# MetaTPT: Meta Test-time Prompt Tuning for Vision-Language Models

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

Vision-language models (VLMs) such as CLIP exhibit strong zero-shot generalization but remain sensitive to domain shifts at test time. Test-time prompt tuning (TPT) mitigates this issue by adapting prompts with fixed augmentations, which may falter in more challenging settings. In this work, we propose Meta Test-Time Prompt Tuning (MetaTPT), a meta-learning framework that learns a self-supervised auxiliary task to guide test-time prompt tuning. The auxiliary task dynamically learns parameterized augmentations for each sample, enabling more expressive transformations that capture essential features in target domains. MetaTPT adopts a dual-loop optimization paradigm: an inner loop learns a self-supervised task that generates informative views, while the outer loop performs prompt tuning by enforcing consistency across these views. By coupling augmentation learning with prompt tuning, MetaTPT improves test-time adaptation under domain shifts. Extensive experiments demonstrate that MetaTPT achieves state-of-the-art performance on domain generalization and cross-dataset benchmarks.

## 1 Introduction

Vision-language models (VLMs) [25, 34, 2, 63] such as CLIP [45] have exhibited strong zero-shot generalization. By pretraining on large-scale image-text corpora, CLIP learns a joint embedding space that aligns visual and textual representations. In zero-shot classification, an image is assigned to the class whose textual description—often instantiated using a template such as *"a photo of a {class}"*—has the highest similarity with the image embedding. While this approach obviates the need for task-specific fine-tuning, zero-shot performance is contingent on the target domain following a similar distribution to its source domain. When there is a domain shift, its performance on the target domain will drop substantially.

To mitigate the impact of domain shifts, Test-Time Adaptation (TTA) have been proposed to enable models to self-adapt during inference. Instead of freezing the model after training, TTA updates certain model components by optimizing unsupervised objectives [58, 35, 36] without access to labeled data from source domain. Among these, Test-Time Training (TTT) [55] introduces a self-supervised auxiliary task—such as image rotation prediction—optimized at test time, allowing the model to adjust its representations to new domains.

Building on this paradigm, recent work [11, 1, 64, 26, 66] such as Test-time Prompt Tuning (TPT) [51] extend TTA to VLMs, which adapts learnable prompts [68] on-the-fly while keeping the pretrained image and text encoders frozen. During inference, TPT employs fixed data augmentations [18] to generate a batch of augmented views for each test sample, exposing the model to diverse visual variations during adaptation. The learnable prompts is then optimized through entropy minimization across these views, encouraging the model to produce confident predictions in the new domain.

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

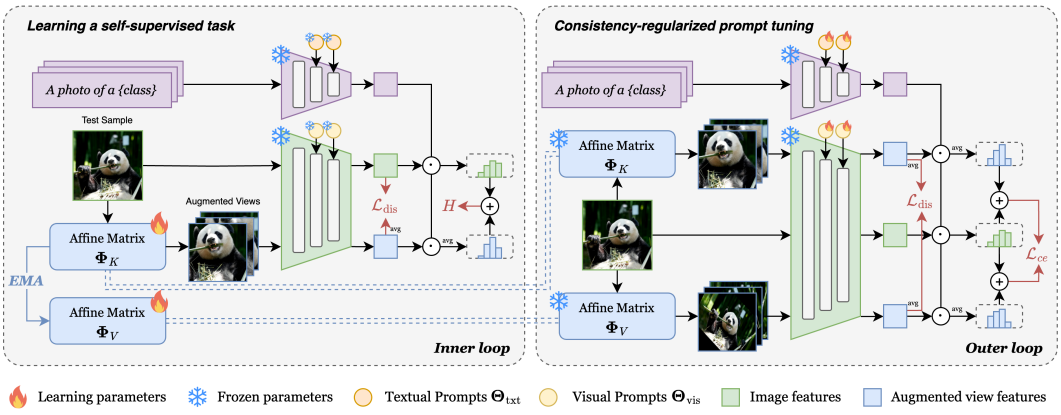


Figure 1: **Overview of MetaTPT**. This framework is formulated as a dual-loop meta-adaptation process. In the *inner loop*, a self-supervised auxiliary task is learned to dynamically optimize parameterized augmentations: $\mathbf{\Phi}_K$ is updated through a joint objective combining $H$ and $\mathcal{L}_{\text{dis}}$ to produce informative views, while $\mathbf{\Phi}_V$ employs an EMA of $\mathbf{\Phi}_K$ to prevent collapse. In the *outer loop*, the learnable prompts $\mathbf{\Theta} = \{\mathbf{\Theta}_{\text{txt}}, \mathbf{\Theta}_{\text{vis}}\}$ by enforcing cross-view consistency, using a combined loss of $\mathcal{L}_{ce}$ and $\mathcal{L}_{\text{dis}}$ over the generated views.

Inspired by TTT [55], we propose an extension to TPT [51] that introduces a learnable self-supervised auxiliary task at test time. We observe the fix augmentations used in TPT may fail to capture the nuanced features necessary for discrimination, leading to unreliable adaptation especially in challenging target domains. To overcome these limitations, the self-supervised auxiliary task is designed to dynamically learn augmentations during adaptation rather than manually defined. These augmentations are parameterized as differentiable affine transformations, enabling spatial operations—such as scaling, shearing, flipping, and translation. This adaptability allows the model to capture complex, sample-specific variations such as subtle spatial distortions, shape deformations, or domain-specific artifacts that are common in challenging domains.

To effectively co-adapt parameterized augmentations with prompts, we formulate the adaptation as a bi-level optimization problem, naturally interpreted as a form of dual-loop meta-learning [12]. Building on this formulation, we introduce **Meta** **T**est-time **P**rompt **T**uning (**MetaTPT**) for vision-language models. In this framework, the *inner loop* optimizes a self-supervised task that dynamically generates adaptive augmentations, while the *outer loop* updates the prompts by enforcing consistency across these views. Following TPT [51], our adaptation is performed online, with learnable parameters optimized for each test sample. Thus, MetaTPT serves as a meta-adaptation mechanism within a single test stream, where both loops operate per sample to improve generalization under domain shift.

Empirically, we evaluate MetaTPT on two zero-shot generalization benchmarks (domain generalization and cross-dataset evaluation) against four ImageNet-trained models (CLIP [45], CoOP [68], MaPLe [27] and MMRL [15]). Compared with TPT, MetaTPT demonstrates strong performance to domain shift and achieves more improvements on challenging datasets, highlighting the effectiveness of learnable augmentations in adapting to complex distribution changes. Ablation studies further confirm the effectiveness of MetaTPT's design. First, compared to the fixed augmentations used in TPT, our learnable augmentations outperforms fixed augmentations all four base models, validating their generality across target domains. Second, our dual-loop meta-learning framework (interleaved optimization of augmentations and prompts) outperforms one-stage joint optimization, emphasizing the benefits of meta-learning. Finally, our online adaptation strategy (meta-adapts augmentations and prompts per sample) surpasses an offline variant (meta-train augmentations across all samples before adapting prompts individually), underscoring the importance of online augmentation adaptation.

In summary, (1) We propose MetaTPT, a novel meta-learning framework that jointly adapts learnable prompts and a self-supervised auxiliary task at test time, thereby enhancing zero-shot generalization of vision-language models. (2) We introduce a self-supervised task to dynamically optimize augmentations, parameterized as affine transformations, enabling the capture of fine-grained variability within target domains. (3) We develop consistency-regularized prompt tuning that fosters stable and robust adaptation by enforcing alignment of representations across the learned views.

## 2 Preliminaries

**Prompt tuning for VLMs.** CLIP [45] consists of two main components: a text encoder $g$ and an image encoder $f$. Let $\mathbf{t}_y$ denote a handcrafted prompt, such as *"A photo of a {class}"*, and let $\mathbf{x}$ represent an image. The probability of class $y$ given the image $\mathbf{x}$ is:

$$P(\mathbf{x}) = \frac{\exp\left(\text{sim}\left(f\left(\mathbf{x}\right), g(\mathbf{t}_y)\right)/\tau\right)}{\sum_{i=1}^{N_c} \exp\left(\text{sim}\left(f\left(\mathbf{x}\right), g(\mathbf{t}_i)\right)/\tau\right)}\,. \tag{1}$$

where $\text{sim}(a,b) = \frac{a^\top b}{\|a\|\|b\|}$ represents the cosine similarity, $\tau$ denotes the temperature, and $N_c$ is the class number. To adapt CLIP to downstream tasks, prompt-learning methods introduce learnable parameters that modify the input embeddings of the frozen encoders. In CoOp [68], a set of learnable text prompt embeddings $\mathbf{\Theta} = \{\mathbf{\Theta}^1, \mathbf{\Theta}^2, \cdots, \mathbf{\Theta}^n\}$ replaces the handcrafted context. MaPLe [27] extends this paradigm by incorporating learnable prompts $\mathbf{\Theta} = \{\mathbf{\Theta}_{\text{txt}}, \mathbf{\Theta}_{\text{vis}}\}$ for both modalities, where $\mathbf{\Theta}_{\text{txt}} = \{\mathbf{\Theta}_{\text{txt}}^{(l)}\}_{l=1}^{L}$ and $\mathbf{\Theta}_{\text{vis}} = \{\mathbf{\Theta}_{\text{vis}}^{(l)}\}_{l=1}^{L}$ are inserted across multiple transformer layers, and cross-modal alignment is enforced via a linear projection. MMRL [15] further generalizes this framework by inserting learnable prompts in the higher layers of both encoders, with $\mathbf{\Theta}_{\text{txt}} = \{\mathbf{\Theta}_{\text{txt}}^{(l)}\}_{l=J}^{L}$ and $\mathbf{\Theta}_{\text{vis}} = \{\mathbf{\Theta}_{\text{vis}}^{(l)}\}_{l=J}^{L}$, allowing independent optimization while maintaining alignment through the contrastive objective.

**Test-time prompt tuning.** TPT [51] is a TTA method that optimizes the the learnable prompts $\mathbf{\Theta}$ during inference. For each test sample, it employs fix data augmentation [18] $\mathbf{\Phi}$ to generate $N$ augmented views and selects top-$\rho$ confident views for optimization:

$$\min_{\mathbf{\Theta}} H\left(\tilde{P}_{\mathbf{\Theta}}\left(\mathbf{\Phi}(\mathbf{x})\right)\right), \tag{2}$$

$$\text{where} \quad \tilde{P}_{\mathbf{\Theta}}\left(\mathbf{\Phi}(\mathbf{x})\right) = \frac{1}{\rho N}\sum_{i=1}^{N} \mathbb{1}\left[H(P_i) \leqslant \delta\right] P_{\mathbf{\Theta}}\left(\mathbf{\Phi}^i(\mathbf{x})\right). \tag{3}$$

Here, $H(P) = \sum_{i=1}^{N_c} P_i \log P_i$. $\mathbf{\Phi}^i(\mathbf{x})$ denotes the $i$-th augmented view of test sample $\mathbf{x}$, and $\delta$ is the entropy threshold at the $\rho$-th confidence percentile. The learnable prompts $\mathbf{\Theta}$ are optimized by minimizing the average entropy of the selected augmented views.

**Meta-learning.** MAML (Model Agnostic Meta Learning) [12] learns a model initialization $\mathbf{\Theta}$ that can quickly adapt to new tasks through a dual-loop optimization:

$$\min_{\mathbf{\Theta}} \ \mathbb{E}_{\mathcal{T}_i \sim p(\mathcal{T})}\left[\mathcal{L}_{\mathcal{T}_i}(\mathbf{\Phi}_i)\right], \tag{4}$$

$$\text{s.t. } \mathbf{\Phi}_i = \arg\min_{\mathbf{\Phi}} \mathcal{L}_{\mathcal{T}_i}(\mathbf{\Phi}). \tag{5}$$

Here, Eq. (5) represents the *inner loop*, corresponding to task-specific adaptation, where $\mathbf{\Phi}_i$ is optimized for task $\mathcal{T}_i$. Eq. (4) defines the *outer loop*, updating the meta-parameters $\mathbf{\Theta}$ to minimize the loss across tasks after adaptation. *Sun et al.* [54] extend this framework to the test-time regime. By modeling the RNN hidden state itself with a learnable model parameterized by low-rank matrices $\mathbf{\Phi}_K$ and $\mathbf{\Phi}_V$, which serve as an *inner-loop* learner updated dynamically with incoming inputs. The *outer loop* then optimizes the model parameters $\mathbf{\Theta}$ so that the inner-loop updates lead to effective sequence modeling.

## 3 Method

In this section, we introduce **Meta** **T**est-time **P**rompt **T**uning (**MetaTPT**), a dual-loop meta-learning framework for adapting vision-language models. The *inner loop* (Sec. 3.1) learns a self-supervised auxiliary task to optimize parameterized augmentations generating confident and semantically consistent views. The *outer loop* (Sec. 3.2) updates the learnable prompts by enforcing prediction- and feature-level consistency across these views, enhancing robustness and generalization under domain shifts.

### 3.1 Learning a self-supervised task

Inspired by TTT [55], we introduce a self-supervised auxiliary task for test-time prompt tuning (TPT) [51] to improve its adaptability on unseen target domain. Although TPT effectively adjusts frozen models through prompt optimization, its reliance on fixed augmentation [18] limits the effectiveness in capturing discriminative features. To mitigate it, we formulate the self-supervised task as a learnable augmentation strategy. Rather than applying pre-defined transformations, we parameterize the augmentation as an affine transformation $\Phi(\mathbf{x})$:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \Phi \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \quad \text{where } \Phi = \begin{bmatrix} a & b & t_x \\ c & d & t_y \end{bmatrix}. \tag{6}$$

where $(x, y)$ is the original coordinates of image $\mathbf{x}$, and $(x', y')$ is the transformed coordinate. The affine matrix $\Phi \in \mathbb{R}^{2\times3}$ allows the model to optimize spatial transformations, such as flipping, scaling, aspect ratio adjustments (controlled by $a$ and $d$), shearing (controlled by $b$ and $c$), and translation (controlled by $t_x$ and $t_y$) in a differentiable manner, producing augmented views that are most beneficial for prompt adaptation on each test sample. Following TPT [51], we generate a batch of $N$ augmented views per sample to introduce diversity, denoting the set of affine matrices as $\mathbf{\Phi} = \{\Phi^i\}_{i=1}^N \in \mathbb{R}^{N\times2\times3}$.

To ensure the augmented views effectively guide subsequent prompt adaptation, $\mathbf{\Phi}$ are optimized to satisfy two objectives. First, they should generate augmented views with high confidence by minimizing the prediction entropy:

$$H(\mathbf{x}; \mathbf{\Phi}; \mathbf{\Theta}) = -\sum_{i=1}^{N_c} \hat{P}^i_{\mathbf{\Theta}}(\mathbf{\Phi}(\mathbf{x})) \log \hat{P}^i_{\mathbf{\Theta}}(\mathbf{\Phi}(\mathbf{x})), \tag{7}$$

$$\text{where} \quad \hat{P}_{\mathbf{\Theta}}(\mathbf{\Phi}(\mathbf{x})) = P_{\mathbf{\Theta}}(\mathbf{x}) + \tilde{P}_{\mathbf{\Theta}}(\mathbf{\Phi}(\mathbf{x})). \tag{8}$$

Here, the composite entropy over both the input $\mathbf{x}$ and its augmented views $\mathbf{\Phi}(\mathbf{x})$ empirically outperforms optimizing the views alone. Second, the augmentations should avoid semantic bias by preserving the intrinsic content of the original image. This is enforced by minimizing the feature-level discrepancy between the original input $\mathbf{x}$ and the averaged features of its augmented views $\mathbf{\Phi}(\mathbf{x})$:

$$\mathcal{L}_{\text{dis}}(\mathbf{x}; \mathbf{\Phi}; \mathbf{\Theta}) = \left\| f_{\mathbf{\Theta}}(\mathbf{x}) - \bar{f}_{\mathbf{\Theta}}(\mathbf{\Phi}(\mathbf{x})) \right\|_2, \tag{9}$$

$$\text{where} \quad \bar{f}(\mathbf{\Phi}_{\mathbf{\Theta}}(\mathbf{x})) = \frac{1}{N} \sum_{i=1}^{N} f_{\mathbf{\Theta}}(\mathbf{\Phi}^i(\mathbf{x})). \tag{10}$$

These two items are combined to optimize $\mathbf{\Phi}$ in the inner loop:

$$\mathcal{L}_{\text{inner}} = H(\mathbf{x}; \mathbf{\Phi}; \mathbf{\Theta}) + \mathcal{L}_{\text{dis}}(\mathbf{x}; \mathbf{\Phi}; \mathbf{\Theta}). \tag{11}$$

We introduce two sets of affine matrices, $\mathbf{\Phi}_K$ and $\mathbf{\Phi}_V$, each initialized with distinct augmentation types to generate diverse input patterns for prompt adaptation. Directly optimizing may lead them to converge toward similar transformation patterns, reducing augmentation diversity. To prevent this, we optimize $\mathbf{\Phi}_K$ via Eq. (11), while $\mathbf{\Phi}_V$ is updated using an Exponential Moving Average (EMA) [21] of $\mathbf{\Phi}_K$:

$$\mathbf{\Phi}_V \leftarrow \alpha \mathbf{\Phi}_V + (1 - \alpha) \mathbf{\Phi}_K. \tag{12}$$

where $\alpha$ is the momentum coefficient. Note that in the inner loop, only the affine matrices $\mathbf{\Phi}_K$ and $\mathbf{\Phi}_V$ are updated, while the learnable prompts $\mathbf{\Theta}$ remain frozen.

### 3.2 Consistency-regularized prompt tuning

Entropy-based objectives, as employed in TPT, can promote confident predictions; however, under challenging domain shifts, they often produce overconfident yet inaccurate outputs. To mitigate this, MetaTPT enforces consistency between two sets of learned views, $\mathbf{\Phi}_K(\mathbf{x})$ and $\mathbf{\Phi}_V(\mathbf{x})$, providing a reliable supervisory signal for optimizing the learnable prompts $\mathbf{\Theta}$.

Predictive consistency $\mathcal{L}_{ce}$ is first enforced in the probability space through a cross-entropy loss between the soft predictions from two distinct sets of augmented views:

$$\mathcal{L}_{ce}(\mathbf{x}; \mathbf{\Phi}_K; \mathbf{\Phi}_V; \mathbf{\Theta}) = -\sum_{i=1}^{N_c} \hat{P}^i_{\mathbf{\Theta}}(\mathbf{\Phi}_K(\mathbf{x})) \log \hat{P}^i_{\mathbf{\Theta}}(\mathbf{\Phi}_V(\mathbf{x})). \tag{13}$$

Semantic consistency $\mathcal{L}_{\text{dis}}$ is further enforced in the feature space by minimizing the distance between the average feature representations of the two augmented views:

$$\mathcal{L}_{\text{dis}}(\mathbf{x}; \mathbf{\Theta}_K; \mathbf{\Theta}_V; \mathbf{R}) = \|\bar{f}_{\mathbf{\Theta}}(\mathbf{\Phi}_K(\mathbf{x})) - \bar{f}_{\mathbf{\Theta}}(\mathbf{\Phi}_V(\mathbf{x}))\|_2. \tag{14}$$

The learnable prompts $\mathbf{\Theta}$ are jointly optimized by two objectives, while $\mathbf{\Phi}_K$ and $\mathbf{\Phi}_V$ remain fixed:

$$\mathcal{L}_{\text{outer}} = \mathcal{L}_{ce}(\mathbf{x}; \mathbf{\Phi}_K; \mathbf{\Phi}_V; \mathbf{\Theta}) + \mathcal{L}_{\text{dis}}(\mathbf{x}; \mathbf{\Phi}_K; \mathbf{\Phi}_V; \mathbf{\Theta}). \tag{15}$$

### 3.3 Dual-loop meta adaptation

For each test sample $\mathbf{x}$ from the target domain, the model is adapted using a dual-loop meta-learning framework:

$$\min_{\mathbf{\Theta}} \mathbb{E}_{\mathbf{x}\sim p(\mathcal{X})}\left[\mathcal{L}_{\text{outer}}(\mathbf{x}; \mathbf{\Phi}; \mathbf{\Theta})\right], \tag{16}$$

$$\text{s.t. } \mathbf{\Phi} = \arg\min_{\mathbf{\Phi}} \mathcal{L}_{\text{inner}}(\mathbf{x}; \mathbf{\Phi}; \mathbf{\Theta}). \tag{17}$$

Here, Eq. (17) defines the *inner loop*, which minimizes Eq. (11) to learn a self-supervised task, thereby optimizing the learnable augmentations $\mathbf{\Phi}$ to generate informative views for sample $\mathbf{x}$. Eq. (16) represents the *outer loop*, which updates the learnable prompts $\mathbf{\Theta}$ by minimizing Eq. (15). This framework enables sample-wise adaptation of the test stream, enhancing zero-shot generalization.

After adaptation, the final prediction is obtained by aggregating information from the original input $\mathbf{x}$ and its selected augmented views.

**Algorithm 1** MetaTPT: Meta Test-Time Prompt Tuning

**Require:** Target distribution $p(\mathcal{X})$; learnable prompts $\mathbf{\Theta}$; batch size $N$; loop steps $T$, $M$; learning rate $\eta_i, \eta_o$; EMA momentum $\alpha$; weights $\lambda_K, \lambda_V$.

**for** all test sample $\mathbf{x} \sim p(\mathcal{X})$) **do**
  Initialize $\mathbf{\Theta}$ from a ImageNet-pretrained model.
  Initialize $\mathbf{\Phi}_K^{i\in[1,N]}$ using Eq. (20), and $\mathbf{\Phi}_V^{i\in[1,N]}$ using Eq. (19).
  **for** $m = 1$ to $M$ **do**
    **for** $t = 1$ to $T$ **do**
      // Inner loop: learning a self-supervised task
      $\mathcal{L}_{\text{inner}} = H(\mathbf{x}; \mathbf{\Phi}_K; \mathbf{\Theta}) + \mathcal{L}_{\text{dis}}(\mathbf{x}; \mathbf{\Phi}_K; \mathbf{\Theta})$.
      Update $\mathbf{\Phi}_K \leftarrow \mathbf{\Phi}_K - \eta_i \nabla_{\mathbf{\Phi}_K} \mathcal{L}_{\text{inner}}$.
      Update $\mathbf{\Phi}_V \leftarrow \alpha \mathbf{\Phi}_V + (1-\alpha)\mathbf{\Phi}_K$.
    **end for**
    // Outer loop: consistency-regularized prompt tuning
    $\mathcal{L}_{\text{outer}} = \mathcal{L}_{ce}(\mathbf{x}; \mathbf{\Phi}; \mathbf{\Theta}) + \mathcal{L}_{\text{dis}}(\mathbf{x}; \mathbf{\Phi}; \mathbf{\Theta})$.
    Update $\mathbf{\Theta} \leftarrow \mathbf{\Theta} - \eta_o \nabla_{\mathbf{\Theta}} \mathcal{L}_{\text{outer}}$.
  **end for**
  $\hat{P}(\mathbf{x}) = P_{\mathbf{\Theta}}(\mathbf{x}) + \lambda_K \tilde{P}_{\mathbf{\Theta}}(\mathbf{\Phi}_K(\mathbf{x})) + \lambda_V \tilde{P}_{\mathbf{\Theta}}(\mathbf{\Phi}_V(\mathbf{x}))$.
**end for**

$$\hat{P}_{\mathbf{\Theta}}(\mathbf{x}) = P_{\mathbf{\Theta}}(\mathbf{x}) + \lambda_K \tilde{P}_{\mathbf{\Theta}}(\mathbf{\Phi}_K(\mathbf{x})) + \lambda_V \tilde{P}_{\mathbf{\Theta}}(\mathbf{\Phi}_V(\mathbf{x})). \tag{18}$$

where $\lambda_K$ and $\lambda_V$ weight the contributions of the augmented views.

## 4 Experiments

### 4.1 Experimental Setup

**Implementation details.** Following the test-time adaptation (TTA) protocol, we adapt the source-trained model to the target domain data during inference. Following TPT [51], we use ImageNet as the target domain and fine-tune the learnable prompts in a few-shot setting for three vision-language models: CoOp, MaPLe, and MMRL. We then evaluate MetaTPT on these models and the zero-shot CLIP across target datasets. Unless otherwise specified, all models employed CLIP-ViT-B/16 as the visual backbone with default hyperparameters. In Alg. 1, we set $M = T = 1$, adopting a single inner and outer optimization loop for a fair comparison. Both loops use the AdamW optimizer, with learning rates $\eta_i = \eta_o = 0.0001$ for domain generation and $0.001$ for cross-dataset evaluation. The EMA momentum was set to $\alpha = 0.9$, and a grid search is performed over $\lambda_K$ and $\lambda_V$. Following TPT [51], we generate $N = 64$ augmented views for both $\mathbf{\Phi}_K$ and $\mathbf{\Phi}_V$, each with different initializations. Specifically, $\mathbf{\Phi}_V$ is initialized to mimic the *rotation* task used in TTT [55]:

$$\mathbf{\Phi}_V^i = \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \end{bmatrix}, \; i = \{1, \cdots, N\}. \tag{19}$$

Each $\mathbf{\Phi}_V^{i\in[1,64]}$ is initialized with Eq. (19), with a rotation angle $\gamma$ sampled uniformly from $(0°, 30°)$, enabling in-plane rotations of up to 30 degrees. $\mathbf{\Phi}_K$ is initialized to mimic the *Random Resize Crop*

Table 1: Comparison of MetaTPT on **Domain Generation** is conducted across four variants.

| | ImageNet-V2 | ImageNet-Sketch | ImageNet-A | ImageNet-R | *Average* |
|---|---|---|---|---|---|
| CLIP [45] | 60.86 | 46.09 | 47.87 | 73.98 | 57.20 |
| CLIP + TPT [51] | 63.45 | 47.94 | 54.77 | 77.06 | 60.81 |
| **CLIP + MetaTPT** | 63.87 | 47.97 | 60.04 | 76.73 | 62.15 |
| CoOp [68] | 64.20 | 47.99 | 49.71 | 75.21 | 59.28 |
| CoOp + TPT [51] | 66.83 | 49.29 | 57.95 | 77.27 | 62.84 |
| **CoOp + MetaTPT** | **67.03** | 49.36 | 62.80 | 77.78 | 64.24 |
| MaPLe [27] | 64.07 | 49.15 | 50.90 | 76.98 | 60.28 |
| MaPLe + TPT [51] | 64.87 | 48.16 | 58.08 | 78.12 | 62.31 |
| MaPLe + PromptAlign [1] | 65.29 | 50.23 | 59.37 | 79.33 | 63.56 |
| **MaPLe + MetaTPT** | 66.46 | 51.37 | **62.83** | 79.63 | **65.07** |
| MMRL [15] | 64.47 | 49.17 | 51.20 | 77.53 | 60.59 |
| MMRL + TPT | 64.49 | 49.09 | 50.26 | 77.31 | 60.29 |
| **MMRL + MetaTPT** | 66.40 | **51.49** | 58.47 | **80.31** | 64.17 |

and *Random Horizontal Flip* augmentations from PromptAlign [1]:

$$\mathbf{\Phi}_K^i = \begin{bmatrix} \frac{\text{flip}\cdot w}{\text{width}} & 0 & t_x \\ 0 & \frac{h}{\text{height}} & t_y \end{bmatrix}, \ i = \{1 \cdots N\}. \tag{20}$$

where $\text{flip} \in \{-1, 1\}$ is a random $x$-axis flipping factor, emulating *Random Horizontal Flip*. $\text{width}$ and $\text{height}$ are the original dimensions of image $\mathbf{x}$. $w$ and $h$ represent the new dimensions: $w = \sqrt{\text{targetArea} \cdot \text{ratio}}$, $h = \sqrt{\frac{\text{targetArea}}{\text{ratio}}}$, where $\text{targetArea} = \text{scale} \cdot \text{width} \cdot \text{height}$. Here, the hyperparameters $\text{scale}$ and $\text{ratio}$ mimic the input parameters of *Random Resize Crop*. If valid dimensions $w$ and $h$ are found, random starting coordinates $(i, j)$ are selected to prevent the image from being cropped beyond its boundaries: $i \in [0, \text{height} - h]$, $j \in [0, \text{width} - w]$. $t_x$ and $t_y$ translate the center of the original image $\left(\frac{\text{width}}{2}, \frac{\text{height}}{2}\right)$ to the new center $\left(j + \frac{w}{2}, i + \frac{h}{2}\right)$. Each $\mathbf{\Phi}_K^{i \in [1,64]}$ is initialized via Eq. (20), where $\text{scale}$ and $\text{ratio}$ are randomly sampled. Specifically, $\text{scale}$ is drawn from the interval $(0.2, 1.0)$, resizing the image to $20\% \sim 100\%$ of its original area. The aspect $\text{ratio}$ is sampled from $(\frac{3}{4}, \frac{4}{3})$, introducing mild geometric distortions. All initialization hyperparameters are fixed and shared across target datasets. Notably, during inner-loop optimization, we update the affine matrices $\mathbf{\Phi}_K$ and $\mathbf{\Phi}_V$ directly, rather than tuning their initialization parameters. These hyperparameters are used only to construct the initial affine matrices. We implement MetaTPT on a NVIDIA A800 80GB GPU. For CoOp, MaPLe and MMRL, all results are averaged over three independent runs.

### 4.2 Results

**Domain generalization.** Table 1 reports the performance of four vision-language models across four out-of-distribution ImageNet variants. The results demonstrate that our MetaTPT outperforms other adaptation methods across all base models. By contrast, TPT exhibits limited robustness and may underperform under certain conditions; for example, for MMRL, TPT slightly decreases performance on ImageNet-R by -0.22%, whereas MetaTPT improves it by +2.78%. Similar gains are observed on ImageNet-A and ImageNet-Sketch, indicating that MetaTPT effectively mitigates performance degradation and enhances adaptation to diverse target domains.

**Cross-dataset evaluation.** Table 2 presents the performance of various vision-language models across ten image classification datasets. The results underscore the superior adaptability of our MetaTPT, particularly in challenging domain shifts. For example, while TPT fails to effectively adapt MaPLe on DTD (-0.62%) and EuroSAT (-0.26%), MetaTPT achieves substantial gains of +1.86% and +1.43%, respectively. Furthermore, TPT exhibits limited adaptability for MMRL, whereas MetaTPT consistently improves performance across all datasets, thereby demonstrating its efficacy in cross-domain adaptation.

Table 2: Comparison of MetaTPT on **Cross-Dataset Evaluation** is conducted across ten datasets.

| | Caltech101 | OxfordPets | StanfordCars | OxfordFlowers | Food101 | FGVCAircraft | SUN397 | DTD | EuroSAT | UCF101 | *Average* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CLIP [45] | 93.35 | 88.25 | 65.48 | 67.44 | 83.65 | 23.67 | 62.59 | 44.27 | 42.01 | 65.13 | 63.58 |
| CLIP + TPT [51] | 94.16 | 87.79 | 66.87 | 68.98 | 84.67 | 24.78 | 65.50 | 47.75 | 42.44 | 68.04 | 65.12 |
| **CLIP + MetaTPT** | 94.81 | 90.57 | 68.71 | 70.77 | 86.71 | 26.52 | 66.45 | 48.17 | 41.96 | 69.92 | 66.46 |
| CoOp [68] | 93.70 | 89.14 | 64.51 | 68.71 | 85.30 | 18.47 | 64.15 | 41.92 | 46.39 | 66.55 | 63.88 |
| CoOp + TPT [51] | 93.15 | 89.48 | 66.77 | 68.48 | 86.48 | 20.51 | 66.06 | 43.32 | 37.73 | 68.91 | 64.09 |
| **CoOp + MetaTPT** | 93.59 | 90.76 | 67.24 | 69.14 | 86.81 | 21.97 | 66.77 | 45.37 | 41.74 | 70.43 | 65.38 |
| MaPLe [27] | 93.53 | 90.46 | 65.57 | 72.23 | 86.20 | 24.74 | 67.01 | 46.49 | 48.06 | 68.69 | 66.30 |
| MaPLe + TPT [51] | 93.59 | 90.72 | 66.50 | 72.37 | 86.64 | 24.70 | 67.54 | 45.87 | 47.80 | 69.19 | 66.49 |
| MaPLe + PromptAlign [1] | 94.01 | 90.76 | 68.50 | 72.39 | 86.65 | 24.80 | 67.54 | 47.24 | 47.86 | 69.47 | 66.92 |
| **MaPLe + MetaTPT** | 94.31 | 90.82 | 68.69 | 72.70 | 87.28 | 26.41 | 68.16 | 48.35 | 49.49 | 69.87 | 67.61 |
| MMRL [15] | 94.67 | 91.43 | 66.10 | 72.77 | 86.40 | 26.30 | 67.57 | 45.90 | 53.10 | 68.27 | 67.25 |
| MMRL + TPT | 94.40 | 91.31 | 66.49 | 72.89 | 86.18 | 26.23 | 67.27 | 46.41 | 47.18 | 69.04 | 66.74 |
| **MMRL + MetaTPT** | **94.90** | **92.79** | **69.50** | **74.22** | **87.61** | **29.05** | **69.17** | **48.88** | **54.26** | **72.24** | **69.26** |

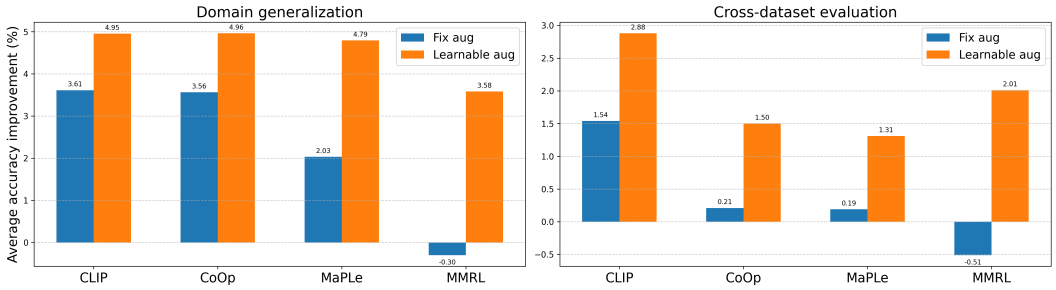


Figure 2: **Effect of learnable augmentation**: comparison of the average performance gains achieved by learnable augmentations in MetaTPT versus fixed augmentations in TPT across the base models.

### 4.3 Ablation Studies

**Effect of learnable augmentations.** Figure 2 presents the effectiveness of learnable augmentations within MetaTPT for adapting diverse vision-language models. The base model is trained on the source domain and evaluated directly on the target domain, providing a baseline for measuring adaptation performance. To mitigate domain shift, TPT employs fixed augmentations for test-time prompt tuning, which can improve target-domain performance in many cases; however, its benefits are not universal. For instance, TPT yields negative gains on MMRL, with -0.30% in domain generalization and -0.51% in cross-dataset evaluation. In contrast, MetaTPT introduces learnable augmentations that are dynamically optimized at test time, consistently improving performance across all base models in both cross-dataset evaluation and domain generalization.

**Effect of dual-loop optimization.** Figure 3 illustrates the effectiveness of dual-loop optimization within MetaTPT. Compared to one-stage training, where the learnable prompts $\mathbf{\Theta}$ and learnable augmentations $\mathbf{\Phi}$ are jointly optimized under a unified objective, $\mathcal{L} = \mathcal{L}_{\text{inner}} + \mathcal{L}_{\text{outer}}$, the dual-loop approach decouples the updates of $\mathbf{\Phi}$ and $\mathbf{\Theta}$ into dedicated inner and outer loops, respectively. While the one-stage optimization is straightforward, it fails to capture the hierarchical dependency between primary and auxiliary tasks. In contrast, the dual-loop design aligns with the meta-learning principle of nested optimization, facilitating more effective task adaptation and generalization. Empirically, dual-loop optimization increases the average accuracy for MMRL from 68.78% to 69.26%, demonstrating the benefits of respecting the intrinsic hierarchical structure of meta-learning.

**Effect of online augmentation adaptation.** Figure 4 demonstrates the effectiveness of online adaptation of learnable augmentations in MetaTPT. Following the online setting of TPT, MetaTPT

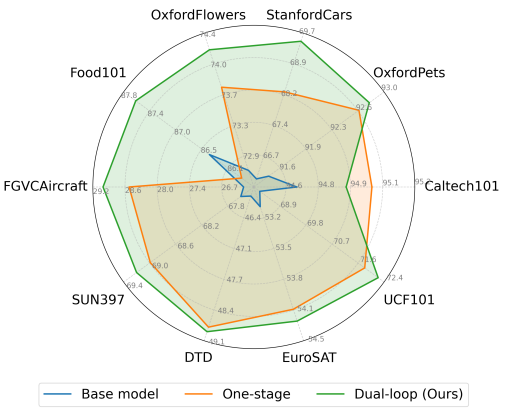


Figure 3: **Effect of dual-loop optimization**: the "one-stage" scheme updates $\mathbf{\Phi}$ and $\mathbf{\Theta}$ simultaneously, whereas our "dual-loop" scheme alternates their updates in an interleaved manner.

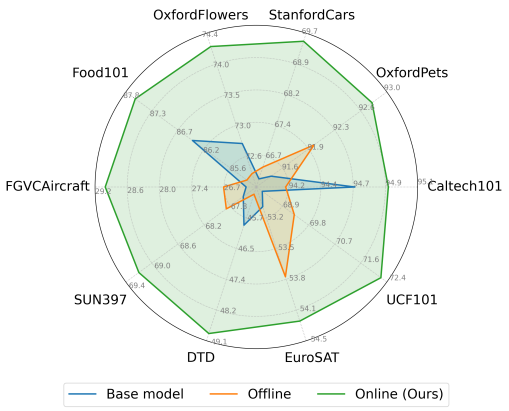


Figure 4: **Effect of online adaptation**: In the "offline" setting, $\mathbf{\Phi}$ is meta-trained prior to prompt adaptation, whereas in our "online" setting, $\mathbf{\Phi}$ and $\mathbf{\Theta}$ are meta-adapted per test sample.

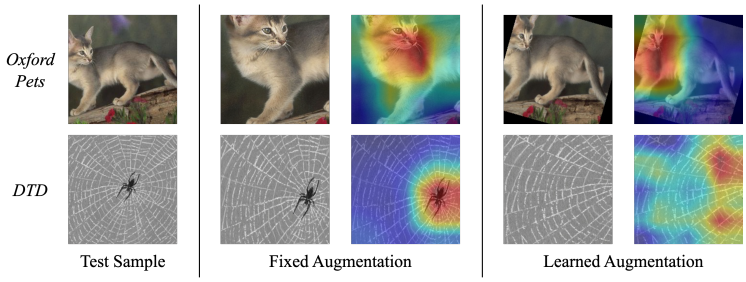


Figure 5: **Visualization of learnable augmentations** and corresponding attention maps.

meta-adapts both prompts $\mathbf{\Theta}$ and augmentations $\mathbf{\Phi}$ on a per-sample basis within a single test stream. To provide a comparative baseline, we introduce an offline variant, in which augmentations $\mathbf{\Phi}$ are first meta-trained for each target domain and subsequently applied to each test sample during meta-testing with prompt tuning. Notably, the offline variant exhibits a performance degradation of 0.39% relative to the online approach, underscoring the superiority of online adaptive augmentation in enhancing test-time generalization.

**Visualization of learnable augmentations.** Figure 5 visualizes the augmented views and their corresponding attention maps. In natural-object domains such as OxfordPets, fixed augmentations may crop views that still retain informative object content, allowing the model to focus its attention on relevant regions. However, in more challenging domains like DTD, fixed augmentations are more likely to crop out domain-relevant features—for example, producing a view that highlights a spider rather than a cobweb (the ground-truth class). In contrast, our learnable augmentation generates views that better preserve domain-specific semantics, such as emphasizing cobweb textures to guide the model's attention. Interestingly, in OxfordPets, the learnable augmentation frequently rotates objects (e.g., Abyssinian) to a horizontal orientation, making it easier for the model to recognize.

## 5 Conclusion

In this work, we proposed MetaTPT, a meta-learning framework for test-time adaptation that facilitates zero-shot generalization. By jointly optimizing parameterized augmentations and learnable prompts via dual-loop optimization, MetaTPT effectively addresses the limitations of previous methods that rely on fixed augmentation strategies. Experiments on two benchmarks show that MetaTPT consistently outperforms existing methods under distribution shifts.

# References


[1] J. Abdul Samadh, M. H. Gani, N. Hussein, M. U. Khattak, M. M. Naseer, F. Shahbaz Khan, and S. H. Khan. Align your prompts: Test-time prompting with distribution alignment for zero-shot generalization. *Advances in Neural Information Processing Systems*, 36:80396–80413, 2023.

[2] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

[3] A. Antoniou, H. Edwards, and A. Storkey. How to train your maml. In *ICLR*, 2019.

[4] Y. Balaji, S. Sankaranarayanan, and R. Chellappa. Metareg: Towards domain generalization using meta-regularization. In *NeurIPS*, 2018.

[5] S. Bechtle, A. Molchanov, Y. Chebotar, E. Grefenstette, L. Righetti, G. S. Sukhatme, and F. Meier. Meta learning via learned loss. In *ICPR*, 2020.

[6] L. Bossard, M. Guillaumin, and L. Van Gool. Food-101–mining discriminative components with random forests. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part VI 13*, pages 446–461. Springer, 2014.

[7] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3606–3613, 2014.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[9] Y. Du, J. Shen, X. Zhen, and C. G. Snoek. Emo: episodic memory optimization for few-shot meta-learning. In *Conference on Lifelong Learning Agents*, pages 1–20. PMLR, 2023.

[10] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *2004 conference on computer vision and pattern recognition workshop*, pages 178–178. IEEE, 2004.

[11] C.-M. Feng, K. Yu, Y. Liu, S. Khan, and W. Zuo. Diverse data augmentation with diffusions for effective test-time prompt tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2704–2714, 2023.

[12] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, pages 1126–1135. JMLR. org, 2017.

[13] P. Gao, S. Geng, R. Zhang, T. Ma, R. Fang, Y. Zhang, H. Li, and Y. Qiao. Clip-adapter: Better vision-language models with feature adapters. *International Journal of Computer Vision*, 132(2):581–595, 2024.

[14] E. Grant, C. Finn, S. Levine, T. Darrell, and T. Griffiths. Recasting gradient-based meta-learning as hierarchical bayes. In *ICLR*, 2018.

[15] Y. Guo and X. Gu. Mmrl: Multi-modal representation learning for vision-language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 25015–25025, 2025.

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

[17] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8340–8349, 2021.

[18] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019.

[19] D. Hendrycks, K. Zhao, S. Basart, J. Steinhardt, and D. Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15262–15271, 2021.

[20] S. Hochreiter, A. S. Younger, and P. R. Conwell. Learning to learn using gradient descent. In *ICANN*, 2001.

[21] C. C. Holt. Forecasting seasonals and trends by exponentially weighted moving averages. *International journal of forecasting*, 20(1):5–10, 2004.

[22] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey. Meta-learning in neural networks: A survey. *TPAMI*, 44(9):5149–5169, 2021.

[23] D. Hwang, J. Park, S. Kwon, K. Kim, J.-W. Ha, and H. J. Kim. Self-supervised auxiliary learning with meta-paths for heterogeneous graphs. In *NeurIPS*, 2020.

[24] D. Hwang, J. Park, S. Kwon, K.-M. Kim, J.-W. Ha, and H. J. Kim. Self-supervised auxiliary learning for graph neural networks via meta-learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[25] C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. Le, Y.-H. Sung, Z. Li, and T. Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pages 4904–4916. PMLR, 2021.

[26] A. Karmanov, D. Guan, S. Lu, A. El Saddik, and E. Xing. Efficient test-time adaptation of vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14162–14171, 2024.

[27] M. U. Khattak, H. Rasheed, M. Maaz, S. Khan, and F. S. Khan. Maple: Multi-modal prompt learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19113–19122, 2023.

[28] M. U. Khattak, S. T. Wasim, M. Naseer, S. Khan, M.-H. Yang, and F. S. Khan. Self-regulating prompts: Foundational model adaptation without forgetting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15190–15200, 2023.

[29] D. Ko, J. Choi, H. K. Choi, K.-W. On, B. Roh, and H. J. Kim. Meltr: Meta loss transformer for learning to fine-tune video foundation models. In *CVPR*, 2023.

[30] G. Koch. Siamese neural networks for one-shot image recognition. In *ICML Workshop*, 2015.

[31] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013.

[32] K. Lee, S. Maji, A. Ravichandran, and S. Soatto. Meta-learning with differentiable convex optimization. In *CVPR*, 2019.

[33] D. Li, Y. Yang, Y.-Z. Song, and T. Hospedales. Learning to generalize: Meta-learning for domain generalization. In *AAAI*, 2018.

[34] J. Li, D. Li, C. Xiong, and S. Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pages 12888–12900. PMLR, 2022.

[35] J. Liang, D. Hu, and J. Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International Conference on Machine Learning (ICML)*, pages 6028–6039, 2020.

[36] J. Liang, D. Hu, Y. Wang, R. He, and J. Feng. Source data-absent unsupervised domain adaptation through hypothesis transfer and labeling transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2021. In Press.

[37] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. *HAL - INRIA*, 2013.

[38] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel. A simple neural attentive meta-learner. In *ICLR*, 2018.

[39] T. Munkhdalai and H. Yu. Meta networks. In *ICML*, 2017.

[40] T. Munkhdalai, X. Yuan, S. Mehri, and A. Trischler. Rapid adaptation with conditionally shifted neurons. In *ICML*, 2018.

[41] A. Nichol, J. Achiam, and J. Schulman. On first-order meta-learning algorithms. *arXiv:1803.02999*, 2018.

[42] M.-E. Nilsback and A. 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.

[43] J. Park, J. Ko, and H. J. Kim. Prompt learning via meta-regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26940–26950, 2024.

[44] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE, 2012.

[45] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[46] S. Ravi and H. Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2016.

[47] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar. Do imagenet classifiers generalize to imagenet? In *International conference on machine learning*, pages 5389–5400. PMLR, 2019.

[48] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[49] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap. Meta-learning with memory-augmented neural networks. In *ICML*, pages 1842–1850, 2016.

[50] J. Shu, Q. Xie, L. Yi, Q. Zhao, S. Zhou, Z. Xu, and D. Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. In *NeurIPS*, 2019.

[51] M. Shu, W. Nie, D.-A. Huang, Z. Yu, T. Goldstein, A. Anandkumar, and C. Xiao. Test-time prompt tuning for zero-shot generalization in vision-language models. *Advances in Neural Information Processing Systems*, 35:14274–14289, 2022.

[52] J. Snell, K. Swersky, and R. Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, pages 4077–4087, 2017.

[53] K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *CRCV-TR*, 2012.

[54] Y. Sun, X. Li, K. Dalal, J. Xu, A. Vikram, G. Zhang, Y. Dubois, X. Chen, X. Wang, S. Koyejo, et al. Learning to (learn at test time): Rnns with expressive hidden states. In *ICML*, 2025.

[55] Y. Sun, X. Wang, Z. Liu, J. Miller, A. Efros, and M. Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pages 9229–9248. PMLR, 2020.

[56] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales. Learning to compare: Relation network for few-shot learning. In *CVPR*, pages 1199–1208, 2018.

[57] O. Vinyals, C. Blundell, T. Lillicrap, and D. Wierstra. Matching networks for one shot learning. In *NeurIPS*, pages 3630–3638, 2016.

[58] D. Wang, E. Shelhamer, S. Liu, B. Olshausen, and T. Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021.

[59] H. Wang, S. Ge, Z. Lipton, and E. P. Xing. Learning robust global representations by penalizing local predictive power. *Advances in Neural Information Processing Systems*, 32, 2019.

[60] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3485–3492. IEEE, 2010.

[61] L. Yang, R.-Y. Zhang, Y. Wang, and X. Xie. Mma: Multi-modal adapter for vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23826–23837, 2024.

[62] H. Yao, L. Zhang, and C. Finn. Meta-learning with fewer tasks through task interpolation. *International Conference on Learning Representations*, 2021.

[63] J. Yu, Z. Wang, V. Vasudevan, L. Yeung, M. Seyedhosseini, and Y. Wu. Coca: Contrastive captioners are image-text foundation models. *Transactions on Machine Learning Research*, 2022.

[64] J. Zhang, J. Huang, X. Zhang, L. Shao, and S. Lu. Historical test-time prompt tuning for vision foundation models. *Advances in Neural Information Processing Systems*, 2024.

[65] R. Zhang, R. Fang, W. Zhang, P. Gao, K. Li, J. Dai, Y. Qiao, and H. Li. Tip-adapter: Training-free clip-adapter for better vision-language modeling. *arXiv preprint arXiv:2111.03930*, 2021.

[66] T. Zhang, J. Wang, H. Guo, T. Dai, B. Chen, and S.-T. Xia. Boostadapter: Improving vision-language test-time adaptation via regional bootstrapping. *Advances in Neural Information Processing Systems*, 2024.

[67] K. Zhou, J. Yang, C. C. Loy, and Z. Liu. Conditional prompt learning for vision-language models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16816–16825, 2022.

[68] K. Zhou, J. Yang, C. C. Loy, and Z. Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022.

[69] L. Zintgraf, K. Shiarli, V. Kurin, K. Hofmann, and S. Whiteson. Fast context adaptation via meta-learning. In *ICML*, pages 7693–7702, 2019.

# A Related Work

**Vision-language models (VLMs).** Contrastive Language–Image Pretraining (CLIP) [45] is a vision-language model demonstrating strong zero-shot classification via alignment of test images with handcrafted prompts. However, adapting CLIP to downstream tasks through full fine-tuning often degrades its zero-shot generalization. To mitigate this, various parameter-efficient fine-tuning (PEFT) methods have been proposed. CoOp [68] formulates prompt tuning by introducing learnable textual prompts, while CoCoOp [67] introduces sample-specific prompts to reduce overfitting to base classes. MaPLe [27] adopts a prefix-tuning approach by injecting learnable prefixes into shallow layers of both encoders, preserving handcrafted prompts. Building upon this, PromptSRC [28] incorporates self-regulating losses to constrain prefix learning. ProMetaR [43] leverages meta-learning to optimize these regularization terms, enhancing robustness across diverse tasks. MMRL [15] diverges by inserting learnable tokens into deep encoder layers to better retain general representations. In parallel, adapter-based methods provide an alternative PEFT paradigm. CLIP-Adapter [13] appends lightweight MLP adapters, while Tip-Adapter [65] leverages a cache-based design. MMA [61] introduces shared-space adapters in deep layers to enhance multimodal integration. While these methods primarily target few-shot settings, our work focuses on test-time tuning to improve zero-shot performance without access to training data.

**Test-time adaptation (TTA).** TTA addresses distribution shifts by adapting a source-pretrained model to the target domain at test time. Recent studies have extended TTA to VLMs along two lines: Training-required approaches fine-tune a subset of model parameters at test time. TPT [51] proposes test-time prompt tuning by adapting CoOp [68]'s learnable prompts via augmented views of test samples. DiffTPT [11] improves view diversity using diffusion [48]-based generation. PromptAlign [1] adopts test-time prefix tuning to align feature distributions, aligning feature distributions to adapt MaPLe [27]. Training-free approaches refine predictions via non-parametric mechanisms. TDA [26] constructs positive and negative caches from previous test samples for test-time inference. BoostAdapter [66] improves this by incorporating sample-specific augmented views into the cache. Our method follows the training-required paradigm, addressing limitations of prompt tuning.

**Meta-learning.** Meta-learning, or "learning to learn", focuses on rapidly adapting models to new tasks by leveraging prior knowledge, as described by Hospedales *et al.* [22]. This approach has been applied across various domains, including the design of loss functions [50, 4, 5], the development of task-specific initializations [12], and the enablement of few-shot learning [30, 56, 52]. Meta-learning techniques are typically classified into metric-based [52, 56, 57, 32], memory-based [38, 49, 39, 40, 20, 9], and gradient-based methods [46, 14, 41, 33]. Since the emergence of Model-Agnostic Meta-Learning (MAML) [12], gradient-based meta-learning has received substantial attention; however, these approaches often encounter meta-overfitting issues, especially with limited meta-training tasks [3, 62, 69, 23, 24, 29]. In light of these challenges, we introduce a meta-learning test-time adaptation framework that dynamically adapts VLMs to diverse testing scenarios.

# B Additional Implementation Details

**Datasets.** Following TPT [51], we conduct experiments on two zero-shot generalization benchmarks. For *domain generalization*, we evaluate across four out-of-distribution (OOD) variants of ImageNet [8]: ImageNet-V2 [47], ImageNet-Sketch [59], ImageNet-A [19] and ImageNet-R [17]. For *cross-dataset evaluation*, we conduct evaluations on ten image classification datasets: Caltech101 [10], OxfordPets [44], OxfordFlowers [42], StanfordCars [31], FGVC-Aircraft [37], Food101 [6], SUN397 [60], DTD [7], EuroSAT [16] and UCF101 [53].

**Baselines.** We evaluate MetaTPT against several baselines: (1) Zero-shot CLIP [45]. (2) Few-shot prompt learning methods: CoOp [68], MaPLe [27] and MMRL [15]. (3) Test-time prompt tuning methods: TPT [51] and PromptAlign [1].

# C Additional Ablation Studies

**Loss ablation.** Figure 6 depicts a loss ablation study conducted on StanfordCars, SUN397, and UCF101, comparing different loss configurations for augmentation tuning (left) and prompt tuning

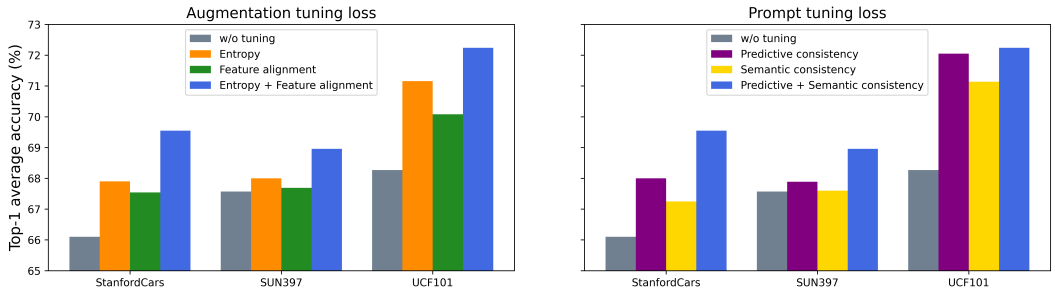


Figure 6: **Ablation of Loss Components** on augmentation tuning loss $\mathcal{L}_{\text{inner}}$ and prompt tuning loss $\mathcal{L}_{\text{outer}}$, evaluated on StanfordCars, SUN397 and UCF101.

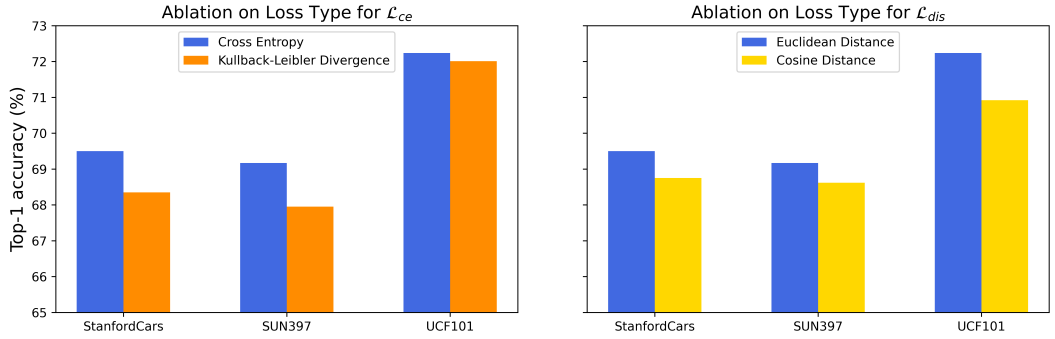


Figure 7: **Ablation study on loss types for predictive consistency loss $\mathcal{L}_{ce}$ and feature alignment loss $\mathcal{L}_{\text{dis}}$**, evaluated on StanfordCars, SUN397 and UCF101.

(right). For augmentation tuning, we evaluate four settings: MMRL [15] without tuning, tuning $\mathbf{\Phi}$ with only entropy loss $H$, only feature alignment loss $\mathcal{L}_{\text{dis}}$, and our proposed combination of both. While individually applying entropy or feature alignment loss improves performance relative to MMRL, their joint optimization consistently achieves superior results. Similarly, for prompt tuning, we assess MMRL without tuning, tuning $\mathbf{\Theta}$ with only predictive consistency $\mathcal{L}_{ce}$, only semantic consistency $\mathcal{L}_{\text{dis}}$, and our combined predictive and semantic consistency losses. The trend is consistent: combining predictive and semantic consistency losses achieves superior adaptation compared to using either loss individually.

**Ablation on loss types in test-time prompt tuning.** Figure 7 evaluates the impact of different loss functions on test-time prompt tuning (Section 3.2). For predictive consistency $\mathcal{L}_{ce}$, we compare cross-entropy and Kullback–Leibler (KL) divergence. Cross-entropy leads to more reliable alignment of soft predictions than KL divergence, improving output stability. For semantic consistency $\mathcal{L}_{\text{dis}}$, we evaluate Euclidean distance and cosine distance. Euclidean distance outperforms cosine distance by achieving stronger semantic alignment between features. Taken together, these findings suggest that combining cross-entropy with Euclidean distance provides the most effective objective for test-time prompt tuning.

**Ablation on numbers of augmented views.** Figure 8 reports an ablation study on the number of augmented views on model performance. Performance improves steadily as the number of views increases from 8 to 64, indicating that greater augmentation diversity benefits representation learning. Beyond 64 views, gains become marginal while computational overhead rises notably. These results highlight 64 views as an effective trade-off between accuracy and computational efficiency.

**Ablation on confidence selection threshold.** Figure 9 reports an ablation study on the confidence selection threshold $\rho$ defined in Eq. (3), evaluated across a range of values from 0.01 to 0.9. Per-

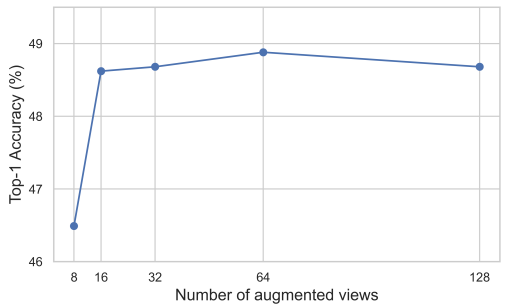


Figure 8: **Ablation study on the number of augmented views** $N$, evaluated on DTD.

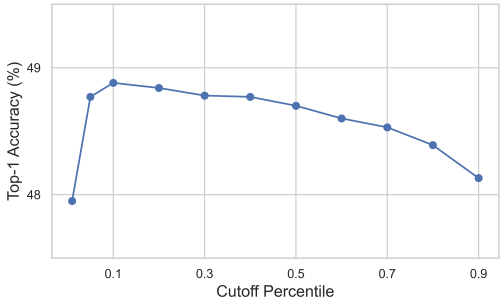


Figure 9: **Ablation study on the cutoff percentile $\rho$ in confidence selection**, evaluated on DTD.

formance peaks at $\rho = 0.1$, indicating an optimal balance between retaining sufficient samples and maintaining label quality. Thresholds below 0.1 introduce noisy pseudo-labels, degrading performance, while higher thresholds excessively reduce the training signal by filtering out too many samples.

**Results of different CLIP backbones.** Table 3 and Table 4 report the performance of MetaTPT using the ViT-B/32 backbone on cross-dataset and domain generalization benchmarks, respectively. On the cross-dataset benchmark (Table 3), MetaTPT achieves the highest average accuracy of 65.57%, surpassing strong baselines including MaPLe [27], PromptAlign [1], and MMRL [15]. Improvements are consistent across most datasets, demonstrating the effectiveness of MetaTPT in handling distribution shifts. On the domain generalization benchmark (Table 4), which includes four ImageNet variants, MetaTPT again outperforms all competitors, reaching an average accuracy of 54.55%. These results demonstrate that MetaTPT consistently enhances generalization performance across domain-shift settings, highlighting its effectiveness regardless of backbone capacity.

Table 3: **Comparison of MetaTPT in Cross-Dataset Evaluation on ViT-B/32**, conducted across ten datasets.

| | Caltech | Pets | Cars | Flowers | Food101 | Aircraft | SUN397 | DTD | EuroSAT | UCF101 | *Average* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| MaPLe [27] | 92.50 | 88.13 | 59.93 | 65.33 | 81.00 | 17.53 | 65.00 | 41.70 | 40.80 | 63.63 | 61.56 |
| MaPLe + TPT [51] | 91.44 | 88.47 | 59.35 | 66.08 | 82.08 | 18.71 | 66.07 | 40.01 | 39.67 | 61.63 | 61.35 |
| PromptAlign [1] | 92.10 | 88.44 | 63.48 | 66.14 | 82.07 | 18.76 | 66.08 | 42.54 | 39.68 | 65.57 | 62.49 |
| MMRL [15] | 92.40 | 88.70 | 60.78 | 66.94 | 80.59 | 20.33 | 65.44 | 46.93 | **48.86** | 65.63 | 63.66 |
| **MMRL + MetaTPT** | **93.12** | **89.79** | **64.41** | **68.47** | **82.56** | **22.75** | **68.05** | **50.24** | 47.10 | **69.26** | **65.57** |

Table 4: **Comparison of MetaTPT in Domain Generation on ViT-B/32**, conducted on four ImageNet variants.

| | ImageNet-V2 | ImageNet-Sketch | ImageNet-A | ImageNet-R | *Average* |
|---|---|---|---|---|---|
| MaPLe [27] | 57.63 | 42.15 | 32.12 | 67.64 | 49.89 |
| MaPLe + TPT [51] | 60.01 | 43.77 | 37.52 | 71.11 | 53.10 |
| PromptAlign [1] | 60.43 | 44.24 | 38.02 | 71.44 | 53.53 |
| MMRL [15] | 58.22 | 42.59 | 32.09 | 68.26 | 50.29 |
| **MMRL + MetaTPT** | **61.31** | **45.63** | **38.65** | **72.54** | **54.53** |

# D Additional Discussions

One limitation of MetaTPT is its sample-specific optimization overhead at test time. While we adopt amortized and parallelizable augmentation updates, the dual-loop structure and per-sample affine

parameters still introduce extra latency compared to training-free or batch-level adaptation methods. Future work could explore more efficient augmentation parameterizations, shared initialization across similar samples, or learning lightweight controllers to modulate augmentation without per-sample gradient steps.

# NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The contributions and scope of this paper are claimed in the abstract. Detailed paper's contribution can be found in Section 1.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We provide a "limitation" in Appendix D.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [No]

Justification: This paper does not include theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide the experimental details in Section 4 for reproduction.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: We will open the source code once accepted.

Guidelines:

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

6. **Experimental setting/details**

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

Answer: [Yes]

Justification: Our experimental dataset is split following TPT [51]. Detailed information can be found in Section 4 and Appendix C.

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [Yes]

Justification: Following the standard experimental setup, we repeat each experiment over 3 random seeds and report the mean of the results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We provide the computing resources in Section 4.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We reviewed and followed the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We provide the potential impacts in Appendix D.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [Yes]

    Justification: The data and models pose no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We cite the original papers that produced the code package and datasets.

    Guidelines:

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

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [No]

    Justification: This paper does not involve the new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

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

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

    Answer: [No]

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

    Guidelines:

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

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

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

    Answer: [No]

    Justification: This paper does not mention participant risks, disclosure, or IRB/ethical approval.

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [Yes]

Justification: We use LLM solely for grammar checking.

Guidelines:

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