Statistics Caching Test-Time Adaptation for Vision-Language Models

Zenghao Guan[1,2,3], **Yucan Zhou**[4], **Wu Liu**[5], **Xiaoyan Gu**[1,2,3*]
[1]Institute of Information Engineering, Chinese Academy of Sciences,
[2]School of Cyber Security, University of Chinese Academy of Sciences,
[3]State Key Laboratory of Cyberspace Security Defense,
[4]Tianjin University, [5]University of Science and Technology of China
`zenghaoguan.cs@gmail.com, guxiaoyan@iie.ac.cn`

Abstract

Test-time adaptation (TTA) for Vision-Language Models (VLMs) aims to enhance performance on unseen test data. However, existing methods struggle to achieve robust and continuous knowledge accumulation during test time. To address this, we propose Statistics Caching test-time Adaptation (SCA), a novel cache-based approach. Unlike traditional feature-caching methods prone to forgetting, SCA continuously accumulates task-specific knowledge from all encountered test samples. By formulating the reuse of past features as a least squares problem, SCA avoids storing raw features and instead maintains compact, incrementally updated feature statistics. This design enables efficient online adaptation without the limitations of fixed-size caches, ensuring that the accumulated knowledge grows persistently over time. Furthermore, we introduce adaptive strategies that leverage the VLM's prediction uncertainty to reduce the impact of noisy pseudo-labels and dynamically balance multiple prediction sources, leading to more robust and reliable performance. Extensive experiments demonstrate that SCA achieves compelling performance while maintaining competitive computational efficiency. The code is available at this link.

1 Introduction

Recently, large-scale vision-language models (VLMs), such as CLIP [1], have shown strong generalization across a wide range of tasks, thanks to pre-training on billions of image-text pairs from the web. These models learn powerful cross-modal representations, enabling impressive zero-shot performance without task-specific supervision [1, 2, 3, 4]. However, despite their success, adapting VLMs to real-world tasks remains challenging. Most existing approaches rely on a small number of labeled samples from the target domain for fine-tuning [5, 6, 7], which limits their use in scenarios where annotation is expensive or unavailable [8, 9, 10, 11]. This motivates the need for test-time adaptation (TTA) for VLMs, where the model adapts to new domains on the fly using only unlabeled test data, improving robustness without extra labeling cost [12, 13, 14, 15, 16].

Existing TTA methods for vision-language models (VLMs) generally fall into two categories: (1) test-time prompt tuning methods and (2) cache-based methods. Test-time prompt tuning [12, 13, 17, 18, 19] adapts the prompt to each individual sample in the test data stream using an entropy minimization objective on randomly augmented samples. To avoid prompt degradation caused by errors in previous unsupervised optimization steps, these methods typically process each test sample independently and reset the model after evaluating it. These approaches, while simple, fail to leverage

[*]Corresponding author is Xiaoyan Gu.

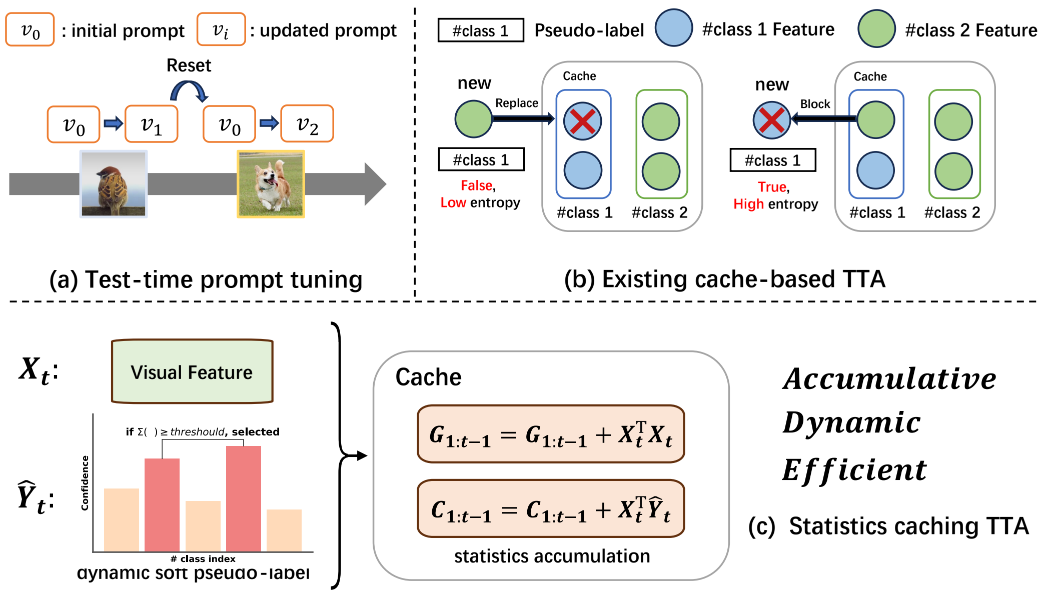

Figure 1: **Comparison of our SCA (c) with test-time prompt tuning methods (a) and existing cache-based methods (b).** We use each test sample's feature and soft pseudo-label to update the cached feature statistics, enabling continuous knowledge accumulation without forgetting.

information from previously seen samples. As a result, the model may repeatedly forget useful patterns, which reduces its ability to adapt and generalize across the test set. Moreover, test-time tuning methods require backpropagation through the text encoder at every test-time training step to learn the text prompt, leading to substantial computational overhead.

Another type of work is cache-based methods [14, 20, 15, 21]. These methods use a class-wise feature cache to retain information during inference. Since ground-truth labels are unavailable, test samples are assigned to cache slots based on zero-shot pseudo-labels. To improve cache quality under limited capacity, low-entropy samples are retained while high-entropy ones are discarded.

Cache-based methods have garnered significant attention due to their potential for knowledge accumulation and computational efficiency. However, a critical question arises: is this acclaimed knowledge accumulation process truly robust and effective over time? Since low entropy does not guarantee correctness, confidently incorrect samples may enter and persist in the cache. Previous studies have primarily highlighted this detrimental impact on knowledge quality, often reflected in decreased classification accuracy. However, they often overlook a more insidious consequence: whether and how such errors fundamentally disrupt and hinder the process of accumulating new knowledge. Here, we analyze the issue from two perspectives:

Firstly, a misclassified low-entropy sample can overwrite a previously correct entry in the cache, leading to the **forgetting** of useful knowledge that had already been acquired. To demonstrate this effect, we conducted a controlled experiment on the Flowers102 [22] using a SOTA cache-based method, DPE [14]. Specifically, we construct a controlled setup with two cache update orders: (1) using only a set of standard samples, and (2) using the same set followed by a few deliberately selected low-entropy misclassified samples that are guaranteed to enter the cache. The second sequence simulates the injection of "forgetting triggers," allowing us to measure the performance drop by directly comparing the performance before and after forgetting. As shown in Fig. 2, DPE suffers from forgetting, which causes a decline in performance as useful information is overwritten.

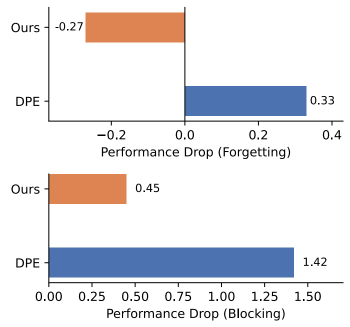

Figure 2: Performance drop on Flowers using ViT-B/16. Negative values mean the performance improves instead of drops.

Secondly, a misclassified low-entropy sample, once admitted, might persistently occupy a cache slot due to its high confidence score. This can **block** potentially correct but higher-entropy samples from entering the cache, thereby hindering the intended knowledge accumulation and refinement. To demonstrate this, we simulated a 'bad case' scenario, where several low-entropy but incorrect samples were placed at the beginning of the online test sample sequence. As depicted in Fig.2, under these conditions, the cache was consistently dominated by incorrect features, preventing effective knowledge accumulation and resulting in a significant performance drop for DPE compared to a standard evaluation order.

The preceding analysis reveals that while existing TTA methods attempt to adapt, they face significant hurdles in achieving robust and continuous knowledge accumulation. Test-time prompt tuning resets for each sample, inherently discarding prior information and preventing any meaningful accumulation of knowledge from the test stream. Existing cache-based methods, while aiming for accumulation, are susceptible to forgetting due to limited cache capacity and can suffer from blocked accumulation when noisy feature-label pairs lead to the entrenchment of incorrect information. This highlights a critical need for an approach that can consistently learn from all past samples and is robust to pseudo-label noise.

To this end, we propose Statistics Caching test-time Adaptation (SCA), a simple yet effective cache-based test-time approach for VLM. As illustrated in Fig. 1, SCA is designed to effectively accumulate task-specific knowledge from all previously encountered test samples without the forgetting issues inherent in feature-caching methods. We achieve this by modeling the effective use of previous test sample features as a least squares problem. Crucially, solving this problem requires only the corresponding feature statistics, not the full set of historical raw features. Therefore, unlike prior cache-based methods that store a limited subset of raw features, SCA maintains and incrementally updates these compact feature statistics in the cache. This core design choice enables the continuous integration of information from every incoming sample, addressing the problem of forgetting due to fixed-size feature caches and ensuring that the accumulated knowledge grows persistently over time.

Furthermore, to ensure the quality of this continuously accumulated knowledge, especially in the presence of noisy pseudo-labels, we introduce a dynamic soft pseudo-label assignment strategy. This strategy intelligently utilizes the uncertainty information from the VLM's initial predictions to generate more robust pseudo-labels for updating the feature statistics. Finally, we propose an instance-level adaptive fusion strategy that combines the cache (statistics-derived) logit and the textual (zero-shot) logit using prediction entropy. This strategy estimates the confidence of the statistics-derived classifier and adaptively adjusts its weight relative to the VLM's textual logits. As a result, it enables more robust and flexible predictions with relatively little hyperparameter tuning effort.

We evaluate SCA on 15 diverse datasets across out-of-domain and cross-domain benchmarks, where it achieves notable performance gains over state-of-the-art prompt tuning and cache-based methods. Moreover, SCA avoids time-consuming backpropagation and achieves over $10\times$ computational efficiency compared to TPT [12], making it much more practical for resource-limited settings. Our contributions are summarized as follows:

- We propose Statistics Caching test-time Adaptation (SCA), a novel cache-based method that stores feature statistics instead of raw features, enabling effective accumulation of task-specific knowledge from all previously seen test samples without forgetting.
- We introduce a dynamic soft pseudo-labeling strategy that leverages uncertainty to reduce error accumulation. Additionally, we propose an instance-level adaptive fusion mechanism based on prediction entropy, enabling robust predictions without extensive hyperparameter tuning.
- Extensive experimental results show that SCA achieves strong performance across 15 diverse datasets, matching or surpassing state-of-the-art methods while maintaining high computational efficiency.

2 Preliminaries

Zero-shot classification in CLIP. CLIP [1] enables zero-shot image classification via pre-trained visual $\Phi_I(\cdot)$ and textual $\Phi_T(\cdot)$ encoders mapping inputs to a shared D-dimensional space $\mathbb{R}^D$. To classify an image I into one of K classes $\{c_1, ..., c_K\}$, text prompts T_k are generated for each class (e.g., "`a photo of [CLS]`"). The image and prompts are embedded:

$$f_I = \Phi_I(I) \quad \text{and} \quad f_{T_k} = \Phi_T(T_k) \quad \text{for } k = 1, ..., K. \tag{1}$$

Here, f_I and f_{T_k} are the image and text features, respectively. CLIP computes the cosine similarity $\text{cosine}(f_I, f_{T_k})$ between the image feature and each text feature. These similarities determine class probabilities using a softmax function with temperature κ:

$$p(c_k|I) = \frac{\exp\left(\text{cosine}(f_{T_k}, f_I)/\kappa\right)}{\sum_{j=1}^{K} \exp\left(\text{cosine}(f_{T_j}, f_I)/\kappa\right)}. \tag{2}$$

The image I is assigned the class c_k corresponding to the maximum probability $p(c_k|I)$.

Test-time prompt tuning. These methods [12, 13, 23, 17, 18, 19] optimize learnable prompts for each test sample. As ground-truth labels are unavailable during testing, the optimization is guided by the prediction entropy of selected sample augmentations:

$$\min_v \mathcal{L}_(v; I) = \min_v -\sum_{k=1}^{K} p(\hat{y} = c_k \mid \mathcal{A}(I), v) \log p(\hat{y} = c_k \mid \mathcal{A}(I), v), \tag{3}$$

where v denotes the learnable prompt, $\mathcal{A}(I)$ represents the set of selected augmentations with lower prediction entropy for test sample I, and $p(\hat{y} = c_k \mid \mathcal{A}(I), v)$ denotes the averaged predicted probability over these augmentations.

Cache-based test-time adaptation. These methods [14, 20, 15, 21] maintain a fixed-size cache of M entries per class to store features from test samples. This cache acts as an external memory, allowing the model to reference past examples during inference. By continuously updating the cache based on minimal-entropy filtering strategy, the model can gradually accumulate useful information and improve prediction consistency. Final predictions are obtained by combining zero-shot logits from the textual classifier with similarity scores from the cache:

$$\begin{aligned} \mathbf{z}_k &= \alpha_0 \mathbf{z}_{\text{text}} + \alpha_1 \mathbf{z}_{\text{cache}} \\ &= \alpha_1 f_I^\top f_{T_k} + \alpha_2 \sum_{i=1}^{M} \exp(-\beta(1 - f_I^\top f_i^{(k)})), \end{aligned} \tag{4}$$

where $\mathbf{z}_{\text{cahce}}$ denotes the logit for class k, and $f_i^{(k)}$ is the i^{th} feature of category k in the cache. Their performance relies heavily on fusion hyperparameters α_1 and α_2.

3 Statistics Caching Test-time Adaptation

To avoid forgetting of previously learned knowledge, we aim to make full use of past test samples observed up to time step t to guide model adaptation. Denote $\boldsymbol{X}_{1:t} \in \mathbb{R}^{n_{1:t} \times d}$ as the features extracted from all test samples up to time t, and $\boldsymbol{Y}_{1:t} \in \mathbb{R}^{n_{1:t} \times K}$ as their corresponding one-hot encoded ground-truth labels. Here, $n_{1:t}$ represents the cumulative number of test samples up to time t. We make full use of historical samples by formulating it as a least squares problem:

$$\min_{\boldsymbol{W}_t} \|\boldsymbol{X}_{1:t}\boldsymbol{W}_t - \boldsymbol{Y}_{1:t}\|_{\text{F}}^2 + \gamma\|\boldsymbol{W}_t\|_{\text{F}}^2. \tag{5}$$

Here, we use γ to control the strength of the regularization. The closed-form solution of Eq. 5 is:

$$\boldsymbol{W}_t = (\boldsymbol{X}_{1:t}^\top \boldsymbol{X}_{1:t} + \gamma \boldsymbol{I})^{-1} \boldsymbol{X}_{1:t}^\top \boldsymbol{Y}_{1:t}. \tag{6}$$

$\boldsymbol{W}_t$ denotes the classifier learned by leveraging the feature representations of all previously observed test samples up to time t. It captures accumulated knowledge across test instances, enabling more informed and robust predictions.

However, in practical test-time scenarios, storing all historical sample features $\boldsymbol{X}_{1:t}$ is typically infeasible due to increasing storage overhead or potential privacy concerns. Moreover, ground-truth labels $\boldsymbol{Y}_{1:t}$ are not accessible during testing. Existing methods typically rely on CLIP's zero-shot predictions to assign pseudo-labels for cache construction. However, these pseudo-labels are often noisy, leading to the inclusion of unreliable feature-label pairs that degrade the quality of the updated classifier. To tackle these challenges, we propose the following parts.

Statistics Accumulation. In fact, computing $\boldsymbol{W}_t$ does not require storing the growing feature matrix $\boldsymbol{X}_{1:t}$ as test samples accumulate. Instead, it is sufficient to maintain two compact feature statistics that are independent of the sample size: $\boldsymbol{G}_{1:t} = \boldsymbol{X}_{1:t}^\top \boldsymbol{X}_{1:t} \in \mathbb{R}^{d \times d}$ and $\boldsymbol{C}_{1:t} = \boldsymbol{X}_{1:t}^\top \boldsymbol{Y}_{1:t} \in \mathbb{R}^{d \times K}$. Gram matrix $\boldsymbol{G}_{1:t}$ represents the second-order feature statistics, capturing the pairwise relationships between the features, and $\boldsymbol{C}_{1:t}$ represents the first-order statistics, which is essentially the weighted

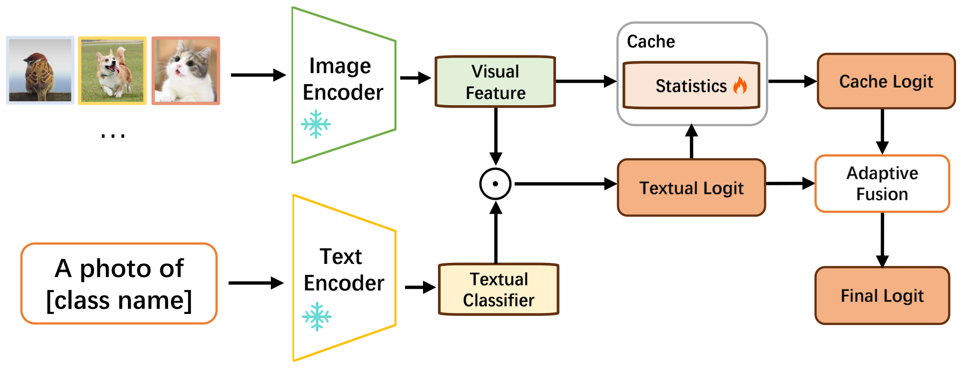

Figure 3: Framework of our SCA. Each test sample updates the feature statistics in the cache. Based on these statistics, we build a closed-form classifier that captures knowledge from all test samples. Finally, we apply a instance-level adaptive fusion strategy to combine the textual and cache logits.

sum of the features and the labels. Due to the linear characteristics of these feature statistics, we can update them additively to support online updates during test-time scenarios:

$$\boldsymbol{G}_{1:t} = \boldsymbol{G}_{1:t-1} + \boldsymbol{X}_t^\top \boldsymbol{X}_t, \quad \boldsymbol{C}_{1:t} = \boldsymbol{C}_{1:t-1} + \boldsymbol{X}_t^\top \boldsymbol{Y}_t. \tag{7}$$

Here, $\boldsymbol{X}_t \in \mathbb{R}^{n_t \times d}$ and $\boldsymbol{Y}_t \in \mathbb{R}^{n_t \times K}$ denote the features of the current test sample and its corresponding label, respectively. Finally, these two statistics are then used to compute $\boldsymbol{W}_t$ as follows:

$$\boldsymbol{W}_t = (\boldsymbol{G}_{1:t} + \gamma \boldsymbol{I})^{-1} \boldsymbol{C}_{1:t}. \tag{8}$$

This approach efficiently accumulates knowledge and builds a strong classifier that captures information from all previously seen samples.

Dynamic soft pseudo-label assignment. The above discussion assumes access to true labels when accumulating feature statistics. In practice, we typically rely on pseudo-labels generated by CLIP's zero-shot predictions to update $\boldsymbol{C}_{1:t}$. A common but problematic practice [14, 15] is to convert these predictions into hard pseudo-labels by selecting the single most confident class as ground truth. This approach ignores the nuance of the full probability distribution, collapsing all uncertainty into a binary decision, and can amplify errors when the top prediction is only marginally more probable or even incorrect [24]. A continuous stream of such errors will inevitably skew the learned cache classifier and degrade its accuracy.

To mitigate this issue, we dynamically generate the soft pseudo-label $\hat{\boldsymbol{Y}}_t$ for each test sample $\boldsymbol{X}_t$ based on its predicted class probabilities $\mathbf{p}_{\text{text}}$ from the CLIP textual (zero-shot) classifier. Specifically, we first identify a candidate set of classes based on each sample. Without loss of generality, we assume the class probabilities $\mathbf{p}_{\text{text}} = (\mathbf{p}_{\text{text}}^{(1)}, \mathbf{p}_{\text{text}}^{(2)}, ..., \mathbf{p}_{\text{text}}^{(K)})$ are sorted in descending order. In this case, $\mathbf{p}_{\text{text}}^{(1)}$ denotes the probability of the class initially deemed most likely by the zero-shot classifier. We select the smallest number of classes, denoted by Ω, such that their cumulative probability meets or exceeds a threshold τ:

$$\Omega = \min \left\{ k \;\middle|\; \sum_{k=2}^{K} \mathbf{p}_{\text{text}}^{(k)} \geq \tau (1 - \mathbf{p}_{\text{text}}^{(1)}) \right\} \cup \{1\}. \tag{9}$$

Then the soft pseudo-label $\hat{\boldsymbol{Y}}_t$ is generated as:

$$\hat{\boldsymbol{Y}}_t^{(k)} = \frac{\exp\left(\mathbf{p}_{\text{text}}^{(k)} \cdot \mathbb{I}(k \in \Omega)\right)}{\sum_{l \in \Omega} \exp\left(\mathbf{p}_{\text{text}}^{(l)}\right)}. \tag{10}$$

By allowing $|\Omega|$ vary with τ and the sample's confidence, our method produces softer pseudo-labels for uncertain samples and sharper labels when the model is confident. This dynamic assignment strategy better captures the sample-specific uncertainty inherent in the zero-shot predictions and effectively curbs the overconfidence often associated with hard labeling schemes. Finally, the pseudo-label $\hat{\boldsymbol{Y}}_t$ is used to update the first-order statistics $\hat{\boldsymbol{C}}_{1:t}$ as shown in Eq.7. The cache classifier $\hat{\boldsymbol{W}}_t$ is then computed as Eq.8 to obtain the cache logits $\mathbf{z}_{\text{cache}} = \boldsymbol{X}_t \hat{\boldsymbol{W}}_t$.

Table 1: **Experimental results on the cross-domain benchmark with two backbones of CLIP.** The best and second-best results are shown **in bold** and underlined, respectively.

Method	Aircraft	Caltech	Cars	DTD	EuroSAT	Flower	Food101	Pets	SUN397	UCF101	Average
CLIP-ResNet-50	15.66	85.88	55.70	40.37	23.69	61.75	73.97	83.57	58.80	58.84	55.82
Ensemble	16.11	87.26	55.89	40.37	25.79	62.77	74.82	82.97	60.85	59.48	56.63
CoOp [5] [IJCV'22]	15.12	86.53	55.32	37.29	26.20	61.55	75.59	87.00	58.15	59.05	56.18
CoCoOp [26] [CVPR'22]	14.61	87.38	56.22	38.53	28.73	65.57	76.20	**88.39**	59.61	57.10	57.23
TPT [12] [NeurIPS'22]	17.58	87.02	58.46	40.84	28.33	62.69	74.88	84.49	61.46	60.82	57.66
DiffTPT [13] [ICCV'23]	17.60	86.89	**60.71**	40.72	41.04	63.53	**79.21**	83.40	62.72	62.67	59.85
TDA [14] [CVPR'24]	17.61	89.70	57.78	43.74	42.11	68.74	77.75	86.18	62.53	64.18	61.03
DPE [15] [NeurIPS'24]	19.80	**90.83**	59.26	50.18	41.67	67.60	77.83	85.97	64.23	61.98	61.93
BCA [27] [CVPR'25]	**19.89**	89.70	58.13	48.58	**42.12**	66.30	77.19	85.58	63.38	63.51	61.44
SCA (Ours)	20.82	91.03	59.73	**53.31**	43.33	**71.09**	76.51	86.15	**64.50**	**66.40**	**63.29**
CLIP-ViT-B/16	23.67	93.35	65.48	44.27	42.01	67.44	83.65	88.25	62.59	65.13	63.58
Ensemble	23.22	93.55	66.11	45.04	50.42	66.99	82.86	86.92	65.63	65.16	64.59
CoOp [5] [IJCV'22]	18.47	93.70	64.51	41.92	46.39	68.71	85.30	89.14	64.15	66.55	63.88
CoCoOp [26] [CVPR'22]	22.29	93.79	64.90	45.45	39.23	70.85	83.97	90.46	66.89	68.44	64.63
TPT [12] [NeurIPS'22]	24.78	94.16	66.87	47.75	42.44	68.98	84.67	87.79	65.50	68.04	65.10
DiffTPT [13] [ICCV'23]	25.60	92.49	67.01	47.00	43.13	70.10	**87.23**	88.22	65.74	62.67	65.47
TDA [14] [CVPR'24]	23.91	94.24	67.28	47.40	58.00	71.42	86.14	88.63	67.62	70.66	67.53
DPE [15] [NeurIPS'24]	**28.95**	94.81	67.31	54.20	55.79	75.07	86.17	91.14	70.07	70.44	69.40
Dynaprompt [23] [ICLR'25]	24.33	94.32	67.65	47.96	42.28	69.95	85.42	88.28	66.32	68.72	65.52
BCA [27] [CVPR'25]	28.59	94.69	66.86	53.49	56.63	73.12	85.97	90.43	68.41	67.59	68.59
SCA (Ours)	28.50	**94.85**	**68.49**	**57.09**	**57.16**	**76.09**	86.09	**91.44**	**70.27**	**73.43**	**70.34**

Instance-level Adaptive Fusion. Previous methods [25, 15, 20] fuse zero-shot (textual) logit and cache logit using a fixed dataset-level coefficient, which typically requires extensive hyperparameter tuning on the validation set or even the test set. In contrast, we propose an instance-level fusion strategy that leverages entropy to estimate classifier confidence and adaptively weights the logits, enabling more robust and adaptive predictions.

For each test sample, we compute the prediction entropy for each classifier and denote them as $\mathcal{H}_{\text{cache}}$ and $\mathcal{H}_{\text{text}}$. Lower entropy indicates higher confidence, so we derive confidence weights as follows:

$$[\alpha_1, \alpha_2] = \text{softmax}\big(-\beta\,[\mathcal{H}_{\text{cache}}, \mathcal{H}_{\text{text}}]\big), \tag{11}$$

where β controlling the sharpness of the distribution. This enables the model to evaluate their relative confidence, relying more on the zero-shot model when the cache model is uncertain and shifting toward the cache model as its confidence increases. Entropy is used as it is the most common confidence metric. We then compute the aggregated logits as:

$$\mathbf{z}_{\text{final}} = \alpha_1\,\mathbf{z}_{\text{cache}} + \alpha_2\,\mathbf{z}_{\text{text}} \tag{12}$$

By automatically and adaptively adjusting the fusion coefficient for each test sample, our method captures instance-level uncertainty and improves robustness across diverse data distributions. We also provide an option to amplify the cache logits using the same sharpness parameter β in a divisive manner. Using default settings ($\beta = 0.5$) gives strong performance across multiple datasets. In other words, a single set of hyperparameters can be applied to fuse predictions across different datasets, reducing the effort required to tune the fusion parameters α_1, α_2.

4 Experiments

4.1 Experimental Settings

Datasets. Following prior work [12, 13, 15, 27], we evaluate our method using two established benchmarks: the cross-domain benchmark and the out-of-distribution (OOD) benchmark. (1) The cross-domain benchmark assesses the model's ability to generalize across various domains, each with its own set of classes. This benchmark includes 10 diverse image classification datasets from distinct domains: FGVCAircraft [28], Caltech101 [29], StanfordCars [30], DTD [31], EuroSAT [32],

Table 2: **Experimental results on the OOD benchmark with two backbones of CLIP.** The best and second-best results are shown **in bold** and <u>underlined</u>, respectively.

Method	ImageNet	ImageNet-A	ImageNet-V2	ImageNet-R	ImageNet-S	Average	OOD Average
CLIP-ResNet-50 [1]	58.16	21.83	51.41	56.15	33.37	44.18	40.69
Ensemble	59.81	23.24	52.91	60.72	35.48	46.43	43.09
CoOp [5] [IJCV'22]	63.33	23.06	55.40	56.60	34.67	46.61	42.43
CoCoOp [26] [CVPR'22]	62.81	23.32	55.72	57.74	34.48	46.81	42.82
TPT [12] [NeurIPS'22]	60.74	26.67	54.70	59.11	35.09	47.26	43.89
DiffTPT [13] [ICCV'23]	60.80	**31.06**	55.80	58.80	37.10	48.71	45.69
TDA [14] [CVPR'24]	61.35	30.29	55.54	62.58	38.12	49.58	46.63
DMN-ZS [20] [CVPR'24]	**63.87**	28.57	56.12	61.44	<u>39.84</u>	49.97	46.49
DPE [15] [NeurIPS'24]	<u>63.41</u>	30.15	**56.72**	**63.72**	**40.03**	50.81	**47.66**
BCA [27] [CVPR'25]	61.81	30.35	56.58	<u>62.89</u>	38.08	49.94	46.98
SCA (Ours)	63.11	<u>30.64</u>	<u>56.51</u>	62.85	39.33	<u>50.49</u>	<u>47.33</u>
CLIP-ViT-B/16 [1]	66.73	47.87	60.86	73.98	46.09	59.11	57.20
Ensemble	68.34	49.89	61.88	77.65	48.24	61.20	59.42
CoOp [5] [IJCV'22]	71.51	49.71	64.20	75.21	47.99	61.72	59.28
CoCoOp [26] [CVPR'22]	71.02	50.63	64.07	76.18	48.75	62.13	59.91
TPT [12] [NeurIPS'22]	68.98	54.77	63.45	77.06	47.94	62.44	60.81
DiffTPT [13] [ICCV'23]	70.30	55.68	65.10	75.00	46.80	62.28	60.52
TDA [14] [CVPR'24]	69.51	60.11	64.67	80.24	50.54	65.01	63.89
DMN-ZS [20] [CVPR'24]	**72.25**	58.28	65.17	78.55	**53.20**	65.49	63.80
DPE [15] [NeurIPS'24]	<u>71.91</u>	59.63	**65.44**	80.40	52.26	<u>65.93</u>	<u>64.43</u>
Dynaprompt [23] [ICLR'25]	69.61	56.17	64.67	78.17	48.22	63.37	61.81
BCA [27] [CVPR'25]	70.22	**61.14**	64.90	<u>80.72</u>	50.87	65.37	64.16
SCA (Ours)	71.75	<u>60.33</u>	<u>65.38</u>	**80.85**	<u>52.50</u>	**66.16**	**64.77**

Flowers102 [22], Food101 [33], OxfordPets [34], SUN397 [35], and UCF101 [36]. (2) The OOD benchmark measures the model's robustness against natural shifts in data distribution. It evaluates performance on ImageNet [37] and four challenging variants: ImageNet-A [9], ImageNet-V2 [10], ImageNet-R [38], and ImageNet-Sketch [39].

Implementation Details. Consistent with prior work [15, 25], our experiments adopt ResNet-50 [40] and ViT-B/16 [41] as the visual encoders for CLIP, with a batch size of 1 to satisfy online processing requirements. For textual prompts, we employ the prompt ensembling strategy as used in previous works [15, 42, 20, 25]. For data augmentation, we adopt the approach from DPE [12], generating 63 randomly resized crops per test image for the OOD benchmark. No data augmentation is applied in the cross-domain benchmark. By default, we set the ridge coefficient γ to 1e4, threshold τ to 0.1, sharpness coefficient β to 0.5. All experiments are conducted on a single NVIDIA A100 RTX GPU, using top-1 accuracy to measure classification performance.

Baselines. We compare our method with zero-shot CLIP using either a simple prompt or prompt ensemble, few-shot methods such as CoOp [5], with 16-shot annotated samples per class, alongside the following established CLIP test-time adaptation approaches: (1) prompt-tuning methods: TPT [12], DiffTPT [13], Dynaprompt[23]; (2) cache-based methods: TDA [14], DMN-ZS [20], DPE [15]. We also include recent baseline BCA [27] that adapts likelihood using Bayes rules. To ensure a fair comparison, we exclude RLCF [43] and COSMIC [21], which rely on auxiliary strong models (e.g., ViT-L-14) during test-time adaptation. We also exclude some transductive TTA baselines [44, 45] that rely on utilizing inter-sample relations within a large batch. The results of these baselines are directly taken from their respective original papers.

4.2 Results and Discussions

Cross-Domain Benchmark. Table 1 presents results on the cross-domain benchmark. While CoOp [5] and CoCoOp [26] improve CLIP's generalization, they rely on training-based adaptation with labeled data, which restricts their practical applicability. In contrast, our SCA outperforms CoOp by an average of 7.11% and 5.71%, CoCoOp by an average of 6.06% and 6.46% on two backbones in a training-free manner, demonstrating superior generalization without relying on labeled examples. Compared to test-time prompt tuning methods, SCA achieves a 4.82% gain in average accuracy over the strongest baseline, Dynaprompt [23]. It also consistently outperforms cache-based approaches, surpassing TDA [14] by 2.25% and 2.81%, and DPE [15] by 1.36% and 0.94% on ResNet-50 and ViT-B/16 backbones, respectively. Notably, unlike existing cache-based methods [15, 14] that rely on dataset-specific fusion hyperparameters (α_1, α_2) for optimal performance, our method uses the same configuration for all datasets within a benchmark-backbone pair, while different benchmarks or backbones use different configurations, highlighting its strong generalization.

OOD Benchmark. In Table 2, we further evaluate the generalization capability of our proposed method on the OOD benchmark by comparing it with state-of-the-art approaches. Our proposed method consistently outperforms the CoOp [5] and CoCoOp [26] with substantial improvements across all datasets. Specifically, it improves average accuracy by about 5% on both backbones. These results demonstrate the strong generalization capability of our approach and its effectiveness in enhancing performance without any additional supervised training. On the ResNet-50 backbone, our method performs slightly below the strongest baseline DPE [15]; however, it achieves this level of performance without any training overhead, unlike DPE, which requires additional training. In contrast, on the ViT-B/16 backbone, SCA consistently outperforms all existing methods, with average accuracy improvements ranging from 0.23% to 3.72% across various OOD datasets.

Computation Efficiency. We measure time per image for several baselines on SUN397 (19,850 samples) using a single 80 GB NVIDIA RTX A100 GPU as an example of computational efficiency. Table 3 presents the corresponding results. Since our proposed SCA avoids time-consuming backpropagation, it offers a notable advantage in computational efficiency compared to prompt tuning methods. Specifically, our approach is over 10× faster than TPT [12]. Compared to the cache-based methods TDA [14], our approach is slightly less efficient but delivers notable performance gains, especially on the cross-domain benchmark.

Table 3: **Computation efficiency comparison on SUN397 using ViT-B/16.**

Method	BP-free	Time Per Image (s)
CLIP [1]	✓	0.007
TPT [12]	✗	0.218
TDA [14]	✓	0.009
DPE [15]	✗	0.043
SCA (Ours)	✓	0.012

Storage Overhead. Previous cache-based methods that store feature-label pairs with fixed size M have a storage cost of $M \times K \times d$. Our cached feature statistics accumulate knowledge from all samples and require $(K + d) \times d$ storage. While this may be less efficient when the number of classes K is small, our method offers lower storage overhead as K grows, for example on large-scale datasets like ImageNet.

4.3 Ablation Studies

Effectiveness on statistics accumulation and dynamic soft label assignment. To investigate the roles of our statistics accumulation and dynamic soft label assignment, we report the performance on 10 fine-grained recognition datasets involved in cross-domain datasets using ViT-B/16 in Table 4. [W/o statistics accumulation] means that we follow [14, 15] by using a fixed-size cache of M feature-label pairs per class to build our classifier (Eq. 6). [W/o dynamic soft label assignment] means that we use hard pseudo-labels for statistics accumulation. We have the following key observations: (1) Replacing statistics accumulation with direct feature caching leads to a performance drop compared to our method. While increasing the cache size gradually improves performance, the gains eventually plateau. Interestingly, when the cache size reaches 8, it outperforms the state-of-the-art baseline DPE [15] (69.40). This improvement stems from our use of second-order feature information (the Gram matrix G) to construct the cache-based classifier, whereas prior cache-based methods rely solely on first-order information. Studies in other areas of machine learning similarly conclude that second-order feature statistics significantly enhance performance [46, 47, 48, 49]. (2) Using hard labels to update statistics leads to a performance drop. This highlights the importance of the extra information in soft labels and demonstrates the effectiveness of our dynamic label assignment.

Table 4: **Effectiveness on statistics accumulation and dynamic soft label assignment.** We report the averaged results on 10 datasets in cross-domain benchmark.

Method	Avg
w/o statistics accumulation ($M = 4$)	69.31
w/o statistics accumulation ($M = 8$)	69.47
w/o statistics accumulation ($M = 16$)	69.75
w/o statistics accumulation ($M = 32$)	69.72
w/o dynamic soft label assignment	69.32
with both	**70.34**

Ablations on fusion coefficients. In our SCA, the fusion coefficient is automatically adjusted for each test sample. We perform an ablation study by manually setting a fixed fusion coefficient, varying α_1 from 0.1 to 0.9. As shown in Fig. 4, the optimal fusion value differs across datasets. While the adaptive coefficients may not be optimal for every case, they consistently outperform most fixed coefficients across different datasets. Compared to some cache-based methods [14, 15] that require costly parameter searches to tune fusion hyperparameters for each dataset, our approach uses a single set of hyperparameters for all datasets to fuse the prediction and still deliver strong performance. This achieves a good trade-off between performance and tuning effort.

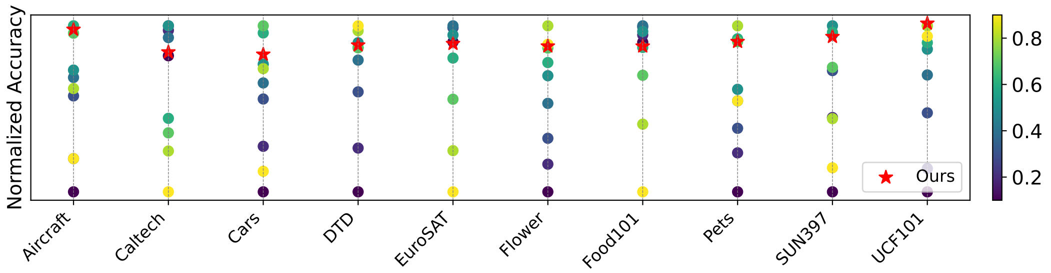

Figure 4: Normalized accuracy with varying fusion parameters on ViT-B/16.

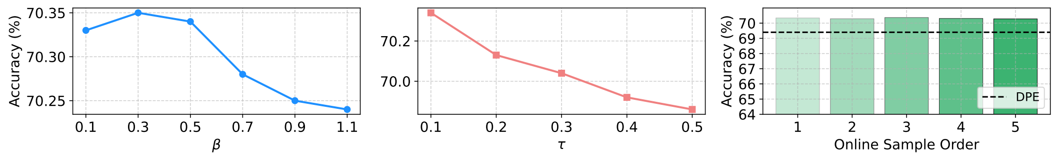

Figure 5: **Left:** hyperparameter analysis of sharpness coefficient β. **Middle:** hyperparameter analysis of threshould τ. **Right:** sensitivity analysis of test-time sample order.

Hyperparameter Analysis. Here, we present the results of hyperparameter analysis for the threshould τ and the sharpness coefficient β on 10 fine-grained recognition datasets. As shown in Fig. 5 (Left and Middle), the performance remains relatively stable when the parameters are set within a reasonable range. This suggests that our method is not sensitive to these hyperparameters, making it more robust and easier to apply in practice.

Sensitivity to test time sample order. Since our SCA is designed for online test-time adaptation, its performance may be affected by the order in which test samples are encountered. To assess this sensitivity, we conduct experiments on 10 datasets from the cross-domain benchmark, each with varying test-time sample orders. As shown in Fig. 5 (Right), our method demonstrates stable performance across different orders and consistently outperforms the state-of-the-art baseline DPE [15], highlighting its robustness to input sequence variations in online settings.

5 Related Work

Test-time prompt tuning. These methods tune the text prompts using the test samples. TPT [12] optimizes the prompt by minimizing prediction entropy to promote consistency across augmented views of the same input. DiffTPT [13] follows the TPT's framework and uses a pre-trained diffusion model to generate more diverse augmentations. C-TPT [18] optimizes prompts by minimizing calibration error, while RLCF [43] introduces an additional CLIP model as a reward signal to guide test-time prompt tuning. Despite their effectiveness, they typically treat each test sample in isolation, resetting the model for every instance without leveraging information from previously seen samples. To effectively accumulate previously learned knowledge, it is essential to incorporate a cache module. For instance, Dynaprompt [23] uses a prompt buffer and introduces a dynamic selection strategy to update the appropriate prompt stored in the buffer. Although this approach facilitates the accumulation of historical knowledge, it, like other test-time tuning methods, requires backpropagation through the text encoder at each test-time step, resulting in significant computational overhead.

Cache-based test-time adaptation. These methods store high-confidence visual features during test time and utilize them to provide extra information for current prediction. TDA [14] and DMN [20] establish visual caches from test data and pseudo-labels to retrieve class prototypes during test time. Similarly, DPE [15] maintains a dynamic cache that facilitates the training of residual dual prototypes. COSMIC [21] enhances the semantic diversity of pseudo-labels by leveraging fine-grained visual features from DINOv2. These methods demonstrate high computational efficiency and strong performance due to the rich information provided by features stored in the cache. However, they suffer from forgetting and blocking issues during knowledge accumulation. Additionally, fusing the predictions derived from the cache features with the VLM's zero-shot predictions often requires extensive and careful tuning. Our SCA overcomes these challenges through statistics accumulation combined with dynamic label assignment and fusion strategy.

6 Conclusion

We propose Statistics Caching test-time Adaptation (SCA), a cache-based method that stores feature statistics instead of raw features, allowing effective accumulation of task-specific knowledge without forgetting. We also introduce a dynamic soft pseudo-labeling strategy that uses uncertainty to reduce error accumulation, and an instance-level adaptive fusion mechanism based on prediction entropy for robust predictions without heavy hyperparameter tuning. Experiments on 15 diverse datasets show that SCA matches or exceeds state-of-the-art performance with high computational efficiency.

While our method enables test-time accumulation of task-specific knowledge without training and achieves strong performance, it still shares some limitations with existing methods. Firstly, storing feature statistics introduces some storage overhead. Secondly, our method only mitigates error accumulation during test time to a certain extent, and the performance still falls short of the ideal setting where ground-truth labels are available to build the classifier. Moreover, similar to existing CLIP TTA methods, our approach mainly focuses on the single-domain scenario, where all test samples come from a single dataset. Extending CLIP-based TTA to the multi-domain setting where test samples from multiple domains are mixed is an important yet underexplored direction. In such cases, naive accumulation may even lead to negative effects due to domain discrepancies. Since feature representations from different domains often exhibit distinctive patterns, a promising direction is to identify each sample's domain and accumulate feature statistics separately for each domain. In future work, we plan to further investigate more effective ways to leverage feature statistics and develop improved strategies for robust knowledge accumulation across domains.

Acknowledgements

This work is supported by the Chinese Academy of Sciences under grant No. XDB0690302, the National Key Research and Development Program of China (NO. 2024YFE0203200), and the National Nature Science Foundation of China (NO. U24A20329).

References

[1] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*. PMLR, 2021, pp. 8748–8763. 1, 3, 7, 8

[2] J. Yu, Z. Wang, V. Vasudevan, L. Yeung, M. Seyedhosseini, and Y. Wu, "Coca: Contrastive captioners are image-text foundation models," *Transactions on Machine Learning Research*, 2022. [Online]. Available: https://openreview.net/forum?id=Ee277P3AYC 1

[3] X. Zhai, X. Wang, B. Mustafa, A. Steiner, D. Keysers, A. Kolesnikov, and L. Beyer, "Lit: Zero-shot transfer with locked-image text tuning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 18 123–18 133. 1

[4] Y. Li, F. Liang, L. Zhao, Y. Cui, W. Ouyang, J. Shao, F. Yu, and J. Yan, "Supervision exists everywhere: A data efficient contrastive language-image pre-training paradigm," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=zq1iJkNk3uN 1

[5] K. Zhou, J. Yang, C. C. Loy, and Z. Liu, "Learning to prompt for vision-language models," *International Journal of Computer Vision*, vol. 130, no. 9, pp. 2337–2348, 2022. 1, 6, 7, 8

[6] R. Zhang, W. Zhang, R. Fang, P. Gao, K. Li, J. Dai, Y. Qiao, and H. Li, "Tip-adapter: Training-free adaption of clip for few-shot classification," in *European Conference on Computer Vision*. Springer, 2022, pp. 493–510. 1

[7] T. Yu, Z. Lu, X. Jin, Z. Chen, and X. Wang, "Task residual for tuning vision-language models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 10 899–10 909. 1

[8] D. Hendrycks and T. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=HJz6tiCqYm 1

[9] D. Hendrycks, K. Zhao, S. Basart, J. Steinhardt, and D. Song, "Natural adversarial examples," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 262–15 271. 1, 7, 15, 16

[10] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do imagenet classifiers generalize to imagenet?" in *International Conference on Machine Learning*. PMLR, 2019, pp. 5389–5400. 1, 7, 15, 16

[11] C. Zhang, S. Stepputtis, J. Campbell, K. Sycara, and Y. Xie, "Hiker-sgg: Hierarchical knowledge enhanced robust scene graph generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 28 233–28 243. 1

[12] M. Shu, W. Nie, D.-A. Huang, Z. Yu, T. Goldstein, A. Anandkumar, and C. Xiao, "Test-time prompt tuning for zero-shot generalization in vision-language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 14 274–14 289, 2022. 1, 3, 4, 6, 7, 8, 9, 16, 17

[13] C.-M. Feng, K. Yu, Y. Liu, S. Khan, and W. Zuo, "Diverse data augmentation with diffusions for effective test-time prompt tuning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 2704–2714. 1, 4, 6, 7, 9, 16

[14] A. Karmanov, D. Guan, S. Lu, A. E. Saddik, and E. Xing, "Efficient test-time adaptation of vision-language models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 1, 2, 4, 5, 6, 7, 8, 9, 14, 16

[15] C. Zhang, S. Stepputtis, K. Sycara, and Y. Xie, "Dual prototype evolving for test-time generalization of vision-language models," *Advances in Neural Information Processing Systems*, vol. 37, pp. 32 111–32 136, 2024. 1, 2, 4, 5, 6, 7, 8, 9, 14, 16, 17

[16] Z. Han, J. Yang, G. Wang, J. Li, Q. Xu, M. Z. Shou, and C. Zhang, "Dota: Distributional test-time adaptation of vision-language models," *arXiv preprint arXiv:2409.19375*, 2024. 1

[17] J. Abdul Samadh, M. H. Gani, N. Hussein, M. U. Khattak, M. M. Naseer, F. Shahbaz Khan, and S. H. Khan, "Align your prompts: Test-time prompting with distribution alignment for zero-shot generalization," *Advances in Neural Information Processing Systems*, vol. 36, pp. 80 396–80 413, 2023. 1, 4

[18] H. S. Yoon, E. Yoon, J. T. J. Tee, M. Hasegawa-Johnson, Y. Li, and C. D. Yoo, "C-tpt: Calibrated test-time prompt tuning for vision-language models via text feature dispersion," *arXiv preprint arXiv:2403.14119*, 2024. 1, 4, 9

[19] D.-C. Zhang, Z. Zhou, and Y.-F. Li, "Robust test-time adaptation for zero-shot prompt tuning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 15, 2024, pp. 16 714–16 722. 1, 4

[20] Y. Zhang, W. Zhu, H. Tang, Z. Ma, K. Zhou, and L. Zhang, "Dual memory networks: A versatile adaptation approach for vision-language models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 28 718–28 728. 2, 4, 6, 7, 9, 14, 16, 17

[21] F. Huang, J. Jiang, Q. Jiang, H. Li, F. N. Khan, and Z. Wang, "Cosmic: Clique-oriented semantic multi-space integration for robust clip test-time adaptation," *arXiv preprint arXiv:2503.23388*, 2025. 2, 4, 7, 9

[22] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *Indian Conference on Computer Vision, Graphics and Image Processing*. IEEE, 2008, pp. 722–729. 2, 7, 15, 16

[23] Z. Xiao, S. Yan, J. Hong, J. Cai, X. Jiang, Y. Hu, J. Shen, C. Wang, and C. G. Snoek, "Dynaprompt: Dynamic test-time prompt tuning," in *The Thirteenth International Conference on Learning Representations*. 4, 6, 7, 9, 16

[24] J. Zhang, Q. Wei, F. Liu, and L. Feng, "Candidate pseudolabel learning: Enhancing vision-language models by prompt tuning with unlabeled data," in *Forty-first International Conference on Machine Learning*. 5

[25] Z. Deng, Z. Chen, S. Niu, T. Li, B. Zhuang, and M. Tan, "Efficient test-time adaptation for super-resolution with second-order degradation and reconstruction," *Advances in Neural Information Processing Systems*, vol. 36, pp. 74 671–74 701, 2023. 6, 7

[26] K. Zhou, J. Yang, C. C. Loy, and Z. Liu, "Conditional prompt learning for vision-language models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 816–16 825. 6, 7, 8

[27] L. Zhou, M. Ye, S. Li, N. Li, X. Zhu, L. Deng, H. Liu, and Z. Lei, "Bayesian test-time adaptation for vision-language models," *arXiv preprint arXiv:2503.09248*, 2025. 6, 7, 16

[28] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi, "Fine-grained visual classification of aircraft," *arXiv preprint arXiv:1306.5151*, 2013. 6, 15, 16

[29] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," *Computer Vision and Image Understanding*, vol. 106, no. 1, pp. 59–70, 2007. 6, 15, 16

[30] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3d object representations for fine-grained categorization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2013, pp. 554–561. 6, 15, 16

[31] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi, "Describing textures in the wild," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3606–3613. 6, 15, 16

[32] P. Helber, B. Bischke, A. Dengel, and D. 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*, vol. 12, no. 7, pp. 2217–2226, 2019. 6, 15, 16

[33] L. Bossard, M. Guillaumin, and L. Van Gool, "Food-101–mining discriminative components with random forests," in *European Conference on Computer Vision*. Springer, 2014, pp. 446–461. 7, 15, 16

[34] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. Jawahar, "Cats and dogs," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3498–3505. 7, 15, 16

[35] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2010, pp. 3485–3492. 7, 15, 16

[36] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012. 7, 15, 16

[37] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255. 7, 15, 16

[38] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo *et al.*, "The many faces of robustness: A critical analysis of out-of-distribution generalization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 8340–8349. 7, 15, 16

[39] H. Wang, S. Ge, Z. Lipton, and E. P. Xing, "Learning robust global representations by penalizing local predictive power," in *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 10 506–10 518. 7, 15, 16

[40] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778. 7

[41] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=YicbFdNTTy 7

[42] S. Pratt, I. Covert, R. Liu, and A. Farhadi, "What does a platypus look like? generating customized prompts for zero-shot image classification," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 15 691–15 701. 7

[43] S. Zhao, X. Wang, L. Zhu, and Y. Yang, "Test-time adaptation with CLIP reward for zero-shot generalization in vision-language models," 2024. 7, 9

[44] C. Zhang, K. Xu, Z. Liu, Y. Peng, and J. Zhou, "Scap: Transductive test-time adaptation via supportive clique-based attribute prompting," *arXiv preprint arXiv:2503.12866*, 2025. 7

[45] M. Zanella, C. Fuchs, C. De Vleeschouwer, and I. B. Ayed, "Realistic test-time adaptation of vision-language models," *arXiv preprint arXiv:2501.03729*, 2025. 7

[46] H. Li, Y. Zhou, X. Gu, B. Li, and W. Wang, "Diversified semantic distribution matching for dataset distillation," in *Proceedings of the 32nd ACM International Conference on Multimedia*, 2024, pp. 7542–7550. 8

[47] Z. Guan, Y. Zhou, and X. Gu, "Capture global feature statistics for one-shot federated learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 16, 2025, pp. 16 942–16 950. 8

[48] M. D. McDonnell, D. Gong, A. Parvaneh, E. Abbasnejad, and A. Van den Hengel, "Ranpac: Random projections and pre-trained models for continual learning," *Advances in Neural Information Processing Systems*, vol. 36, pp. 12 022–12 053, 2023. 8

[49] Z. Guan, G. Zhu, Y. Zhou, W. Liu, W. Wang, J. Luo, and X. Gu, "Stsa: Federated class-incremental learning via spatial-temporal statistics aggregation," *arXiv preprint arXiv:2506.01327*, 2025. 8

[50] D. Osowiechi, M. Noori, G. Vargas Hakim, M. Yazdanpanah, A. Bahri, M. Cheraghalikhani, S. Dastani, F. Beizaee, I. Ayed, and C. Desrosiers, "Watt: Weight average test time adaptation of clip," *Advances in neural information processing systems*, vol. 37, pp. 48 015–48 044, 2024. 16

[51] D. Wang, E. Shelhamer, S. Liu, B. Olshausen, and T. Darrell, "Tent: Fully test-time adaptation by entropy minimization," in *International Conference on Learning Representations*. 16

[52] S. Cui, J. Xu, Y. Li, X. Tang, J. Li, J. Zhou, F. Xu, F. Sun, and H. Xiong, "Bayestta: Continual-temporal test-time adaptation for vision-language models via gaussian discriminant analysis," *arXiv preprint arXiv:2507.08607*, 2025. 17

[53] M. U. Khattak, H. Rasheed, M. Maaz, S. Khan, and F. S. Khan, "Maple: Multi-modal prompt learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 19 113–19 122. 17

Statistics Caching Test-Time Adaptation for Vision-Language Models

Appendix

A Details on forgetting and blocking issues

Since low-entropy zero-shot predictions are not always correct (as shown in the following figure), they can lead to forgetting and blocking issues that hinder the accumulation of new knowledge in existing cache-based methods [14, 15, 20]. We conduct experiments to illustrate this (Figure 2 in the Introduction). Below, we present the experimental settings and analysis in detail.

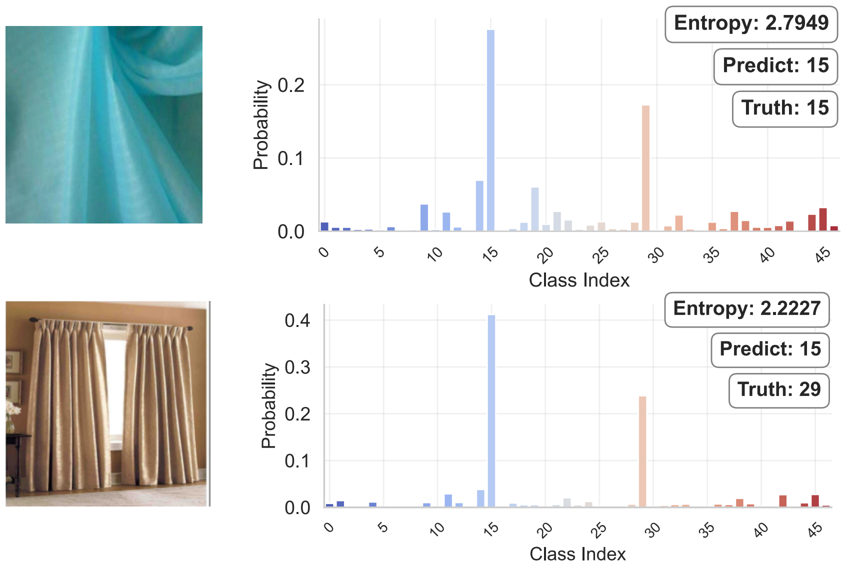

Figure A1: An example of wrong high confidence sample.

Forgetting: a misclassified low-entropy sample can overwrite a previously correct entry in the cache, leading to the forgetting of useful knowledge that had already been acquired. To better illustrate forgetting issues, we conduct an additional set of control experiments. Firstly, we identify a small set of "forgetting trigger samples." For each dataset, we randomly select 10 classes and choose the single lowest-entropy misclassified sample for each class. To ensure these 10 trigger samples are admitted by the cache, we filter out any correctly classified samples from the entire test set that have even lower entropy. From the remaining pool of test samples, we randomly draw 500 samples to serve as our fixed "final test samples." All other remaining samples are designated as "initial samples," which allow the cache to accumulate sufficient class-specific knowledge. We then consider the following two sequences for updating the model's cache:

- Sequence A: `[initial samples]`
- Sequence B: `[initial samples, forgetting trigger samples]`

The forgetting trigger samples, with their low entropy, are designed to infiltrate the cache built from the initial samples, causing it to forget previously learned information. In other words, the cache updated with Sequence A represents the state before forgetting, while the cache updated with Sequence B represents the state after forgetting. To quantify the impact of forgetting, we measure the performance difference between Sequence A and Sequence B on the final test samples, where the performance drop is defined as the accuracy of Sequence A minus the accuracy of Sequence B. The results are summarized in the following figure.

We have several observations: (1) The SOTA baseline DPE is highly vulnerable to forgetting. (2) SCA is inherently robust to forgetting, as its accumulation design prevents error samples from fully

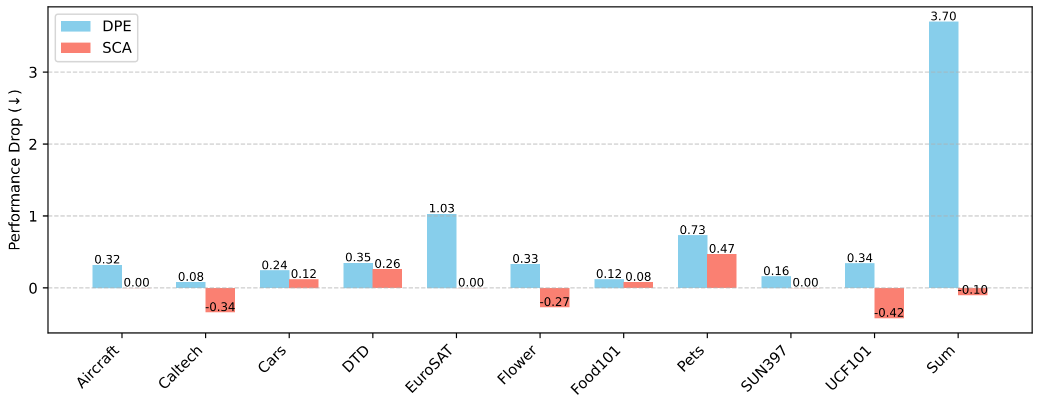

Figure A2: Forgetting issues.

overwriting the stored knowledge. (3) Remarkably, SCA even leverages these "forgetting triggers" to improve performance (a negative drop), turning a vulnerability into a strength.

Blocking: A misclassified sample with low entropy can, once inserted into the cache, occupy a cache slot persistently due to its high confidence score. This prevents correct but higher-entropy samples from being stored, thereby impeding subsequent knowledge accumulation and refinement. To illustrate this effect, we simulate blocking cases by randomly selecting 20% of all classes and, for each selected class, identifying the single misclassified sample with the lowest entropy. These samples are placed at the start of the test sequence, and the performance drop is measured against the standard evaluation order. As shown in the figure below, the performance degradation caused by blocking is considerably smaller for SCA than for DPE. This advantage arises because SCA's statistics-based accumulation mechanism is not constrained by fixed cache slots, thus avoiding the persistence of early high-confidence errors and enabling effective learning from later samples.

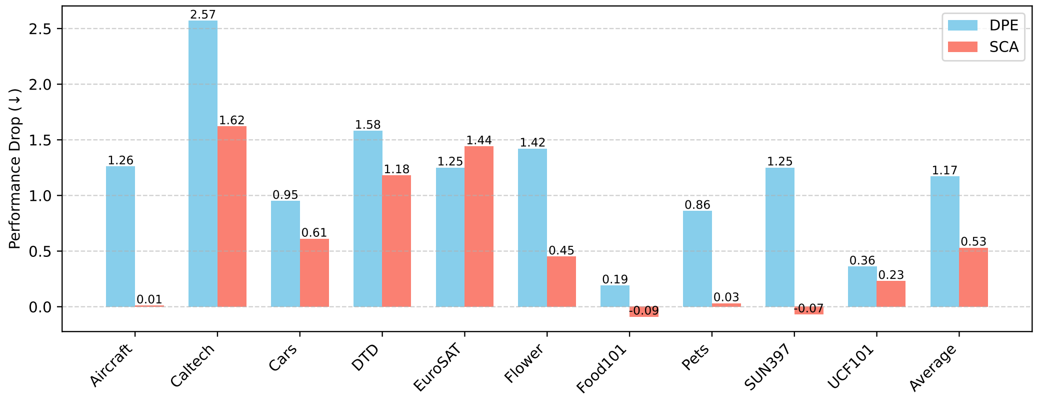

Figure A3: Blocking issues.

B Experiment

B.1 Details on Dataset

We evaluate our methods on 15 datasets, including Caltech101 [29], DTD [31], EuroSAT [32], FGVCAircraft [28], Flowers102 [22], Food101 [33], OxfordPets [34], StanfordCars [30], SUN397 [35], UCF101 [36], ImageNet [37], ImageNet-V2 [10], ImageNet-Sketch [39], ImageNet-A [9], and ImageNet-R [38]. Table B1 provides detailed statistics for each dataset, including the number of classes, the sizes of the training, validation, and testing sets, as well as the corresponding original task for each dataset.

Table B1: **Summary of the 15 image classification datasets used in experiments**. The last four ImageNet variant datasets are designed for evaluation only and contain no training or validation splits.

Benchmark	**Dataset**	**Classes**	**Splits**			**Task**
			train	*val*	*test*	
Cross-Domain	Caltech101 [29]	100	4,128	1,649	2,465	Object recognition
	DTD [31]	47	2,820	1,128	1,692	Texture recognition
	EuroSAT [32]	10	13,500	5,400	8,100	Satellite imagery
	FGVCAircraft [28]	100	3,334	3,333	3,333	Fine-grained aircraft recognition
	Flowers102 [22]	102	4,093	1,633	2,463	Fine-grained flower recognition
	Food101 [33]	101	50,500	20,200	30,300	Fine-grained food recognition
	OxfordPets [34]	37	2,944	736	3,669	Fine-grained pet recognition
	StanfordCars [30]	196	6,509	1,635	8,041	Fine-grained car recognition
	SUN397 [35]	397	15,880	3,970	19,850	Scene recognition
	UCF101 [36]	101	7,639	1,898	3,783	Action recognition
Out-of-Distribution	ImageNet [37]	1,000	1.28M	-	50,000	Object recognition
	ImageNet-V2 [10]	1,000	-	-	10,000	Robustness (collocation shift)
	ImageNet-Sketch [39]	1,000	-	-	50,889	Robustness (sketch domain)
	ImageNet-A [9]	200	-	-	7,500	Robustness (adversarial attack)
	ImageNet-R [38]	200	-	-	30,000	Robustness (multi-domain)

B.2 Baseline Details

- TPT [12], a prompt tuning method designed to minimize self-entropy across predictions from multiple augmented views.
- DiffTPT [13], an improved variant of TPT that leverages diffusion-based augmentations to refine prompt optimization.
- TDA [14], a cache-based method that builds positive and negative caches to improve the prediction of the current test sample.
- DMN-ZS [20], a cache-based method that constructs memory banks from test data and pseudo-labels to enhance the prediction of the current test sample.
- DPE [15], a cache-based method that trains multi-modal prototypes with historical test data to accumulate task-specific knowledge for test-time adaptation.
- BCA [27], a method that not only updates class embeddings to adapt the likelihood but also updates each class's prior using the posterior of incoming samples.
- Dynaprompt [23], a prompt-tuning method that maintains a prompt buffer and introduces a dynamic selection strategy to adaptively leverage relevant information from previous test samples for test-time adaptation.

B.3 Comparison with other baselines

DMN-ZS [20] is a cache-based method that builds memory banks using test data and pseudo-labels to boost the prediction accuracy of the current test sample. In the original paper, DMN-ZS [20] uses a large cache size ($M = 50$) and conducts extensive hyperparameter search with ground-truth labels from the test set to determine the fusion hyperparameter for each downstream task. Removing hyperparameter search leads to a significant drop in performance. We provide a version that uses the same small cache size ($M = 3$) as other cache-based methods and applies a fixed fusion hyperparameter across all datasets. As shown in Table B2, our SCA substantially outperforms DMN-ZS under a fixed fusion hyperparameter and a small cache size ($M = 3$), and exceeds the performance of DMN-ZS configured with a large cache and per-dataset hyperparameter tuning as reported in the original paper.

Additionally, we provide results of other two gradient-based baselines WATT [50] and TENT [51] on the cross-domain benchmark using CLIP ViT-B/16.

Table B2: **Experimental results on the cross-domain benchmark with CLIP ViT-B/16.** DMN-ZS (M=50) refers to the original results, where the fusion hyperparameter is exhaustively tuned for each dataset. DMN-ZS (M=3, fixed) uses the same small cache size as other cache-based methods and applies a fixed fusion hyperparameter across all datasets. The results of TENT and WATT are directly derived from [52].

Method	Aircraft	Caltech	Cars	DTD	EuroSAT	Flower	Food101	Pets	SUN397	UCF101	Average
DMN-ZS (M=50)	**30.03**	**95.38**	67.96	55.85	**59.43**	74.49	85.08	**92.04**	70.18	72.51	70.30
DMN-ZS (M=50, fixed)	29.76	94.60	67.53	55.08	56.79	75.27	81.60	89.89	69.95	72.22	69.27
DMN-ZS (M=3)	28.56	94.77	66.73	53.84	56.75	75.49	84.50	91.03	68.99	72.59	69.33
DMN-ZS (M=3, fixed)	25.59	91.52	55.57	46.63	46.14	74.83	71.57	72.17	60.77	65.16	61.00
TENT	23.52	93.87	65.96	45.51	49.36	67.40	82.89	87.35	65.58	65.08	64.65
WATT	24.27	93.27	66.37	46.39	55.95	68.21	83.26	88.09	65.89	66.01	65.77
SCA (Ours)	28.50	94.85	**68.49**	**57.09**	57.16	**76.09**	**86.09**	91.44	**70.27**	**73.43**	**70.34**

B.4 Impact of prompt initialization

We follow the existing cache-based methods that use prompt ensembling strategy [20, 15] to initialize the prompt. This practice is widely adopted in the literature to ensure a robust and fixed zero-shot classifier, which serves as the foundation for these feature-space adaptation methods.

For prompt-tuning based methods, the use of a single prompt for methods like TPT and DiffTPT is also the standard protocol, as their focus is on demonstrating adaptation of the prompt itself.

Additionally, we present results using high-quality prompts obtained through training as the enhanced prompt initialization for prompt-tuning based TTA [12]. Specifically, we use the prompts trained by MaPLe [53] as the initialization, a widely adopted approach in prompt-tuning based TTA. As shown in the table below, the performance of prompt-tuning based methods is significantly lower than that of cache-based methods.

Table B3: **Experimental results on the cross-domain benchmark with CLIP ViT-B/16.**

Method	Aircraft	Caltech	Cars	DTD	EuroSAT	Flower	Food101	Pets	SUN397	UCF101	Average
TPT	24.78	94.16	66.87	47.75	42.44	68.98	84.67	87.79	65.50	68.04	65.10
TPT+MaPLe	24.70	93.59	66.50	45.87	47.80	72.37	86.64	90.72	67.54	69.19	66.49
SCA (Ours)	28.50	94.85	**68.49**	**57.09**	57.16	**76.09**	**86.09**	91.44	**70.27**	**73.43**	**70.34**

B.5 Impact of the number of classes and test samples on performance

Impact of the number of classes on performance: The number of classes inevitably impacts performance, as this dictates the inherent difficulty of the task. However, our results show that there is no consistent inverse relationship between the performance gain of SCA and the number of classes. In other words, the effectiveness of our SCA does not systematically degrade as the number of classes increases.

Impact of the number of test samples on performance: The performance of any cache-based TTA method, including strong baselines like DPE and our SCA, is indeed influenced by the amount of test data available for adaptation. Here, we evaluate both SCA and DPE on subsets of the test data (10%, 50%, and 100%) for two datasets: DTD (1692 testing samples) and SUN397 (19,850 testing samples). As shown in the figure below, while both methods experience performance degradation with fewer test samples, SCA shows a smaller drop. This is because SCA leverages every encountered testing sample to incrementally update its feature statistics. Unlike existing cache-based methods that may filter or discard samples, SCA ensures that no information is wasted. By fully leveraging all encountered test samples, SCA builds a more comprehensive and accurate statistical model, resulting in a notable performance advantage under data-scarce scenarios.

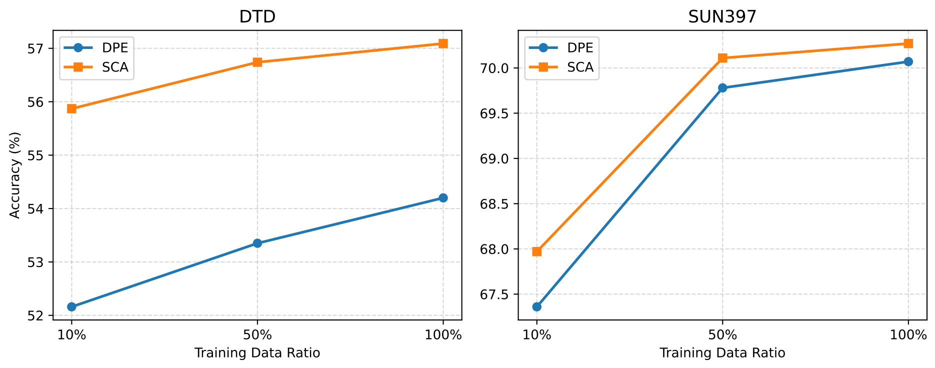

Figure B4: Performance curve.

C Broader Impacts

This paper focuses on test-time adaptation (TTA) for vision-language models (VLMs). In real-world applications, data often come from domains that differ significantly from the training distribution, leading to performance degradation. TTA enables VLMs to adjust to these unseen distributions on the fly, without requiring access to labeled data. Our research improves both the effectiveness and efficiency of TTA on VLMs, paving the way for its wider adoption in practical applications.

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 main claims made in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Limitations in appendix.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [NA]

   Justification: No theory assumptions.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We report the setup throughout the paper in the Section 4.1.

   Guidelines:

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

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

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

   Answer: [No]

   Justification: Code will be released after being accepted.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.

- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: In Section 4.1, we provide detailed descriptions of data splits and the proposed method's hyperparameters.

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

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: We do not report the error bars in our experiments.

   Guidelines:

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

8. **Experiments compute resources**

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

Answer: [Yes]

Justification: We have provided the experiment setting and hardware and software details in Section 4.1.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

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

Answer: [Yes]

Justification: This work does not incorporate any ethic concerns of NeurIPS. The datasets and models are commonly used in the community, and the method does not incorporate potential concerns.

Guidelines:

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

10. **Broader impacts**

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

Answer: [Yes]

Justification: In appendix.

Guidelines:

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

- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [NA]

    Justification: This paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: The datasets and baselines, used libraries are properly credited.

    Guidelines:

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

13. **New assets**

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

    Answer: [Yes]

    Justification: The datasets and baselines, used libraries are well documented and cited.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.

- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

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

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

    Answer: [NA]

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

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: [NA]

    Guidelines:

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

16. **Declaration of LLM usage**

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

    Answer: [NA]

    Justification: The core method development in this paper does not involve LLMs as any important components.

    Guidelines:

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