks how a pretrained VLM can convert delayed deployment-time labels into reusable evidence. VLMs are a natural fit for this question because their embedding spaces make semantically related examples retrievable across time, allowing a queried sample to serve as a labeled anchor for later uncertain samples.

To instantiate this setting, we propose **ACT**ive VLM-TTA for **O**nline **R**etrieval (ACTOR), a training-free plug-in module for Active VLM-TTA. For each uncertain sample, ACTOR makes a three-way routing decision: use the base TTA prediction, query a labeler for future use, or correct the prediction from previously queried labels. The correction rule is intentionally conservative. In its main correction rule, a retrieved label is used only when the two nearest labeled anchors agree and are sufficiently close to the test sample in the VLM embedding space. This consistency-and-reliability check is a simple safeguard against the main failure mode of memory-based adaptation: propagating a sparse label beyond the neighborhood where it is trustworthy.

Empirically, ACTOR provides a strong first baseline for Active VLM-TTA. Across 10 cross-domain and 5 OOD benchmarks, it consistently improves CLIP zero-shot prediction and strong VLM-TTA methods on both ResNet and ViT backbones; with ViT-B/16, for example, it improves zero-shot CLIP ($61.94\% \rightarrow 70.67\%$) and TDA ($68.14\% \rightarrow 72.66\%$) on the cross-domain benchmark. Since labels are an explicit resource in our setup, we also compare against supervised prompt learning under the same queried labeled set: training-free ACTOR outperforms Online-CoOp and Replay-CoOp by 2.50 and 2.45 percentage points, respectively, across 15 datasets, while adding less than 1 ms of inference overhead. Our contributions are:

- We introduce **Active VLM-TTA**, a streaming evaluation protocol for pretrained VLMs in which queried labels are budgeted and future-only, separating delayed human feedback from both label-free TTA and supervised online learning.

- We propose ACTOR, a training-free plug-in framework that converts sparse queried labels into reusable memory through three-way routing and a consistency-and-reliability correction rule.
- We provide the first broad empirical study of this regime, including label-matched supervised adaptation, budget, latency, hyperparameter, and VLM-family analyses.

2 Preliminary

2.1 Test-Time Adaptation with VLMs (VLM-TTA)

CLIP [1] consists of an image encoder f_{img} and a text encoder f_{txt}, and performs classification by matching an image to class-specific text prompts. Given an input image x_t and a class set $\mathcal{C}$, the similarity score for class $c \in \mathcal{C}$ is defined as

$$s_c = \cos(f_{\text{img}}(x_t),\ f_{\text{txt}}(t_c))\,, \tag{1}$$

where t_c denotes the text prompt for class c, and $\cos(\cdot,\cdot)$ is the cosine similarity. The corresponding prediction probability is

$$z_t^c = \frac{\exp(s_c/\tau)}{\sum_{i=1}^{|\mathcal{C}|}\exp(s_i/\tau)}, \tag{2}$$

where τ is a temperature parameter.

In the test-time adaptation (TTA) setting, we write the prediction of a VLM-TTA method in a unified form as

$$p(x_t) = \mathcal{T}_\xi(x_t; f_{\text{img}}, f_{\text{txt}}), \tag{3}$$

where $\mathcal{T}_\xi(\cdot)$ denotes a TTA procedure built on the pretrained encoders f_{img} and f_{txt}, together with any method-specific context variables ξ such as learnable prompts (See Appendix A for examples).

A common limitation of existing VLM-TTA methods is that their adaptation signals are derived from the model itself, such as pseudo-labels, confidence scores, or entropy. When the pretrained model is poorly calibrated under distribution shift, these self-generated signals can reinforce incorrect predictions rather than correct them.

2.2 Problem Formulation: Active VLM-TTA

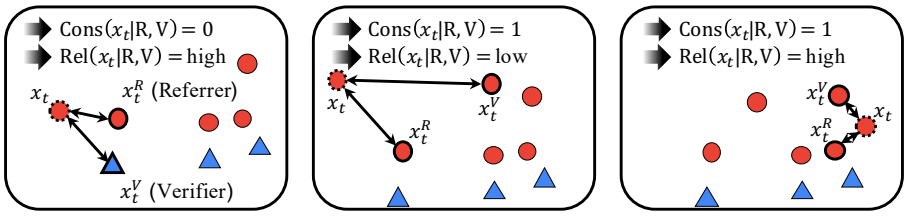
We study a deployment setting in which the fully unsupervised assumption is relaxed, but only in a controlled way. The model may occasionally query ground-truth labels for incoming test samples under a limited budget. Crucially, the queried label is not available for revising the current prediction: the model must predict first, and any queried supervision can be used only for future samples. The label budget is therefore an explicit resource, and the queried labels are delayed evidence rather than free corrections. We formulate this setting as **Active VLM-TTA**, a streaming test-time adaptation protocol for frozen vision-language models with *future-only supervision*.

Definition 2.1 (Active VLM-TTA)**.** For each test sample $x_t \in \mathfrak{D}_{\text{test}}$ arriving at time t, a pretrained VLM must first output a prediction for x_t. It may then optionally query the ground-truth label y_t under a labeling budget $\mathfrak{B}$. If x_t is queried, its label is returned after a latency $\Delta_t \geq 1$. Until then, the queried sample is stored in a pending set rather than in the usable labeled memory. At time t, the labeled memory is updated only with labels whose latency has expired:

$$D_l(t+1) = D_l(t) \cup \{(x_s, y_s) : s + \Delta_s \leq t+1,\ x_s \text{ was queried}\}.$$

Thus, the queried label for x_t cannot affect the prediction on x_t and becomes available only for samples arriving after its return time.

Active VLM-TTA differs from standard online active learning in that the objective is not to improve a continually updated classifier through label-efficient training, but to improve online test-time inference while preserving a frozen VLM. It also differs from prior active test-time adaptation for conventional classifiers [11], which relies on online parameter updates. This makes the setting complementary to label-free VLM-TTA: the question is not whether labels help, but how a small, explicitly costed set of queried labels should be reused under streaming constraints.



(a) (b) (c)

Figure 3: **Conceptual illustration of retrieval-augmented correction. (a)** Low consistency. **(b)** High consistency, but low reliability. **(c)** High consistency, and high reliability. ACTOR operates under the philosophy of reflecting predictions based on proximity to a given test sample, provided the labels between neighboring samples are consistent.

3 ACTOR: ACTive VLM-TTA for Online Retrieval

The key challenges of Active VLM-TTA are (1) deciding which uncertain samples are worth spending the label budget on and (2) reusing the resulting labels without over-propagating them to unrelated future samples. ACTOR is designed as a conservative training-free baseline for this protocol: it changes neither the VLM parameters nor the base TTA method, and intervenes only when delayed labeled evidence is sufficiently local and consistent. Its process is summarized as follows (see Appendix B for the full design).

1. Filter uncertain samples likely to be misclassified. ▷ Section 3.1
2. Among uncertain samples, build $\mathcal{D}_l(t)$ until it reaches size N.
   (a) If consistency and reliability are high, apply retrieval-augmented correction. ▷ Section 3.2
   (b) Otherwise, query the label and add the sample to $\mathcal{D}_l(t)$ for future use once the annotation is returned. In case of budget exhaustion, apply retrieval-augmented correction under related criteria ▷ Section 3.3

3.1 Uncertainty-based Filtering

We first choose the samples that the model is most likely to misclassify. For the test sample x_t, we first compute the base TTA prediction probability $q_t = T_\xi(x_t; f_{\text{img}}, f_{\text{txt}})$. If T_ξ outputs logits, we apply softmax normalization to obtain q_t. We then calculate entropy as

$$\mathcal{H}(q_t) = -\sum_{c \in C} q_{t,c} \log q_{t,c}. \tag{4}$$

Since the entropy is a measure of *how uncertain a model is in its prediction for a given sample*, a sample with a high-entropy value is more likely to be predicted incorrectly by the model [12]. Therefore, we conduct retrieval-augmented correction only for high-entropy samples.

3.2 Retrieval-augmented Correction

We employ a retrieval-augmented correction strategy that leverages $\mathfrak{D}_l(t)$ to correct predictions made by the base TTA method. However, a sparse queried label is useful only within the part of the embedding space where it remains trustworthy; applying nearest-neighbor correction to every uncertain sample can easily introduce new errors. We therefore use two criteria before accepting a memory-based correction: **consistency**, which checks whether the labeled samples nearest to a test sample agree in label space, and **reliability**, which measures how close those labeled samples are to the test sample. Intuitively, if the nearest labeled samples share a label and are close to the current sample, their shared label is more plausible as delayed evidence for the current prediction. For simplicity, we refer to the first and second nearest labeled samples as a *Referrer* $\mathrm{R} = (x_t^R, y_t^R)$ and a *Verifier* $\mathrm{V} = (x_t^V, y_t^V)$, respectively.

Figure 3 depicts three common cases for the Referrer-Verifier interactions. In Figure 3a, the Referrer and Verifier disagree on the label, so no retrieval-augmented correction is applied. In Figure 3b, the labels match, but the Verifier is far from the test sample, making its endorsement of the Referrer's label less trustworthy. In contrast, Figure 3c shows both high consistency (same label) and high

reliability (all close), thus allowing retrieval-augmented correction to proceed confidently. Based on these common cases, we now provide the detailed mathematical formulation as follows.

Consistency. To ensure that the candidate label from the Referrer is reliable, we define the *consistency* metric to check whether the Verifier's label agrees with the Referrer's. Formally, it is defined as

$$\mathrm{Cons}(x_t|\mathrm{R},\mathrm{V}) = \mathbb{1}\{y_t^R = y_t^V\}, \tag{5}$$

where $\mathbb{1}\{\cdot\}$ is the indicator function. A value of 1 indicates full agreement between the Referrer and the Verifier, thus confirming the candidate label. This check is motivated by a simple design principle: agreement between two nearby labeled anchors is less likely to reflect an isolated retrieval error than relying on a single nearest neighbor alone.

Reliability. To quantify the joint proximity of the Referrer and Verifier to a test sample x_t, we define reliability as the average cosine similarity between the test sample and its two nearest labeled neighbors. Formally, this metric is given by

$$\mathrm{Rel}(x_t|\mathrm{R},\mathrm{V}) = \frac{1}{2}\Big(\cos\big(f_{\mathrm{img}}(x_t), f_{\mathrm{img}}(x_t^R)\big) + \cos\big(f_{\mathrm{img}}(x_t), f_{\mathrm{img}}(x_t^V)\big)\Big). \tag{6}$$

The reliability threshold further restricts this agreement to local neighborhoods in the VLM embedding space, reducing the chance that a queried label is propagated too far from the region where it is informative.

Overall correction criterion. The condition for retrieval-augmented correction is formulated as

$$C(x_t \mid R, V) = \mathrm{Cons}(x_t \mid R, V) \cdot \mathrm{Rel}(x_t \mid R, V). \tag{7}$$

This multiplicative formulation ensures that if the Verifier disagrees with the Referrer, i.e., $\mathrm{Cons}(x_t \mid R, V) = 0$, no retrieval-based correction is applied. Taken together, the consistency check suppresses locally noisy retrievals, while the reliability threshold restricts correction to nearby labeled evidence. This motivates our conservative routing rule, which activates retrieval-based correction only when both conditions are satisfied.

3.3 Correction after Label Budget Exhaustion

Once the labeling budget $\mathfrak{B}$ is exhausted, no additional human feedback can be obtained. In this regime, `ACTOR` retains the consistency-and-reliability correction rule in Section 3.2 and applies a relaxed Top-K plausibility check as a fallback when the original criterion is not satisfied. Let q_t be the class-probability vector predicted by the base TTA model for sample x_t, and let y_t^R denote the label suggested by the Referrer. We accept y_t^R only if it belongs to $\mathrm{Top}\text{-}K(q_t)$, the set of classes with the K largest probabilities under q_t:

$$\hat{p}_t = \begin{cases} \mathrm{one_hot}(y_t^R), & \text{if } y_t^R \in \mathrm{Top}\text{-}K(q_t), \\ q_t, & \text{otherwise.} \end{cases} \tag{8}$$

This fallback is a label-space plausibility check after the annotation budget is exhausted, not the full spatial reliability test used before budget exhaustion. It avoids unrestricted nearest-neighbor correction by accepting the retrieved label only when it appears among the K highest-confidence labels of the base TTA prediction q_t. We set $K = 5$ across all datasets, following the standard Top-5 criterion in large-scale image classification [13].

4 Experiment

We provide details of the experimental setup, including datasets, baselines, and implementation details, in Appendix C due to space constraints. This section focuses on CLIP-based main results and ablations; Appendix D includes results on BLIP-2 [4] and SigLIP [14], as well as an ablation over uncertainty metrics. Since Active VLM-TTA explicitly allows a limited amount of delayed human supervision, we report the labeling budget as part of the protocol rather than treating labels as free. Unless otherwise stated, all ablations use CLIP-ResNet-50 with a fixed validation-free setting: $\tau_u = 0.2$, $\tau_r = 0.85$, $N = 5$, $\Delta_t = 1$, and a 10% labeling budget. Other backbones are explicitly indicated where used.

Method	IN	IN-A	IN-V2	IN-R	IN-S	Avg (All5)	Avg(OOD)
ResNet-50							
Zero-shot	56.47	26.96	50.83	55.87	31.50	44.33	41.29
+ACTOR	57.66±0.06	28.11±0.19	50.76±0.07	56.93±0.07	36.94±0.04	46.08	43.18
TPT	60.68±0.01	25.12±0.07	54.33±0.23	59.11±0.04	35.29±0.04	46.91	43.46
+ACTOR	61.20±0.10	26.28±0.05	54.34±0.02	59.65±0.18	39.67±0.18	48.23	44.98
C-TPT	60.42±0.06	23.25±0.06	54.13±0.13	57.72±0.01	34.76±0.01	46.05	42.46
+ACTOR	61.10±0.11	24.76±0.30	53.88±0.03	58.66±0.12	39.62±0.02	47.60	44.23
TDA†	57.78±0.01	27.84±0.01	51.10±0.03	57.01±0.01	33.85±0.11	45.52	42.45
+ACTOR	58.64±0.01	28.73±0.28	51.31±0.09	57.84±0.06	39.00±0.01	47.10	44.22
TDA	61.72±0.04	30.59±0.21	**55.10**±0.26	62.76±0.04	38.05±0.04	49.64	46.62
+ACTOR	**62.06**±0.04	**31.00**±0.15	54.99±0.29	**63.25**±0.12	**42.73**±0.10	**50.80**	**47.99**
ViT-B/16							
Zero-shot	63.52	54.65	59.80	75.36	44.05	59.48	58.47
+ACTOR	65.90±1.03	55.85±0.74	59.69±0.14	76.33±0.02	50.29±0.13	61.61	60.54
TPT	68.76±0.01	52.99±0.06	63.20±0.04	76.97±0.10	47.86±0.02	61.95	60.25
+ACTOR	69.27±0.12	54.89±0.05	61.94±1.49	77.18±0.08	51.30±0.94	62.91	61.32
C-TPT	68.31±0.01	50.71±0.04	62.50±0.04	75.68±0.08	47.46±0.06	60.93	59.08
+ACTOR	69.09±0.01	53.09±0.17	62.14±0.03	76.39±0.13	52.13±0.01	62.57	60.94
TDA†	65.81±0.81	55.42±0.06	60.06±0.13	75.99±0.13	46.01±0.08	60.66	59.37
+ACTOR	67.48±0.03	56.49±0.20	60.09±0.12	76.70±0.08	51.23±0.02	62.40	61.13
TDA	70.08±0.88	60.13±0.33	64.41±0.16	80.52±0.06	50.96±0.06	65.22	64.01
+ACTOR	**70.88**±0.11	**60.73**±0.60	**64.54**±0.10	**80.83**±0.11	**55.01**±0.07	**66.40**	**65.28**

Table 1: **Results on the out-of-distribution benchmark.** All the compared methods are built upon CLIP-ResNet-50 or CLIP-ViT-B/16 baselines. The two average metrics *All 5* and *OOD* are calculated by taking the mean accuracy across all five datasets and four OOD datasets excluding ImageNet (IN). Note that the **bold** type represents the best performance overall. † indicates that the method uses the *unified* prompt template (*i.e.,* a single "a photo of a {class}.") that is shared with TPT and C-TPT, rather than TDA's original per-dataset hand-crafted templates.

4.1 Main Results

Consistent gains under costed delayed labels. ACTOR can be integrated into any VLM-TTA pipeline, yielding consistent improvements under the same future-only labeling protocol. As shown in Table 1, adding ACTOR to a baseline VLM-TTA method improves top-1 accuracy on the OOD benchmark, especially on ImageNet-S (50.96%→55.01%), which raises the Avg(OOD) score (64.01%→65.28%). On the cross-domain benchmark in Table 2, ACTOR also delivers consistent gains, with a particularly large improvement on Flowers102 (71.62%→82.28%). These results should be interpreted as the value of explicitly budgeted, delayed label reuse in Active VLM-TTA, not as a label-free replacement for unsupervised TTA.

Architecture-agnostic efficacy. ACTOR is effective across different backbone architectures. On both the OOD and cross-domain benchmarks, integrating ACTOR consistently improves performance regardless of whether the base model is ResNet-based or Transformer-based. For the OOD benchmark in Table 1, applying ACTOR to TDA with a ResNet-50 backbone improves the average accuracy across all five datasets (49.64%→50.80%), while the ViT-B/16 backbone shows a comparable gain (65.22%→66.40%). Likewise, on the cross-domain benchmark in Table 2, the ResNet-50 model enjoys a substantial accuracy gain (60.78%→66.16%), and the ViT-B/16 model achieves a similar boost (68.14%→72.66%). These results indicate that ACTOR is a robust enhancement for both CNN- and Transformer-based VLM backbones.

Streaming comparison with online CoOp-style prompt tuning under matched supervision. To verify that ACTOR's gains are not merely due to access to queried labels, we compare it with two supervised prompt-tuning baselines under the same Active VLM-TTA protocol. Online-CoOp performs at most one gradient update per queried sample, whereas Replay-CoOp reuses accumulated labeled samples for subsequent updates. Both baselines share the same queried labels, labeling budget, and future-only constraint as ACTOR, but rely on gradient-based prompt optimization rather than training-free retrieval memory. As shown in Table 3, ACTOR improves accuracy across 15 datasets by 2.50 and 2.45 percentage points (pp) over Online-CoOp and Replay-CoOp, respectively. These results indicate that ACTOR's gains cannot be explained solely by label access or repeated label reuse.

Inference cost. For a fair comparison, we run all experiments on Intel Xeon Gold 6326 CPUs with a single NVIDIA RTX 4090. After a warm-up with 100 samples, we measure the time taken to process

Method	Aircraft	Caltech101	Cars	DTD	EuroSAT	Flowers102	Food101	Pets	SUN397	UCF101	Average
ResNet50											
Zero-shot	15.30	80.61	56.27	37.53	25.89	56.44	73.45	80.46	58.16	59.16	54.33
+ACTOR	18.16±0.25	84.61±0.26	57.54±0.42	43.35±1.05	**74.16**±1.12	72.55±0.06	73.73±0.08	81.71±0.11	59.41±0.08	70.15±0.62	63.53
TPT	16.02±0.47	85.36±0.40	57.95±0.16	39.69±0.37	28.04±0.03	60.22±0.40	73.63±0.02	78.39±0.12	59.86±0.12	60.77±0.34	55.99
+ACTOR	18.60±0.25	87.73±0.83	**60.31**±0.52	45.16±0.50	62.16±0.69	70.61±0.40	73.89±0.04	80.86±0.02	61.45±0.13	69.20±0.52	62.99
C-TPT	13.70±0.15	85.82±0.20	56.51±0.06	40.58±0.29	23.13±0.45	61.29±0.21	73.13±0.04	80.04±0.13	59.54±0.04	59.31±0.02	55.30
+ACTOR	16.65±0.51	87.59±0.69	59.35±0.11	45.89±0.96	60.76±0.78	73.02±1.23	73.64±0.03	81.85±0.38	61.28±0.45	69.83±0.25	62.98
TDA†	15.93±0.81	85.13±0.03	57.17±0.08	39.34±0.29	36.71±1.01	59.24±0.11	74.62±0.04	80.69±0.10	59.68±0.11	60.75±0.26	56.92
+ACTOR	17.75±0.74	85.52±0.23	58.48±0.09	44.18±0.79	70.81±1.08	71.64±0.66	74.63±0.19	81.84±0.06	60.40±0.04	68.86±0.98	63.41
TDA	16.79±0.23	89.36±0.43	57.36±0.17	43.91±0.08	41.66±0.61	68.17±0.28	**77.78**±0.11	86.29±0.27	62.48±0.01	64.03±0.15	60.78
+ACTOR	**19.01**±0.02	**89.57**±0.01	58.67±0.33	**48.17**±0.08	71.08±1.29	**77.19**±0.63	77.28±0.20	**86.62**±0.50	**62.81**±0.11	**71.27**±1.12	**66.16**
ViT-B/16											
Zero-shot	22.59	85.76	65.61	40.60	44.25	64.07	82.68	83.81	63.68	66.30	61.94
+ACTOR	27.33±0.71	90.31±0.12	67.77±0.25	47.94±0.42	**80.63**±0.54	80.53±0.20	83.57±0.06	87.10±0.13	65.30±0.03	76.28±0.54	70.67
TPT	23.43±0.08	93.21±1.07	66.52±0.20	46.33±1.10	42.69±0.16	68.25±0.93	83.94±1.18	84.42±0.98	65.19±0.47	67.62±0.41	64.16
+ACTOR	25.56±0.47	93.37±0.08	67.97±0.04	49.53±0.01	73.03±0.33	74.63±0.80	83.54±0.04	85.96±0.16	66.49±0.06	72.67±0.23	69.27
C-TPT	24.09±0.25	92.98±0.69	65.24±0.78	45.01±1.21	42.28±0.10	70.36±0.75	83.22±0.11	87.82±0.72	64.07±0.59	65.58±0.79	64.06
+ACTOR	27.78±0.47	93.27±0.18	66.88±0.37	50.59±0.17	74.72±0.79	77.49±0.43	83.73±0.05	87.70±0.02	66.22±0.05	74.65±0.64	70.30
TDA†	23.48±0.28	88.54±0.08	66.59±0.11	42.97±0.67	54.82±0.77	65.27±0.49	83.82±0.08	85.12±0.27	65.14±0.01	69.63±0.15	64.54
+ACTOR	25.74±0.21	90.41±0.43	67.94±0.04	49.41±0.34	75.40±0.54	79.20±0.49	84.20±0.06	87.37±0.10	66.18±0.01	76.01±0.62	70.18
TDA	25.29±0.30	**94.06**±0.08	66.39±1.72	45.48±0.79	63.95±0.92	71.62±0.23	**86.14**±0.06	89.85±0.13	67.70±0.04	70.94±0.28	68.14
+ACTOR	**27.84**±0.08	**94.06**±0.03	**68.76**±0.13	**51.63**±0.88	79.61±1.40	**82.28**±0.54	86.09±0.06	**90.75**±0.18	**68.20**±0.11	**77.38**±1.01	**72.66**

Table 2: **Results on the cross-domain benchmark.** The *Average* is calculated by taking the mean accuracy across all ten datasets. Note that the **bold** type represents the best performance overall. † indicates that the method uses the *unified* prompt template (*i.e.,* a single "a photo of a {class}.") that is shared with TPT and C-TPT, rather than TDA's original per-dataset hand-crafted templates.

Method	Cross (10)	OOD (4)	OOD (5)	Avg. (15)
CLIP (Zero-shot)	61.94	58.47	59.48	61.12
CLIP + Online-CoOp	67.43	58.61	60.58	65.15
CLIP + Replay-CoOp	69.03	56.24	57.54	65.20
CLIP + ACTOR (Train-free)	**70.67**	**60.54**	**61.61**	**67.65**

Table 3: **Label-matched comparison with online CoOp baselines (ViT-B/16).** Cross and OOD denote average accuracy across 10 and 4/5 datasets, respectively. Online-CoOp performs one update per queried label, while Replay-CoOp reuses accumulated labels. All label-using methods share identical queried labels under future-only supervision (See Appendix D.3).

ACTOR	Zero-shot	TPT	C-TPT	TDA
✗	10.07	509.87	511.33	10.84
✓	10.16	510.22	511.55	11.27
Rel.	+0.9%	+0.07%	+0.04%	+4.0%

Table 4: **Per-sample cost analysis.** ACTOR introduces less than 0.5 ms of additional computational overhead across the evaluated TTA methods.

the next 100 samples, as reported in Table 4. This table measures computational inference overhead; human labeling cost is accounted for separately through the explicit budget and latency analyses. Training-based TTA methods (*e.g.,* TPT, C-TPT) incur an additional cost of roughly 500 ms per sample over train-free methods (*e.g.,* TDA). In contrast, ACTOR introduces only a negligible computational overhead of *below 1 ms* per sample.

4.2 Further Analysis Results

Performance across label budgets. ACTOR's retrieval relies on the labeled memory $\mathfrak{D}_l(t)$, so a larger budget enlarges this memory and enables more reliable retrieval. Figure 4 shows that the gains are not confined to the 10% endpoint used in our main tables: across all three datasets, ACTOR improves throughout the budget sweep, indicating that queried supervision is useful before the final budget is exhausted and that additional labels continue to sharpen retrieval.

Comparison with diverse sample selection methods. We compare ACTOR with standard streaming query-selection strategies—Random, Coreset [15], and Entropy [12]—under the same labeling budget (See Figure 5). To isolate the effect of which samples are queried, all baselines use the same delayed-label protocol and the same retrieval memory once queried labels become available; they differ only in the criterion used to select samples for annotation. Random selects queried samples uniformly, Coreset prioritizes samples that are far from the current labeled memory in the embedding space, and Entropy prioritizes samples with high predictive entropy (See Appendix E).

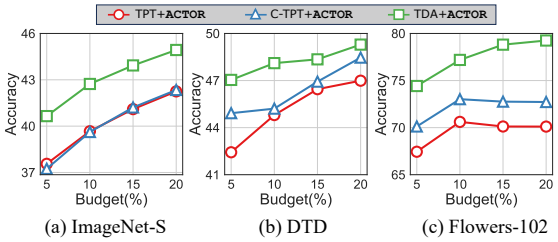


Figure 4: **Performance variation with budget across three datasets.** TDA+ACTOR achieves the best performance under varying budget constraints.

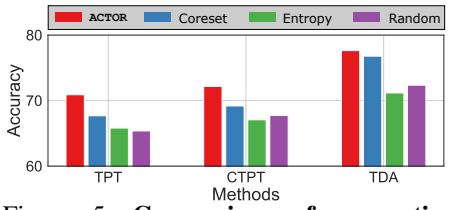


Figure 5: **Comparison of conventional sample selection methods on Flowers-102.** All the active learning methods are adapted to the stream setup (see Appendix E).

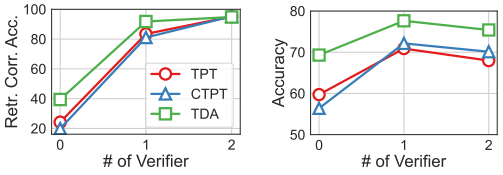


Figure 6: **Performance changes as the number of Verifiers increases.** (a) indicates the accuracy specifically for retrieval-augmented correction, whereas (b) refers to the accuracy calculated over all samples.

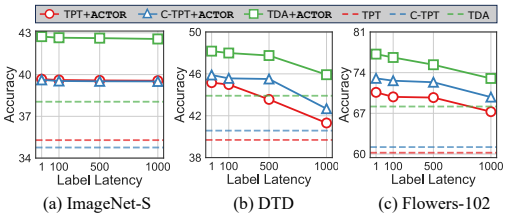


Figure 7: **Robustness to label latency in ACTOR.** *Labeling latency* is defined as the number of test samples that elapse between the moment ACTOR decides to query x_t and the moment the oracle returns y_t.

Impact of a Verifier (x_t^V, y_t^V)**.** Verifiers minimize retrieval-augmented correction errors (Section 3.2). Varying the number of Verifiers in Figure 6, we observe that introducing a Verifier ($0 \rightarrow 1$) significantly boosts correction accuracy (Figure 6a), with further gains as Verifiers increase. However, Figure 6b shows overall accuracy dips when $1 \rightarrow 2$ because Equation 7 becomes overly strict, reducing the number of corrected samples.

Sensitivity of labeling latency of ACTOR. In this analysis, we explicitly vary Δ_t to test delayed annotation. Some delay in human labeling is unavoidable. Figure 7 shows that longer delays gradually reduce the performance gap of ACTOR over TTA, yet ACTOR still matches TTA's performance at worst. We test delays of $\{1, 100, 500, 1000\}$ samples on ImageNet-S (50k), DTD ($\sim$2k), and Flowers-102 ($\sim$2k). On the large ImageNet-S, performance is stable (around 43% for TDA+ACTOR and 40% for C-TPT+ACTOR). On the smaller DTD and Flowers-102, the same delays produce larger drops (*e.g.,* DTD: $46\% \rightarrow 42\%$) , confirming that degradation occurs when latency constitutes a significant fraction of the stream; larger streams mitigate this effect.

ACTOR corrects predictions in local embedding neighborhoods. Figure 8 illustrates how ACTOR corrects misclassified samples using labeled anchors in local neighborhoods of the visual embedding space: the examples with black outlines (selected Referrers/Verifiers) exert noticeable influence in their neighborhoods, where many initially misclassified samples (crosses in Figure 8(left)) are corrected to align with the ground truth (circles in Figure 8(right)).

Conservative reuse reduces harmful corrections. Table 5 isolates the effect of different label reuse rules by fixing the test stream, queried samples, returned labels, labeling budget, and annotation latency across all methods, while varying only the retrieval-based correction rule. Using TDA with CLIP-ViT-B/16, we evaluate average accuracy across Flowers102, DTD, Caltech101, IN-S, and IN-V2. The full consistency-and-reliability rule achieves 69.50% accuracy, improving upon base TDA (65.31%) by 4.19 percentage points (pp) and the strongest plain kNN baseline, 5-NN majority voting (67.22%), by 2.28 pp. It also reduces the harmful-correction rate, defined as the fraction of applied corrections that turn a correct base prediction into an incorrect one (13.68% $\rightarrow$ 5.28%), an 8.40 pp reduction relative to 5-NN majority voting. Consistency-only reuse improves accuracy to 68.78% but retains a relatively high harm rate of 11.99%, while reliability-only reuse yields limited improvement (67.30% accuracy and 13.56% harm).

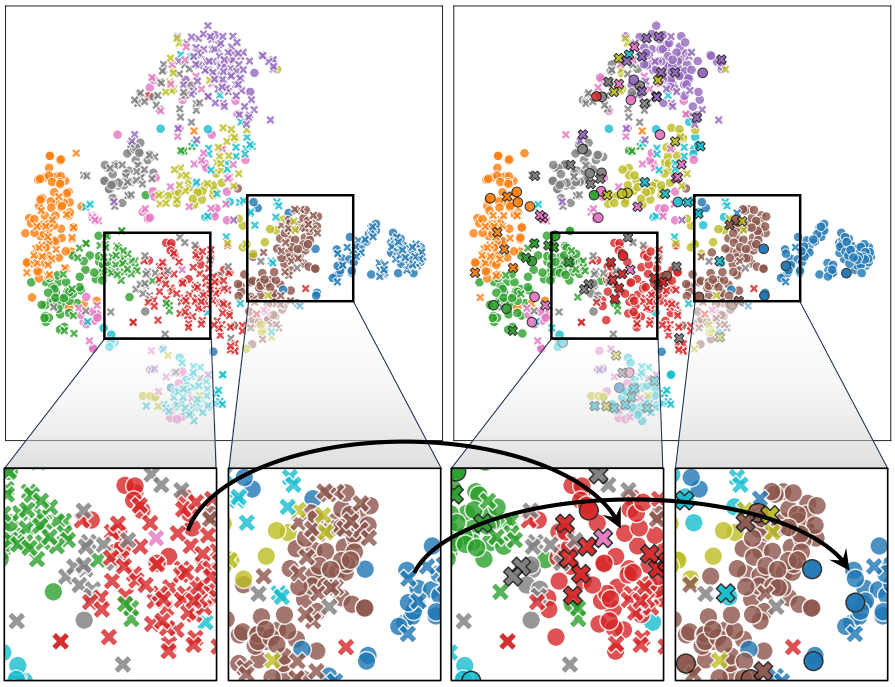


Figure 8: **Visualization of retrieval-augmented prediction correction on EuroSAT.** Circles (○) indicate correctly classified samples, while crosses (×) indicate misclassified samples. Black outlines denote samples queried and labeled by ACTOR.

Hyperparameter sensitivity. ACTOR introduces two hyperparameters, τ_u and τ_r, that decide whether a sample should undergo retrieval-augmented correction. First, τ_u dictates how many samples rely on TTA predictions. Specifically, lowering τ_u makes more samples appear *uncertain*, diverting them away from direct TTA-based decisions, whereas raising τ_u increases the likelihood of accepting TTA outputs. Second, τ_r governs which of these uncertain samples undergo retrieval-augmented correction. A lower τ_r includes more samples in retrieval-augmented correction, while a higher τ_r imposes stricter conditions, causing more samples to label.

To assess potential sensitivity of ACTOR, we tested different values of τ_u and τ_r, as reported in Figure 9. Results across three datasets show that each one behaves differently due to varying levels of difficulty in each dataset. Nonetheless, on every dataset and for every TTA method, incorporating ACTOR consistently surpasses the baseline (*i.e., without* ACTOR), regardless of the specific hyperparameter settings. This observation confirms the robustness for a reasonable range of the thresholds.

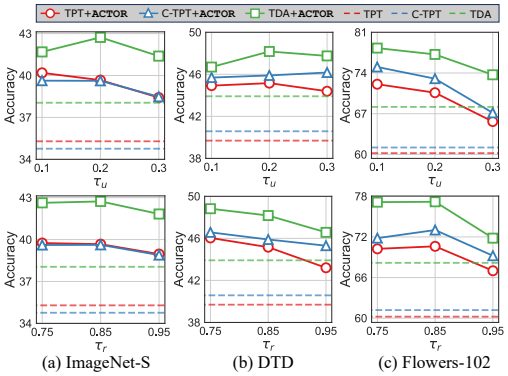
5 Related Work

Active learning. Active learning aims to minimize labeling cost by selecting informative samples [16, 17], using uncertainty-based criteria [18, 19, 12], diversity-based criteria [15, 20], or hybrids such as BADGE [21]. PCB [22] specializes active learning to VLMs, while online active learning [23, 24] often still assumes continual retraining; in both cases, queried labels mainly serve to update a model or improve a final pool-trained learner. Active test-time adaptation is closer in spirit because it allows limited human feedback during deployment, but existing methods remain update-centric: for example, SimATTA [11] uses queried labels for batch-level online parameter updates, so labels affect future predictions only after annotation and a subsequent update step. In contrast, ACTOR keeps the VLM frozen and stores each returned label in memory as soon as it arrives, enabling retrieval-augmented correction for subsequent samples while explicitly modeling human label delay.

Vision-language models. Large-scale image–text pretraining has produced a family of VLMs including CLIP [1], ALIGN [25], BLIP/BLIP-2 [3, 4], LiT [26], SigLIP [14], and FILIP [27], which align images and text in a shared embedding space and support zero-shot classification. Beyond zero-shot use, prior work has adapted VLMs with offline labeled support through prompt learning

Reuse rule	Avg. (↑)	Harm. (↓)
Base TDA	65.31±0.26	–
Hard 1-NN	67.00±0.38	14.98±0.96
5-NN majority	67.22±0.28	13.68±0.04
Weighted 5-NN	67.21±0.28	13.69±0.06
Consistency only	68.78±0.23	11.99±0.98
Reliability only	67.30±0.34	13.56±0.67
ACTOR (both)	**69.50**±0.32	**5.28**±0.26

Table 5: **Effect of correction reuse rules under a shared memory.** Average accuracy (Avg.) and harmful-correction rate (Harm.) across five datasets using TDA with CLIP-ViT-B/16. All methods share the same queried samples and returned labels under an identical labeling budget and annotation latency, differing only in their label reuse rules. Harm. measures the fraction of applied corrections that change a correct base prediction into an incorrect one.



(a) ImageNet-S (b) DTD (c) Flowers-102

Figure 9: **Hyperparameter sensitivity.** We vary the uncertainty threshold τ_u with $\tau_r = 0.85$ (upper) and the retrieval-augmented correction threshold τ_r with $\tau_u = 0.2$ (lower).

methods such as CoOp [28], CoCoOp [29], and MaPLe [30], as well as feature/cache-based adapters such as CLIP-Adapter [31] and Tip-Adapter [32]. These methods assume a labeled support set is already available before deployment and then optimize how to fit or reuse it. ACTOR exploits the same shared-space property for retrieval, but differs in that its labeled memory is not given in advance and cannot be used to revise the queried sample itself: it is built on the fly from delayed deployment-time queries under a streaming budget.

Test-time adaptation for VLMs. To bridge the pretraining–deployment gap, TPT [8] tunes prompts per test sample via augmentation-based entropy minimization, DiffTPT [7] augments the input with diffusion-generated views, and C-TPT [9] calibrates the prediction uncertainty of TPT. RLCF [33] observes that entropy-based losses can get stuck on wrong predictions and proposes a CLIP-score-based objective. Moving away from per-sample optimization, TDA [10] introduces training-free TTA via Tip-Adapter [32]-style caching, reducing inference cost while still adapting online. These methods are label-free and are therefore complementary to our setting. More broadly, retrieval-based adaptation has also been studied outside our setting [34, 35], but these methods do not model label acquisition from the deployed stream under an explicit query budget.

6 Conclusion

We introduced **Active VLM-TTA**, a streaming protocol between fully unsupervised TTA and supervised online learning. In this setting, a frozen VLM may spend an explicit label budget, but queried labels are delayed and can only improve future predictions. We instantiated this protocol with ACTOR, a training-free plug-in framework that reuses sparse queried labels through retrieval-augmented correction. Across OOD and cross-domain benchmarks, ACTOR consistently improves strong VLM-TTA baselines on both ResNet and ViT backbones with minimal computational overhead, and a label-matched CoOp comparison shows that the gains are not merely due to label access.

Limitations

Active VLM-TTA is intended to complement, rather than replace, label-free TTA, because it explicitly models deployment scenarios where limited delayed human feedback is available. ACTOR is designed for settings in which queried annotations are reliable and the labeled memory can provide local evidence for future samples. Although our latency and budget analyses show that the method remains robust across the tested ranges, its effectiveness may decrease when the returned labels are noisy, the stream provides highly uneven class coverage, or the annotation budget is exhausted and correction must rely more heavily on Top-K plausibility. Extending ACTOR with noise-aware memory filtering, class-balanced querying, distance-aware fallback, and stream-order-aware memory management is an important direction for future work.

Acknowledgment

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2020-II200862, DB4DL: High-Usability and Performance In-Memory Distributed DBMS for Deep Learning, 40%, No. RS-2022-II220157, Robust, Fair, Extensible Data-Centric Continual Learning, 40%, No. RS-2026-25614738, AI Star Fellowship Support Program, 5%, and No. RS-2025-25464461, AI's Vision of Harmony: A Fair and Transparent Multimodal Agentic Platform for Conflict Mediation, 5%) and the InnoCORE program of the Ministry of Science and ICT (AI Meta-Scientist, N10260110, 10%).

References

[1] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *Proceedings of International Conference on Machine Learning*, pages 8748–8763, 2021.

[2] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *ArXiv Preprint*, 2023.

[3] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *Proceedings of International Conference on Machine Learning*, pages 12888–12900. PMLR, 2022.

[4] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *Proceedings of International Conference on Machine Learning*, pages 19730–19742. PMLR, 2023.

[5] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Proceedings of Advances in Neural Information Processing Systems*, volume 36, 2023.

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

[7] Chun-Mei Feng, Kai Yu, Yong Liu, Salman Khan, and Wangmeng Zuo. Diverse Data Augmentation with Diffusions for Effective Test-time Prompt Tuning. In *Proceedings of IEEE/CVF International Conference on Computer Vision*, pages 2704–2714, 2023.

[8] Manli Shu, Weili Nie, De-An Huang, Zhiding Yu, Tom Goldstein, Anima Anandkumar, and Chaowei Xiao. Test-time Prompt Tuning for Zero-shot Generalization in Vision-Language Models. In *Proceedings of Advances in Neural Information Processing Systems*, pages 14274–14289, 2022.

[9] Hee Suk Yoon, Eunseop Yoon, Joshua Tian Jin Tee, Mark Hasegawa-Johnson, Yingzhen Li, and Chang D Yoo. C-TPT: Calibrated Test-Time Prompt Tuning for Vision-Language Models via Text Feature Dispersion. In *Proceedings of International Conference on Learning Representations*, 2024.

[10] Adilbek Karmanov, Dayan Guan, Shijian Lu, Abdulmotaleb El Saddik, and Eric Xing. Efficient test-time adaptation of vision-language models. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14162–14171, 2024.

[11] Shurui Gui, Xiner Li, and Shuiwang Ji. Active test-time adaptation: Theoretical analyses and an algorithm. In *Proceedings of International Conference on Learning Representations*, 2024.

[12] Alex Holub, Pietro Perona, and Michael C Burl. Entropy-based active learning for object recognition. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshop*, pages 1–8, 2008.

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[14] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of IEEE/CVF International Conference on Computer Vision*, pages 11975–11986, 2023.

[15] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *Proceedings of International Conference on Learning Representations*, 2018.

[16] Burr Settles. Active learning literature survey. Technical report, Department of Computer Sciences, University of Wisconsin-Madison, 2009.

[17] Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B Gupta, Xiaojiang Chen, and Xin Wang. A survey of deep active learning. *ACM Computing Surveys (CSUR)*, 54(9):1–40, 2021.

[18] David D Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of International Conference on Machine Learning*, pages 148–156, 1994.

[19] Dan Roth and Kevin Small. Margin-based active learning for structured output spaces. In *Proceedings of European Conference on Machine Learning*, pages 413–424, 2006.

[20] Amin Parvaneh, Ehsan Abbasnejad, Damien Teney, Gholamreza Reza Haffari, Anton Van Den Hengel, and Javen Qinfeng Shi. Active learning by feature mixing. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12237–12246, 2022.

[21] Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. *ArXiv Preprint*, 2019.

[22] Jihwan Bang, Sumyeong Ahn, and Jae-Gil Lee. Active prompt learning in vision language models. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27004–27014, 2024.

[23] Jiongming Qin, Cong Wang, Qinhong Zou, Yubin Sun, and Bin Chen. Active learning with extreme learning machine for online imbalanced multiclass classification. *Knowledge-Based Systems*, 231:107385, 2021.

[24] Giulia DeSalvo, Claudio Gentile, and Tobias Sommer Thune. Online active learning with surrogate loss functions. In *Proceedings of Advances in Neural Information Processing Systems*, volume 34, pages 22877–22889, 2021.

[25] 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 *Proceedings of International Conference on Machine Learning*, pages 4904–4916, 2021.

[26] Xiaohua Zhai, Xiao Wang, Basil Mustafa, Andreas Steiner, Daniel Keysers, Alexander Kolesnikov, and Lucas Beyer. Lit: Zero-shot transfer with locked-image text tuning. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18123–18133, 2022.

[27] 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. *ArXiv Preprint*, 2021.

[28] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022.

[29] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16816–16825, 2022.

[30] Muhammad Uzair Khattak, Hanoona Rasheed, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Maple: Multi-modal prompt learning. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19113–19122, 2023.

[31] 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. *International Journal of Computer Vision*, pages 1–15, 2023.

[32] Renrui Zhang, Wei Zhang, Rongyao Fang, Peng Gao, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-adapter: Training-free adaption of clip for few-shot classification. In *Proceedings of European Conference on Computer Vision*, pages 493–510. Springer, 2022.

[33] Shuai Zhao, Xiaohan Wang, Linchao Zhu, and Yi Yang. Test-time Adaptation with CLIP Reward for Zero-shot Generalization in Vision-Language Models. In *Proceedings of International Conference on Learning Representations*, 2024.

[34] Luca Zancato, Alessandro Achille, Tian Yu Liu, Matthew Trager, Pramuditha Perera, and Stefano Soatto. Train/test-time adaptation with retrieval. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15911–15921, 2023.

[35] Youngjun Lee, Doyoung Kim, Junhyeok Kang, Jihwan Bang, Hwanjun Song, and Jae-Gil Lee. Ra-tta: Retrieval-augmented test-time adaptation for vision-language models. In *Proceedings of International Conference on Learning Representations*, 2025.

[36] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[37] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *Proceedings of International Conference on Machine Learning*, pages 5389–5400. PMLR, 2019.

[38] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of IEEE/CVF International Conference on Computer Vision*, pages 8340–8349, 2021.

[39] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. *Proceedings of Advances in Neural Information Processing Systems*, 32, 2019.

[40] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *ArXiv Preprint*, 2013.

[41] 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 *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshop*, pages 178–178, 2004.

[42] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of IEEE/CVF International Conference on Computer Vision Workshop*, pages 554–561, 2013.

[43] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3606–3613, 2014.

[44] 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*, 12(7):2217–2226, 2019.

[45] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Proceedings of Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729, 2008.

[46] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *Proceedings of European Conference on Computer Vision*, pages 446–461, 2014.

[47] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3498–3505, 2012.

[48] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3485–3492, 2010.

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

[50] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. *Proceedings of Advances in Neural Information Processing Systems*, 33:21464–21475, 2020.

A Conventional TTA

Test-time adaptation (TTA) using vision-language models (VLMs) can be classified into two types: prompt tuning-based approaches (*e.g.,* TPT [8] and C-TPT [9]) and training-free approaches (*e.g.,* TDA [10]). Depending on the approach, we can formulate the TTA operator $\mathcal{T}_\xi(\cdot)$ as follows. Note that C-TPT is omitted here due to its conceptual similarity to TPT.

A.1 Test-time Prompt Tuning (TPT)

TPT is achieved by fine-tuning a learnable prompt p using a single test sample x_t. Given multiple augmented views of x_t, denoted by $A_i(x_t)$ for $i = \{1, \dots, M\}$, the goal is to minimize the entropy (as defined in Equation 4) of the model's predictions across these augmentations.

Objective Function. The probability distribution over classes generated by the CLIP model with prompt p on the i-th augmented view can be formulated as

$$p^{(i)}(x_t) = p(A_i(x_t)) = \frac{\exp(\mathbf{s}^{(i)}/\tau_{\text{temp}})}{\sum_{k=1}^{\mathcal{C}} \exp(s_k^{(i)}/\tau_{\text{temp}})},$$

where $\mathbf{s}^{(i)} = \{s_c^{(i)} | c \in \{1, \dots, \mathcal{C}\}\}$, τ_{temp} denotes the temperature, and the similarity $s_c^{(i)}$ is defined as

$$s_c^{(i)} = \cos\left(f_{\text{img}}(A_i(x_t)), f_{\text{txt}}([p; t_c])\right),$$

where t_c is the text prompt of class c. Note that it is similarly calculated by Equation 1, but the difference is that the text prompt concatenates learnable prompt p and text prompt t_c for class c, and the input image is augmented when calculating the similarity score. Then, the averaged prediction $\tilde{p}_p(x_t)$ (with confidence selection) is defined as

$$\tilde{p}_p(x_t) = \frac{1}{\rho M} \sum_{i=1}^{M} \mathbf{1}\left[H(p^{(i)}(x_t)) \leq \tau_{\text{ent}}\right] p^{(i)}(x_t),$$

where $\mathcal{H}(\cdot)$ denotes the entropy function, τ_{ent} is the entropy threshold, and ρ is the proportion of augmented views selected.

The optimization problem is then given by

$$p^* = \arg\min_p \mathcal{L}_{\text{TPT}}(p; x_t),$$

with the loss defined as the marginal entropy,

$$\mathcal{L}_{\text{TPT}}(p; x_t) = -\sum_{y \in \mathcal{Y}} \tilde{p}_{p,y}(x_t) \log \tilde{p}_{p,y}(x_t).$$

After optimization, the test-time adapted prediction function is defined as

$$\mathcal{T}_\xi(x_t) = \mathcal{T}_{p^*}(x_t) \triangleq p_{p^*}(x_t).$$

A.2 Training-free Dynamic Adapter (TDA)

In contrast to prompt tuning-based approaches, TDA leverages a non-parametric dynamic cache to adapt predictions without backpropagation. Here, ξ comprises the components of the dynamic cache,

$$\xi = \{Q_p,\, \hat{L}_p,\, Q_n,\, \hat{L}_n\},$$

where $(Q_p, \hat{L}_p)$ denotes the keys and corresponding pseudo labels in the *positive cache* (collected from high-confidence predictions), and $(Q_n, \hat{L}_n)$ denotes the keys and corresponding negative pseudo labels in the *negative cache* (collected from low-confidence predictions).

Then, the final prediction is computed as

$$\mathcal{T}_\xi(x_t) = f_{\text{img}}(x_t) W_T^c + \mathcal{A}\big(f_{\text{img}}(x_t) Q_p^T\big)\, \hat{L}_p - \mathcal{A}\big(f_{\text{img}}(x_t) Q_n^T\big)\, \hat{L}_n,$$

where $W_T^c = \left[f_{\text{txt}}(t_c) \mid c \in \mathcal{C}\right]$ represents the text embedding matrix computed from the class names, and $\mathcal{A}(\cdot)$ is an adapter function (*e.g.,* an exponential weighting function as used in Tip-Adapter [32]).

B Algorithm Details

Algorithm 1 outlines our proposed method, ACTOR. For each test sample x_t, our method first assesses its prediction uncertainty. If the sample is certain, ACTOR directly returns the base TTA prediction. For uncertain samples, it employs a hierarchical strategy to manage its explicit labeling budget. The process begins with a warm-up phase, during which uncertain samples are queried until at least N returned labels are available in the labeled memory $\mathfrak{D}_l$, subject to the labeling budget $\mathfrak{B}$. Pending annotations are counted toward the budget but are incorporated into memory only after the specified latency. Importantly, each prediction is emitted before its annotation is requested, ensuring that queried labels cannot revise the predictions of their corresponding samples.

Once sufficient labeled samples are available, ACTOR's core **retrieval-augmented correction** mechanism is activated. It first attempts a budget-free correction by retrieving labeled anchors from $\mathfrak{D}_l$ and checking whether their labels are consistent and their features are sufficiently close to the current sample. If the correction criterion is not satisfied and the remaining labeling budget permits, the algorithm queries the oracle for future use. After budget exhaustion, it instead applies the Top-K plausibility check as a fallback when the original correction criterion is not satisfied. This strategy enables conservative label reuse while preserving the future-only constraint of Active VLM-TTA.

Algorithm 1: ACTOR

Input: Labeled memory $\mathfrak{D}_l(t)$, Pending annotations $\mathcal{P}(t)$, Number of issued queries $Q(t)$, Uncertainty threshold τ_u, Latency Δ, Correction threshold τ_r, Labeling budget $\mathfrak{B}$, Minimum labeled samples N, TTA method $\mathcal{T}_\xi(\cdot)$, Human annotator $\texttt{Human}(\cdot)$, Test sample x_t

Incorporate returned annotations from $\mathcal{P}(t)$ into $\mathfrak{D}_l(t)$
$q_t \leftarrow \mathcal{T}_\xi(x_t; f_{\text{img}}, f_{\text{txt}})$
Normalize q_t with softmax if needed
$u \leftarrow \mathcal{H}(q_t)$ from Equation 4
$\hat{p}_t \leftarrow q_t$
query $\leftarrow$ False
# **Uncertain sample**: Apply ACTOR
if $u \geq \tau_u$ **then**
  # **Warm-up**: Construct labeled memory
  if $|\mathfrak{D}_l(t)| < N$ **then**
    if $Q(t) < \mathfrak{B}$ **then**
      query $\leftarrow$ True
    end
  else
    Retrieve $\mathrm{R}, \mathrm{V} \in \mathfrak{D}_l(t)$
    Compute $\mathcal{C}(x_t|\mathrm{R}, \mathrm{V})$ from Equation 7
    # Retrieval-augmented correction ▷ Section 3.2
    if $\mathcal{C}(x_t|\mathrm{R}, \mathrm{V}) > \tau_r$ **then**
      $\hat{p}_t \leftarrow \texttt{one_hot}(y_t^R)$
    else if $Q(t) < \mathfrak{B}$ **then**
      query $\leftarrow$ True
    else
      # Correction after budget exhaustion ▷ Section 3.3
      Compute $\hat{p}_t$ using Equation 8
    end
  end
end
Emit prediction before requesting annotation
$\texttt{Emit}(\hat{p}_t)$
if query **then**
  Request annotation $\texttt{Human}(x_t)$
  Add the request to $\mathfrak{P}(t)$ with return time $t + \Delta$
  $Q(t+1) \leftarrow Q(t) + 1$
end

C Experimental Setup

C.1 Datasets

Our experiments were conducted on two benchmarks: the out-of-distribution (OOD) benchmark and the cross-domain benchmark, both used in previous research on adapting VLMs during test time. The OOD benchmark evaluates the robustness of our approach on ImageNet [13] and four ImageNet-derived distribution-shift datasets: ImageNet-A [36], ImageNet-V2 [37], ImageNet-R [38], and ImageNet-S [39]. In contrast, the cross-domain benchmark examines the model's adaptability across different domains by evaluating it on ten diverse image classification datasets, each representing a distinct class space. These datasets include Aircraft [40], Caltech101 [41], Stanford Cars [42], DTD [43], EuroSAT [44], Flowers102 [45], Food101 [46], Oxford Pets [47], SUN397 [48], and UCF101 [49]. Using these two benchmarks, we provide a comprehensive evaluation of the model's generalization capability and adaptability during test time. We summarize the statistics of datasets in Table 6, and the details are provided below.

C.1.1 OOD Benchmarks

ImageNet [13] contains 14,197,122 annotated images according to the WordNet hierarchy. This dataset has been used in the ImageNet large scale visual recognition challenge (ILSVRC), a benchmark in image classification and object detection, since 2010.

ImageNet-A [36] is a subset of 7,500 visually similar but naturally perturbed ImageNet images of 200 classes.

ImageNet-V2 [37] consists of 10,000 images and 1,000 ImageNet classes, and was collected by applying an updated natural data collection pipeline to the original ImageNet dataset.

ImageNet-R [38] includes 30,000 images belonging to 200 categories of the ImageNet dataset, but with diverse artistic styles.

ImageNet-S [39] consists of 50,889 sketches of 1,000 class objects from the ImageNet dataset, and represents a domain shift from natural images to sketches.

C.1.2 Cross Domain Benchmarks

FGVC-Aircraft [40] comprises a total of 10,000 images depicting various aircraft, organized into 100 distinct classes. Each class corresponds to a specific aircraft model variant and contains 100 images. The dataset is divided into training, validation, and test sets, with the test set comprising 3,333 images across all 100 classes. The class labels specify the aircraft manufacturer, model, and variant, such as Boeing 737-76J.

Caltech101 [41] originally comprises 9,144 images spanning 101 object categories and an additional background category. These categories encompass a wide range of objects, including animals, vehicles, and household items. Following previous evaluation protocols, we exclude the `BACKGROUND` and `Faces easy` categories, resulting in 100 classes. The test set consists of 2,465 images distributed unevenly across these classes. Notably, Caltech101 exhibits substantial class imbalance due to the varying numbers of images across its categories.

Stanford Cars [42] consists of a collection of 16,185 images categorized into 196 different classes, with each class typically representing specific car make, model, and year, *e.g.,* 2012 Tesla Model S.

DTD [43], abbreviated from Describable Texture Dataset, is designed for texture classification task. This dataset consists of 47 distinct classes, including categories like fabrics and natural materials. In total, DTD comprises 5,640 samples. Notably, when examining the performance reported in the CLIP [1], it becomes evident that DTD poses a challenging problem for pre-trained CLIP models, as textures are not typical, easily recognizable objects.

EuroSAT [44] comprises 10 distinct classes that represent various land use and land cover categories. In total, this dataset includes 27,000 satellite images, with 2,700 images allocated to each of the 10 classes. Notably, each class contains an equal number of images, ensuring a balanced distribution within the dataset.

Dataset	# of Classes	# of Test Instances
ImageNet	1000	50,000
ImageNet-A	200	7,500
ImageNetV2	1000	10,000
ImageNet-R	200	30,000
ImageNet-S	1000	50,889
FGVC Aircraft	100	3,333
Caltech101	100	2,465
Stanford Cars	196	8,041
DTD	47	1,692
EuroSAT	10	8,100
Flowers102	102	2,463
Food101	101	30,300
Oxford Pets	37	3,669
SUN397	397	19,850
UCF101	101	3,783

Table 6: **Profiles of the datasets used for the experiments.**

Flowers102 [45] consists of 102 different categories of flowers, each representing a distinct flower species such as roses, sunflowers, and daisies. There are 8,189 image-label pairs in total. Some categories have more images than the others, which means that it is imbalanced as typical real-world datasets; each category contains at least 40 and at most 258 samples.

Food-101 [46] consists of 101 food categories with 750 training and 250 test images per category, summing up to 101K images. The labels for the test images have been manually cleaned, while the training set contains some noise.

Oxford Pets [47] consists of 37 different pet categories, including various dogs and cats. This dataset contains 7,400 samples. In particular, it has 4,978 dog images and 2,371 cat images. We use only class labels even though the dataset has segmentation, *i.e.,* both RoI and class.

SUN397 [48] is the Scene UNderstanding (SUN) database that contains 899 categories and 130,519 images. There are 397 well-sampled categories to evaluate numerous state-of-the-art algorithms for scene recognition.

UCF101 [49] is an extension of UCF50 and consists of 13,320 video clips, which are classified into 101 categories. These 101 categories can be classified into 5 types (body motion, human-human interactions, human-object interactions, playing musical instruments, and sports). The total length of these video clips is over 27 hours. All the videos are collected from YouTube and have a fixed frame rate of 25 FPS with the resolution of 320×240. In this work, the middle frame of each video is fed to the image encoder.

C.2 Baselines

We compare ACTOR against representative strong TTA methods for VLMs, which can be categorized into prompt-tuning-based (*e.g.,* TPT, C-TPT) and training-free approaches (*e.g.,* TDA).

Prompt tuning-based TTA. We select TPT [8] and its concurrent extension C-TPT [9] as representatives of this category. These methods adapt a set of learnable prompt vectors for each individual test sample. The optimization is guided by a self-supervised objective that minimizes the entropy of the model's predictions across multiple augmented views of that sample, thereby fine-tuning the prompt without requiring true labels.

Training-free TTA. As a training-free counterpart, we use TDA [10]. In contrast to prompt tuning, TDA is a non-parametric method that requires no backpropagation at test time. It maintains a dynamic cache that stores features from past predictions, separating them into positive and negative caches using confidence scores. These caches are then used to adjust the prediction scores of the current test sample, effectively adapting the model on the fly.

C.3 Implementation Details

All models in our main experiments are built upon the pre-trained CLIP model, which consists of an image encoder and a text encoder. The image encoder can be either a ResNet or a Vision Transformer

Model	Aircraft	Caltech101	Cars	DTD	EuroSAT	IN	IN-A
CLIP	22.59	85.76	65.61	40.60	44.25	63.52	54.65
+ ACTOR	**27.33**	**90.31**	**67.77**	**47.94**	**80.63**	**65.90**	**55.85**
BLIP-2	12.75	92.58	78.54	53.49	48.88	60.23	63.35
+ ACTOR	**22.20**	**92.60**	**79.92**	**53.66**	**56.58**	**61.83**	**63.89**
SigLIP	36.66	95.86	88.47	61.70	33.88	75.14	43.49
+ ACTOR	**39.48**	**96.06**	**88.50**	**63.30**	**34.30**	**75.22**	**44.24**

Table 7: **Generalization of ACTOR across different VLM backbones.** We report top-1 classification accuracy (%) on seven datasets using CLIP, BLIP-2, and SigLIP, with and without ACTOR. Bold indicates the better result for each backbone.

(ViT), while the text encoder is a Transformer. TTA is performed in a single-image setting with a batch size of 1. Specifically, the thresholds of uncertainty (See Equation 4) and retrieval-augmented correction (See Equation 7) are set to 0.2 and 0.85, respectively, the initial number of labeled examples N is set to 5, and the explicit label budget $\mathfrak{B}$ is set to 10% of the test stream. Queried labels are inserted into memory only for future samples. For evaluation, we use the *top-1 accuracy* (%) as the standard classification metric. All experiments were repeated three times with different test sample orders generated using three fixed random seeds. We report the mean accuracy and standard deviation across the three runs. All experiments were conducted on a single NVIDIA RTX 4090.

Hyperparameter protocol (validation-free). A common concern in the test-time adaptation literature is that strong results may arise from per-dataset hyperparameter tuning, which is not realistic under a streaming deployment assumption. We take the position that, by definition, Active VLM-TTA should not assume access to a per-task labeled validation split: the test stream itself is the only signal available at deployment, and the labeling budget is explicitly accounted for as a queried cost. We therefore fix a single set of hyperparameters $(\tau_u, \tau_r, N, \mathfrak{B}/|\mathfrak{D}_{\text{test}}|) = (0.2, 0.85, 5, 10\%)$ *before* seeing any benchmark-specific performance and use exactly this configuration on all 15 datasets (5 OOD + 10 cross-domain), both ResNet-50 and ViT-B/16 backbones, and all baselines (Zero-shot, TPT, C-TPT, TDA). The sensitivity study in Figure 9 then shows that, for any choice within a reasonable range around these defaults, ACTOR still consistently outperforms the underlying TTA baseline, which suggests that the reported gains are not an artifact of a narrow, carefully tuned operating point.

D Further Analyses

D.1 Various VLMs

While recent vision-language models (VLMs) such as BLIP-2 [4] and SigLIP [14] have emerged, the majority of prior TTA research has been conducted on CLIP. Therefore, for baseline consistency and fair comparison, we adopted CLIP in our main experiments (as presented in Table 1 and Table 2). To further evaluate the generalizability of ACTOR, we extend our experiments to BLIP-2 and SigLIP, using visual features extracted from their respective pretrained image encoders for retrieval-based correction without additional training. As shown in Table 7, ACTOR consistently improves performance across these architectures, demonstrating its applicability to different VLM families.

D.2 Ablation on Uncertainty Metric

To validate our choice of *entropy* as the uncertainty metric, we performed an ablation by replacing it with two other common metrics: Max Softmax probability and Energy Score [50]. In Table 8, the results indicate that entropy is the most effective choice, achieving the highest accuracy on six out of seven benchmark datasets. The gain is particularly pronounced on DTD, where entropy surpasses the next best metric by a large margin. While Max Softmax yields a stronger result on EuroSAT, the consistently top-tier performance of entropy across a diverse range of benchmarks indicates its general robustness.

Metric	Aircraft	Caltech101	Cars	DTD	EuroSAT	IN	IN-A
Max Softmax	27.69	85.76	59.25	48.05	**82.14**	64.54	55.89
Energy	25.74	94.04	67.57	44.74	61.07	70.70	60.42
Entropy (Ours)	**27.84**	**94.06**	**68.76**	**51.63**	79.61	**70.88**	**60.73**

Table 8: **Performance with various uncertainty metrics.** Since entropy, maximum softmax probability, and energy score have different numerical scales, we use metric-specific uncertainty thresholds chosen to match the same query budget, while keeping $\tau_r = 0.85$, $N = 5$, and the 10% label budget fixed.

	Cross-domain										Out-of-distribution					
Method	Aircraft	Caltech	Cars	DTD	EuroSAT	Flowers	Food	Pets	SUN	UCF	IN	IN-A	IN-V2	IN-R	IN-S	**Avg.**
Zero-shot CLIP	22.59	85.76	65.61	40.60	44.25	64.07	82.68	83.81	63.68	66.30	63.52	54.65	59.80	75.36	44.05	61.12
Online-CoOp	26.22	89.17	66.68	42.61	72.43	70.77	83.22	86.29	**67.74**	69.13	**68.50**	52.47	59.73	76.09	46.13	65.15
Replay-CoOp (online)	**27.63**	89.82	**68.26**	**49.29**	76.58	76.05	83.40	82.61	64.50	72.20	62.73	54.27	**61.23**	69.92	39.53	65.20
CLIP + ACTOR	27.33	**90.31**	67.77	47.94	**80.63**	**80.53**	**83.57**	**87.10**	65.30	**76.28**	65.90	**55.85**	59.69	**76.33**	**50.29**	**67.65**

Table 9: **Full results of label-matched comparisons with online CoOp baselines.** Top-1 accuracy (%) across 15 datasets using CLIP-ViT-B/16. All label-using methods share identical queried labels and follow the future-only supervision protocol. Best results are in **bold**.

D.3 Comparison with Online Prompt Tuning

Table 9 provides the full results of the label-matched comparison across 15 datasets. Replay-CoOp extends Online-CoOp by reusing accumulated labeled samples for subsequent prompt updates. Despite this additional label reuse, ACTOR achieves the highest average accuracy (67.65%), outperforming both Online-CoOp (65.15%) and Replay-CoOp (65.20%).

E Conventional Sample Selection

To evaluate the sample-selection component of ACTOR, we compare it against three standard streaming query-selection baselines: Random, Coreset, and Entropy. All methods follow the same Active VLM-TTA setup, use the same labeling budget, and reuse returned labels through the same retrieval memory. They differ only in how they select samples for annotation.

Random. This baseline selects samples for annotation uniformly at random until the labeling budget is exhausted.

Coreset [15]. This baseline prioritizes samples that are far from the current labeled memory in the VLM embedding space. For a test sample x_t, we compute its maximum cosine similarity to the labeled memory and query samples with low similarity:

$$S_{\text{core}}(x_t) = - \max_{(x_i, y_i) \in D_l(t)} \cos(f_{\text{img}}(x_t), f_{\text{img}}(x_i)).$$

Samples with larger $S_{\text{core}}(x_t)$ are prioritized for annotation.

Entropy [12]. This baseline prioritizes samples with high predictive uncertainty. Let q_t denote the base TTA probability vector for x_t. The selection score is

$$S_{\text{ent}}(x_t) = H(q_t) = - \sum_{c \in C} q_{t,c} \log q_{t,c}.$$

Samples with larger $S_{\text{ent}}(x_t)$ are prioritized for annotation under the same labeling budget. Entropy is used only for query selection; retrieval-augmented correction still follows the same memory-based correction rule used by ACTOR.

F Societal Impact

Active VLM-TTA may improve the robustness of vision-language systems deployed in changing real-world environments by selectively incorporating limited human supervision, which could reduce repeated failures under distribution shift and make adaptation more label-efficient. Such improvements may be beneficial in applications where robustness matters, but the same paradigm also

Asset	Type	Citation	License / Terms
CLIP	Model/code	[1]	MIT License
BLIP-2	Model/code	[4]	BSD-3-Clause / model-card terms
SigLIP	Model	[14]	Apache-2.0
TPT	Baseline code	[8]	MIT License
C-TPT	Baseline code	[9]	MIT License
TDA	Baseline code	[10]	MIT License
CoOp	Baseline code	[28]	MIT License
ImageNet	Dataset	[13]	ImageNet terms of access
ImageNet-A	Dataset	[36]	MIT License
ImageNet-V2	Dataset	[37]	MIT License
ImageNet-R	Dataset	[38]	MIT License
ImageNet-Sketch	Dataset	[39]	MIT License
FGVC-Aircraft	Dataset	[40]	Non-commercial research use
Caltech101	Dataset	[41]	CC BY 4.0
Stanford Cars	Dataset	[42]	Research/ImageNet-style terms
DTD	Dataset	[43]	Research use
EuroSAT	Dataset	[44]	MIT License; Sentinel data terms
Flowers102	Dataset	[45]	Research use
Food101	Dataset	[46]	Scientific fair-use / Foodspotting terms
Oxford Pets	Dataset	[47]	CC BY-SA 4.0
SUN397	Dataset	[48]	Academic/research use
UCF101	Dataset	[49]	Research/fair-use style terms

Table 10: **Existing assets used in our experiments and their licenses or terms of use.**

introduces risks: annotation errors or human bias can propagate through reused feedback, and limited labeling budgets may cause adaptation quality to differ across classes, scenarios, or user groups. As a result, deploying Active VLM-TTA in real-world systems requires careful attention to feedback quality, selection policy, and potential disparities in who benefits from adaptation.

G Licenses and Terms of Use for Existing Assets

We use publicly available pretrained models, baseline implementations, and benchmark datasets as shown in Table 10. All assets are used for research evaluation, and we do not redistribute third-party datasets.

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: The abstract and introduction accurately describe the paper's main contributions: a streaming Active VLM-TTA setting with explicit future-only label supervision, the proposed training-free ACTOR framework, and empirical improvements under the stated budgeted protocol. These claims are supported by the method in Section 3, the main results in Section 4, and additional analyses in Appendix D.

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: The paper explicitly discusses limitations in Section 6.

   Guidelines:

   - The answer [N/A] 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: [N/A]

Justification: The paper does not make formal theoretical claims.

Guidelines:

- The answer [N/A] 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: The paper provides the datasets, baselines, algorithm description, hyperparameters, evaluation protocol, and implementation details needed to reproduce the main results. These are described in Section 4, Appendix B, and Appendix C, including Algorithm 1, dataset lists, thresholds, label budget, and model settings.

Guidelines:

- The answer [N/A] 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: The paper provides detailed experimental settings, including datasets, baselines, and implementation details. The source code and instructions for reproducing the experiments are publicly available at `https://github.com/kaist-dmlab/ACTOR`.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies the datasets, baselines, model family, batch size, thresholds, initial labeled-sample count, budget, evaluation metric, and hardware setting. Additional algorithmic details and benchmark descriptions are provided in Appendix B and Appendix C.

   Guidelines:

   - The answer [N/A] 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: The main experimental tables report mean accuracy with standard deviations across three runs for the adaptive methods, and the implementation details explicitly state that all experiments were conducted three times and averaged with standard deviation.

Guidelines:

- The answer [N/A] 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: The paper described the hardware spec (See Appendix C.3), and reported the results of per-sample inference cost (See Table 4).

   Guidelines:

   - The answer [N/A] 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: To the best of our knowledge, the research complies with the NeurIPS Code of Ethics. The work uses standard benchmark datasets and model evaluation procedures, does not involve deceptive data collection or unsafe intervention, and does not report experiments involving protected or vulnerable human populations.

   Guidelines:

   - The answer [N/A] 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: The paper discusses both potential benefits and risks of active test-time adaptation in Appendix F.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a high-risk generative model, scraped dataset, or other asset that would require special safeguards for responsible release. The work studies adaptation methods on existing benchmark datasets and pretrained vision-language models.

    Guidelines:

    - The answer [N/A] 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: All existing assets used in the paper, including pretrained models, baseline codebases, and benchmark datasets, are cited and their licenses or terms of use are explicitly listed in Appendix G. We use these assets only for research evaluation and do not redistribute third-party datasets.

Guidelines:

- The answer [N/A] 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: We introduce `ACTOR`, a new algorithmic framework for **Active VLM-TTA**. The implementation and documentation are publicly available through our GitHub repository at `https://github.com/kaist-dmlab/ACTOR`.

Guidelines:

- The answer [N/A] 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: [N/A]

Justification: The paper does not conduct crowdsourcing experiments or human-subject studies.

Guidelines:

- The answer [N/A] 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: [N/A]

    Justification: The paper does not involve human-subject data collection or human-subject experimentation requiring IRB review.

    Guidelines:

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

16. **Declaration of LLM usage**

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

    Answer: [N/A]

    Justification: In this paper, the LLM is used only for writing and editing.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.