Stability-Aware Self-Training for CLIP under Cross-Modal Anchoring Mismatch

Siyuan Liu[1], Xinyang Chen[1]✉, Xiucheng Li[1], Weili Guan[2], Liqiang Nie[1]
[1]School of Computer Science and Technology, Harbin Institute of Technology (Shenzhen)
[2]School of Information Science and Technology, Harbin Institute of Technology (Shenzhen)
{liusiyuan666777, chenxinyang95, nieliqiang}@gmail.com,
{lixiucheng, guanweili}@hit.edu.cn

Abstract

Self-training is a practical way to adapt CLIP under limited supervision, but its performance is highly sensitive to pseudo-label reliability. We identify a key failure mode in CLIP adaptation: temporal prediction instability, in which a target sample repeatedly switches its prediction across epochs. We offer a perspective based on cross-modal anchoring mismatch: the decision geometry induced by pre-trained text anchors can deviate from the local visual cluster structure of target domain. This mismatch can make ambiguous target samples susceptible to prediction switches during adaptation, whereas target-aligned anchoring can produce more stable pseudo-labels. Motivated by this observation, we propose Stability-Aware Self-Training (**SAST**), centered on Stability-Weighted Image Prototypes (**SWIP**). SWIP builds class-wise visual anchors from temporally stable target samples to provide more reliable adaptation references. We further add a lightweight text-side refinement to reduce confusion-induced instability and combine the two streams with an adaptive fusion rule. Across six benchmarks and three adaptation settings, SAST consistently outperforms strong baselines. The results suggest that explicitly modeling temporal stability is an effective route to robust CLIP adaptation under self-training. Our code is available at `https://github.com/lsy-2003/SAST`.

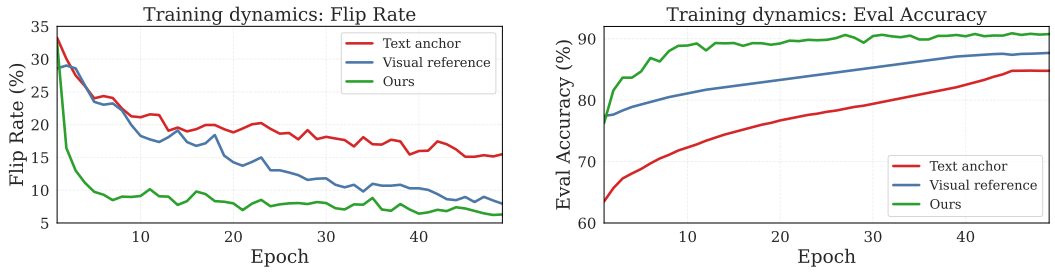
1 Introduction

Large-scale vision-language models (VLMs) [22, 16, 30] have shown strong transferability across a wide range of visual recognition tasks. To adapt these models to downstream domains with low annotation cost, self-training has become a practical paradigm that iteratively exploits unlabeled target data through pseudo-label supervision [25, 34, 7]. Its effectiveness depends critically on pseudo-label reliability [25, 4, 29]. In CLIP [22] adaptation, however, this reliability can be fragile: predictions that appear confident at one epoch may change later, causing pseudo-label to fluctuate during training.

In this work, we focus on an important yet less-studied failure mode in CLIP self-training: prediction instability, i.e., changes in predicted labels across epochs. Such instability is harmful because inconsistent predictions can be repeatedly reused as pseudo-labels, injecting unreliable supervision into subsequent adaptation [25, 9]. Existing methods typically evaluate pseudo-label reliability at a single point during training, by regulating the number of pseudo-labels, the sharpness of confidence, or calibrating probabilities [4, 8]. However, a pseudo-label that is confident at one epoch can still be unreliable if its prediction is temporally unstable. This motivates our view that stability should not only determine which pseudo-labels to trust, but what targets the model should adapt toward.

Our study is motivated by the modality gap [17, 23], which shows that image and text embeddings in vision-language models can exhibit systematic separation. In this paper, we focus on a related but distinct issue during adaptation, and we refer to this as **Cross-Modal Anchoring Mismatch**:

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



(a) Switch rate.

(b) Accuracy.

Figure 1: Training dynamics under SSL setting. We compare three variants: a text anchor adaptation, a target-aligned visual anchor reference and our SAST. For detailed protocol, refer to Appendix A.2.

pre-trained text anchors (produced by the text encoder) may not align well with the target-domain visual clusters. Although text anchors remain valuable because they provide zero-shot semantic knowledge, their local classification boundaries may still be imperfect for the target domain. When a target sample lies near the boundary induced by misaligned text anchors, small changes in the adapted representation can frequently change its nearest class anchor, leading to temporal prediction switches.

To explore how the arrangement of class anchors is related to prediction stability, we compare the training dynamics of three variants under the 2-shot SSL setting: a standard adaptation based on text anchors, a target-aligned visual anchor reference (initialized by visual features of the 2-shot samples), and our SAST framework. We provide further analysis and interpretation in Appendices A and B.

As shown in Figure 1, the target-aligned visual reference exhibits consistently lower switch rates and higher accuracy throughout the self-training process. The resulting prediction trajectories become more stable, suggesting that instability is not simply caused by insufficient training, but is instead closely related to the anchors used during adaptation. Moreover, the stability gain persists across the entire adaptation process, indicating that target-aligned anchoring affects not only the initial decision geometry but also the subsequent pseudo-label dynamics. This observation motivates a central design principle of our method: instead of relying only on fixed text anchors, CLIP adaptation can benefit substantially from target-aligned visual references that are constructed from reliable target samples.

Therefore, we propose **Stability-Aware Self-Training (SAST)**, which uses temporal stability as a structural signal to construct class anchors and to guide fusion. The core component, **Stability-Weighted Image Prototypes (SWIP)**, builds class-wise visual anchors from temporally stable target samples. To further reduce instability among semantically nearby classes, we introduce **Confusion-Aware Text Prompt Augmentation (CTPA)** and combine these two streams with an adaptive fusion rule. In Figure 1, SAST achieves the lowest switch rate and the highest accuracy among the three variants, supporting our hypothesis that stability-aware visual anchoring can effectively mitigate prediction instability. We evaluate SAST on CLIP adaptation benchmarks that emphasize domain shift and fine-grained ambiguity, where pseudo-label reliability remains an important bottleneck.

Our contributions are summarized as follows. First, we identify **prediction instability** as an important yet underexplored failure mode in CLIP self-training and provide diagnostic evidence that associates it with anchoring mismatch in adaptation. Second, we propose **SWIP**, which constructs class-wise visual anchors from temporally stable samples to serve as target-aligned references during self-training. Third, based on SWIP and a lightweight text refinement, we develop **SAST**, a stability-aware adaptation framework, and show consistent gains across multiple CLIP adaptation settings.

2 Related Work

Pseudo-label-based CLIP adaptation. Recent CLIP adaptation methods increasingly exploit unlabeled target data through pseudo-label self-training [11, 20, 32, 29]. Representative approaches improve pseudo-label usage via confidence-based sample selection, candidate-label modeling, or class balancing. These are the most direct baselines to our work, since they operate under the same practical setting of adapting CLIP with unlabeled or weakly labeled target data. In particular, we focus

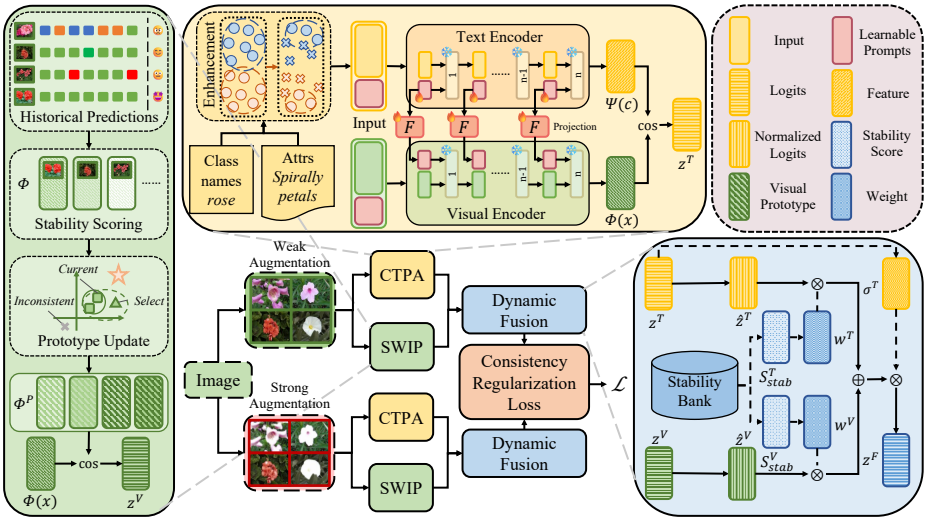


Figure 2: Our SAST framework. **Green:** SWIP selects class-wise image prototypes from temporally stable samples to provide visual anchors. **Yellow:** CTPA augments confusion-prone class prompts with discriminative attributes to enlarge local text-side separation. **Blue:** The rule to obtain a fused prediction. The CLIP encoders are frozen; only the learnable prompts (following MaPLe) are trained.

on a specific and underexplored failure mode in this regime: pseudo-labels that remain temporally unstable across training, which can inject unreliable supervision and therefore degrade adaptation.

Pseudo-label reliability and consistency. In semi-supervised learning, pseudo-label reliability is commonly improved through techniques such as confidence thresholding, calibration, consistency regularization, and curriculum strategies [14, 25, 31, 4, 8, 3, 28]. Related analyses of self-training dynamics also highlight the accumulation of pseudo-label errors during adaptation [9]. Our setting differs in two important respects. First, while consistency regularization in prior SSL work mainly targets perturbation-level invariance, we focus instead on epoch-level prediction switches during CLIP adaptation under anchoring mismatch. Second, temporal stability in SAST serves as a structural signal that guides visual anchor construction, unlabeled-sample partitioning, and text–visual fusion.

Prompt and cache-based CLIP adaptation. Prompt tuning, adapter tuning, and cache/prototype methods have been widely studied for CLIP adaptation with labeled support sets [35, 34, 13, 15, 12, 1, 33]. Related cache-based or training-free methods, such as Tip-Adapter [33] and TFUP [18], also improve adaptation by constructing representative memories or prompts from target data. SWIP repeatedly identifies temporally stable target samples and uses them to refresh class-wise visual anchors over the course of self-training, instead of depending on support prototypes or static caches.

Attribute-guided prompting. Several recent methods enrich CLIP text representations by incorporating semantic attributes, class descriptions, or confusion-aware prompting strategies [29, 2, 26, 24]. In contrast, our text-side refinement component is considerably lighter-weight: CTPA performs a fixed offline augmentation and selects neighbor-discriminative attributes only for confusion-prone classes, rather than relying on end-to-end prompt tuning or more complex attribute fusion schemes.

3 Methodology

Problem Definition. We adapt a pre-trained CLIP model to a downstream domain with label space $\mathcal{Y} = \{1, \ldots, C\}$. The training data consist of a large unlabeled set $\mathcal{D}_{\mathrm{UL}} = \{\mathbf{x}_i\}_{i=1}^{N}$ and, optionally, a small labeled set $\mathcal{D}_{\mathrm{L}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{M}$. Let $\phi(\cdot;\boldsymbol{\theta})$ and $\psi(\cdot;\boldsymbol{\theta})$ denote the visual and text encoders, respectively. Given an image $\mathbf{x}$ and a class prompt prompt_c, CLIP then predicts by cosine similarity:

$$p(y = c \mid \mathbf{x}) \propto \exp\big(\cos\big(\phi(\mathbf{x};\boldsymbol{\theta}),\, \psi(\mathrm{prompt}_c;\boldsymbol{\theta})\big)/\tau\big)\,.$$

We adapt the model using labeled data when available, and unlabeled data otherwise, with pseudo-labels generated during training. Following CAP [29], we use MaPLe [13] as the adaptation scheme.

Overview. SAST is designed to improve CLIP self-training stability. It centers on SWIP (§3.1), is complemented by CTPA (§3.2) and combines the two streams through an adaptive fusion rule (§3.3).

3.1 Stability-Weighted Image Prototypes

Motivation. SWIP mitigates sample-level prediction instability by constructing class-wise visual anchors directly from target data. Since support-derived prototypes are unavailable in fully unlabeled settings, we build them in a self-training manner from temporally stable samples. This yields anchors that better align with the underlying target distribution and are more reliable during self-training.

Visual prototype classifier. Given an image $\mathbf{x}$, we use its ℓ_2-normalized visual embedding $\tilde{\phi}(\mathbf{x}) = \phi(\mathbf{x})/\|\phi(\mathbf{x})\|_2$. Let $\mathbf{\Phi}^P = [\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_C]$ denote the normalized prototypes. SWIP produces logits by

$$\mathbf{z}^V = \tilde{\phi}(\mathbf{x})(\mathbf{\Phi}^P)^\top/\tau. \tag{1}$$

Since $\mathbf{\Phi}^P$ are refreshed from temporally stable target-domain samples, they provide more reliable visual references that are typically less sensitive to transient prediction switches than text anchors.

Stability-based prototype construction. We maintain a memory bank $\mathcal{Q}$ that stores prediction history and lightweight statistics. At the beginning of each epoch, SWIP refreshes the prototypes by selecting candidate samples with high temporal stability. For a sample $\mathbf{x}$, let $\{\hat{y}^{(l)}\}_{l=1}^{L}$ denote the available historical predicted classes of $\mathbf{x}$ stored in $\mathcal{Q}$ up to the last epoch L, and let $\hat{y}^{(L+1)}$ be its current prediction. We estimate its prediction instability by computing the disagreement frequency

$$S_{\text{inst}}(\mathbf{x}) = \frac{1}{L}\sum_{l=1}^{L}\mathbb{I}\Big(\hat{y}^{(l)} \neq \hat{y}^{(L+1)}\Big). \tag{2}$$

For samples with insufficient history ($L < 2$), we assign a neutral frequency $S_0 = 0.5$ (an ablation is provided in Table 9). Then the frequency is converted into a stability score by a normalized ranking,

$$S(\mathbf{x}) = \mathcal{R}(1 - S_{\text{inst}}(\mathbf{x})) - \lambda_c \Delta_{\text{cons}}(\mathbf{x}), \tag{3}$$

where $\mathcal{R}(\cdot)$ maps the scores in the current candidate pool to $[0, 1]$. Here, Δ_{cons} discourages prototype drift away from the text-side anchor. For a candidate $\mathbf{x}$ with pseudo-label c, we form a tentative update $\boldsymbol{\mu}_{\text{new}} = \text{Normalize}(\boldsymbol{\mu}_c^{(t)} + \tilde{\phi}(\mathbf{x}))$ and measure its similarity drop to the text prototype $\boldsymbol{\psi}_c$:

$$\Delta_{\text{cons}}(\mathbf{x}) = \max\Big(0,\, \cos(\boldsymbol{\mu}_c^{(t)}, \boldsymbol{\psi}_c) - \cos(\boldsymbol{\mu}_{\text{new}}, \boldsymbol{\psi}_c)\Big). \tag{4}$$

Samples with high $S(\mathbf{x})$ are less affected by the prediction switches and are therefore more suitable to serve as visual anchors. For each class c, we collect unlabeled samples whose current pseudo-label is c. To avoid class imbalance and reduce computational cost, the selection is performed class-wise. Denoting the selected unlabeled set by $\mathcal{S}_{\text{UL},c}^{(t)}$ and the labeled set by $\mathcal{D}_{\text{L},c}$, we update the prototype as

$$\boldsymbol{\mu}_c^{(t+1)} = \text{Normalize}\left(\frac{1}{|\mathcal{S}_c^{(t)}|}\sum_{\mathbf{x}\in\mathcal{S}_c^{(t)}}\tilde{\phi}(\mathbf{x})\right), \tag{5}$$

where $\mathcal{S}_c^{(t)} = \mathcal{D}_{\text{L},c} \cup \mathcal{S}_{\text{UL},c}^{(t)}$ contains labeled samples of class c (when such labels are available) together with the selected unlabeled target samples. We update the prototypes progressively, starting from highly stable core samples and gradually broadening the candidate pool as training proceeds.

In this way, SWIP leverages temporal stability as the selection criterion to progressively convert unlabeled target samples into increasingly reliable visual anchors over the course of self-training.

3.2 Confusion-Aware Text Prompt Augmentation

Motivation. Beyond sample-level instability, prediction switches concentrate on semantically confusing classes (Appendix A.5), suggesting that standard text prompts may insufficiently separate nearby categories. CTPA therefore refines prompts for those classes to enlarge the local text separation.

Confusion-aware prompt augmentation. We first define the confusion neighbor set of class c as

$$\mathcal{N}(c) = \{c' \in \mathcal{Y} \mid c' \neq c,\ \cos(\psi_{\text{base}}(c), \psi_{\text{base}}(c')) > \delta\}, \tag{6}$$

where $\psi_{\text{base}}(c)$ is the text feature of the base class name. For each class, an LLM generates a candidate attribute pool $\mathcal{A}_c$ offline, after which the prompts remain fixed. We then rank attributes by

$$\rho(a; c) = \max_{c' \in \mathcal{N}(c)} \cos(\psi(a), \psi_{\text{base}}(c')), \tag{7}$$

and select the attributes with the lowest values that are least similar to the confusing neighbors. We construct the retained subset $\mathcal{A}_c^*$ greedily: starting from the base class prompt, we add the candidate attribute with the lowest $\rho(a; c)$ among the remaining candidates at each step and stop once the prompt length limit is reached. For classes without confusion neighbors, we keep the prompt unchanged.

The enhanced text feature is formed by concatenating the class name with the selected attributes in $\mathcal{A}_c^*$. Stacking the resulting class-wise features yields $\mathbf{\Psi} \in \mathbb{R}^{C \times D}$. The corresponding text logits are

$$\mathbf{z}^T = \phi(\mathbf{x})\mathbf{\Psi}^\top / \tau. \tag{8}$$

By selectively injecting such confusion-aware attributes, CTPA reduces instability from the textual side, complementing SWIP's visual anchoring. Attributes for all classes are generated offline from class names only, without access to any target samples, labels, or statistics, matching the information available to the baselines. Details of LLM usage with more experiments are provided in Appendix C.

3.3 Prediction Rule and Optimization

Prediction Rule. Since SAST includes both visual and text-based predictors, we use an adaptive fusion rule to combine their outputs. This adaptivity is driven by sample-level branch reliability. For each stream $\text{Str} \in \{T, V\}$, we define a nonnegative reliability score $R^{\text{Str}}(\mathbf{x})$. During training, recurring unlabeled target samples provide prediction histories over successive epochs, so $R^{\text{Str}}(\mathbf{x}) = 1 - S_{\text{inst}}(\mathbf{x})$ is computed analogously to SWIP from the disagreement frequency between the current branch prediction and its branch-specific prediction history accumulated up to the preceding epoch. At evaluation time, histories are unavailable, so we use branch confidence as an inductive surrogate.

Due to the scale discrepancy between the two streams, we normalize their logits: $\hat{\mathbf{z}}^{\text{Str}} = (\mathbf{z}^{\text{Str}} - \mu^{\text{Str}})/\sigma^{\text{Str}}$, where μ^{Str} and σ^{Str} are the instance mean and standard deviation. We then convert the reliability scores into normalized fusion weights $w^{\text{Str}} \propto R^{\text{Str}}$, and rescale the fused logits to σ^T so that they remain on the same scale as the native CLIP logits. The fused logits are computed as

$$\mathbf{z}^F = \sigma^T \left(w^T\, \hat{\mathbf{z}}^T + w^V\, \hat{\mathbf{z}}^V \right). \tag{9}$$

The resulting prediction rule thus preserves the complementary benefits of SWIP and CTPA while reducing the risk that a less reliable stream dominates the final prediction for a particular sample.

Optimization. The optimization follows standard pseudo-label training. We optimize the model with

$$\mathcal{L} = \mathcal{L}_{\text{sup}}(\mathcal{D}_{\text{L}}) + \mathcal{L}_{\text{unsup}}(\mathcal{U}), \tag{10}$$

where $\mathcal{L}_{\text{sup}}$ is the standard cross-entropy loss. For the unlabeled batch $\mathcal{U}$, we use a standard consistency regularization with CE loss. During optimization, for an unlabeled sample $\mathbf{x}$, we compute the fused prediction $\mathbf{z}^F(\mathbf{x})$ and obtain its maximum probability $p_{\max}(\mathbf{x})$. We maintain an EMA-updated confidence threshold $\mu_{\text{conf}}^{(t)}$ initialized at μ_{init}. The training batch is formed as:

$$\mathcal{U}^{(t)} = \left\{ \mathbf{x}_i \in \mathcal{D}_{\text{UL}} : p_{\max}(\mathbf{x}_i) \geq \mu_{\text{conf}}^{(t)} \right\}. \tag{11}$$

4 Experiments

We evaluate SAST in terms of overall adaptation performance and prediction-instability control. We describe the settings (§4.1), report the main results (§4.2), and provide ablations and analyses (§4.3).

Table 1: Comparison of accuracy (%). Results are reported as mean_{std}. The best results are in **bold**.

	FLOWERS102			RESISC45			DTD		
METHODS	UL	SSL	TRZSL	UL	SSL	TRZSL	UL	SSL	TRZSL
CLIP [22]	$63.67_{0.00}$	$63.67_{0.00}$	$63.40_{0.00}$	$54.48_{0.00}$	$54.48_{0.00}$	$54.46_{0.00}$	$43.24_{0.00}$	$43.24_{0.00}$	$43.45_{0.00}$
FPL [20]	$65.67_{0.23}$	$75.96_{0.74}$	$80.97_{0.00}$	$63.07_{0.38}$	$68.13_{0.55}$	$72.11_{0.00}$	$44.96_{0.55}$	$37.10_{5.45}$	$46.30_{0.03}$
GRIP [20]	$69.84_{1.06}$	$83.60_{0.48}$	$86.26_{0.00}$	$70.55_{0.88}$	$74.11_{0.68}$	$81.07_{0.00}$	$46.09_{1.06}$	$56.07_{0.85}$	$65.30_{0.01}$
CPL [32]	$72.90_{0.78}$	$89.66_{0.36}$	$87.35_{0.76}$	$77.39_{0.44}$	$80.98_{0.11}$	$85.85_{0.49}$	$51.91_{0.71}$	$61.21_{0.56}$	$68.00_{0.34}$
CAP [29]	$76.80_{0.84}$	$89.96_{0.46}$	$89.53_{0.70}$	$81.48_{0.45}$	$83.32_{0.58}$	$88.82_{0.18}$	$55.29_{0.31}$	$62.33_{0.58}$	$69.55_{0.51}$
OURS	$\mathbf{77.60}_{1.28}$	$\mathbf{90.88}_{0.30}$	$\mathbf{90.69}_{0.19}$	$\mathbf{82.62}_{0.26}$	$\mathbf{85.33}_{0.79}$	$\mathbf{88.92}_{0.21}$	$\mathbf{58.28}_{0.29}$	$\mathbf{63.35}_{0.83}$	$\mathbf{69.92}_{0.32}$

	EUROSAT			CUB			FGVC-AIRCRAFT		
METHODS	UL	SSL	TRZSL	UL	SSL	TRZSL	UL	SSL	TRZSL
CLIP [22]	$32.88_{0.00}$	$32.88_{0.00}$	$30.54_{0.00}$	$51.82_{0.00}$	$51.82_{0.00}$	$51.57_{0.00}$	$17.58_{0.00}$	$17.58_{0.00}$	$17.86_{0.00}$
FPL [20]	$48.96_{1.49}$	$62.05_{1.64}$	$53.70_{26.87}$	$53.04_{0.53}$	$55.29_{0.59}$	$55.44_{0.20}$	$16.62_{0.67}$	$20.02_{0.77}$	$17.55_{0.37}$
GRIP [20]	$57.21_{1.77}$	$58.66_{2.64}$	$92.33_{0.69}$	$51.42_{0.21}$	$56.65_{0.33}$	$59.48_{0.38}$	$15.22_{0.71}$	$16.98_{0.82}$	$26.08_{0.25}$
CPL [32]	$67.26_{0.47}$	$77.51_{0.80}$	$93.78_{0.12}$	$53.47_{0.36}$	$58.53_{0.24}$	$66.20_{0.50}$	$18.35_{0.27}$	$22.48_{0.63}$	$30.86_{0.70}$
CAP [29]	$75.01_{1.94}$	$92.78_{0.34}$	$96.64_{0.27}$	$55.76_{0.26}$	$58.04_{0.23}$	$61.35_{0.05}$	$18.42_{0.13}$	$21.79_{0.16}$	$29.03_{0.49}$
OURS	$\mathbf{95.30}_{0.26}$	$\mathbf{94.70}_{0.16}$	$\mathbf{98.17}_{0.34}$	$\mathbf{60.72}_{0.50}$	$\mathbf{66.12}_{0.72}$	$\mathbf{72.30}_{0.22}$	$\mathbf{18.84}_{1.92}$	$\mathbf{25.21}_{0.30}$	$\mathbf{31.65}_{1.33}$

4.1 Experimental Settings

Datasets and Metrics. Following previous studies [20, 32, 29], we evaluate on the same six widely used benchmarks: RESISC45 [5], DTD [6], EuroSAT [10], FGVC-Aircraft [19], CUB [27], and Flowers102 [21]. These datasets encompass diverse visual domains, ranging from satellite imagery and textures to fine-grained object recognition, and are standard in recent pseudo-label-based CLIP adaptation work, where pseudo-label reliability under target-domain mismatch is the central concern. We report the top-1 accuracy averaged across three independent runs with different random seeds.

Learning Paradigms. We evaluate SAST under three target adaptation settings: (1) **Unsupervised Learning (UL):** adaptation uses only unlabeled target data $\mathcal{D}_{\text{UL}}$. (2) **Semi-Supervised Learning (SSL):** following CAP [29], we provide two labeled samples per class together with the remaining unlabeled target data. (3) **Transductive Zero-Shot Learning (TRZSL):** following prior work [20, 32, 29], we split the label space into 62% seen and 38% unseen classes. During adaptation, seen classes are trained with full supervision, whereas unseen classes rely solely on unlabeled target data. We report generalized performance by the harmonic mean of top-1 accuracy on the seen and unseen splits. Here, "transductive" is used in the sense of prior work and does not imply that test samples are used for training. It only refers to the availability of unlabeled samples from unseen classes during adaptation, which yields an imbalanced supervision setting. Evaluation remains inductive at test time.

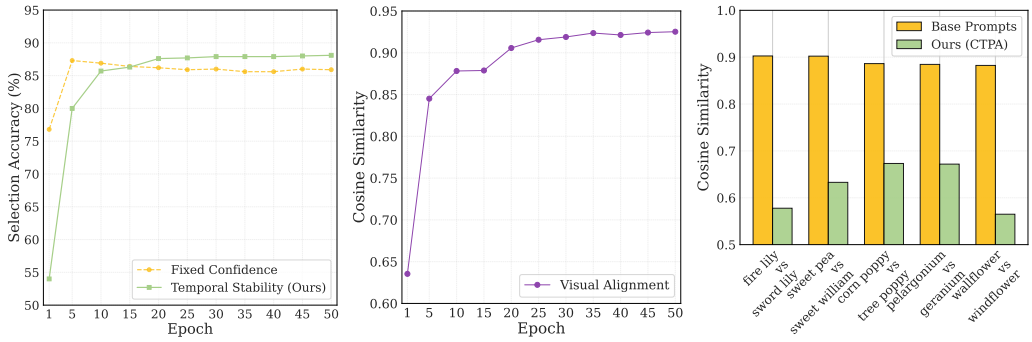
Implementation. We use pre-trained CLIP (ViT-B/32) [22] as the backbone. The model is trained for 50 epochs with SGD (learning rate of 0.02, momentum of 0.9) and a cosine annealing scheduler. All experiments are conducted on an NVIDIA A100 GPU. Training details are provided in Appendix D.

Baselines. We compare SAST with zero-shot CLIP and several recent CLIP adaptation methods under the same pseudo-label-based target adaptation regime, including FPL/GRIP [20] (NeurIPS 2023), CPL [32] (ICML 2024), and CAP [29] (ICML 2025). These are our most direct baselines, since they likewise rely on unlabeled or weakly labeled target data and are similarly affected by pseudo-label reliability under domain shift. By contrast, few-shot methods built on labeled support sets (e.g., prompt-, adapter-, or cache-based tuning) are not protocol-matched and are excluded.

4.2 Main Results

Table 1 summarizes the quantitative results on six benchmarks under UL, SSL, and TRZSL. Across all 18 dataset-setting combinations, SAST improves over the strongest baseline. These results support the central claim of our paper: accounting for prediction instability improves pseudo-label-based CLIP adaptation, especially when text anchors are misaligned with target-domain visual structure.

Unsupervised Learning (UL). SAST shows substantial improvements in the fully unlabeled setting, where adaptation depends entirely on pseudo-label supervision and prediction instability is particularly costly because mistakes can be repeatedly recycled. With no labeled anchor to correct a drifting



(a) Prototype selections. (b) Prototype alignment. (c) Similarity reduction.

Figure 3: Analysis on the effects of the components of SAST.

boundary, the improvements on EuroSAT, Flowers102, and CUB are consistent with the intended role of SWIP, namely substituting temporal stability for supervision when no labeled anchor is available.

Semi-Supervised Learning (SSL). SAST also improves consistently in the low-label setting. The gains are largest on CUB and FGVC-Aircraft, two fine-grained datasets where two labeled samples per class are insufficient to resolve target-domain ambiguity. This suggests that stability-aware prototype refresh is most useful when scarce supervision cannot reliably anchor the decision boundary by itself.

Transductive Zero-Shot Learning (TRZSL). SAST remains effective in generalized transductive zero-shot learning. The improvements are largest on CUB, EuroSAT, and FGVC-Aircraft, where the seen/unseen split introduces stronger semantic mismatch than in the UL and SSL settings. This suggests that stability-aware anchoring helps maintain the seen/unseen balance under this mismatch.

4.3 Analysis

We now analyze why SAST works, focusing on component contributions, the relation between stability and confidence, and sensitivity to hyperparameters and backbone scale. Additional analyses, including the CTPA robustness analysis and the overhead, are provided in Appendices C.3 and E.

Is Temporal Stability an Effective Selection Signal? To evaluate whether temporal stability provides a reliable criterion for constructing image prototypes in SWIP, we track pseudo-label accuracy during training on RESISC45 under the UL setting. The pseudo-label accuracy is computed against ground-truth labels for analysis only and is not used during training. We compare the standard confidence-based strategy with our stability-based ranking in Figure 3a. The confidence baseline quickly plateaus at around 86%, while the stability-based strategy shows a stronger upward trend after the early stage of training and eventually exceeds 88% pseudo-label accuracy. This result supports our claim that temporal stability is an effective criterion for identifying reliable target samples.

Do SWIP Prototypes Better Align with Target Visual Clusters? To examine whether SWIP progressively captures the target-domain visual structure, we visualize the alignment between the learned image prototypes and the target visual clusters on RESISC45 under the SSL setting. Figure 3b reports the average cosine similarity between the learned prototypes and the class-wise target cluster centers, where the latter are computed using ground-truth labels. The initial similarity (0.64) reflects the alignment under the pre-trained CLIP before adaptation, indicating a substantial mismatch between the initial CLIP representation and the target-domain feature distribution. As training proceeds, the similarity rises steadily and stabilizes around 0.93. This trend suggests that SWIP progressively selects more stable candidates and builds prototypes that better match the target visual structure, which is consistent with its role in mitigating prediction instability under anchoring mismatch.

Does CTPA Improve Text-Side Stability Geometry? We analyze whether CTPA reduces confusion-induced instability by enlarging local text-side separation among semantically nearby classes on Flowers102. Figure 3c shows the cosine similarity of the text features for five representative confusing class pairs before and after prompt augmentation. In these cases, the enhanced prompts yield lower similarity than the base prompts. Notable reductions are observed for pairs such as *fire lily* vs. *sword lily* ($\Delta = -0.325$) and *wallflower* vs. *windflower* ($\Delta = -0.318$). These results indicate that

Table 2: Ablation of SWIP and CTPA.

SWIP	CTPA	EuroSAT	CUB	DTD
×	×	91.7	58.4	59.6
×	✓	91.8	59.9	62.8
✓	×	93.2	64.7	61.1
✓	✓	**94.7**	**66.1**	**63.4**

Table 3: Swap experiment for text prompt.

Method	CUB	RESISC45
CAP	58.0	83.3
SAST + CAP prompt	64.0	86.2
CAP + our CTPA	56.9	83.7
SAST	66.1	85.3

Table 4: CTPA activation statistics and its accuracy gains.

Dataset	Fraction $\mathcal{N}(c) \neq \emptyset$	Mean $\lvert\mathcal{N}(c)\rvert$	CTPA gain
Flowers102	0.40	1.0	$+0.5$
CUB	0.70	3.7	$+2.1$
EuroSAT	1.00	6.6	$+2.7$
RESISC45	0.97	18.1	-0.9
DTD	0.98	24.8	$+4.3$

confusion-aware attribute augmentation pushes nearby text features farther apart, improving local text-side separation and thereby helping reduce prediction switches and improving temporal stability.

Component Ablation. To examine the roles of the components in SAST, we perform ablations on EuroSAT, CUB, and DTD under the SSL setting. Table 2 reports the results of removing SWIP and CTPA from the full framework. Both components improve over the baseline, and their combination yields the best performance, supporting their complementary roles in the proposed stability-aware framework. This complementarity is not uniform, and we observe dataset-dependent trends: SWIP contributes more on CUB and EuroSAT, where stability-based visual anchoring is particularly helpful under fine-grained ambiguity, while CTPA brings larger gains on DTD, where reducing text-side confusion appears more beneficial. The ablation indicates that SWIP accounts for most of the gain on ambiguity-sensitive datasets such as CUB, supporting our claim that stability-aware anchoring is the primary driver of SAST. CTPA provides complementary refinement for confusion-prone classes.

Attribution of CTPA Gains. The ablation shows that CTPA contributes complementarily to SWIP, but it does not rule out that the gain comes from stronger text descriptions rather than from the stability-aware integration. We therefore cross-combine the two adaptation frameworks (CAP[29] and SAST) with the two text prompt sources (the CAP-style prompt and our CTPA attributes), keeping all other components unchanged. As shown in Table 3, SAST consistently outperforms CAP under both prompt sources, whereas feeding our attributes directly to CAP yields no comparable gain. This indicates that the improvement stems primarily from how the text branch is integrated into the stability-aware framework, rather than from any prompt text alone. We therefore view CTPA as a semantic refinement whose benefit is realized through the stability-guided adaptation framework.

When Does CTPA Help? To understand when CTPA is beneficial, we compute two statistics: the fraction of classes with a non-empty confusion set $\mathcal{N}(c)$, which reflects how often CTPA is activated, and the mean $|\mathcal{N}(c)|$, which characterizes the degree of semantic ambiguity, both under $\delta = 0.8$. As shown in Table 4, with the exception of RESISC45, CTPA brings larger gains as more classes participate in refinement and as the neighborhoods become denser, with DTD and EuroSAT showing the clearest gains. For RESISC45, we attribute this deviation to the compositional nature of its categories, such as *railway* vs. *railway station*, which share substantial semantic components rather than distinctive object attributes. Even when prompted for discriminative attributes, the candidate pool retains considerable semantic overlap. Since CTPA performs confusion-aware selection rather than attribute generation, its benefit is bounded by the discriminability of the candidate attribute pool.

Single-Branch vs. Fused Prediction. Table 5 compares the single-branch and fused predictors on four datasets under the UL setting. Across all datasets, the adaptive fusion consistently exceeds the better single branch. This suggests that the two streams are not redundant and that the adaptive fusion can effectively exploit their combination. For example, on CUB, the fused predictor reaches 60.7%, outperforming both the text branch (56.9%) and the visual branch (58.0%). A similar trend is observed on RESISC45, DTD, and Flowers102, indicating that the benefit of fusion is consistent.

Table 5: Single-branch vs. fused prediction.

DATASET	TEXT	VISUAL	FUSED
CUB	56.9	58.0	**60.7**
RESISC45	81.3	81.9	**82.6**
DTD	57.9	55.4	**58.3**
FLOWERS102	75.7	77.1	**77.6**

Table 6: Comparison of fusion strategies.

FUSION VARIANT	DTD	CUB
SWIP:CTPA = 1:0	60.1	64.7
SWIP:CTPA = 0:1	62.9	59.8
SWIP:CTPA = 0.5:0.5	62.4	64.1
GLOBAL LEARNED SCALAR α	58.2	63.9
OURS	**63.4**	**66.1**

Table 7: Replacing stability with confidence.

CONFIG	RESISC45	DTD	CUB
STAB. IN BOTH	**83.6**	**57.5**	**60.7**
SWIP $\rightarrow$ CONF.	76.6	55.6	57.9
FUSION $\rightarrow$ CONF.	81.5	56.9	60.4
CONF. IN BOTH	75.6	55.7	57.7

Table 8: Correlations.

PROXY	ACC.	STAB.
STAB.	0.566	1.000
CONF.	0.518	0.624
ENTROPY	-0.513	-0.606
MARGIN	0.485	0.767

Comparison of Fusion Strategies. Having established that fusing the two branches is beneficial, we next ask how they should be fused. We compare the proposed adaptive fusion with several alternative integration strategies on DTD and CUB under the SSL setting. As shown in Table 6, our method consistently outperforms two single-stream variants (SWIP-only and CTPA-only), arithmetic average, and globally learned fusion rule. The global learned scalar yields inferior performance, suggesting that a single dataset-level trade-off is insufficient to capture the instance-dependent reliability of the two streams. These results support our design of estimating branch reliability on a per-sample basis.

Is Stability More Than Confidence? A natural question is whether temporal stability merely reparameterizes confidence. We examine this question at two levels: the component level and the sample level. At the component level, we replace stability with confidence in the two components that use it: SWIP prototype construction and adaptive fusion. As shown in Table 7, replacing either component reduces accuracy, and using confidence in both yields the lowest accuracy on all three datasets. When all components rely on confidence, the same signal is used for pseudo-label filtering, prototype selection, and fusion, making these components partially redundant and thereby reducing accuracy. This indicates that stability contributes information that confidence alone does not provide.

At the sample level, we analyze pseudo-label correctness on CUB by grouping target samples according to their final-round stability scores. As shown in Figure 4, the pseudo-label error rate decreases monotonically as stability increases, from 97.6% in the lowest-stability bin to 7.8% in the highest-stability bin. We further condition on both signals. For high-confidence samples (confidence ≥ 0.9), the overall accuracy is 84.2% ($n = 2021$). However, among low-confidence yet high-stability samples (confidence $< 0.9, S(\mathbf{x}) \geq 0.8$), accuracy remains remarkably high at 82.6% ($n = 310$), nearly comparable to that of high-confidence samples. This pattern indicates that stability identifies a subset of low-confidence predictions that are nonetheless reliable, a distinction that confidence cannot make.

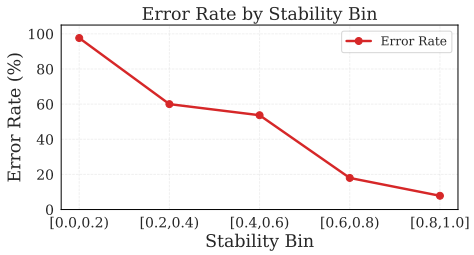


Figure 4: Pseudo-label error rate vs. stability.

How Should Reliability Signals Be Assigned? The above results show that stability captures information beyond confidence. Overall, these results suggest that stability is not a replacement for confidence-based filtering, nor merely a proxy for confidence, but a complementary structural signal: confidence serves as the per-sample filtering criterion, while stability determines which samples form visual anchors and how the two streams are weighted. Since high-stability samples are relatively limited in number, they are naturally suited to serve as anchors, where precision matters more than coverage, whereas high-confidence samples are much more abundant and can provide broader pseudo-label supervision. Table 8 quantifies this complementarity. Temporal stability shows the strongest correlation with pseudo-label accuracy (0.566) and is therefore used during training when prediction histories are available. Among the instantaneous proxies, confidence and entropy

Table 9: Ablation of hyperparameters. Default values are shown in **bold**.

PARAMETER	VALUES	FLOWERS102	FGVC-AIRCRAFT
S_0	0 / **0.5** / 1.0	90.3 / 90.9 / 90.6	24.6 / 25.2 / 24.9
λ_c	0.05 / **0.1** / 0.2	90.9 / 90.9 / 90.6	24.9 / 25.2 / 25.1
k	4 / **8** / 12	91.0 / 90.9 / 90.6	24.8 / 25.2 / 24.9
δ	0.7 / **0.8** / 0.9	90.7 / 90.9 / 91.0	25.2 / 25.2 / 24.4
μ_{INIT} (FLOWERS102)	0.85 / **0.90** / 0.95	89.8 / 90.9 / 90.4	–
μ_{INIT} (FGVC-AIRCRAFT)	0.50 / **0.60** / 0.70	–	24.9 / 25.2 / 25.1
MOMENTUM	0.8 / **0.9** / 0.99	91.0 / 90.9 / 89.8	24.0 / 25.2 / 24.3

Table 10: Comparison of accuracy using ViT-L/14 as backbone. The best results are in **bold**.

DTD				RESISC45			
METHODS	SSL	UL	TRZSL	METHODS	SSL	UL	TRZSL
ZERO-SHOT CLIP	52.45	52.45	51.61	ZERO-SHOT CLIP	62.67	62.67	62.13
FPL	60.61	52.99	60.77	FPL	79.01	70.85	77.69
GRIP	60.91	54.40	64.92	GRIP	81.53	76.86	86.88
CPL	69.82	57.20	71.97	CPL	87.75	80.88	89.73
CAP	70.26	65.85	73.61	CAP	88.09	84.46	90.21
OURS	**70.64**	**66.38**	**75.40**	**OURS**	**91.16**	**85.34**	**93.02**

behave similarly (0.518 and 0.513) and are strongly anti-correlated with each other, indicating that they capture closely related uncertainty information. The logit margin is a notable case: although its correlation with accuracy (0.485) is slightly weaker than that of confidence, it shows the strongest correlation with temporal stability (0.767) among the instantaneous proxies. We attribute this to the fact that margin is closely tied to local decision-boundary robustness and hence to the difficulty of prediction switching. However, as a single-step signal, margin is more easily affected by samples that are consistently but incorrectly predicted, whereas temporal stability benefits from long-horizon aggregation and is therefore more robust as a reliability measure. When prediction histories are unavailable, as is the case for unseen test samples, we therefore retain confidence as the default single-pass proxy, while using temporal stability as the primary reliability signal during training.

Hyperparameter Sensitivity. We evaluate the robustness of SAST to key hyperparameters, including the initial disagreement frequency S_0, the penalty weight λ_c, the initial prototype quota k, the text-similarity threshold δ used to define confusion neighbors, the base pseudo-label threshold μ_{init}, and the momentum. Each hyperparameter is varied independently while keeping the others fixed at their default values. As shown in Table 9, SAST remains reasonably stable across the tested ranges on Flowers102 and FGVC-Aircraft, suggesting that SAST does not rely heavily on precise tuning.

Comparison Across Different Encoders. To further evaluate the generalizability of SAST across different backbone scales, we replace the default ViT-B/32 with a larger ViT-L/14 visual backbone and conduct additional experiments on DTD and RESISC45 under SSL, UL, and TRZSL settings. As shown in Table 10, our method consistently outperforms all baselines across all settings on both datasets. These results suggest that the proposed stability-aware design is not tied to a specific encoder scale, and can continue to provide gains even when starting from a stronger visual representation.

5 Conclusion

We study prediction instability in CLIP self-training through cross-modal anchoring mismatch: classifying target-domain visual features against pre-trained text anchors misaligned with the local target structure yields a fragile decision geometry under cosine-based classification. We propose a Stability-Aware Self-Training (SAST) framework, built around Stability-Weighted Image Prototypes (SWIP) for constructing visual anchors from temporally stable samples, and Confusion-Aware Text Prompt Augmentation (CTPA) for improving text-side separation among confusing categories, combined by an adaptive fusion rule. Our experiments across six benchmarks and three learning settings show that explicitly modeling prediction instability consistently improves CLIP adaptation.

Acknowledgments and Disclosure of Funding

This work was supported by the National Natural Science Foundation of China (62676119, 62236003), Guangdong Provincial Key Laboratory of Artificial General Intelligence and Large Models (2026B1212010005), Guangdong Science and Technology Program (2025B0101130003), Shenzhen College Stability Support Plan (GXWD20231130151329002), and Shenzhen Science and Technology Program (KQTD20240729102207002, KQTD2025120109171700).

References

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

[2] Chengyi Cai, Zesheng Ye, Lei Feng, Jianzhong Qi, and Feng Liu. Attribute-based visual reprogramming for vision-language models. *arXiv preprint arXiv:2501.13982*, 2025.

[3] Paola Cascante-Bonilla, Fuwen Tan, Yanjun Qi, and Vicente Ordonez. Curriculum labeling: Revisiting pseudo-labeling for semi-supervised learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6912–6920, 2021.

[4] Hao Chen, Ran Tao, Yue Fan, Yidong Wang, Jindong Wang, Bernt Schiele, Xing Xie, Bhiksha Raj, and Marios Savvides. Softmatch: Addressing the quantity-quality trade-off in semi-supervised learning. *arXiv preprint arXiv:2301.10921*, 2023.

[5] Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883, 2017.

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

[7] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. Clip-adapter: Better vision-language models with feature adapters. *International Journal of Computer Vision*, 132(2):581–595, 2024.

[8] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017.

[9] Yifei He, Haoxiang Wang, Bo Li, and Han Zhao. Gradual domain adaptation: Theory and algorithms. *Journal of Machine Learning Research*, 25(361):1–40, 2024.

[10] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

[11] Tony Huang, Jack Chu, and Fangyun Wei. Unsupervised prompt learning for vision-language models. *arXiv preprint arXiv:2204.03649*, 2022.

[12] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision*, pages 709–727. Springer, 2022.

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

[14] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016.

[15] Dongjun Lee, Seokwon Song, Jihee Suh, Joonmyeong Choi, Sanghyeok Lee, and Hyunwoo J Kim. Read-only prompt optimization for vision-language few-shot learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1401–1411, 2023.

[16] 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*, 34:9694–9705, 2021.

[17] Victor Weixin Liang, Yuhui Zhang, Yongchan Kwon, Serena Yeung, and James Y Zou. Mind the gap: Understanding the modality gap in multi-modal contrastive representation learning. *Advances in Neural Information Processing Systems*, 35:17612–17625, 2022.

[18] Sifan Long, Linbin Wang, Zhen Zhao, Zichang Tan, Yiming Wu, Shengsheng Wang, and Jingdong Wang. Training-free unsupervised prompt for vision-language models. *arXiv preprint arXiv:2404.16339*, 2024.

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

[20] Cristina Menghini, Andrew Delworth, and Stephen Bach. Enhancing clip with clip: Exploring pseudolabeling for limited-label prompt tuning. *Advances in Neural Information Processing Systems*, 36:60984–61007, 2023.

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

[22] 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*, pages 8748–8763. PmLR, 2021.

[23] Simon Schrodi, David T Hoffmann, Max Argus, Volker Fischer, and Thomas Brox. Two effects, one trigger: On the modality gap, object bias, and information imbalance in contrastive vision-language models. *arXiv preprint arXiv:2404.07983*, 2024.

[24] Maoyuan Shao, Yutong Gao, Xinyang Huang, Chuang Zhu, Lijuan Sun, and Guoshun Nan. Capt: Confusion-aware prompt tuning for reducing vision-language misalignment. *arXiv preprint arXiv:2603.02557*, 2026.

[25] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in Neural Information Processing Systems*, 33:596–608, 2020.

[26] Xinyu Tian, Shu Zou, Zhaoyuan Yang, and Jing Zhang. Argue: Attribute-guided prompt tuning for vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 28578–28587, 2024.

[27] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[28] Yidong Wang, Hao Chen, Qiang Heng, Wenxin Hou, Yue Fan, Zhen Wu, Jindong Wang, Marios Savvides, Takahiro Shinozaki, Bhiksha Raj, et al. Freematch: Self-adaptive thresholding for semi-supervised learning. *arXiv preprint arXiv:2205.07246*, 2022.

[29] Yuchen Wang, Xuefeng Bai, Xiucheng Li, Weili Guan, Liqiang Nie, and Xinyang Chen. Handling imbalanced pseudolabels for vision-language models with concept alignment and confusion-aware calibrated margin. In *International Conference on Machine Learning*, pages 62357–62372. PMLR, 2025.

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

[31] Bowen Zhang, Yidong Wang, Wenxin Hou, Hao Wu, Jindong Wang, Manabu Okumura, and Takahiro Shinozaki. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *Advances in Neural Information Processing Systems*, 34:18408–18419, 2021.

[32] Jiahan Zhang, Qi Wei, Feng Liu, and Lei Feng. Candidate pseudolabel learning: Enhancing vision-language models by prompt tuning with unlabeled data. In *International Conference on Machine Learning*, pages 60004–60020. PMLR, 2024.

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

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

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

Appendix Contents

- A. More Analysis on Prediction Instability
- B. A Geometric Interpretation of Temporal Prediction Switch
- C. LLM Usage in CTPA
- D. Training Settings
- E. Complexity, Overhead, and Statistical Significance
- F. Limitation
- G. Social Impact

A More Analysis on Prediction Instability

A.1 CLIP Self-Training versus a Unimodal Classifier

To assess whether the observed instability is specific to cross-modal adaptation rather than a generic property of self-training, we compare CLIP self-training with a standard unimodal classifier under matched settings. The unimodal classifier uses a ResNet-50 backbone pre-trained on ImageNet with a linear classification head, trained on the same target-domain data with the same rule and budget. The only intended difference is the prediction mechanism: a single visual encoder with a learned linear head, versus a frozen vision-language model with text anchors.

We report the temporal switch rate, i.e., the average fraction of unlabeled target samples whose predicted class changes between consecutive epochs, computed over the same epoch range for both models. Under this matched protocol, CLIP self-training exhibits a switch rate of 0.205, compared with 0.133 for the unimodal classifier. This gap suggests that cross-modal anchoring introduces an additional source of instability beyond the noise typically associated with pseudo-label self-training.

A.2 Measurement Protocol

Unless otherwise specified, all experiments in this section are conducted on Flowers102 under the SSL protocol, using the same labeled support set $\mathcal{D}_{\mathrm{L}}$, unlabeled set $\mathcal{D}_{\mathrm{UL}}$, CLIP ViT-B/32 backbone, and self-training framework. Both settings use the same SSL supervision budget, namely 2 shots per class.

Matched-support anchor diagnostics. We fix the target split and the support set $\mathcal{D}_{\mathrm{L}}$, and compare (i) standard text anchors from CLIP text embeddings under the base prompt, and (ii) target-aligned anchors constructed from $\mathcal{D}_{\mathrm{L}}$.

For the visual-anchor setting, we encode each support image (2-shot per class) using the pre-trained CLIP image encoder $\phi(\cdot)$ and compute a class-wise image prototype by averaging the normalized support features:

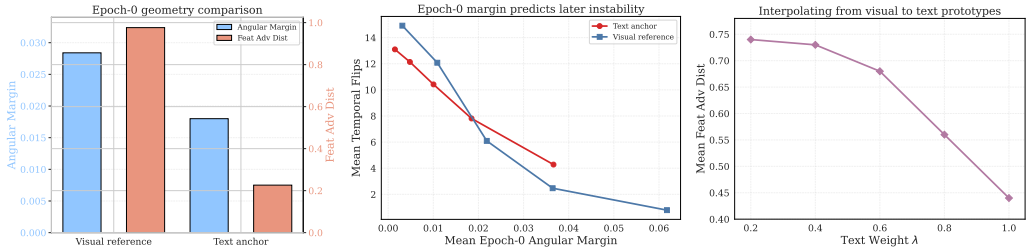
$$\mathbf{w}_c^{\mathrm{vis}} = \mathrm{Normalize}\left(\frac{1}{|\mathcal{D}_{L,c}|}\sum_{\mathbf{x}\in\mathcal{D}_{L,c}}\phi(\mathbf{x})\right), \tag{12}$$

where $\mathcal{D}_{L,c}$ denotes the labeled support images of class c. These prototypes form a target-aligned visual classifier weight matrix $\mathbf{W}^{\mathrm{vis}} = [\mathbf{w}_1^{\mathrm{vis}}, \ldots, \mathbf{w}_C^{\mathrm{vis}}]^\top$. Classification logits are computed by cosine similarity,

$$z_c^{\mathrm{vis}}(\mathbf{x}) = \mathrm{Normalize}(\phi(\mathbf{x}))^\top \mathrm{Normalize}(\mathbf{w}_c^{\mathrm{vis}}), \tag{13}$$

Geometry metrics. We use two geometric metrics: (i) angular margin, i.e., the difference between top-1 and top-2 cosine similarities; and (ii) approximate feature switch distance, i.e., the smallest targeted perturbation norm in feature space that changes the predicted class. Larger values indicate a more robust local decision geometry.

The approximate feature switch distance is computed by an iterative targeted attack in feature space: starting from a clean feature, we optimize an additive perturbation toward the runner-up class and record the first ℓ_2 norm at which the predicted class changes. If no switch occurs within the attack budget, the sample is marked as non-switched.



(a) Epoch-0 geometry. (b) Initial margin vs. later switches. (c) Robustness under interpolation.

Figure 5: Diagnostic analyses of prediction instability.

Table 11: Matched diagnostic training summary on Flowers102.

METRIC	TEXT ANCHOR	VISUAL REFERENCE
EPOCH-0 ACC.	60.64	77.07
FINAL ACC.	84.78	87.69
EPOCH-0 MARGIN	0.0180	0.0284
FINAL MARGIN	0.0360	0.0593
EPOCH-0 SWITCH DIST.	0.2263	0.9762
FINAL SWITCH DIST.	0.3346	1.0736
FINAL SWITCH RATE	15.49%	7.94%

A.3 Initialization Geometry and Training Stability

We compare text anchors and target-aligned visual anchors under matched settings, so that only anchor construction differs.

At epoch 0, before self-training starts, the visual reference has a mean angular margin about $1.58\times$ larger, an approximate feature switch distance more than $4\times$ larger, and an epoch-0 accuracy about 16 points higher than the text anchor baseline (Figures 5a and 1b). This suggests that the source of class anchors strongly affects the initial local decision geometry on target-domain samples.

The visual reference remains consistently more stable throughout adaptation, ending with a lower switch rate (7.94% vs. 15.49%) and higher accuracy (87.69% vs. 84.78%) (Table 11). Target-aligned anchoring thus provides more stable pseudo-label trajectories than standard text anchoring.

We further group target samples into five bins by their epoch-0 angular margin and measure the mean number of temporal switches accumulated during training. Figure 5b shows a clear monotonic trend for both initializations: samples with larger epoch-0 margins undergo fewer switches later in training. Under text anchors, the mean number of switches decreases from 13.10 in the lowest-margin bin to 4.28 in the highest; under the visual reference, it drops from 14.92 to 0.79. The same trend holds when using approximate feature switch distance instead of angular margin. These results suggest that early local geometry is a useful indicator of which samples are likely to remain reliable during self-training.

A.4 Interpolating between Target-Aligned and Text Anchors

The previous analyses compare two endpoint references: purely visual and purely text anchors. To examine whether stability changes gradually with anchor construction, we interpolate between the two. For each class c, we form

$$\mathbf{w}_c^{(\alpha)} = \mathrm{Normalize}\big((1-\alpha)\mathbf{w}_c^{\mathrm{vis}} + \alpha \mathbf{w}_c^{\mathrm{text}}\big), \tag{14}$$

where $\alpha = 0$ corresponds to fully visual anchors and $\alpha = 1$ to purely text anchors.

As shown in Figure 5c, the mean feature switch distance decreases monotonically as the classifier becomes more text-dominated, from about 0.74 at $\alpha = 0$ to 0.44 at $\alpha = 1$. This graded dependence is consistent with anchor construction: as class references become less target-aligned and more text-dominated, local robustness on target-domain samples weakens progressively.

A.5 CTPA and Local Text Geometry

The analyses above explain why visual anchoring can improve stability. We next complement this view from the text side and examine whether improving local text geometry also reduces instability. To avoid selection bias, we define a fixed confusion-prone subset once using the base-prompt text similarities before any attribute augmentation, and keep it unchanged when comparing prompt variants.

Before augmentation, the average switch frequency of samples associated with these confusing classes is 17.6% higher than the overall dataset average; after applying CTPA, this relative excess decreases to 7.26%. On this fixed subset in Flowers102, CTPA reduces the average pairwise similarity of confusing class pairs (0.8478 $\rightarrow$ 0.7387) and increases the corresponding local margin (0.1633 $\rightarrow$ 0.2613), accompanied by a reduction in temporal switch rate from 10.7% to 9.1% (about a 15% relative decrease).

These results support the intended role of CTPA: improving local text-side separation precisely in those neighborhoods where prediction switch is most concentrated. Taken together, the analyses in this appendix support a coherent picture: anchoring mismatch is associated with fragile local decision geometry and higher prediction instability in CLIP self-training. Improving visual anchoring through stability-guided prototypes is associated with lower switch and more reliable pseudo-labels, while text-side refinement provides a complementary effect that is most beneficial in confusion-dense label spaces and is realized in combination with visual adaptation.

B A Geometric Interpretation of Temporal Prediction Switch

This section gives a mechanism-level account of why target-domain visual features can exhibit fragile local decision geometry when classified against fixed text anchors. Our argument links class-conditional anchor offset to (i) a smaller effective angular margin, (ii) a smaller local logit gap, (iii) higher sensitivity to epoch-to-epoch perturbations, and thus (iv) a higher tendency of prediction switch. This is consistent with the matched-support diagnostics in Appendix A.

B.1 CLIP Classification on the Unit Hypersphere

Let $\tilde{\boldsymbol{\phi}}(\mathbf{x})$ and $\tilde{\boldsymbol{\psi}}_c$ denote the ℓ_2-normalized image and class-c text features. CLIP computes the class logit by scaled cosine similarity,

$$z_c(\mathbf{x}) = \frac{\tilde{\boldsymbol{\phi}}(\mathbf{x})^\top \tilde{\boldsymbol{\psi}}_c}{\tau} = \frac{\cos\theta_c}{\tau}, \tag{15}$$

where τ is the temperature and θ_c is the angle between the two features. For a competing pair of classes (c, c'), define the pairwise margin

$$m_{c,c'}(\mathbf{x}) = \langle \tilde{\boldsymbol{\phi}}(\mathbf{x}), \tilde{\boldsymbol{\psi}}_c - \tilde{\boldsymbol{\psi}}_{c'} \rangle. \tag{16}$$

Since the logits are cosine scores scaled by τ, the corresponding pairwise logit gap is exactly

$$z_c(\mathbf{x}) - z_{c'}(\mathbf{x}) = m_{c,c'}(\mathbf{x})/\tau. \tag{17}$$

B.2 Class-Conditional Anchor Offset

Inspired by prior work on modality gap [17], we use a class-conditional anchor offset between target visual features and text anchors as a geometric measure for anchoring mismatch. Our focus is classification-time geometry on a target domain, where each sample is evaluated against fixed text anchors. For each class c, we define

$$\Delta^{(c)}_{\text{anchor}} = \left\| \bar{\boldsymbol{\phi}}_c - \tilde{\boldsymbol{\psi}}_c \right\|_2, \qquad \bar{\boldsymbol{\phi}}_c = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_c}\left[\tilde{\boldsymbol{\phi}}(\mathbf{x})\right], \tag{18}$$

where $\mathcal{D}_c$ denotes the target-domain samples of class c. This quantity measures how far, on average, the target visual cluster of class c is displaced from its text anchor. When $\Delta^{(c)}_{\text{anchor}}$ is large, the text anchor becomes less dominant relative to nearby competing classes, making the local class ranking more fragile on target-domain samples.

B.3 Anchor Offset and Pairwise Margin

We next relate the class-conditional anchor offset to the expected pairwise margin under the text classifier. Taking expectation of the pairwise margin over $\mathbf{x} \sim \mathcal{D}_c$ and writing the class center as $\bar{\boldsymbol{\phi}}_c = \tilde{\boldsymbol{\psi}}_c + \mathbf{e}_c$ gives

$$\mathbb{E}_{\mathbf{x}\sim\mathcal{D}_c}\big[m_{c,c'}(\mathbf{x})\big] = 1 - \left\langle \tilde{\boldsymbol{\psi}}_c, \tilde{\boldsymbol{\psi}}_{c'} \right\rangle + \left\langle \mathbf{e}_c, \tilde{\boldsymbol{\psi}}_c - \tilde{\boldsymbol{\psi}}_{c'} \right\rangle. \tag{19}$$

The first term depends only on the separation between the two text features and becomes small when c and c' are semantically similar. The second term captures how the target visual center is displaced relative to the text anchor: when $\mathbf{e}_c$ has a component toward nearby competing classes, this correction is negative and reduces the expected margin.

Applying Cauchy–Schwarz yields the lower bound

$$\mathbb{E}_{\mathbf{x}\sim\mathcal{D}_c}\big[m_{c,c'}(\mathbf{x})\big] \geq 1 - \left\langle \tilde{\boldsymbol{\psi}}_c, \tilde{\boldsymbol{\psi}}_{c'} \right\rangle - \|\mathbf{e}_c\|_2 \cdot \|\tilde{\boldsymbol{\psi}}_c - \tilde{\boldsymbol{\psi}}_{c'}\|_2. \tag{20}$$

Since $\|\mathbf{e}_c\|_2 = \Delta^{(c)}_{\text{anchor}}$, a larger class-conditional anchor offset weakens this lower bound, especially when the competing class c' is already close in text space.

B.4 From Margin to Logit Gap

By Eq. (17), any reduction in pairwise margin directly reduces the corresponding logit gap. The small-angle case further explains why semantically or geometrically close class pairs often produce especially small gaps.

Let θ_c and $\theta_{c'}$ denote the angles from the image feature to the matched class and a competing class, respectively, and let $\Delta\theta = \theta_{c'} - \theta_c$. For small $\Delta\theta$,

$$z_c - z_{c'} \approx \frac{\sin\theta_c \cdot \Delta\theta}{\tau}, \tag{21}$$

which follows from the first-order expansion of $\cos\theta_{c'} = \cos(\theta_c + \Delta\theta)$. This local form makes the small-gap regime explicit: when the matched class and a confusing competitor are angularly close, the corresponding logit gap under cosine classification is also small.

Empirical support for the small-angle regime. We performed an additional analysis on Flowers102 with zero-shot CLIP. For each sample whose local class ranking switches in our perturbation analysis in Appendix A, we extract the top-1 and top-2 logits and compute the corresponding angles θ_c and $\theta_{c'}$. We find that the mean angular gap $\Delta\theta = \theta_{c'} - \theta_c$ of these samples is only 0.21°. Moreover, for Eq. (21), the absolute residual of the first-order approximation is 2.00×10^{-6}, accounting for only 0.06% of the corresponding logit value. This suggests that the small-$\Delta\theta$ approximation is well aligned with the local regime most relevant to prediction switch, and thus serves as a sanity check for Eq. (21) in the switch-prone regime.

B.5 From Small Logit Gap to Higher Local Sensitivity

The following proposition formalizes a local robustness property of the pairwise logit gap: when the gap is small, only a small feature-space perturbation is needed to change the local pairwise ordering between a matched class and a nearby competitor. We consider an additive perturbation $\boldsymbol{\delta}$ in normalized feature space; re-normalization preserves the sign of the gap because the normalization factor is strictly positive, so the condition below also applies to the exact re-normalized cosine classifier.

Proposition B.1. *Let*

$$g_{c,c'}(\mathbf{x};\boldsymbol{\delta}) = \frac{\left\langle \tilde{\boldsymbol{\phi}}(\mathbf{x}) + \boldsymbol{\delta}, \tilde{\boldsymbol{\psi}}_c - \tilde{\boldsymbol{\psi}}_{c'} \right\rangle}{\tau} \tag{22}$$

denote the pairwise logit gap between classes c and c' under perturbation $\boldsymbol{\delta}$, where $\tilde{\boldsymbol{\phi}}(\mathbf{x})$ and $\tilde{\boldsymbol{\psi}}_c$ are ℓ_2-normalized image and text features. Then

$$|g_{c,c'}(\mathbf{x};\boldsymbol{\delta}) - g_{c,c'}(\mathbf{x};\mathbf{0})| \leq \frac{\|\boldsymbol{\delta}\|_2 \cdot \|\tilde{\boldsymbol{\psi}}_c - \tilde{\boldsymbol{\psi}}_{c'}\|_2}{\tau}. \tag{23}$$

In particular, if $g_{c,c'}(\mathbf{x};\mathbf{0}) > 0$ *and*

$$\|\boldsymbol{\delta}\|_2 < \frac{\tau\, g_{c,c'}(\mathbf{x};\mathbf{0})}{\|\tilde{\boldsymbol{\psi}}_c - \tilde{\boldsymbol{\psi}}_{c'}\|_2}, \tag{24}$$

then $g_{c,c'}(\mathbf{x};\boldsymbol{\delta}) > 0$.

Proof. By definition,

$$g_{c,c'}(\mathbf{x};\boldsymbol{\delta}) - g_{c,c'}(\mathbf{x};\mathbf{0}) = \frac{\left\langle \boldsymbol{\delta}, \tilde{\boldsymbol{\psi}}_c - \tilde{\boldsymbol{\psi}}_{c'} \right\rangle}{\tau}. \tag{25}$$

Cauchy–Schwarz gives Eq. (23). Now assume $g_{c,c'}(\mathbf{x};\mathbf{0}) > 0$ and Eq. (24) holds. Then $|g(\boldsymbol{\delta}) - g(\mathbf{0})| < g(\mathbf{0})$, so

$$g_{c,c'}(\mathbf{x};\boldsymbol{\delta}) \geq g_{c,c'}(\mathbf{x};\mathbf{0}) - |g_{c,c'}(\mathbf{x};\boldsymbol{\delta}) - g_{c,c'}(\mathbf{x};\mathbf{0})| > 0. \tag{26}$$

Hence the sign of the pairwise gap is preserved. □

Proposition B.1 gives a deterministic local robustness criterion: the critical perturbation radius in Eq. (24) is proportional to the original pairwise logit gap. Samples in a small-gap regime therefore admit a smaller robustness radius. In self-training, such perturbations need not be adversarial; they may arise from epoch-to-epoch feature drift, stochastic optimization, or feedback from imperfect pseudo-label supervision. Samples that repeatedly remain in a small-gap regime under text anchor classification are thus plausibly more prone to changing their top-1 class across epochs, exhibiting temporal prediction switch.

B.6 Geometric Interpretation

Text anchors partition the unit hypersphere into spherical Voronoi cells. A large class-conditional anchor offset does not mean that target samples sit near cell boundaries. It does mean that target features are, on average, less centered around their text anchors, so more of them can fall in off-center regions of the cell, where the margin against a nearby class is small. This matches the matched-support diagnostic in Appendix A: swapping text anchors for visual anchors enlarges angular margin and feature-space robustness, and lowers switch rates during training.

Proposition B.1 shows that a small logit gap leaves little room before a perturbation flips the class ranking. The analysis above shows that anchor offset makes such small-gap configurations more common on target-domain samples. Anchoring mismatch therefore raises the prevalence of fragile configurations that are prone to temporal prediction switch, without determining the outcome for any single sample. This is the regime SAST is designed to detect and mitigate.

B.7 Stability and Pseudo-Label Noise

The geometric analysis above explains why temporal prediction switch arises. Here we ask why reducing it helps adaptation. On CUB, the pseudo-label error rate falls monotonically with stability, from 97.6% in the lowest-stability bin to 7.8% in the highest (Figure 4), and stability is complementary to confidence: among high-confidence samples, accuracy is 93.13% for high-stability samples but only 70.55% for low-stability ones. High-stability samples are thus a cleaner source of pseudo-label supervision than confidence alone would suggest.

Prioritizing them reduces the pseudo-label noise that enters self-training. Pseudo-label errors can accumulate across training steps and degrade target performance [9]; lower switch rates indicate more stable predictions, which in turn yield more reliable pseudo-labels and less error accumulation.

C LLM Usage in CTPA

C.1 LLM Setup

We use **Gemini 2.5 Pro** as the LLM for offline attribute generation. For each class, the LLM returns a JSON list of candidate attributes, from which CTPA selects the subset that is least compatible with

Table 12: CTPA performance with different attribute-generating LLMs.

LLM	CUB	FLOWERS102	DTD	EUROSAT	RESISC45	FGVC-AIRCRAFT
GEMINI-2.5-PRO	66.1	90.9	63.4	94.7	85.3	25.2
QWEN3.5-0.8B	64.5	90.0	62.5	93.9	85.1	24.8

the confusion neighbors of the class according to Eq. (6)–(7) in the main text. The LLM is instructed to produce attributes that emphasize what makes a class visually recognizable and different from other classes in the same domain.

In all settings, including TRZSL, the attribute pool is generated from class names only, without access to instance labels, ground-truth descriptions, or any held-out annotations. For unseen classes in TRZSL, attributes are generated before adaptation begins and use only the class names; no unseen images, pseudo-labels, or target-domain statistics are provided to the LLM. This matches the information available to the baselines. Generated attribute pools are released with the code.

C.2 Prompt

We use the following prompt template for offline attribute generation:

> Your goal is to create a single JSON object containing a list of 10–15 distinctive attributes for each class provided in the input. Distinctive attributes are features that uniquely identify a class and differentiate it from other classes within its domain. Focus on what makes it special, different, or easily recognizable compared to others.
>
> Example:
>
> Input: "Wilson Warbler"
>
> Output: { "Wilson Warbler": ["The roundness of the cap on the top of the head", "The olive-green on the back is a slightly dark shade", "The little yellow bird with a black cap"] }
>
> New Task:
>
> Now, generate the distinctive attributes for all the following classes. Ensure that you provide 10–15 attributes for each class in a single JSON object.

C.3 LLM- and attribute-level robustness of CTPA

Robustness to the Choice of LLM. We repeat CTPA using attributes generated by Qwen3.5-0.8B, a 1B-scale model, under the same prompting protocol. Replacing Gemini-2.5-Pro with Qwen3.5-0.8B decreases average accuracy by only 0.8% across the six datasets (Table 12). The largest degradation appears on fine-grained datasets such as CUB and Flowers102, where the smaller model produces more generic attributes; on DTD and EuroSAT, performance remains close to Gemini-2.5-Pro. Even with this weaker LLM, CTPA still improves over CAP by 2.1% on average, suggesting that the gain of SAST does not rely on a frontier LLM.

Which Types of Attributes Are Effective? For a confusing class pair (A, B), we measure the similarity drop

$$\Delta_{\text{sim}} = \text{sim}_{\text{base}}(A, B) - \text{sim}_{\text{typed}}(A, B), \tag{27}$$

when only one type of attribute is used. As shown in Table 13, the most effective type is domain- and pair-dependent: on Flowers102, color attributes produce the largest separation, while on CUB no single type dominates. Selecting a small discriminative subset also outperforms stacking all attributes, which supports the design of CTPA.

D Training Settings

The detailed training settings used in the main experiments are summarized in Table 14.

Table 13: Δ_{sim} by attribute types (10 confusing pairs each).

FLOWERS102						
TYPE	COLOR	SHAPE	TEXTURE	PART	COUNT	HABIT
MEAN Δ_{sim}	0.297	0.184	0.171	0.244	0.164	0.179

CUB					
TYPE	COLOR	SHAPE	TEXTURE	PATTERN	ALL ATTRIBUTES
MEAN Δ_{sim}	0.056	0.039	0.224	0.049	0.033

Table 14: Detailed settings.

	Flowers102	RESISC45	DTD	CUB	EuroSAT	FGVC-Aircraft
Class number	102	45	47	200	10	100
Training set size	2040	6300	3760	5594	27000	6667
Testing set size	6149	25200	1880	5794	5000	3333
Network	ViT-B/32					
Batch size	32					
Epochs	50					
Optimizer	SGD					
SGD momentum	0.9					
Learning rate	0.02					
Weight decay	0.05					
Scheduler	CosineAnnealingLR					

E Complexity, Overhead, and Statistical Significance

All measurements below are on DTD using a single NVIDIA A100 GPU under the same backbone and training budget.

Computational cost. SAST introduces two additional sources of overhead: (i) epoch-wise prototype refresh in SWIP, and (ii) dual-stream logit computation followed by fusion. The refresh is performed once per epoch with class-wise ranking, so its cost is moderate. Table 15 reports the comparison: FLOPs are nearly identical to CAP (188.297 vs. 188.275 GFLOPs), and training time is 64 min on DTD, comparable to CAP (62 min).

Memory and storage overhead. The additional memory cost comes from the prediction-history queue and the class-wise prototype matrix. The storage complexity is $\mathcal{O}(NQ + CD)$, where $Q \leq 50$ is the history length and C is the number of classes. Because the queue stores only predicted class labels and lightweight statistics rather than full features, increasing the budget beyond the current 8 GiB would not change what is stored or how stability is estimated; we therefore do not expect additional accuracy gains from a larger budget. Peak GPU memory is 8206 MiB, compared with 8019 MiB for CAP (Table 15).

Statistical significance of small gains. For two settings where the improvement over CAP is small, we repeat each run with 10 random seeds (1–10) under identical configurations. Paired t-tests yield $p = 0.0004$ (DTD-TRZSL, ViT-B/32, mean gain $+0.334$) and $p = 0.0276$ (DTD-SSL, ViT-L/14, mean gain $+0.457$), with Cohen's d of 1.56 and 0.697, respectively. The absolute gains are small but statistically significant and occur on already strong baselines.

F Limitation

This work has several limitations. First, our analysis is partly formal at the local geometric level, but it is not yet a full theoretical characterization of self-training dynamics. In particular, the interaction among temporal stability, pseudo-label noise accumulation, branch-wise reliability, and

Table 15: Complexity comparison.

METHOD	GFLOPS	TRAINING TIME	PEAK GPU MEMORY
CAP	188.275	62 MIN	8019 MIB
SAST	188.297	64 MIN	8206 MIB

Table 16: Storage cost.

DATASET	COST
CUB	4.5 MB
EUROSAT	21 MB

representation drift remains to be studied more systematically. Second, although SAST performs well on the evaluated benchmarks, its scalability to larger label spaces, larger unlabeled pools, and broader deployment settings has not yet been fully established. In particular, settings where prediction histories are unavailable, costly to maintain, or only partially observable may require modified stability estimators. Third, because SAST is implemented as a PEFT-based adaptation method with pseudo-label self-training, its strongest gains should be expected in regimes where the target domain still lies within the transferable range of the pre-trained representation and the main bottleneck is adaptation rather than missing semantics. Accordingly, its ability to correct severe class-level mismatch inherited from pretraining may remain limited when the target domain is too far from CLIP pretraining or when the initial text-side semantics are strongly biased relative to the target domain. Exploring stronger combinations with explicit alignment or representation-level adaptation is an important direction for future work.

G Social Impact

By improving the reliability of self-training for vision-language models, this work may help reduce annotation cost and improve adaptation to new visual domains. However, better adaptation can also make such systems easier to deploy in high-stakes or weakly monitored settings. If the target data contain systematic biases, stability-aware filtering may preserve or even reinforce biased pseudo-labels rather than remove them. In addition, LLM-generated attributes may inherit biases from the language model. We therefore recommend careful evaluation before deployment, especially in sensitive or high-impact applications.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

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 include detailed information in Section 1.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The paper includes a dedicated limitations section (Appendix F).

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: The paper includes a local robustness proposition together with its assumptions and a complete proof in Appendix B.5 (Proposition B.1). The geometric interpretation and assumptions used for the analysis are described in Appendix B.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: The paper specifies the datasets, learning settings, backbone, training protocol, optimization setup, ablations, and diagnostic analyses needed to reproduce the reported results. Key implementation details are given in Section 4 and Appendix D.

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: Our code is available at `https://github.com/lsy-2003/SAST`.

   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 provide anonymized code, configuration files, and instructions for dataset preparation and execution, together with the implementation details in Section 4 and Appendix D.

   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: Main experimental results are reported as mean $\pm$ standard deviation over three independent runs with different random seeds, as stated in Sec. 4 and Table 1. This quantifies variability due to stochastic training and data sampling.

   Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

Justification: The paper reports the hardware used for the experiments (single NVIDIA A100 GPU), and Appendix E provides runtime, GFLOPs, peak GPU memory, and storage overhead for representative runs.

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: We have reviewed the NeurIPS Code of Ethics and, to the best of our knowledge, the research complies with it. The submission uses public datasets and pretrained models and does not involve deceptive, unsafe, or non-consensual experimental procedures.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: The paper includes a social impact section (Appendix G) discussing both potential benefits and possible risks.

    Guidelines:

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

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The paper does not release high-risk generative models, scraped datasets, or other assets that would require special misuse safeguards. The method is an adaptation framework built on existing public vision-language models and benchmark datasets.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: All data, models, and code in the paper respect the license.

    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: [N/A]

    Justification: The paper does not release new assets.

    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 experiments or 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 human subjects research or crowdsourcing, so IRB approval or equivalent review was not required.

    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: [Yes]

    Justification: LLM usage is part of the method: Gemini 2.5 Pro is used offline to generate candidate class attributes for CTPA. Details, prompt template, and reproducibility notes are provided in Appendix C.

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.