Efficient Test-time Adaptation through Candidate Verification and Divergence Shifts

Seungmin Oh
Ajou University
seungminoh@ajou.ac.kr

Seunghun Kang
Ajou University
seunghun.kang@ajou.ac.kr

Jongbin Ryu*
Ajou University
jongbinryu@ajou.ac.kr

Abstract

Vision-language models (VLMs) achieve strong zero-shot transferability but remain vulnerable to target-domain shifts at inference time. Test-time adaptation (TTA) offers a practical remedy, yet most existing VLM-TTA methods follow a prediction-side adaptation paradigm. They use test samples to adjust logits, prototypes, caches, priors, or feature statistics, often incurring additional computational overhead. In this paper, we take a different perspective and reframe VLM-TTA as candidate verification rather than prediction adjustment. We propose **Test-Time Correction (TTC)**, a hypothesis-based correction framework guided by a simple principle: *hypothesize, reconstruct, correct*. Given a test feature and its top-k candidate labels, TTC treats each candidate label as a hypothesis, reconstructs the feature within the corresponding latent subspace stored in a memory bank, and measures the resulting **divergence shift**. This shift quantifies how much the candidate subspace and its relations to other candidates change after the hypothetical insertion of the test feature. A correct candidate hypothesis induces only a small shift, whereas an incorrect one perturbs the subspace more strongly. TTC therefore corrects the prediction by selecting the candidate with the minimum aggregated divergence shift. This training-free candidate-verification mechanism avoids iterative optimization and provides a favorable accuracy-efficiency trade-off. Across five TTA settings and 15 benchmark datasets, including zero-shot classification, domain generalization, few-shot classification, base-to-novel generalization, and cross-dataset evaluation, TTC consistently improves accuracy over state-of-the-art VLM-TTA methods while achieving up to $2\times$ speedup, over $3\times$ lower CPU memory usage, and up to $1.4\times$ lower GPU memory usage than the lowest-memory training-free baseline.[2]

1 Introduction

Vision-language models (VLMs) [2, 16, 21, 29, 33, 40] have shown strong zero-shot transferability across diverse visual recognition tasks by aligning image and text representations in a shared embedding space. However, they remain vulnerable to target-domain shifts, including changes in style, texture, viewpoint, and data distribution. Test-time adaptation (TTA) has therefore emerged as a practical strategy for improving VLM predictions at inference time using only unlabeled test samples.

Existing VLM-based TTA methods adapt different components of the prediction pipeline. Early approaches [1, 11, 22, 23, 31, 32, 38, 44–46] optimize learnable prompts or feature representations, often using multiple augmented views of each test sample. Recent training-free methods [12, 17, 49] reduce this overhead through cache retrieval, prototype refinement, or distribution modeling. Despite their differences, most methods share the same principle. They use test samples to adjust logits, prototypes, or feature statistics, mainly adapting on the prediction side.

*Corresponding author.

[2] Our code is available at https://github.com/mmai-laboratory/ttc

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

In this work, we take a different perspective. Rather than using test samples to adjust predictions, we verify whether the test sample is geometrically compatible with each candidate class. Given a test feature and its top-k candidate labels, we treat each candidate as a hypothesis and examine whether its class subspace can accommodate the test feature with minimal perturbation. This reframes TTA as candidate verification rather than prediction adjustment. To instantiate this idea, we propose **Test-Time Correction (TTC)**, a hypothesis-based correction framework for VLM-TTA. TTC follows a simple principle: *hypothesize, reconstruct, correct*. It first selects the top-k candidates from the original VLM prediction. For each candidate hypothesis, TTC reconstructs the test feature within the corresponding latent subspace stored in a memory bank and measures the resulting **divergence shift**. This shift quantifies how much the candidate subspace and its relations to other candidate subspaces change after the hypothetical insertion. If the hypothesis is correct, the insertion should induce only a small shift; otherwise, it should perturb the subspace more strongly. TTC then corrects the prediction by selecting the candidate with the minimum aggregated divergence shift.

This candidate-verification view distinguishes TTC from prior VLM-TTA methods. Unlike confidence-based approaches, TTC does not assume that high logits remain reliable under domain shift. It explicitly tests whether each candidate class subspace can accommodate the test feature with minimal structural perturbation. The divergence-shift criterion is motivated by subspace perturbation analysis [8, 18, 41]. Adding an in-class sample should preserve the corresponding class subspace, whereas adding an out-of-class sample should cause a larger perturbation. Since TTC operates only over a small top-k candidate set and avoids iterative optimization or heavy test-time processing, it offers a favorable accuracy-efficiency trade-off, achieving up to $2\times$ speedup and over $3\times$ lower CPU memory usage than prior VLM-TTA methods while retaining a distinctive candidate-verification mechanism. Across five TTA settings and 15 benchmark datasets, TTC consistently improves accuracy while maintaining competitive inference efficiency and low memory usage. These results show that candidate verification is an effective alternative to conventional prediction-side adaptation. Our contributions are as follows:

- We propose **Test-Time Correction (TTC)**, a hypothesis-based correction framework that reframes VLM-TTA as candidate verification rather than prediction-side adaptation.
- We introduce **divergence shift**, a perturbation-based criterion that verifies each candidate label by measuring how stably the test feature can be reconstructed within the corresponding candidate subspace after its hypothetical insertion into that subspace.
- We demonstrate that TTC achieves a favorable accuracy-efficiency trade-off, attaining state-of-the-art performance with up to $2\times$ speedup, over $3\times$ lower CPU memory, and up to $1.4\times$ lower GPU memory than the most efficient training-free baselines.

2 Related Work

2.1 Test-Time Adaptation in VLMs

Vision-language models (VLMs) have shown strong generalization, motivating the development of parameter-efficient adaptation methods for downstream datasets, such as prompt learning and feature adaptation [28, 47, 48]. Beyond offline adaptation, test-time adaptation (TTA) improves VLM predictions using only pre-trained models and unlabeled test data. Early VLM-TTA methods rely on test-time optimization or augmentation. TPT [32] optimizes learnable prompts using augmented views, while DiffTPT [11] improves view diversity with diffusion-generated images. Later methods refine prediction-side adaptation by calibrating textual features [38], imposing prompt orthogonality [31], aligning prediction distributions [1], or updating auxiliary representations [22, 23, 43–45]. However, these methods require iterative optimization, multiple views, or additional computation.

Recent training-free methods reduce this cost by replacing backpropagation with memory-, prototype-, distribution-, or prior-based adjustment. MTA [42] builds mean-shift prototypes from augmented prediction scores, TDA [17] uses positive and negative caches, OGA [12] models online Gaussian statistics, and BCA [49] updates class-level likelihoods or priors through Bayesian class adaptation. Although these methods improve practicality, they still operate as prediction-side adjustments.

TTC takes a different perspective. Rather than primarily adjusting prediction statistics, it verifies candidate labels by reconstructing the test feature within each candidate subspace and measuring the resulting reconstruction-induced divergence shift.

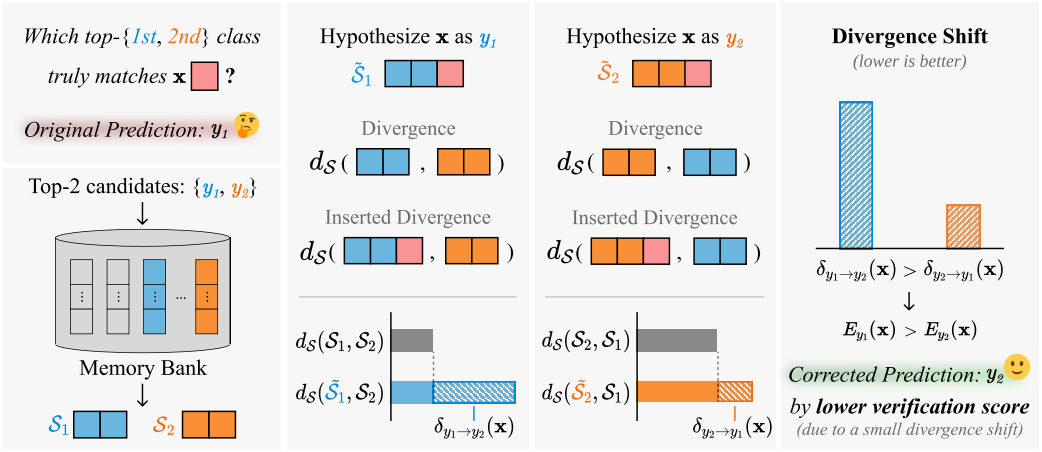


Figure 1: Overview of the Test-Time Correction (TTC) based on divergence shift estimation. TTC first selects the top-2 candidate labels, denoted by y_1 and y_2. Each candidate label is then treated as a hypothesis, and TTC estimates the divergence shifts, $\delta_{y_1 \to y_2}(\mathbf{x})$ and $\delta_{y_2 \to y_1}(\mathbf{x})$, induced by inserting the test feature $\mathbf{x}$ into the corresponding candidate subspace. Although the original top-1 prediction is y_1, TTC corrects the prediction to y_2 because inserting $\mathbf{x}$ into $\mathcal{S}_{y_2}$ induces a smaller divergence shift, yielding the lowest aggregated verification score.

2.2 Nearest Subspace Reconstruction

Nearest-subspace methods [5, 25, 39] measure the proximity of a query to a class-specific subspace rather than to individual data points. A class subspace is constructed from representative features of the same class, and the query is projected onto this subspace to estimate its reconstruction error. Compared with point-wise similarity or nearest-neighbor matching [3, 4], which can be sensitive to noise in high-dimensional spaces, subspace representations provide geometric robustness by capturing class-level structure and suppressing local noise. This makes nearest-subspace methods useful for robust pattern recognition [5, 25] and geometric estimation [39].

TTC is related to nearest-subspace methods, but it uses reconstruction in a different way. Conventional nearest-subspace classifiers assign a query to the class with the smallest absolute reconstruction error [5, 25]. In contrast, TTC uses reconstruction as a perturbation probe for candidate verification. The decision depends not only on the distance between the test sample and a class subspace, but also on how stably the candidate subspace accommodates the sample without disrupting its relations to other candidates. This distinction is central to TTC. An in-class sample should be reconstructed within its corresponding class subspace with only a small perturbation, whereas an out-of-class sample should induce a greater change in subspace relations. This intuition is consistent with subspace perturbation analysis [8, 18, 41]. TTC quantifies this behavior through divergence shift and uses it as a criterion for correcting candidate labels. It thus extends nearest-subspace methods from absolute proximity estimation to candidate verification through reconstruction-induced divergence shift.

3 Method

We present **Test-Time Correction (TTC)**, a candidate-verification framework for VLM-TTA, as illustrated in Figure 1. Given a test feature, TTC obtains the original VLM logits and selects the top-k candidate labels. Instead of directly trusting the top-1 prediction or updating prediction-side statistics, TTC treats each candidate label as a hypothesis and verifies it through subspace stability. For each candidate, TTC hypothetically inserts the test feature into the corresponding class memory, estimates the resulting divergence shift against other candidate subspaces, and selects the candidate with the minimum aggregated verification score. The overall correction rule is

$$\hat{y} = \arg\min_{y \in \mathcal{C}_k(\mathbf{x})} E_y(\mathbf{x}), \tag{1}$$

where $\mathcal{C}_k(\mathbf{x})$ is the top-k candidate set for the test feature $\mathbf{x}$, and $E_y(\mathbf{x})$ denotes the aggregated verification score for candidate label y, formally defined in Definition 2.

3.1 Candidate Hypothesis and Memory Subspace

Let $\mathbf{x} \in \mathbb{R}^d$ be the visual feature of a test image extracted by a frozen VLM, and let $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_{|\mathcal{Y}|}]^\top \in \mathbb{R}^{|\mathcal{Y}| \times d}$ denote the text embeddings of the class label set $\mathcal{Y}$. The original VLM logits are computed as $\mathbf{z} = \mathbf{T}\mathbf{x} \in \mathbb{R}^{|\mathcal{Y}|}$, and the initial prediction is $\hat{y}_0 = \arg\max_{y \in \mathcal{Y}} z_y$. TTC does not directly accept this $\hat{y}_0$. Instead, it constructs the top-k candidate set

$$\mathcal{C}_k(\mathbf{x}) = \mathrm{TopK}_{y \in \mathcal{Y}}(\mathbf{z}), \tag{2}$$

and treats each candidate label $y \in \mathcal{C}_k(\mathbf{x})$ as a candidate hypothesis $h_y : \mathbf{x} \mapsto y$. TTC aims to verify which hypothesis h_y is geometrically compatible with the test feature. To enable this verification, TTC associates each class with a set of representative features stored in a memory bank, and treats the geometric structure induced by these features as the carrier of class identity. For each class $y \in \mathcal{Y}$, let $\mathbf{B}_y = [\mathbf{b}_{y,1}, \ldots, \mathbf{b}_{y,m_y}]^\top \in \mathbb{R}^{m_y \times d}$ denote the memory matrix storing $m_y \leq m$ feature vectors of class y. The associated class memory subspace is $\mathcal{S}_y = \mathrm{span}(\mathbf{B}_y)$. Verifying a candidate label thus amounts to testing whether $\mathbf{x}$ can be stably reconstructed by the corresponding subspace.

Memory update. We maintain $|\mathbf{B}_y| \leq m$ for each class y throughout inference. When a test feature is assigned to class $\hat{y}$ via Eq. (1) and $|\mathbf{B}_{\hat{y}}| < m$, the feature is appended to $\mathbf{B}_{\hat{y}}$. Once the memory is full, $|\mathbf{B}_{\hat{y}}| = m$, the entry with the lowest associated confidence is evicted to make room for the new feature, ensuring that $\mathbf{B}_{\hat{y}}$ remains a high-confidence representation of class $\hat{y}$. In label-free settings, the memory is initialized from confident model predictions on test samples. When labeled training samples are available, their features are stored as fixed memory entries.

3.2 Hypothetical Insertion and Divergence Shift

For each candidate hypothesis h_y, TTC hypothetically inserts the test feature into the candidate class memory and forms an updated subspace

$$\widetilde{\mathbf{B}}_y = [\mathbf{B}_y; \mathbf{x}], \qquad \widetilde{\mathcal{S}}_y = \mathrm{span}(\widetilde{\mathbf{B}}_y). \tag{3}$$

If h_y is correct, $\mathbf{x}$ should already be well explained by $\mathcal{S}_y$, and the updated subspace $\widetilde{\mathcal{S}}_y$ should preserve the original reconstruction relations of $\mathcal{S}_y$. If h_y is incorrect, inserting $\mathbf{x}$ into $\mathcal{S}_y$ introduces a larger out-of-subspace component and disrupts these relations more strongly. This intuition follows the classical subspace perturbation view [8, 18, 41]. To turn this single-subspace perturbation into a verification signal, TTC compares subspace relations *before* and *after* insertion.

Definition 1 (Pairwise Divergence Shift) *Let $d_{\mathcal{S}}(\cdot, \cdot)$ denote a reconstruction-based divergence between subspaces. Given a test feature $\mathbf{x}$ and two candidate labels $y, r \in \mathcal{C}_k(\mathbf{x})$ with $y \neq r$, the* pairwise divergence shift *from y to r induced by inserting $\mathbf{x}$ into the memory of class y is*

$$\delta_{y \to r}(\mathbf{x}) = d_{\mathcal{S}}(\mathcal{S}_y, \mathcal{S}_r) - d_{\mathcal{S}}(\widetilde{\mathcal{S}}_y, \mathcal{S}_r). \tag{4}$$

The reverse shift $\delta_{r \to y}(\mathbf{x})$ is defined analogously by inserting $\mathbf{x}$ into $\mathbf{B}_r$ instead.

The two directions provide complementary evidence. If $\mathbf{x}$ truly belongs to candidate y, then $\delta_{y \to r}(\mathbf{x})$ should remain small because inserting $\mathbf{x}$ into $\mathbf{B}_y$ barely perturbs $\mathcal{S}_y$. At the same time, $\delta_{r \to y}(\mathbf{x})$ should be large because inserting $\mathbf{x}$ into a competing subspace $\mathcal{S}_r$ introduces a substantial orthogonal residual. Combining both directions yields a more discriminative signal than either alone.

3.3 Aggregated Verification Score

We now combine absolute reconstruction compatibility and pairwise divergence shifts into a single score per candidate. Because the score involves only the k candidate memories, each test sample requires $k(k-1)$ pairwise shift evaluations and no gradient updates.

Definition 2 (Aggregated Verification Score) *For each candidate label $y \in \mathcal{C}_k(\mathbf{x})$, the* aggregated verification score *is*

$$E_y(\mathbf{x}) = \underbrace{d_{\mathbf{x}}(\mathcal{S}_y, \mathbf{x})}_{\textit{absolute compatibility}} + \underbrace{\sum_{r \in \mathcal{C}_k(\mathbf{x}) \setminus \{y\}} \big[\delta_{y \to r}(\mathbf{x}) - \delta_{r \to y}(\mathbf{x})\big]}_{\textit{relative stability}}, \tag{5}$$

where $d_{\mathbf{x}}(\mathcal{S}_y, \mathbf{x})$ measures the reconstruction discrepancy between $\mathbf{x}$ and $\mathcal{S}_y$. The TTC prediction is then $\hat{y} = \arg\min_{y \in \mathcal{C}_k(\mathbf{x})} E_y(\mathbf{x})$.

The first term anchors the score to the absolute compatibility between $\mathbf{x}$ and the candidate subspace, ensuring that geometrically distant candidates are penalized even when their pairwise shifts are small. The second term contrasts the stability of candidate y with competing hypotheses r. Specifically, subtracting $\delta_{r \to y}(\mathbf{x})$ rewards candidates for which insertion into competing subspaces causes large perturbations. A correct candidate should achieve both a small absolute discrepancy and a small pairwise divergence shift, resulting in a low $E_y(\mathbf{x})$.

3.4 Subspace Divergence Metric

TTC can be instantiated with different subspace divergence choices. We adopt a nearest-subspace reconstruction error, which directly measures how well one class reconstructs features from another. For a class memory $\mathbf{B}_y$, we define the ridge-regularized projection matrix as

$$\mathbf{P}_y = \mathbf{B}_y^\top \left(\mathbf{B}_y \mathbf{B}_y^\top + \lambda \mathbf{I}\right)^{-1} \mathbf{B}_y, \tag{6}$$

where $\lambda > 0$ is a regularization coefficient. The divergence from $\mathcal{S}_a$ to $\mathcal{S}_b$ is computed as the average row-wise reconstruction error of features in $\mathbf{B}_b$ projected onto $\mathcal{S}_a$ as

$$d_{\mathcal{S}}(\mathcal{S}_a, \mathcal{S}_b) \;=\; \frac{1}{m_b} \left\| \mathbf{B}_b \mathbf{P}_a - \mathbf{B}_b \right\|_{2,1}, \tag{7}$$

where $\| \cdot \|_{2,1}$ denotes the sum of row-wise Euclidean norms. $d_{\mathcal{S}}$ depends on the memories rather than only on their spans; we keep the subspace notation for readability. The feature-to-subspace discrepancy in Eq. (5) is defined analogously

$$d_{\mathbf{x}}(\mathcal{S}_y, \mathbf{x}) \;=\; \left\| \mathbf{x}^\top \mathbf{P}_y - \mathbf{x}^\top \right\|_2. \tag{8}$$

With this metric, the effect of a hypothetical insertion admits a closed form, which we derive and discuss in Appendix A.

4 Experiments

The experiments evaluate whether reconstruction-induced divergence shifts provide an effective and efficient candidate-verification signal across diverse distribution shifts and transfer scenarios. In all settings, TTC keeps the VLM frozen and performs correction solely through candidate verification. We also compare computational efficiency against recent VLM-TTA methods and conduct ablations on the divergence metric, verification score, top-k, and memory design.

4.1 Experimental Setup

To rigorously evaluate our method across diverse transfer learning scenarios, we use a comprehensive suite of five datasets [9, 14, 15, 30, 36] for domain generalization. For zero-shot classification, few-shot classification, base-to-novel generalization, and cross-dataset evaluation, we use 11 downstream datasets [6, 7, 9, 13, 19, 20, 24, 26, 27, 34, 37] with the ResNet50 and ViT-B/16 backbones of CLIP [29]. When using a few training samples, namely in few-shot, base-to-novel, and cross-dataset settings, we strictly follow the standard protocols established in prior work [32, 48]. We repeat every experiment with three random seeds, each of which randomly permutes the entire test stream, and report the average. We use the top-$k = 3$ candidate labels and fix the maximum memory size to $m = 16$. The memory bank initialization protocol depends on the availability of training data. Specifically, when no training samples are available (*i.e.*, zero-shot and domain generalization), the memory bank is initialized using the model's original predictions on test samples. Conversely, when training data is available (*i.e.*, few-shot, base-to-novel, and cross-dataset), the memory bank is initialized using the corresponding training samples. Appendix B provides further details.

4.2 Experimental Results

Zero-shot classification. We evaluate whether TTC improves the initial zero-shot predictions of VLMs. As shown in Table 1, TTC consistently outperforms state-of-the-art VLM-TTA methods [10–12, 17, 31, 32, 35, 38, 43, 49] across both ResNet50 and ViT-B/16 backbones. Among training-free

Table 1: Experimental results of the zero-shot classification task with downstream datasets. The check mark ('✓') denotes the training-free methods.

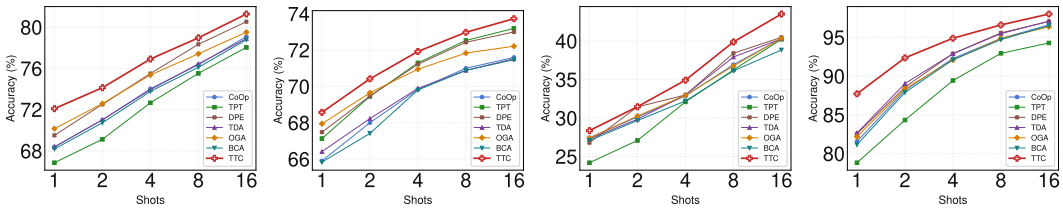
Model	Caltech	DTD	ESAT	Aircraft	Flower	Food	INet	Pets	Cars	SUN	UCF	Avg.
✓ ResNet50$_{(ICML'21)}$	83.68	41.68	27.69	17.40	63.14	74.95	59.90	80.79	58.33	59.53	55.74	56.62
TPT$_{(NeurIPS'22)}$	87.02	40.84	28.33	17.58	62.69	74.88	60.74	84.49	58.46	61.46	60.82	57.94
DiffTPT$_{(ICCV'23)}$	86.89	40.72	41.04	17.60	63.53	**79.21**	60.80	83.40	60.71	62.72	62.67	59.94
C-TPT$_{(ICLR'24)}$	86.90	42.20	27.80	17.00	65.20	74.70	60.20	84.10	56.50	61.00	59.70	57.75
DPE$_{(NeurIPS'24)}$	90.83	50.18	41.67	19.80	67.60	77.83	61.54	85.97	59.26	64.23	61.98	61.90
✓ TDA$_{(CVPR'24)}$	89.70	43.74	42.11	17.61	68.74	77.75	61.35	86.18	57.78	62.53	64.18	61.06
O-TPT$_{(CVPR'25)}$	86.90	41.90	28.40	16.80	65.60	74.20	59.00	83.40	56.40	60.90	58.80	57.48
✓ OGA$_{(CVPR'25)}$	87.91	43.09	43.73	16.86	66.67	76.24	59.65	**86.73**	58.50	62.72	64.37	60.59
✓ BCA$_{(CVPR'25)}$	89.70	48.58	42.12	19.89	66.30	77.19	61.81	85.58	58.13	63.38	63.51	61.47
✓ TTC (Ours)	**91.08**	**53.72**	**47.91**	**21.09**	**71.78**	77.86	**62.58**	86.48	**60.73**	**65.53**	**65.03**	**63.98**
✓ ViT-B/16$_{(ICML'21)}$	90.89	44.87	47.01	24.40	67.65	85.35	68.50	86.20	67.60	64.65	64.36	64.68
TPT$_{(NeurIPS'22)}$	94.16	47.75	42.44	24.78	68.98	84.67	68.98	87.79	66.87	65.50	68.04	65.45
DiffTPT$_{(ICCV'23)}$	92.49	47.00	43.13	25.60	70.10	**87.23**	70.30	88.20	67.01	65.74	68.22	65.91
C-TPT$_{(ICLR'24)}$	93.60	46.00	43.20	24.90	69.80	83.70	68.50	88.20	65.80	64.80	65.70	64.93
✓ ZERO$_{(NeurIPS'24)}$	93.66	46.12	34.33	25.21	67.68	86.53	69.31	87.75	68.04	65.03	67.77	64.68
DPE$_{(NeurIPS'24)}$	94.81	54.20	55.79	**28.95**	75.07	86.17	69.91	91.14	67.31	70.07	70.44	69.44
✓ TDA$_{(CVPR'24)}$	94.24	47.40	58.00	23.91	71.42	86.14	69.51	88.63	67.28	67.62	70.66	67.71
TPS$_{(WACV'25)}$	**95.09**	50.47	44.48	26.34	71.54	85.23	71.45	87.35	69.06	68.98	71.00	67.36
O-TPT$_{(CVPR'25)}$	94.00	45.70	42.80	23.60	70.10	84.10	67.30	88.00	64.50	64.20	64.20	64.41
✓ OGA$_{(CVPR'25)}$	93.20	45.80	**64.50**	25.30	72.70	86.10	68.50	**91.70**	67.80	66.00	71.60	68.47
✓ BCA$_{(CVPR'25)}$	94.69	53.49	56.63	28.59	73.12	85.97	70.22	90.43	66.86	68.41	67.59	68.73
✓ TTC (Ours)	94.24	**55.50**	60.81	28.41	**76.61**	86.62	**71.57**	91.69	**69.99**	**71.43**	**75.26**	**71.10**

Table 2: Experimental results of the domain generalization task. 'Dist.' denotes the average accuracy of distribution-shift datasets. The check mark ('✓') denotes the training-free methods.

Model	# of Aug.	INet	A	R	Ske.	V2	Dist.	Avg.
✓ ResNet50$_{(ICML'21)}$	0	59.90	22.80	60.60	35.51	52.86	42.94	46.33
✓ OGA$_{(CVPR'25)}$	0	59.65	22.71	59.73	36.54	51.02	42.50	45.93
✓ TTC (Ours)	0	62.58	23.44	61.40	38.85	54.85	44.64	48.23
TPT$_{(NeurIPS'22)}$	63	60.74	26.67	59.11	35.09	54.70	43.89	47.26
DiffTPT$_{(ICCV'23)}$	63	60.80	**31.06**	58.80	37.10	55.80	45.69	48.71
C-TPT$_{(ICLR'24)}$	63	60.20	27.00	59.70	35.70	54.80	44.30	47.48
DPE$_{(NeurIPS'24)}$	63	61.54	30.15	**63.72**	40.03	56.72	47.66	50.43
✓ TDA$_{(CVPR'24)}$	63	61.35	30.30	62.60	38.10	55.50	46.63	49.57
O-TPT$_{(CVPR'25)}$	63	59.00	23.07	54.47	33.98	53.11	41.16	44.73
✓ BCA$_{(CVPR'25)}$	63	61.81	30.35	62.89	38.08	56.58	46.98	49.94
✓ TTC (Ours)	10	**62.58**	30.67	63.30	**41.25**	**56.91**	**48.03**	**50.94**
✓ ViT-B/16$_{(ICML'21)}$	0	68.50	49.91	77.64	48.24	61.89	59.42	61.24
✓ OGA$_{(CVPR'25)}$	0	68.50	50.63	78.15	50.45	61.62	60.21	61.87
✓ TTC (Ours)	0	71.57	51.85	78.64	52.35	64.18	61.76	63.72
TPT$_{(NeurIPS'22)}$	63	68.98	53.67	73.90	46.40	64.30	59.57	61.45
DiffTPT$_{(ICCV'23)}$	63	70.30	55.68	75.00	46.80	65.10	60.65	62.58
C-TPT$_{(ICLR'24)}$	63	68.50	52.90	78.00	48.50	63.40	60.70	62.26
✓ ZERO$_{(NeurIPS'24)}$	63	69.31	61.35	77.28	48.29	64.13	62.76	64.07
DPE$_{(NeurIPS'24)}$	63	69.91	59.63	80.40	52.26	65.44	64.43	65.53
✓ TDA$_{(CVPR'24)}$	63	69.51	60.10	80.20	50.50	64.70	63.88	65.00
TPS$_{(WACV'25)}$	63	71.45	60.61	80.20	50.88	64.91	64.15	65.61
O-TPT$_{(CVPR'25)}$	63	67.30	49.87	72.55	47.12	61.65	57.80	59.70
✓ BCA$_{(CVPR'25)}$	63	70.22	61.14	80.72	50.87	64.90	64.41	65.57
✓ TTC (Ours)	10	**71.57**	**61.81**	**80.93**	**54.09**	**66.07**	**65.73**	**66.90**

methods, TTC achieves gains of 2.5 percentage points (pp) and 2.4 pp, respectively. These results demonstrate that candidate verification provides a discriminative correction signal beyond prediction adjustments based on caches, prototypes, or priors.

Domain generalization. We further evaluate TTC under natural distribution shifts. Table 2 reports both zero-augmentation and augmentation-enhanced variants. Although prior methods [17, 32, 43] often use 63 augmented views, TTC uses fewer augmentations while outperforming other training-free



(a) Average (b) ImageNet (c) Aircraft (d) Flowers102

Figure 2: Experimental results of few-shot classification using CoOp trained with a specified shot.

Table 3: Experimental results of the base-to-novel generalization task using a CoOp-tuned ViT-B/16. The results are the harmonic mean of base- and novel-class accuracies.

Model	Caltech	DTD	ESAT	Aircraft	Flower	Food	INet	Pets	Cars	SUN	UCF	Avg.
CoOp	93.36	57.83	69.55	31.57	75.45	85.57	71.88	94.33	67.93	71.13	69.48	72.32
+ TPT	93.71	57.92	60.62	31.82	74.81	84.47	73.67	93.95	69.07	71.51	70.43	71.74
+ DPE	**94.56**	59.80	64.06	31.54	76.29	86.81	73.51	94.37	70.30	72.58	71.52	73.08
+ TDA	94.32	58.91	66.50	31.89	76.59	85.80	72.09	94.02	69.39	72.29	70.73	72.70
+ OGA	93.14	59.16	**76.75**	31.91	76.60	**87.99**	73.33	**94.88**	69.01	74.11	72.08	74.07
+ BCA	93.33	57.74	67.75	30.93	75.32	85.46	71.76	93.99	68.01	71.11	69.36	72.04
+ TTC	94.23	**61.38**	73.89	**32.72**	**78.02**	87.95	**74.30**	94.82	**72.71**	**75.15**	**72.87**	**75.01**

Table 4: Experimental results of the cross-dataset evaluation task. We perform zero-shot prediction using a CoOp-tuned ViT-B/16 trained with 16 shots per class on ImageNet.

Model	INet	Caltech	DTD	ESAT	Aircraft	Flower	Food	Pets	Cars	SUN	UCF	Avg.
CoOp	71.60	92.17	41.43	48.30	20.03	67.13	85.80	89.80	64.87	64.27	67.87	64.84
+ TPT	73.20	93.51	42.38	44.62	20.30	66.69	84.74	89.82	67.06	65.50	68.46	65.12
+ DPE	73.00	91.49	42.22	46.72	19.78	66.95	86.63	89.82	68.24	65.67	69.56	65.46
+ TDA	72.97	91.85	42.71	44.37	**21.01**	67.09	86.58	89.61	67.17	65.40	69.27	65.27
+ OGA	72.21	92.33	42.32	**61.18**	20.21	67.40	86.35	89.94	66.99	66.91	69.75	66.87
+ BCA	71.45	91.95	41.37	48.71	19.95	66.80	85.52	89.53	65.15	64.14	67.30	64.72
+ TTC	**73.73**	**93.87**	**43.36**	56.38	20.13	**70.39**	**86.80**	**90.13**	**68.50**	**67.64**	**72.26**	**67.56**

methods by 1.0 pp with ResNet50 and 1.3 pp with ViT-B/16. Under the zero-augmentation protocol, TTC surpasses existing methods by 2.3 pp and 1.9 pp with ResNet50 and ViT-B/16, respectively.

Few-shot classification. We apply TTC on top of CoOp [48], which is trained with different numbers of shots per class, using the ViT-B/16 CLIP backbone. Figure 2 shows that TTC consistently improves performance across training shot numbers. These gains show that TTC complements few-shot prompt learning. Rather than modifying the learned prompt, TTC corrects each test-time prediction by verifying which candidate subspace most stably accommodates the test feature. This suggests that TTC can provide additional robustness even when the model has already been adapted with a small amount of labeled data. For detailed results, refer to Appendix F.1.

Base-to-novel generalization. Base-to-novel generalization evaluates whether a method can improve novel-class recognition without compromising the knowledge learned from base classes. Table 3 reports the harmonic mean of base and novel accuracies across 11 datasets. TTC achieves the best average harmonic mean, improving over the state-of-the-art by 0.9 pp. This result indicates that candidate verification can mitigate prediction bias in prompt-tuned models while preserving base-class performance. Base- and novel-class results are provided in Appendix F.3.

Cross-dataset evaluation. We evaluate cross-dataset transfer, where CoOp is trained on ImageNet and tested on other downstream datasets. As shown in Table 4, TTC improves the best prior VLM-TTA baseline by approximately 0.7 pp on average. These performance gains show that TTC is not tied to a specific target dataset and remains effective under cross-dataset distribution shifts.

Table 5: Comparison of computing resources such as GPU and CPU memory, throughput (Thr.), wall-clock time (Wall), and accuracy (Acc.) on ImageNet zero-shot classification. All performance measurements are conducted using a single NVIDIA RTX A5000.

Model	ResNet50					ViT-B/16				
	GPU (GB)↓	CPU (MB)↓	Thr. (img/s)↑	Wall (s)↓	Acc. (%)↑	GPU (GB)↓	CPU (MB)↓	Thr. (img/s)↑	Wall (s)↓	Acc. (%)↑
DPE*(NeurIPS'24)*	0.38	1039	16.2	3080.3	61.54	0.20	1009	17.0	2935.5	69.91
TDA*(CVPR'24)*	0.40	520	163.1	306.6	61.35	0.24	520	184.1	271.6	69.51
OGA*(CVPR'25)*	2.26	544	10.6	4712.4	59.65	2.01	545	20.8	2398.7	68.50
BCA*(CVPR'25)*	0.40	472	956.2	52.4	61.81	0.26	471	932.5	53.7	70.22
TTC (Ours)	**0.31**	**142**	**1103.4**	**45.4**	**62.58**	**0.17**	**135**	**2049.0**	**24.5**	**71.57**

Table 6: Top-3 hit rate (Hit@3) and accuracy inside (A_{in}) and outside (A_{out}) the top-3 candidates. Hit@3 is shared by all methods in a column, and $A_{\text{out}} = 0$ for the initial top-1 and Ours (TTC). ZS: zero-shot, DG: domain generalization, FS: few-shot, B2N: base-to-novel, Cross: cross-dataset.

	ZS		DG		FS	B2N		Cross
	ResNet50	ViT-B/16	ResNet50	ViT-B/16	ViT-B/16	Base	Novel	ViT-B/16
Hit@3	80.90	85.39	67.37	81.04	89.24	94.46	81.70	80.30
$A_{\text{in}} = P(\text{correct} \mid \text{GT} \in \textit{top}\text{-}3)$								
Initial top-1	72.62	78.27	68.45	76.24	81.41	86.75	76.94	77.49
DPE*(NeurIPS'24)*	73.83	79.33	69.02	77.03	82.91	87.66	77.30	77.21
TDA*(CVPR'24)*	72.78	77.14	69.06	76.85	81.26	86.55	77.68	77.51
OGA*(CVPR'25)*	71.60	77.76	65.34	75.26	82.37	86.85	79.23	79.28
BCA*(CVPR'25)*	73.94	78.93	68.85	76.55	81.07	86.41	76.59	77.24
TTC (Ours)	**77.26**	**81.77**	**69.75**	**78.14**	**84.73**	**88.51**	**81.00**	**80.96**
$A_{\text{out}} = P(\text{correct} \mid \text{GT} \notin \textit{top}\text{-}3)$								
DPE*(NeurIPS'24)*	1.98	2.79	0.96	0.74	1.13	0.49	1.54	3.79
TDA*(CVPR'24)*	1.14	0.84	0.58	0.64	0.90	0.40	0.86	1.24
OGA*(CVPR'25)*	4.08	4.01	2.35	2.22	4.50	5.50	6.35	4.17
BCA*(CVPR'25)*	0.08	0.09	0.01	0.01	0.03	0.01	0.00	0.08

4.3 Efficiency Comparison

We analyze whether the proposed candidate-verification mechanism can be performed efficiently at test time. Table 5 compares TTC with recent VLM-TTA methods on ImageNet zero-shot classification. To ensure a fair comparison of computational resources, we pre-extract test features and do not use augmentation for all methods, thereby isolating the resources required by each adaptation algorithm. TTC achieves the highest accuracy, uses the least memory, and delivers the highest throughput among compared methods. With ViT-B/16, TTC achieves $2\times$ the throughput of BCA while reducing CPU memory usage by $3\times$. With ResNet50, TTC improves both accuracy and throughput over BCA while requiring substantially less memory. Across both backbones, TTC uses 1.3–$1.4\times$ less GPU memory than the lowest-memory training-free baseline.

4.4 Analysis

Top-k hit rate and decomposition. TTC re-ranks only the top-k candidates, so its accuracy on each dataset decomposes as $\text{Acc} = \text{Hit@}k \times P(\text{correct} \mid \text{GT} \in \text{top-}k)$, where Hit@$k$ is the fraction of samples whose ground truth lies in the top-k set, and the second factor, A_{in}, measures candidate-verification accuracy. As shown in Table 6, in most settings, the top-3 set contains the true label for 80–94% of samples, while the initial prediction misses this label on 13–22% of all samples, leaving substantial room for correction by TTC. Within this space, TTC improves A_{in} over the initial prediction by an average of 3.0 pp, compared with 1.3 pp for the strongest baseline in each setting. By contrast, A_{out} shows that correctly predicting samples outside the top-3 is difficult.

Success and failure study. We partition predictions within the hit set into three cases in Table 7. Overturning a correct initial prediction is rarest at 3.3% and occurs mainly with nearly duplicate

Table 7: Success and failure cases of TTC within the hit set, macro-averaged over the five settings with ViT-B/16. The margin is the median gap between the verification scores of the chosen candidate and the runner-up, and the pairs are representative ImageNet confusions.

Case	% of hit set	Margin	Representative pairs
Success (correct)	82.07	0.434	Alaskan tundra wolf → grey wolf; standard → miniature Schnauzer
Failure (right → wrong)	3.33	0.067	projectile → missile; sunglass → sunglasses
Failure (wrong → wrong)	14.60	0.127	ear of corn → corn; gazelle → impala

Table 8: Ablation study of different divergence metrics.

	Metric	INet	Caltech	DTD	ESAT	Avg.
ResNet50	–	59.90	83.68	41.68	27.69	56.62
	l_2 dist.	52.68	72.70	38.24	27.60	50.01
	Cos. Sim.	61.72	90.63	51.65	35.63	61.58
	NSE	62.58	91.08	53.72	47.91	63.98
ViT-B/16	–	68.50	90.89	44.87	47.01	64.68
	l_2 dist.	62.12	85.11	42.08	32.75	57.60
	Cos. Sim.	70.37	94.56	55.08	56.69	69.61
	NSE	71.57	94.24	55.50	60.81	71.10

Table 9: Ablation study of the three components of the aggregated verification score.

$d_{\mathbf{x}}$	$\delta_{y\to r}$	$\delta_{r\to y}$	ResNet50	ViT-B/16
✓			59.56	67.39
	✓		61.84	69.73
		✓	62.63	70.29
✓	✓		63.60	70.85
✓		✓	63.73	71.04
	✓	✓	63.25	70.72
✓	✓	✓	63.98	71.10

labels such as *sunglass* and *sunglasses*. Both failure cases have verification margins 3.4–6.5× smaller than those of successes, indicating that failures concentrate where competing candidates are nearly tied. These candidates typically represent semantically similar classes such as *ear of corn* and *corn*. The detailed results are provided in Appendix F.4.

4.5 Ablation Study

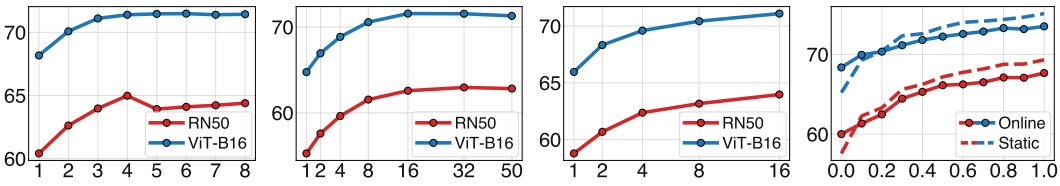
We conduct ablation studies to analyze why TTC works in the zero-shot classification setting. Specifically, we evaluate different divergence metrics, the contribution of each term in the aggregated verification score, and the sensitivity of TTC to top-k, memory size, and memory quality. In these studies, the average denotes the mean accuracy across 11 datasets, and detailed results are provided in Appendices E and F.5.

Divergence metrics. We compare three divergence metrics: l_2 distance, cosine similarity (Cos. Sim.), and nearest-subspace reconstruction error (NSE). When cosine similarity is used, we negate the similarity value so that lower values consistently indicate better compatibility in the aggregated verification score. As shown in Table 8, NSE achieves the best average accuracy for both ResNet50 and ViT-B/16. This confirms that reconstruction-based subspace divergence is well aligned with the candidate-verification principle of TTC.

Aggregated verification score. We analyze the contribution of each component in $E_y(\mathbf{x})$. Table 9 compares the anchor reconstruction term $d_{\mathbf{x}}(\mathcal{S}_y, \mathbf{x})$, the forward divergence shift $\delta_{y\to r}(\mathbf{x})$, and the reverse divergence shift $\delta_{r\to y}(\mathbf{x})$. Each component improves performance when used individually, while combining all three achieves the best accuracy. This result demonstrates that the three terms, absolute compatibility, candidate stability, and competing-hypothesis perturbation, provide complementary evidence for candidate verification.

Top-k. Figure 3a illustrates the effect of varying the number of candidate labels. Increasing k gives TTC more candidate hypotheses to verify, but also increases computation. Accuracy improves until $k = 3$, after which the gains become marginal.

Memory size. Figures 3b and 3c analyze the effect of the maximum per-class memory size m. Since some datasets contain fewer than 16 samples per class, Figure 3b reports the accuracy on



(a) Top-k (b) Memory Size (INet) (c) Memory Size (d) Memory Purity

Figure 3: Ablation studies on top-k and memory under the zero-shot classification setting.

ImageNet, while Figure 3c reports the average accuracy across 11 datasets. More memory provides a richer estimate of each class subspace but also increases reconstruction cost. TTC achieves substantial gains even with a small memory, and its performance saturates around $m = 16$.

Memory quality. We examine how noisy memory initialization affects TTC. This analysis matters in label-free settings because the memory is initialized with pseudo-labels produced by the frozen VLM, which may include misclassified features. TTC is robust to such noise because it does not make decisions based on individual memory entries. Instead, it verifies candidates by aggregating reconstruction-induced divergence shifts over class subspaces.

Figure 3d shows that TTC remains effective even with imperfect memory initialization. The dashed line denotes a static memory that is not updated after initialization and is intentionally kept fixed to preserve its initial purity throughout evaluation. In contrast, the solid line represents the setting in which memory is continuously updated during inference. When initial memory purity is low, online memory updates outperform static memory, suggesting that corrected test samples improve the memory structure. These results indicate that TTC does not require perfectly clean memory. Instead, a coarse and gradually refined class subspace is sufficient for candidate verification.

5 Conclusion

This paper introduced **Test-Time Correction (TTC)**, a candidate-verification framework for VLM test-time adaptation. Rather than adapting prediction-side statistics, TTC treats top-k candidate labels as hypotheses and verifies them by measuring reconstruction-induced divergence shifts after hypothetical insertion into class memories. This reframes test-time adaptation as a geometric verification problem, where a correct candidate should accommodate the test feature with minimal perturbation, whereas an incorrect candidate should more strongly disturb candidate subspace relations. TTC performs correction without updating model parameters or prompts and operates only over a small candidate set with compact class memories. Across zero-shot classification, domain generalization, few-shot classification, base-to-novel generalization, and cross-dataset evaluation, TTC consistently improves VLM predictions while maintaining strong computational efficiency. These results demonstrate that candidate verification provides an effective and practical alternative to conventional prediction-side adaptation.

TTC relies on the original VLM logits to construct its top-k candidate set and therefore cannot recover the correct label when the true class is excluded. Future work could explore adaptive candidate expansion, uncertainty-aware selection, or memory-guided retrieval beyond the initial predictions. Moreover, for unseen classes or short streams with sparse or empty candidate memories, TTC defaults to the initial prediction, limiting its potential gains.

Acknowledgments

This research was supported by the National Research Foundation of Korea (NRF), Electronics and Telecommunications Research Institute(ETRI), and Institute of Information & Communications Technology Planning & Evaluation (IITP), funded by the Korean government [26CS1100, Development of Proprietary Physical AI-based Small-scale Computers and Integrated Soft Suits], and the Korean government(MSIT) (RS-2026-25518808, RS-2026-25617480, and IITP-2026-RS-2023-00255968).

References

[1] Jameel Abdul Samadh, Mohammad Hanan Gani, Noor Hussein, Muhammad Uzair Khattak, Muhammad Muzammal Naseer, Fahad Shahbaz Khan, and Salman H Khan. Align your prompts: Test-time prompting with distribution alignment for zero-shot generalization. *Advances in Neural Information Processing Systems*, 2023.

[2] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 2022.

[3] Alexandr Andoni, Piotr Indyk, and Ilya Razenshteyn. Approximate nearest neighbor search in high dimensions. In *Proceedings of the International Congress of Mathematicians*, 2018.

[4] Sunil Arya, David M Mount, Nathan S Netanyahu, Ruth Silverman, and Angela Y Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM*, 1998.

[5] Ronen Basri, Tal Hassner, and Lihi Zelnik-Manor. Approximate nearest subspace search with applications to pattern recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2007.

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

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

[8] Chandler Davis. The rotation of eigenvectors by a perturbation. *Journal of Mathematical Analysis and Applications*, 1963.

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

[10] Matteo Farina, Gianni Franchi, Giovanni Iacca, Massimiliano Mancini, and Elisa Ricci. Frustratingly easy test-time adaptation of vision-language models. *Advances in Neural Information Processing Systems*, 2024.

[11] Chun-Mei Feng, Kai Yu, Yong Liu, Salman Khan, and Wangmeng Zuo. Diverse data augmentation with diffusions for effective test-time prompt tuning. In *IEEE/CVF International Conference on Computer Vision*, 2023.

[12] Clément Fuchs, Maxime Zanella, and Christophe De Vleeschouwer. Online gaussian test-time adaptation of vision-language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[13] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Introducing eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. In *IEEE International Geoscience and Remote Sensing Symposium*, 2018.

[14] 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 *IEEE/CVF International Conference on Computer Vision*, 2021.

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

[16] 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 *International Conference on Machine Learning*, 2021.

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

[18] Tosio Kato. *Perturbation theory for linear operators*, volume 132. Springer Science & Business Media, 2013.

[19] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *IEEE/CVF International Conference on Computer Vision Workshops*, 2013.

[20] Fei-Fei Li, Marco Andreeto, Marc'Aurelio Ranzato, and Pietro Perona. Caltech 101, 2022.

[21] Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. Align before fuse: Vision and language representation learning with momentum distillation. *Advances in Neural Information Processing Systems*, 2021.

[22] Zichen Liu, Hongbo Sun, Yuxin Peng, and Jiahuan Zhou. Dart: Dual-modal adaptive online prompting and knowledge retention for test-time adaptation. In *AAAI Conference on Artificial Intelligence*, 2024.

[23] Xiaosong Ma, Jie Zhang, Song Guo, and Wenchao Xu. Swapprompt: Test-time prompt adaptation for vision-language models. *Advances in Neural Information Processing Systems*, 2023.

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

[25] Imran Naseem, Roberto Togneri, and Mohammed Bennamoun. Linear regression for face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[26] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*, 2008.

[27] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2012.

[28] Sarah Pratt, Ian Covert, Rosanne Liu, and Ali Farhadi. What does a platypus look like? generating customized prompts for zero-shot image classification. In *IEEE/CVF International Conference on Computer Vision*, 2023.

[29] 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 *International Conference on Machine Learning*, 2021.

[30] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, 2019.

[31] Ashshak Sharifdeen, Muhammad Akhtar Munir, Sanoojan Baliah, Salman Khan, and Muhammad Haris Khan. O-tpt: Orthogonality constraints for calibrating test-time prompt tuning in vision-language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[32] 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. *Advances in Neural Information Processing Systems*, 2022.

[33] Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. Flava: A foundational language and vision alignment model. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

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

[35] Elaine Sui, Xiaohan Wang, and Serena Yeung-Levy. Just shift it: Test-time prototype shifting for zero-shot generalization with vision-language models. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, 2025.

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

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

[38] 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. *International Conference on Learning Representations*, 2024.

[39] Feng Yu, Teng Zhang, and Gilad Lerman. A subspace-constrained tyler's estimator and its applications to structure from motion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[40] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv*, 2022.

[41] Yi Yu, Tengyao Wang, and Richard J Samworth. A useful variant of the davis–kahan theorem for statisticians. *Biometrika*, 2015.

[42] Maxime Zanella and Ismail Ben Ayed. On the test-time zero-shot generalization of vision-language models: Do we really need prompt learning? In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[43] Ce Zhang, Simon Stepputtis, Katia Sycara, and Yaqi Xie. Dual prototype evolving for test-time generalization of vision-language models. *Advances in Neural Information Processing Systems*, 2024.

[44] Chenyu Zhang, Kunlun Xu, Zichen Liu, Yuxin Peng, and Jiahuan Zhou. Scap: Transductive test-time adaptation via supportive clique-based attribute prompting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[45] Jingyi Zhang, Jiaxing Huang, Xiaoqin Zhang, Ling Shao, and Shijian Lu. Historical test-time prompt tuning for vision foundation models. *Advances in Neural Information Processing Systems*, 2024.

[46] Shuai Zhao, Xiaohan Wang, Linchao Zhu, and Yi Yang. Test-time adaptation with clip reward for zero-shot generalization in vision-language models. *International Conference on Learning Representations*, 2024.

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

[48] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 2022.

[49] Lihua Zhou, Mao Ye, Shuaifeng Li, Nianxin Li, Xiatian Zhu, Lei Deng, Hongbin Liu, and Zhen Lei. Bayesian test-time adaptation for vision-language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

A Perturbation View of Divergence Shift

We provide a perturbation-based motivation for the divergence-shift criterion used in TTC. This section does not establish a strict classification guarantee; it explains why hypothetical insertion provides a meaningful candidate-verification signal. The argument proceeds in three steps: (i) a geometric intuition of how inserting a test feature perturbs a candidate subspace, (ii) a closed-form perturbation view relating this change to the residual of the test feature, and (iii) an explanation of how reconstruction-induced divergence shift operationalizes this stability principle.

A.1 Intuition

For each class y, let $\mathbf{B}_y \in \mathbb{R}^{m_y \times d}$ denote the class memory matrix and let $\mathcal{S}_y = \mathrm{span}(\mathbf{B}_y)$ be the class subspace induced by the memory features. Given a test feature $\mathbf{x}$, TTC verifies a candidate hypothesis $h_y : \mathbf{x} \mapsto y$ by hypothetically inserting $\mathbf{x}$ into $\mathbf{B}_y$. This yields the updated memory and the updated subspace as

$$\widetilde{\mathbf{B}}_y = [\mathbf{B}_y; \mathbf{x}], \qquad \widetilde{\mathcal{S}}_y = \mathrm{span}(\widetilde{\mathbf{B}}_y). \tag{9}$$

The change from $\mathcal{S}_y$ to $\widetilde{\mathcal{S}}_y$ depends on how much of $\mathbf{x}$ lies outside the original subspace $\mathcal{S}_y$. Let $\mathrm{Proj}_{\mathcal{S}_y}(\mathbf{x})$ denote the projection of $\mathbf{x}$ onto $\mathcal{S}_y$, and define the residual component as

$$\mathbf{x}_{\perp y} = \mathbf{x} - \mathrm{Proj}_{\mathcal{S}_y}(\mathbf{x}). \tag{10}$$

If $\mathbf{x}$ truly belongs to class y, then it should be geometrically compatible with $\mathcal{S}_y$, and the residual norm $\|\mathbf{x}_{\perp y}\|$ is expected to be small. In this case, inserting $\mathbf{x}$ into $\mathbf{B}_y$ only mildly changes the induced subspace. In contrast, if $\mathbf{x}$ belongs to another class, the residual component with respect to $\mathcal{S}_y$ is expected to be larger, and the updated subspace $\widetilde{\mathcal{S}}_y$ deviates more strongly from the original subspace.

A.2 Subspace-Perturbation View

This intuition aligns with classical subspace perturbation results [8, 41], which bound subspace change by the perturbation size relative to a spectral gap. For exact spans, however, the principal angles between $\mathcal{S}_y$ and $\widetilde{\mathcal{S}}_y$ are not informative: $\mathcal{S}_y \subseteq \widetilde{\mathcal{S}}_y$ holds by construction, and the orthogonal projector onto the span changes by a unit-norm rank-one projector whenever $\mathbf{x} \notin \mathcal{S}_y$, regardless of how small $\|\mathbf{x}_{\perp y}\|$ is.

The ridge-regularized projection matrix $\mathbf{P}_y$ of Section 3.4 removes this discontinuity. Let $\mathbf{r}_y = (\mathbf{I} - \mathbf{P}_y)\mathbf{x}$ be the ridge residual of $\mathbf{x}$, so that $\|\mathbf{r}_y\|_2 = d_{\mathbf{x}}(\mathcal{S}_y, \mathbf{x})$, and let $\widetilde{\mathbf{P}}_y$ be the projection matrix of $\widetilde{\mathbf{B}}_y$. By the Sherman–Morrison formula, the hypothetical insertion changes the projection matrix by a rank-one term along the residual,

$$\widetilde{\mathbf{P}}_y - \mathbf{P}_y \;=\; \frac{\mathbf{r}_y \mathbf{r}_y^\top}{\lambda + \mathbf{x}^\top \mathbf{r}_y}, \tag{11}$$

from which

$$\big\|\widetilde{\mathbf{P}}_y - \mathbf{P}_y\big\|_2 \;=\; \frac{\|\mathbf{r}_y\|_2^2}{\lambda + \mathbf{x}^\top \mathbf{r}_y} \;\le\; \min\Big\{1,\; \tfrac{1}{\lambda}\, d_{\mathbf{x}}(\mathcal{S}_y, \mathbf{x})^2\Big\}. \tag{12}$$

Thus, an in-class insertion with a small residual induces only a small change, whereas an out-of-class insertion tends to perturb the candidate subspace more strongly. As $\lambda \to 0$, $\mathbf{r}_y$ approaches $\mathbf{x}_{\perp y}$, and the change approaches the unit-norm projector above. The ridge term is therefore essential for a graded response.

This view also explains the special case in which $\mathbf{x}$ already lies in $\mathcal{S}_y$. If $\mathbf{x} \in \mathcal{S}_y$, inserting $\mathbf{x}$ does not enlarge the span of the memory matrix, so $\widetilde{\mathcal{S}}_y = \mathcal{S}_y$. The ridge residual is then at most $\lambda/(\sigma_{\min}^2(\mathbf{B}_y) + \lambda) \cdot \|\mathbf{x}\|_2$, where $\sigma_{\min}(\mathbf{B}_y)$ is the smallest nonzero singular value of $\mathbf{B}_y$, so the induced change is small whenever the class memory is well conditioned.

A.3 Connection to Divergence Shift

TTC does not directly use the principal-angle deviation as a verification score. Instead, it uses reconstruction-induced divergence shift as an operational proxy for subspace stability. For candidate

labels y and r, the divergence shift is defined as

$$\delta_{y\to r}(\mathbf{x}) = d_{\mathcal{S}}(\mathcal{S}_y, \mathcal{S}_r) - d_{\mathcal{S}}(\widetilde{\mathcal{S}}_y, \mathcal{S}_r). \tag{13}$$

This quantity compares the reconstruction-based relation between candidate y and reference candidate r before and after inserting $\mathbf{x}$ into the memory of class y.

If $\mathbf{x}$ is compatible with class y, the hypothetical insertion only mildly changes $\mathcal{S}_y$. Consequently, the reconstruction relation between $\mathcal{S}_y$ and other candidate subspaces is expected to remain stable, producing a small divergence shift. If $\mathbf{x}$ is incompatible with class y, the insertion introduces a larger out-of-subspace component and changes the induced subspace more strongly. This, in turn, is likely to alter the reconstruction relation to other candidate subspaces, producing a larger divergence shift. Applying the closed form to each row of $\mathbf{B}_r$ makes this precise:

$$\left|\delta_{y\to r}(\mathbf{x})\right| \;\le\; \frac{d_{\mathbf{x}}(\mathcal{S}_y, \mathbf{x})}{\lambda + \mathbf{x}^\top \mathbf{r}_y} \cdot \frac{1}{m_r} \sum_{j=1}^{m_r} \left|\mathbf{r}_y^\top \mathbf{b}_{r,j}\right|, \tag{14}$$

so a large shift requires a residual that is both sizable and aligned with the memory of class r.

Divergence shift captures the subspace-stability intuition through TTC's reconstruction operator. TTC measures how hypothetical insertion changes reconstruction behavior among candidate subspaces without explicitly computing principal angles or orthonormal bases. The criterion thus aligns directly with the implemented nearest-subspace reconstruction mechanism.

A.4 Interpretation

The perturbation view provides a geometric explanation for TTC's candidate-verification principle. A correct candidate should accommodate the test feature with a small residual-induced perturbation, while an incorrect candidate should induce a larger change in its subspace and reconstruction relations to other candidates. Selecting the candidate with the smallest aggregated divergence shift therefore favors the hypothesis whose class subspace most stably accommodates the test feature.

This interpretation also distinguishes TTC from conventional nearest-subspace classification. A nearest-subspace classifier uses reconstruction error as an absolute measure of proximity between a query and a class subspace. In contrast, TTC uses reconstruction as a perturbation probe to assess how hypothetical insertion of the test feature changes the candidate subspace and its relations to competing candidate subspaces. This perturbation view underlies TTC's candidate-verification mechanism.

B Experimental Details

For performance evaluation on the domain generalization task, we utilize ImageNet [9], ImageNet-A [15], ImageNet-R [14], ImageNet-Sketch [36], and ImageNet-V2 [30]. Furthermore, for zero-shot classification, few-shot classification, base-to-novel generalization, and cross-dataset evaluation tasks, we utilize 11 downstream datasets: Caltech [20], Describable Textures Dataset (DTD) [7], 102 Category Flower Dataset (Flower) [26], ImageNet (INet) [9], UCF101 (UCF) [34], FGVC-Aircraft Benchmark (Aircraft) [24], EuroSAT (ESAT) [13], Food101 (Food) [6], SUN397 (SUN) [37], StanfordCars (Cars) [19], and Oxford-IIIT Pet (Pets) [27]. Table 10 shows the number of samples and classes used in experiments.

All experiments are conducted using an *Intel(R) Xeon(R) Platinum 8270CL CPU @ 2.70GHz* and an *NVIDIA RTX A5000* GPU with PyTorch 2.6.0, CUDA 12.4, and Python 3.12. To thoroughly demonstrate the effectiveness of our method across diverse tasks, we conduct experiments using CLIP [29]. We categorize the experimental tasks by their data requirements. For domain generalization and zero-shot classification, no training samples are used. For few-shot classification, we evaluate performance under multiple data regimes, using 1, 2, 4, 8, and 16 training samples per class. For base-to-novel generalization and cross-dataset evaluation, we use a fixed setting of 16 training samples per class. When using few-shot training samples, we strictly follow the standard protocol adopted in prior work [32, 48]. In this setting, we first train prompts on few-shot samples, then directly evaluate the resulting tuned prompts on classification tasks. For consistency and fair comparison, we use fixed hyperparameters across all experiments: top-$k = 3$ prediction candidates and a memory size of $m = 16$. All reported results are averaged over three random seeds, and the test

Table 10: Detailed dataset information used in experiments. The number of training samples matches that used when training CoOp for few-shot, base-to-novel, and cross-dataset evaluation.

Dataset	Train Samples	Test Samples	Classes
ImageNet	16000	50000	1000
ImageNet-A	–	7500	200
ImageNet-R	–	30000	200
ImageNet-Sketch	–	50889	1000
ImageNet-V2	–	10000	1000
Caltech101	1600	2465	100
DTD	752	1692	47
EuroSAT	160	8100	10
Aircraft	1600	3333	100
Flowers102	1632	2463	102
Food101	1616	30300	101
Oxford Pets	592	3669	37
Stanford Cars	3136	8041	196
SUN397	6352	19850	397
UCF101	1616	3783	101

sample order is always shuffled. Baseline results are taken from the original papers where available. The baselines that we run use their official implementations and published configurations with the same backbones, datasets, and splits as TTC.

C Top-k Candidate Set Hit Rate and Dynamic Top-k

Table 11 shows that dataset-level conditional accuracy of TTC is higher than the initial top-1 candidate on every dataset except ImageNet-A with ResNet50.

Table 11: Per-dataset top-3 hit rate (Hit@3) and $A_{\text{in}} = P(\text{correct} \mid \text{GT} \in \text{top-3})$ of the initial top-1 candidate (Init.) and of Ours (TTC). FS is averaged over 1 to 16 shots and B2N over the base and novel splits, with CoOp-tuned ViT-B/16 prompts, where Init. is the CoOp prediction.

	ZS/DG (ResNet50)			ZS/DG (ViT-B/16)			FS			B2N			Cross		
Dataset	Hit@3	Init.	TTC	Hit@3	Init.	TTC	Hit@3	Init.	TTC	Hit@3	Init.	TTC	Hit@3	Init.	TTC
Caltech	98.17	90.70	**92.77**	99.23	94.44	**94.97**	99.29	94.63	**95.18**	98.92	94.55	**95.40**	99.11	93.00	**94.72**
DTD	73.82	64.21	**72.78**	74.53	70.74	**74.46**	78.12	75.52	**79.58**	79.92	76.90	**80.92**	58.29	71.07	**74.38**
ESAT	75.52	49.45	**63.45**	78.78	68.55	**77.20**	91.93	76.92	**85.25**	92.56	78.80	**82.78**	75.15	64.27	**75.02**
Aircraft	38.97	50.04	**54.12**	51.31	53.68	**55.38**	59.37	56.30	**59.70**	57.16	58.08	**60.44**	37.76	53.04	**53.31**
Flower	81.97	80.39	**87.57**	85.02	85.72	**90.11**	97.57	92.87	**96.21**	88.50	88.82	**91.16**	80.77	83.11	**87.15**
Food	91.10	84.58	**85.47**	95.91	89.50	**90.31**	94.27	87.96	**90.19**	95.34	89.83	**92.29**	95.87	89.50	**90.54**
INet	81.49	75.25	**76.80**	88.08	79.15	**81.25**	87.69	78.98	**81.56**	88.99	80.99	**83.65**	–	–	–
Pets	97.82	86.90	**88.41**	98.99	91.24	**92.62**	99.14	91.94	**92.51**	99.61	94.70	**95.20**	96.29	93.26	**93.60**
Cars	83.27	68.91	**72.92**	90.96	73.19	**76.95**	93.67	78.26	**82.27**	91.66	75.11	**79.93**	89.29	72.65	**76.71**
SUN	85.23	73.88	**76.88**	88.55	76.89	**80.66**	89.22	77.98	**81.53**	89.17	80.66	**84.88**	85.30	75.35	**79.30**
UCF	82.58	74.55	**78.75**	87.92	77.90	**85.60**	91.33	84.12	**88.04**	87.02	81.88	**85.61**	85.19	79.67	**84.83**
Avg.	80.90	72.62	**77.26**	85.39	78.27	**81.77**	89.24	81.41	**84.73**	88.08	81.85	**84.75**	80.30	77.49	**80.96**
INet-A	45.85	**51.99**	51.12	74.13	68.62	**69.95**	–	–	–	–	–	–	–	–	–
INet-R	77.89	78.56	**78.84**	90.35	86.37	**87.04**	–	–	–	–	–	–	–	–	–
INet-Sketch	55.84	65.00	**69.58**	69.79	70.77	**75.01**	–	–	–	–	–	–	–	–	–
INet-V2	75.76	71.45	**72.40**	82.85	76.28	**77.47**	–	–	–	–	–	–	–	–	–
DG Avg.	67.37	68.45	**69.75**	81.04	76.24	**78.14**	–	–	–	–	–	–	–	–	–

Dynamic top-k. We examine an entropy-guided dynamic k. For each sample, we set k by linear interpolation between $k_{\min} = 3$ and $k_{\max}$ based on the Shannon entropy of the softmax distribution. Confident distributions use k near $k_{\min}$, while uncertain distributions use k closer to $k_{\max}$. As shown in Table 12, this approach yields a small, consistent accuracy gain of 0.18–0.24 pp over fixed $k = 3$, at the cost of 20–43% lower throughput. Fixed $k = 3$ therefore remains preferable when throughput is the priority, while dynamic k offers a modest accuracy gain when the additional cost is acceptable.

Table 12: Fixed and entropy-guided dynamic top-k with ViT-B/16, averaged over the 11 zero-shot classification datasets.

Variant	Mean k	Acc.	Thr. (img/s)
Fixed $k = 3$ (default)	3.00	71.10	2049.0 (100%)
Entropy, $k_{\max} = 5$	3.47	71.28	1636 (80%)
Entropy, $k_{\max} = 8$	4.30	71.34	1174 (57%)

D Efficiency Analysis

Verification cost. Each test sample accesses only its $k = 3$ candidate memories with 48 vectors. The per-sample verification cost is $O(k^2m^2d)$, about 5–6 MFLOPs for ViT-B/16 ($d = 512$), three orders of magnitude below one encoder forward pass ($\sim$17 GFLOPs), and independent of $|\mathcal{Y}|$; only the base logit computation scales with the number of classes. The ridge projector inverts only the $m \times m$ Gram matrix, which is cached per class and refreshed only when the class memory changes, and a hypothetical insertion of $\mathbf{x}$ uses a cached block-inverse update at $O(m^2 + md)$, with no new inversion per candidate. Table 13 changes only the divergence metric. Cosine similarity yields only a 1.7$\times$ adaptation-stage speedup at the cost of 0.9–1.2 pp on ImageNet, and TTC with NSE remains the fastest method in Table 5. The cached projection is therefore not the practical bottleneck.

Table 13: Accuracy and throughput with regard to the divergence metric on ImageNet zero-shot task.

	ResNet50		ViT-B/16	
Metric	Acc. (%)↑	Thr. (img/s)↑	Acc. (%)↑	Thr. (img/s)↑
Cos. Sim.	61.72	1890.1	70.37	3578.8
NSE (default)	62.58	1103.4	71.57	2049.0

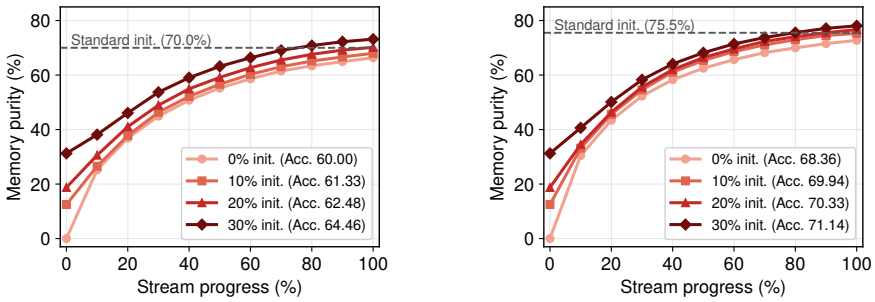
Full-pipeline efficiency. In Table 5, all methods, including TDA, process the same single-view, pre-extracted feature stream to isolate adaptation cost under identical inputs. We estimate full-pipeline time on the 50,000 ImageNet test images by adding encoding time, based on measured encoder throughput, to adaptation time. Encoder throughput is 2,136 img/s for ResNet50 and 1,160 img/s for ViT-B/16. With 63 augmented views and the original view, TDA takes 1,804 s and 3,031 s, respectively, whereas TTC takes 69 s and 68 s. TTC is thus approximately 26$\times$ and 45$\times$ faster than the native protocol of TDA.

Memory footprint. For ViT-B/16 with $d = 512$, the complete ImageNet memory bank requires $|\mathcal{Y}| \times m \times d \times 4\,\mathrm{B} = 1000 \times 16 \times 512 \times 4\,\mathrm{B}$, or approximately 32.8 MB, plus approximately 1 MB for the cached 16×16 Gram inverses. The persistent memory bank and cached inverses reside in CPU memory, while temporary computation is included in the peak GPU memory reported in Table 5. Storage scales linearly with the number of allocated class memories. For 10K open-vocabulary classes, approximately 340 MB of CPU memory is required, which can be reduced with a smaller m or lower-precision storage. Reducing m from 16 to 8 halves the memory bank size at an accuracy cost of 0.8 pp for ResNet50 and 0.7 pp for ViT-B/16 on zero-shot classification.

E Memory Robustness

Recovery from contaminated memory. Figure 3d reports final accuracy under noisy memory initialization but does not show whether memory quality recovers during the stream. We therefore conduct a controlled stress test with initial memory purity ranging from 0% to 30%. Memory purity is the fraction of stored features whose ground-truth class matches the class memory they are stored in. As in Figure 3d, each class memory is initialized with $m = 16$ features from the labeled 16-shot training split. The stated fraction comes from the memory's own class, while the remainder comes from other classes sampled uniformly. We record memory purity at 10% intervals along the stream and macro-average the results across the 11 zero-shot datasets for ResNet50 and ViT-B/16, as shown in Figure 4. Even when all initial entries are incorrect, memory purity increases steadily from 0% to

66% for ResNet50 and 73% for ViT-B/16. TTC does not collapse, retaining accuracies of 60.00% and 68.36%, respectively. Final accuracy increases monotonically with initial purity. Because the controlled memories are initialized with labeled training features and their incorrect entries come from uniformly random classes, this setting isolates the effect of purity and is not directly comparable to the standard initialization, whose measured initial memory purity is 70.00%/75.52%.



(a) ResNet50 (b) ViT-B/16

Figure 4: Memory purity along the stream under contaminated memory initialization, macro-averaged over the 11 zero-shot datasets. The legend gives the final accuracy of each setting, and the dashed line marks the initial purity of the standard initialization.

Three mechanisms reduce sensitivity to contaminated entries. First, confidence-filtered initialization fills each class memory with high-confidence predictions, yielding initial purities of 70.00% and 75.52%. Second, within-class eviction replaces the lowest-confidence entry, removing the model's least reliable entries, consistent with the observed purity recovery. Third, ridge-regularized aggregation uses all $m = 16$ entries per class, so no decision depends on a single stored feature.

Class-imbalanced test streams. Memory is partitioned by class, with eviction restricted to each class, so samples assigned to frequent classes cannot displace rare-class entries. Imbalance can still slow the filling of rare-class memories. We evaluate this effect on long-tailed ImageNet streams sampled without replacement according to a Zipf distribution, yielding head:tail frequency ratios of approximately 3:1 and 6:1. We use ResNet50 and ViT-B/16 with three seeds per ratio, define the tail group as the bottom 20% of classes by realized frequency, and evaluate all training-free baselines, as reported in Table 14. With ViT-B/16, TTC achieves the largest tail-class gains, improving over CLIP by 2.45 and 1.97 pp, respectively, compared with at most 0.46 pp for any baseline. OGA loses 1.43 pp under the 6:1 imbalance. This pattern holds with ResNet50, with gains of 1.41 and 2.15 pp. Overall ViT-B/16 accuracy exceeds CLIP by 1.30 pp on the 3:1 stream but falls below it by 0.08 pp on the 6:1 stream, where performance on the remaining classes offsets the tail-class gain.

Table 14: Accuracy on long-tailed ImageNet streams with ResNet50 and ViT-B/16.

Backbone	Stream	Group	CLIP	TDA	OGA	BCA	TTC
ResNet50	∼21K (3:1)	Overall	61.20	61.82	60.23	61.27	**61.94**
		Tail 20%	60.81	61.55	60.49	61.18	**62.22**
	∼12K (6:1)	Overall	61.26	**61.58**	59.76	61.36	61.21
		Tail 20%	60.32	60.29	58.19	60.80	**62.47**
ViT-B/16	∼21K (3:1)	Overall	69.66	70.12	69.54	69.76	**70.96**
		Tail 20%	69.66	69.97	70.12	69.87	**72.11**
	∼12K (6:1)	Overall	69.74	**70.08**	68.99	69.84	69.66
		Tail 20%	68.92	69.19	67.49	69.17	**70.89**

F Additional Experimental Results

This section provides detailed results that complement Section 4. We report the per-dataset few-shot results, zero-shot classification and domain generalization with additional CLIP backbones, the base- and novel-class accuracies of base-to-novel generalization, the per-setting success and failure analysis, and the per-dataset ablations of the divergence metric and the aggregated verification score. Across these results, the gains of TTC are not confined to a particular backbone, dataset, or evaluation split.

F.1 Few-shot Classification

Figure 5 presents additional few-shot classification results on 11 downstream datasets. TTC outperforms state-of-the-art approaches on most datasets, and its advantage persists with more shots.

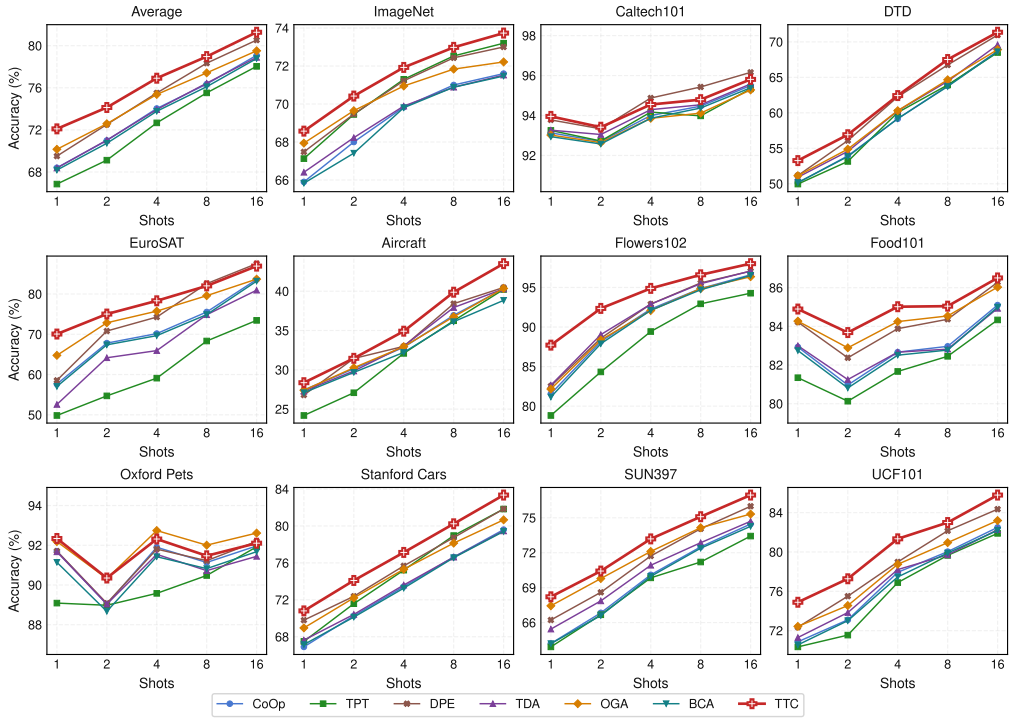


Figure 5: Additional experimental results of the few-shot classification based on CoOp.

F.2 Additional Backbones

We also experiment with lighter and larger backbones, ViT-B/32 and ViT-L/14. Table 15 reports the zero-shot classification results, and Table 16 reports the results of the domain generalization task.

Table 15: Experimental results of the zero-shot classification task on CLIP ViT-B/32 and ViT-L/14.

Model	Caltech	DTD	ESAT	Aircraft	Flower	Food	INet	Pets	Cars	SUN	UCF	Avg.
ViT-B/32 *(ICML'21)*	92.54	49.47	51.23	20.61	66.99	80.23	64.65	86.56	61.65	65.82	64.31	64.01
DPE *(NeurIPS'24)*	93.39	52.36	53.05	21.09	69.59	80.73	65.44	87.27	62.60	67.59	66.38	65.41
TDA *(CVPR'24)*	92.82	51.30	54.32	20.64	69.35	80.45	65.27	86.84	61.71	66.93	65.13	64.98
OGA *(CVPR'25)*	91.56	50.95	53.75	21.06	68.90	79.98	64.60	87.65	62.77	66.78	65.58	64.87
BCA *(CVPR'25)*	92.33	50.12	50.86	21.09	67.11	80.05	64.76	86.89	61.72	66.00	64.66	64.15
TTC (Ours)	**93.79**	**52.96**	**56.74**	**21.63**	**71.99**	**81.17**	**66.00**	**88.39**	**63.60**	**68.64**	**67.51**	**66.58**
ViT-L/14 *(ICML'21)*	96.55	61.88	64.41	35.76	79.37	91.07	76.50	93.51	78.55	71.96	74.23	74.89
DPE *(NeurIPS'24)*	**97.04**	64.36	67.20	**38.13**	81.73	91.34	77.15	93.57	80.28	73.62	76.34	76.43
TDA *(CVPR'24)*	96.31	63.36	67.25	37.47	81.97	91.04	76.88	93.24	78.80	72.74	75.50	75.87
OGA *(CVPR'25)*	96.63	63.06	**68.96**	36.99	82.14	90.81	76.84	94.09	80.87	73.19	76.87	76.40
BCA *(CVPR'25)*	95.94	62.65	62.27	36.60	79.09	90.97	76.44	92.91	79.04	71.98	74.60	74.77
TTC (Ours)	96.84	**65.25**	66.33	37.50	**83.96**	**91.47**	**78.37**	**94.82**	**81.35**	**74.79**	**78.14**	**77.16**

Table 16: Experimental results of the domain generalization task on CLIP ViT-B/32 and ViT-L/14. 'Dist.' denotes the average accuracy of distribution-shift datasets.

Model	# of Aug.	INet	A	R	Ske.	V2	Dist.	Avg.
ViT-B/32 *(ICML'21)*	0	64.65	31.68	69.57	43.04	57.06	50.34	53.20
OGA *(CVPR'25)*	0	64.60	32.12	69.73	44.19	56.53	50.64	53.43
TTC (Ours)	0	66.00	32.15	70.20	45.70	57.59	51.41	54.33
DPE *(NeurIPS'24)*	63	65.44	**45.89**	**74.82**	48.52	60.46	**57.42**	**59.03**
TDA *(CVPR'24)*	63	65.27	45.71	74.21	47.58	**60.47**	56.99	58.65
BCA *(CVPR'25)*	63	64.76	45.33	74.53	47.05	60.12	56.76	58.36
TTC (Ours)	10	**66.00**	45.77	73.37	**48.71**	60.26	57.03	58.82
ViT-L/14 *(ICML'21)*	0	76.50	71.91	88.15	60.37	70.92	72.84	73.57
OGA *(CVPR'25)*	0	76.84	72.43	88.59	61.45	70.42	73.22	73.95
TTC (Ours)	0	78.37	73.65	88.62	62.81	71.30	74.10	74.95
DPE *(NeurIPS'24)*	63	77.15	80.52	**90.73**	63.98	72.26	76.87	76.93
TDA *(CVPR'24)*	63	76.88	80.80	90.65	63.37	72.14	76.74	76.77
BCA *(CVPR'25)*	63	76.44	80.59	90.55	62.83	72.17	76.53	76.51
TTC (Ours)	10	**78.37**	**81.68**	90.34	**64.42**	**72.38**	**77.21**	**77.44**

F.3 Base-to-novel Generalization

Table 17 reports the full base-to-novel generalization results. TTC consistently improves both base-class and novel-class accuracy. Compared with CoOp, it increases the average base and novel accuracies by 1.60 pp and 3.31 pp, respectively. The gain is larger on novel classes, which indicates that candidate verification mitigates the bias of prompt-tuned models toward base classes. TTC also achieves the best average base, novel, and harmonic-mean accuracies among all compared methods.

Table 17: Experimental results of the base-to-novel generalization task using a CoOp-tuned ViT-B/16. 'H.M.' denotes the harmonic mean of base- and novel-class accuracies.

Model	Sets	Caltech	DTD	ESAT	Aircraft	Flower	Food	INet	Pets	Cars	SUN	UCF	Avg.
CoOp	Base	97.80	79.83	91.70	41.30	97.83	88.53	76.40	93.57	78.13	80.73	84.30	82.74
	Novel	89.33	45.33	56.03	25.63	61.47	82.80	67.87	95.10	60.13	63.60	59.23	64.23
	H.M.	93.36	57.83	69.55	31.57	75.45	85.57	71.88	94.33	67.93	71.13	69.48	72.32
+ TPT	Base	98.28	80.44	80.26	40.02	96.20	85.78	77.88	93.51	80.38	79.16	85.33	81.57
	Novel	89.56	45.25	48.70	26.41	61.21	83.20	69.88	94.39	60.56	65.21	59.96	64.03
	H.M.	93.71	57.92	60.62	31.82	74.81	84.47	73.67	93.95	69.07	71.51	70.43	71.74
+ DPE	Base	98.56	82.37	92.91	39.80	98.23	89.51	77.57	93.85	80.72	81.46	86.13	83.74
	Novel	90.87	46.94	48.88	26.11	62.36	84.28	69.86	94.89	62.26	65.45	61.15	64.82
	H.M.	94.56	59.80	64.06	31.54	76.29	86.81	73.51	94.37	70.30	72.58	71.52	73.08
+ TDA	Base	98.00	80.71	89.79	40.80	98.23	88.47	76.29	93.35	78.20	80.86	83.90	82.60
	Novel	90.90	46.38	52.80	26.17	62.77	83.29	68.32	94.69	62.36	65.36	61.13	64.93
	H.M.	94.32	58.91	66.50	31.89	76.59	85.80	72.09	94.02	69.39	72.29	70.73	72.70
+ OGA	Base	97.37	79.40	91.36	40.90	97.78	89.65	76.99	93.78	78.42	81.51	85.52	82.97
	Novel	89.26	47.14	66.17	26.15	62.96	86.39	69.99	96.01	61.62	67.93	62.29	66.90
	H.M.	93.14	59.16	76.75	31.91	76.60	87.99	73.33	94.88	69.01	74.11	72.08	74.07
+ BCA	Base	97.87	79.90	91.44	39.56	97.75	88.42	76.24	93.39	78.13	80.61	83.94	82.48
	Novel	89.23	45.21	53.90	25.49	61.32	82.69	67.78	94.59	60.26	63.65	59.22	63.94
	H.M.	93.33	57.74	67.75	30.93	75.32	85.46	71.76	93.99	68.01	71.11	69.36	72.04
+ TTC	Base	98.26	82.18	93.68	43.72	98.20	89.57	78.08	93.64	80.41	82.76	87.21	**84.34**
	Novel	90.54	48.99	61.15	26.17	64.80	86.40	70.87	96.03	66.44	68.83	62.66	**67.54**
	H.M.	94.23	61.38	73.89	32.72	78.02	87.95	74.30	94.82	72.71	75.15	72.87	**75.01**

F.4 Success and Failure Study

Table 18 reports success and failure cases for each setting. Across all settings, successes account for 78.14–84.75% of the hit set, while overturning a correct initial prediction is the rarest case, occurring in only 2.39–4.46%. Both failure cases have smaller median margins than successes. In particular, the median margin of overturning failures is 5.1–9.0× smaller than that of successes, indicating that TTC overturns correct predictions mainly when competing candidates are nearly tied.

Table 18: Success and failure cases of TTC within the hit set for each setting with ViT-B/16.

	Success (correct)		Failure (right → wrong)		Failure (wrong → wrong)	
Setting	%	Margin	%	Margin	%	Margin
ZS	81.77	0.414	3.78	0.075	14.45	0.122
DG	78.14	0.399	3.57	0.062	18.29	0.139
FS	84.73	0.457	2.45	0.051	12.82	0.118
B2N	84.75	0.492	2.39	0.064	12.86	0.141
Cross	80.96	0.405	4.46	0.080	14.58	0.116
Avg.	82.07	0.434	3.33	0.067	14.60	0.127

F.5 Divergence Metric Functions and Aggregated Verification Score

Table 19 reports the per-dataset zero-shot results for different divergence metrics. NSE achieves the best average accuracy with both backbones and outperforms cosine similarity on all 11 datasets with ResNet50 and on 10 of 11 datasets with ViT-B/16. In contrast, l_2 distance falls below the initial CLIP prediction on average, showing that the choice of divergence is critical for verification.

Table 19: Per-dataset results of the ablation study of different divergence metrics.

	Metric	Caltech	DTD	ESAT	Aircraft	Flower	Food	INet	Pets	Cars	SUN	UCF	Avg.
ResNet50	–	83.68	41.68	27.69	17.40	63.14	74.95	59.90	80.79	58.33	59.53	55.74	56.62
	l_2 dist.	72.70	38.24	27.60	14.58	51.93	72.23	52.68	81.00	48.90	50.16	40.10	50.01
	Cos. Sim.	90.63	51.65	35.63	20.40	68.66	77.17	61.72	85.58	57.94	64.15	63.86	61.58
	NSE	91.08	53.72	47.91	21.09	71.78	77.86	62.58	86.48	60.73	65.53	65.03	63.98
ViT-B/16	–	90.89	44.87	47.01	24.40	67.65	85.35	68.50	86.20	67.60	64.65	64.36	64.68
	l_2 dist.	85.11	42.08	32.75	22.32	55.26	83.79	62.12	87.11	58.09	55.11	49.83	57.60
	Cos. Sim.	94.56	55.08	56.69	28.29	75.96	86.06	70.37	91.11	67.14	69.50	70.98	69.61
	NSE	94.24	55.50	60.81	28.41	76.61	86.62	71.57	91.69	69.99	71.43	75.26	71.10

Table 20 reports the per-dataset zero-shot results for aggregated verification score variants. Every pair of components outperforms each of its components, and combining all three achieves the best average accuracy with both backbones, confirming that the terms provide complementary evidence.

Table 20: Per-dataset results of the ablation study of the three components of the aggregated verification score.

	$d_{\mathbf{x}}$	$\delta_{y\to r}$	$\delta_{r\to y}$	Caltech	DTD	ESAT	Aircraft	Flower	Food	INet	Pets	Cars	SUN	UCF	Avg.
ResNet50	✓			86.21	52.30	41.73	20.49	66.91	72.66	59.41	73.43	56.83	62.60	62.57	59.56
		✓		90.10	49.53	39.57	20.10	68.45	77.65	62.35	85.88	58.86	64.48	63.23	61.84
			✓	91.03	51.30	41.33	20.55	69.31	77.79	62.55	86.26	59.59	64.95	64.26	62.63
	✓	✓		90.99	52.42	47.59	21.09	70.81	77.90	62.81	86.05	60.12	65.47	64.31	63.60
	✓		✓	90.87	53.01	47.46	20.82	71.13	77.89	62.83	86.07	60.34	65.54	65.11	63.73
		✓	✓	91.12	52.48	42.79	21.27	70.20	77.98	62.80	86.64	59.97	65.51	65.00	63.25
	✓	✓	✓	91.08	53.72	47.91	21.09	71.78	77.86	62.58	86.48	60.73	65.53	65.03	63.98
ViT-B/16	✓			89.37	53.25	56.56	28.35	72.11	83.14	68.51	81.00	67.08	69.00	72.93	67.39
		✓		94.32	54.79	55.86	28.95	75.11	86.32	70.80	91.28	68.35	69.66	71.58	69.73
			✓	94.40	55.50	56.42	28.98	76.57	86.58	71.16	91.58	68.87	70.20	72.93	70.29
	✓	✓		94.48	54.73	61.77	28.44	75.72	86.76	71.64	91.33	69.47	71.13	73.91	70.85
	✓		✓	94.52	55.20	60.67	28.68	76.21	86.72	71.66	91.63	69.84	71.36	74.99	71.04
		✓	✓	94.44	54.79	58.33	29.34	76.98	86.65	71.46	91.77	69.64	70.66	73.86	70.72
	✓	✓	✓	94.24	55.50	60.81	28.41	76.61	86.62	71.57	91.69	69.99	71.43	75.26	71.10

G Confidence Intervals for TTC Results

We report 95% confidence intervals for TTC results across three random seeds in Table 21. Since the ImageNet distribution-shift datasets are used only for domain generalization, we report their confidence intervals only in the 'ZS/DG' column.

Table 21: 95% confidence intervals of our experimental results over three random seeds. ZS/DG, FS, B2N, and Cross denote zero-shot classification and domain generalization, few-shot classification, base-to-novel generalization, and cross-dataset evaluation, respectively.

	ZS/DG		FS					B2N			Cross
Dataset	ResNet50	ViT-B/16	1	2	4	8	16	Base	Novel	H.M.	ViT-B/16
Caltech	0.231	0.080	0.256	0.672	0.842	0.417	0.191	0.527	2.428	1.547	0.138
DTD	0.677	0.201	0.470	2.140	1.266	0.278	0.536	2.108	2.714	2.710	1.378
ESAT	0.500	3.164	2.724	3.546	1.102	3.924	1.167	0.880	7.051	5.303	1.910
Aircraft	0.347	0.278	1.842	0.641	2.076	0.706	0.711	0.745	2.652	2.298	1.588
Flower	0.306	0.096	1.255	0.351	1.044	0.253	0.046	0.598	5.018	3.721	2.421
Food	0.138	0.098	0.504	0.845	0.111	0.308	0.150	0.206	2.267	1.267	0.250
INet	0.140	0.138	0.301	0.267	0.115	0.140	0.261	0.147	0.602	0.282	–
Pets	0.252	0.467	0.719	0.294	1.253	1.011	0.526	0.526	0.642	0.495	0.950
Cars	0.418	0.189	0.612	0.693	0.974	0.508	0.559	0.374	4.935	2.995	0.437
SUN	0.149	0.124	0.328	0.505	0.535	0.118	0.194	0.125	1.970	1.207	0.629
UCF	1.115	1.009	0.862	0.765	0.467	0.515	0.096	0.372	4.855	3.163	3.028
INet-A	0.069	0.088	–	–	–	–	–	–	–	–	–
INet-R	0.053	0.081	–	–	–	–	–	–	–	–	–
INet-Sketch	0.079	0.091	–	–	–	–	–	–	–	–	–
INet-V2	0.028	0.205	–	–	–	–	–	–	–	–	–

We also conduct a paired analysis where all methods share the same three stream permutations. Table 22 reports the mean paired difference (Gap) with its 95% confidence interval. Every Gap interval lies entirely above zero, and the Baseline CI is at most ± 0.74, well below the gaps.

Table 22: Paired accuracy differences between TTC and each baseline on the same three stream permutations. We report per-seed differences, their mean with the 95% confidence interval (Gap), and each baseline's own 95% accuracy confidence interval across seeds (Baseline CI).

		Per-seed difference				
Setting	Baseline	Seed 1	Seed 2	Seed 3	Gap $\pm$ 95% CI	Baseline CI
ZS Avg. (ResNet50)	TDA	$+2.19$	$+1.87$	$+2.29$	$+2.12 \pm 0.55$	± 0.17
	OGA	$+2.44$	$+1.92$	$+1.99$	$+2.12 \pm 0.70$	± 0.74
	BCA	$+3.83$	$+3.60$	$+3.91$	$+3.78 \pm 0.41$	± 0.09
ZS Avg. (ViT-B/16)	TDA	$+1.78$	$+1.37$	$+1.83$	$+1.66 \pm 0.63$	± 0.40
	OGA	$+1.60$	$+1.57$	$+1.70$	$+1.62 \pm 0.17$	± 0.21
	BCA	$+3.11$	$+3.28$	$+3.38$	$+3.26 \pm 0.33$	± 0.33

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: We clearly articulate the contributions of our candidate-verification framework, TTC, in both the abstract and Section 1. We also present supporting experimental evidence in Section 4.

   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: Please refer to the limitations of our work as discussed in Section 5.

   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: [Yes]

Justification: We present the full procedural overview of our proposed method in Section 3, and provide a detailed theoretical derivation along with complete proofs in Appendix A. All assumptions are clearly stated and appropriately referenced within the theoretical statements.

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: We provide the full procedure and details of our method to ensure reproducibility in Section 3. The experimental setting for reproduction can be found in Section 4 and Appendix B.

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: All data and code used in our experiments are publicly available.

   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: We describe the experimental setup and hyperparameters starting in Section 4 and present details on model settings, few-shot settings, experiment environments, and downstream task datasets in Appendix B.

   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: All experimental results are averaged over three seeds and reported with 95% confidence intervals. The corresponding confidence intervals are reported in Appendix G.

   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: We report computing resources, including the GPU used, throughput, wall-clock execution time, and memory, in Table 5.

   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: Our paper conforms to the NeurIPS Code of Ethics.

   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: [N/A]

    Justification: This paper introduces new test-time adaptation methods without potential societal impacts.

    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 poses no such risks.

    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: As mentioned in Appendix B, we utilize open-source models and 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 release our code at https://github.com/mmai-laboratory/ttc, with a README that documents the environment, data preparation, and commands to reproduce the main results, along with the license.

    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 involve crowdsourcing nor research with human subjects.

    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 crowdsourcing nor research with human subjects.

    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: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    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.