Test-Time Prompt Tuning for Zero-Shot Generalization in Vision-Language Models

Anonymous Author(s)
Affiliation
Address
email

Abstract

Pre-trained vision-language models (e.g., CLIP) have shown promising zero-shot generalization in many downstream tasks with properly designed text prompts. Instead of relying on hand-engineered prompts, recent works learn prompts using the training data from downstream tasks. While effective, training on domain-specific data reduces a model's generalization capability to unseen new domains. In this work, we propose test-time prompt tuning (TPT), a method that can learn adaptive prompts on the fly with a single test sample. TPT optimizes the prompt by minimizing the entropy with confidence selection so that the model has consistent predictions across different augmented views of each test sample. In evaluating generalization to natural distribution shifts, TPT improves the zero-shot top-1 accuracy of CLIP by 3.6% on average, surpassing previous prompt tuning approaches that require additional task-specific training data. In evaluating cross-dataset generalization with unseen categories, TPT performs on par with the state-of-the-art approaches that use additional training data.

1 Introduction

Recent advances in vision-language pre-training, such as CLIP [1] and ALIGN [2], present a promising direction for developing foundation models for vision tasks [3]. These foundation models encode a wide range of visual concepts after training on millions of noisy image-text pairs and can be applied to downstream tasks in a zero-shot manner without task-specific training data [4–10]. This is made possible by appropriately designed instruction prompts. Take image classification in Figure 1 as an example: We can prepend a category name with a prompt "a photo of a" (*e.g.,* "a photo of a dog"). Images can then be classified by using CLIP to measure their alignment with the various class descriptions. Designing such prompts thus plays a crucial role in applying foundation models to downstream tasks in a zero-shot manner. However, such hand-crafted prompts require domain-specific heuristics and may not be optimal.

Recent works address this by proposing *prompt tuning* to directly learn prompts using training data from downstream tasks [11]. We can fine-tune prompts with training data in the same way we fine-tune model parameters since prompt embeddings are part of the model input and differentiable with respect to the loss function. Such an approach can find better prompts compared to hand-crafted ones, but the learned prompts are limited to the distribution and tasks corresponding to training data, and may have limited generalization beyond that. In addition, this approach requires training data with annotations, which can be expensive and is not available for zero-shot tasks.

Our Approach. To address the aforementioned challenges, we propose **t**est-time **p**rompt **t**uning (TPT) that tunes the prompt on the fly using only the given test sample. The tuned prompt is adapted to each task, making it suitable for zero-shot generalization without requiring any task-specific

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

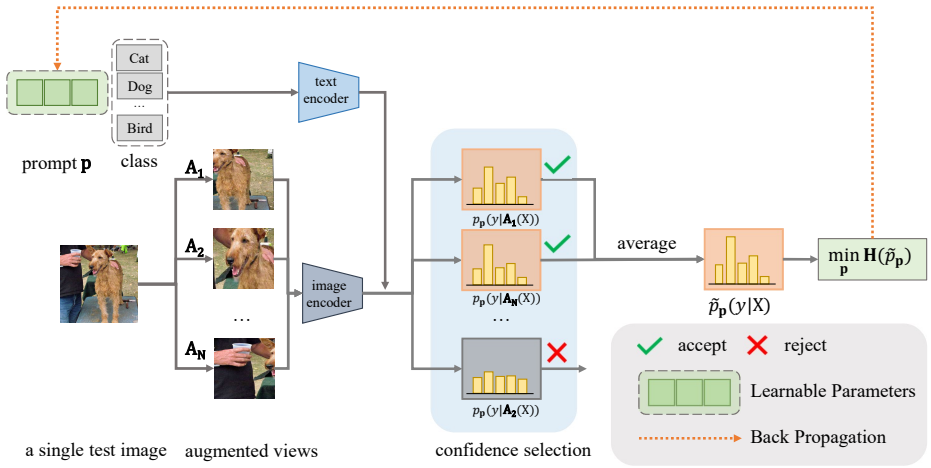


Figure 1: **An overview of Test-time Prompt Tuning (TPT).** We tune adaptive prompts on the fly with a single test sample, without the need for additional training data or annotations. TPT optimizes the prompt to encourage consistent predictions across augmented views of the same test image by minimizing the marginal entropy. We introduce *confidence selection* to filter out noisy augmentations.

training data or annotations. TPT retains the zero-shot generalization setting since no additional training data or annotations are used.

Without loss of generality, we consider CLIP [1] as our vision-language foundation model, for its simplicity in design and its wide applicability [12]. Given a single sample at test time, we perform prompt tuning by generating multiple randomly augmented views, and optimizing the text prompt so that the model has consistent predictions across different augmented views. This is done by minimizing the marginal entropy among the outputs of the augmented views. In addition, since some augmentations may lead to misleading model predictions, we propose *confidence selection* to filter out those "noisy" augmented views. We discard augmented views with a high prediction entropy (*i.e.*, a low confidence), and only include high confidence views in the consistency optimization.

We evaluate the zero-shot generalization of TPT in two image classification settings: natural distribution shifts [13] and cross-dataset generalization [14]. For the setting of evaluating natural distribution shifts, TPT boosts the Top-1 accuracy of CLIP in the zero-shot setting by 3.6% on average compared to using a hand-crafted prompt, achieving on par accuracy with previous prompt tuning methods that require additional training data (*i.e.*, ImageNet). TPT achieves a maximum improvement of 6.9% on ImageNet-A compared to using a hand-crafted prompt, surpassing the existing few-shot prompt tuning method by 5.1%. For the setting of evaluating cross-dataset generalization with possibly unseen categories, TPT achieves on par performance with the state-of-the-art few-shot prompt tuning method [14] without the need for additional training data or annotations.

In addition, we show that TPT is not limited to image classification and is applicable to different downstream tasks. We adapt TPT to context-dependent visual reasoning and outperform the state-of-the-art method [15] by 4.1% on Bongard-HOI benchamrk [16]. We summarize our main contributions as follows:

- We propose test-time prompt tuning (TPT) that does not need any training data or annotations to optimize the prompt. To the best of our knowledge, we are the first to perform prompt tuning on a single test sample in a zero-shot manner.
- We introduce confidence selection as a simple plug-and-play module for TPT. It improves entropy minimization among augmented views by filtering out "noisy" augmentations that lead to low-confidence predictions.
- We conduct extensive experiments on image classification under natural distribution shift and cross-dataset generalization. We find that TPT boosts the performance of CLIP in the zero-shot setting to be on par with prompt tuning methods that require additional training data.

2 Related Work

Prompting for foundation models. Foundation models are those trained on large-scale heterogeneous data, of which the knowledge can be transferred to various downstream tasks in natural language processing [17, 18], computer vision [1, 2, 19], etc. Recent work has proposed different ways to efficiently and effectively transfer such knowledge to downstream task [20–23]. Prompting is a heuristic way to directly apply foundation models to downstream tasks in a zero-shot manner. Prompt works as a part of the text input that instructs the model to perform accordingly on a specific task. However, such zero-shot generalization is highly dependent on a well-designed prompt. Prompt tuning [11, 24, 25] proposes to learn the prompt from downstream data in the continual input embedding space, which presents a parameter-efficient way of fine-tuning foundation models. Although initially developed for language models, prompt tuning has later been applied to other domains, including vision-language models [26, 14, 27] and continual learning [28]. CoOp[26] applies prompt tuning to CLIP. By tuning the prompt on a collection of training data, CoOp effectively improves CLIP's performance on the corresponding downstream tasks. CoCoOp [14] points out that CoOp lacks in generalization to out-of-distribution data, and proposes to alleviate the problem by making the prompt conditioned on model inputs. Despite being effective on the given task, this line of work requires access to downstream training data with annotations, restricting the zero-shot knowledge transfer of foundation models. Another line of work proposes to tune the prompt in an unsupervised manner [29, 30]. However, it requires access to *multiple* samples from either the training or testing split. In this work, we propose test-time prompt tuning that works on a *single* test sample. Our method can directly work with the zero-shot applications of foundation models.

Generalization under data distribution shifts. A reliable machine learning model is supposed to perform well under data distribution shifts for real-world applications. For a model trained on a given set of data, distribution shift refers to the discrepancy between the underlying distributions of the test and the training data. Distribution shifts can occur naturally in the real world due to variations in the environment [31] or the encounter of unseen concepts [32]. For example, in the meta-learning literature [33], each test sample consists of a novel task (i.e., distribution), and the models should be able to quickly adapt to the novel distributions. Even in the standard evaluation protocol for machine learning models, there exists a subtle difference in the data distribution between the training and testing splits [34, 35], which is also one type of distribution shift. Pre-trained vision-language models like CLIP can generalize to downstream tasks with various distribution shifts in a zero-shot manner. Such zero-shot generalization ability presents a promising direction for realizing reliable and generic machine learning models. Our method aims to improve CLIP towards a better generic model in this work, instead of adapting it to specific downstream tasks or target datasets. In our work, we leverage the assumption that a robust model should have decision boundaries lying in low-density data regions [36]. Consistency-regularization-based methods could achieve this goal by making the network outputs invariant to small input perturbations (e.g., augmentations). We use consistency regularization as our test-time prompt tuning objective with the confidence selection module.

Test-time optimization. The idea of adapting machine learning models to test samples on the fly has been applied to different tasks [37–40]. This work mainly focuses on applying the technique to improve the generalization to data distribution shifts. One challenge in this area is to design an effective test-time objective. Test-time training and its variants [41, 42] modify the training process by adding a self-supervised multi-task branch, which computes an optimization objective at test time and adapts the network to the test sample. TENT [43] proposes a test-time objective by minimizing the entropy of the batch-wise prediction probability distributions. Such an objective does not rely on a specific training process, and thus can be applied to a wide range of models. However, TENT needs more than one test sample to get a non-trivial solution. Zhang et al. [44] proposes to bypass the multi-sample requirements using data augmentations. Another major challenge is to choose the right parameter group for optimization. Batch normalization (BN) layers have been shown to capture the domain discrepancies in image data [45, 46]. It is a straightforward way to directly adapt the BN statistics at test time to enhance model robustness against distribution shifts [47]. However, adapting BN layers puts restrictions on model architectures. Another choice is to update the feature extractor while freezing the prediction module [41, 48]. Zhang et al. [44] shows that optimizing the entire model at test time can work as well. Our method addresses both of the challenges above. For the choice of parameter group, we optimize the text prompt while keeping the model intact. Our motivation is to avoid distorting the pre-trained features and to preserve the zero-shot generalization ability of

pre-trained models. In section 5, we empirically show that the prompt works as the most effective parameter group for CLIP. Different from previous single-point method [44], we refine the entropy minimization by proposing *confidence selection*, which helps filtering out noisy augmentations that may lead to misleading predictions.

3 TPT: Test-Time Prompt Tuning

In this section, we first discuss how to apply CLIP to downstream tasks in the zero-shot manner with a hand-crafted prompt. Next, we briefly review recent progress in prompt tuning approaches for CLIP using downstream training data. Finally, we give a detailed introduction of how to apply our method to the image classification taks. We introduce TPT for context-dependent visual reasoning in appendix A.2, along with the background knowledge of this applications.

3.1 Background

Contrastive Language-Image Pre-training (CLIP). CLIP consists of two parallel encoders, one that maps the text input into a feature vector, and the other does the same for the image input. The model is trained with a contrastive loss that promotes similarity between the two vectors so that the text and image align in the feature space. We denote a CLIP model as $\mathcal{F} = \{\mathbf{E}_{\texttt{visual}}, \mathbf{E}_{\texttt{text}}\}$, with $\mathbf{E}_{\texttt{visual}}$ and $\mathbf{E}_{\texttt{text}}$ being the image and text encoders.

We first review how to apply CLIP to downstream tasks in a zero-shot manner with a hand-crafted prompt. We take image classification as an example. Consider a single test image X_{test} of class y, where $X \in \mathbb{R}^{C \times H \times W}$ and $y \in \mathbb{R}^K$ for a K-class classification problem. In the baseline zero-shot setting, we prepend a hand-crafted prompt prefix, such as $\boldsymbol{p}$ =“a photo of a", to every y_i in $\mathcal{Y} = \{y_1, y_2, \ldots, y_K\}$ to form the category-specific text inputs $\{\boldsymbol{p}; y_i\}$. We then feed these class descriptions to the text encoder to get the text features $\{\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_K\}$, where $\boldsymbol{t}_i = \mathbf{E}_{\texttt{text}}(\{\boldsymbol{p}; y_i\})$. Each text feature $\boldsymbol{t}_i$ is paired with the image feature $\boldsymbol{v} = \mathbf{E}_{\texttt{visual}}(X)$ to compute a similarity score $\boldsymbol{s_i} = \texttt{sim}(\boldsymbol{t_i} \cdot \boldsymbol{v})$, where $\texttt{sim}(,)$ denotes the cosine similarity. The prediction probability on X can be denoted by $p(y_i|X) = \frac{\texttt{exp}(\texttt{sim}(\boldsymbol{t_i} \cdot \boldsymbol{v})\tau)}{\sum_{i=1}^{K} \texttt{exp}(\texttt{sim}(\boldsymbol{t_i} \cdot \boldsymbol{v})\tau)}$, where τ is the temperature of the softmax function.

Prompt tuning using downstream training data. Instead of using a hand-crafted prompt, prompt tuning methods train a prompt to maximize performance on a downstream task for which labeled data is available. Prompt tuning optimizes the prompt $\boldsymbol{p} \in \mathbb{R}^{L \times D}$ in the text embedding space, with the number of tokens L and embedding size D, using training data with annotations $\mathcal{D}_{\text{train}} = \{(X_i, y_i)\}$ from the downstream task. The goal is to obtain text inputs $\{\boldsymbol{p}; \mathcal{Y}\} = \{\{\boldsymbol{p}; y_i\} \text{ for } y_i \in \mathcal{Y}\}$ that can provide the model with the most helpful context information about the task. For image classification with cross-entropy loss $\mathcal{L}$, the problem can be formulated as:

$$\boldsymbol{p}^* = \arg\min_{\boldsymbol{p}} \mathbb{E}_{(X,y)\sim\mathcal{D}_{\text{train}}} \mathcal{L}(\mathcal{F}_{\boldsymbol{p}}(X), y), \tag{1}$$

$$\text{where } \mathcal{F}_{\boldsymbol{p}}(X) = \texttt{sim}(\mathbf{E}_{\texttt{text}}(\{\boldsymbol{p}; \mathcal{Y}\}), \mathbf{E}_{\texttt{visual}}(X)). \tag{2}$$

3.2 TPT: Test-Time Prompt Tuning

Why optimize prompts? CLIP contains rich knowledge obtained from pre-training on a massive and diverse dataset. However, how to *more effectively* extract such knowledge remains an open question. A simple strategy is to directly fine-tune the model, either end-to-end or for a subset of layers, on a category of inputs. However, previous work has shown that such fine-tuning strategies result in domain-specific behaviors that lose the out-of-distribution generalization and robustness of foundation models [12, 49]. Prompts, on the other hand, work outside the pre-trained model by modifying the context of the model input, thus do not distort pre-trained features.

In this work, our goal is to leverage the existing knowledge of CLIP to boost its generalization in a zero-shot manner. Therefore, prompt tuning serves as an ideal handle to approach the goal. Furthermore, we regard test-time prompt tuning as a way to provide the model with the context tailored to the single test sample, which helps precisely retrieve the knowledge of CLIP.

At the inference stage, the only information available is the single test sample X_{test} without label information. TPT, therefore, manages to optimize the prompt $\boldsymbol{p}$ at test time based on the single test

sample. In general, our objective can be formulated in the form of

$$\boldsymbol{p}^* = \arg\min_{\boldsymbol{p}} \mathcal{L}(\mathcal{F}, \boldsymbol{p}, X_{\text{test}}) \tag{3}$$

for some carefully constructed loss. Note that, unlike equation (1), our method does not require any labels, or any data beyond the zero-shot test sample.

TPT for image classification. Because labels are not available for test time tuning, we must select an unsupervised loss for prompt tuning. We design our TPT objective to promote the consistency of the model's predictions across different augmented views of a given test image. Specifically, we generate N randomly augmented views of the test image using a family of random augmentations $\mathcal{A}$, and minimize the entropy of the averaged prediction probability distribution:

$$\boldsymbol{p}^* = \arg\min_{\boldsymbol{p}} -\sum_{i=1}^{K} \tilde{p}_{\boldsymbol{p}}(y_i|X_{\text{test}}) \log \tilde{p}_{\boldsymbol{p}}(y_i|X_{\text{test}}), \tag{4}$$

$$\text{where } \tilde{p}_{\boldsymbol{p}}(y_i|X_{\text{test}}) = \frac{1}{N}\sum_{i=1}^{N} p_{\boldsymbol{p}}(y_i|\mathcal{A}_i(X_{\text{test}})). \tag{5}$$

Here, $p_{\boldsymbol{p}}(y|\mathcal{A}_i(X_{\text{test}}))$ is the vector of class probabilities produced by the model when provided with prompt $\boldsymbol{p}$ and the i-th augmented view of the test image.

In addition, to reduce the noise from random augmentations, we propose *confidence selection* to filter out views that generate high-entropy (*i.e.*, low-confidence) predictions. Such views of an image may lack important information needed to classify it correctly, *e.g.*, a random crop may have removed important image content. We select confident samples with a prediction entropy below a threshold τ. We adapt τ for each test sample, by taking the entropy value at the ρ-percentile among the self-entropy of N augmented views ranked from low to high (*i.e.*, confidence from high to low). With τ, the confidence selection can be written as a mask over the augmented samples $\mathbb{1}[\mathbf{H}(p_i) \leq \tau]$, with $\mathbf{H}$ measuring the self-entropy of the prediction on an augmented view. Using confidence selection with a cutoff percentile ρ on N augmented views, the averaged probability in Eq. (4) now becomes:

$$\tilde{p}_{\boldsymbol{p}(y|X_{\text{test}})} = \frac{1}{\rho N}\sum_{i=1}^{N} \mathbb{1}[\mathbf{H}(p_i) \leq \tau] p_{\boldsymbol{p}}(y|\mathcal{A}_i(X_{\text{test}})), \tag{6}$$

4 Experiments

In this section, we describe the tasks and benchmarks used for evaluating our method, along with the implementation details. Our main results cover two aspects of model's generalization: robustness to natural distribution shifts and cross-dataset generalization. We also provide ablation experiments in section 5, analyzing different network components for test-time tuning, and other design choices of our method. Experiments for context-dependent visual reasoning on Bonagrd-HOI benchmark can be found in Appendix A.2.

4.1 Robustness to Natural Distribution Shifts

Datasets. CLIP has shown remarkable robustness to distribution shifts that can occur naturally in real-world scenarios, including variations in image style, data domains, *etc*. We follow the setting in Radford et al. [1] and evaluate model's robustness to natural distribution shifts on 4 ImageNet Variants as follows, which have been considered as out-of-distribution (OOD) data for ImageNet [50] in previous work.

- **ImageNet-V2** [51] is a independent test set containing natural images, collected from different source, including 10,000 images of 1,000 ImageNet categories.
- **ImageNet-A** [52] is a challenging test set of "natural adversarial examples" misclassified by a standard ResNet-50 [53], consisting of 7,500 images of 200 of ImageNet categories.
- **ImageNet-R** [13] collects images of ImageNet categories but with artistic renditions. There are 30,000 images in total, including 200 ImageNet categories.
- **ImageNet-Sketch** [54] is a dataset of black and white sketches, collected independently from the original ImageNet validation set. The dataset includes 50,000 images in total, covering 1,000 ImageNet categories.

Baselines. We compare TPT with existing few-shot prompt tuning methods that are designed for CLIP. CoOp [26] is a few-shot prompt tuning baseline that tunes a fixed dataset-specific prompt on each downstream dataset. CoCoOp [14] is the state-of-the-art prompt tuning method for CLIP. It produces input-dependent prompts with a network module, of which the input is the image feature. The network module of CoCoOp is also trained on downstream data in a dataset-specific way. Following their original configuration, we train both methods on ImageNet using 16-shots training data per category with 4 learnable prompt tokens, and directly test the tuned prompt on OOD benchmarks. We also include two versions of the baseline zero-shot performance of CLIP, using a default prompt "a photo of a", and the ensemble of 80 hand-crafted prompts from Radford et al. [1].

Implementation details. For TPT, we initialize the prompt as the default hand-crafted one "a photo of a", and optimize the corresponding 4 tokens in the text input embedding space based on a single test image. We augment a single test image for 63 times using random resized crops, and construct a batch of 64 images, including the original one. Among the 64 predictions, we select the top 10% (ρ=0.1) confident samples (lowest 10% in self-entropy), and compute the entropy of the averaged probability of the selected predictions (i.e., marginal entropy). We optimize the prompt to minimize the marginal entropy for 1 step, using the AdamW optimizer with a learning rate of 0.005.

Results. In Table 1, the standalone TPT achieves higher accuracy than both prompt ensemble and existing few-shot prompt tuning methods, including CoCoOp. Furthermore, since TPT works at test-time, it is complimentary to existing baseline methods. We show that by applying TPT to prompts learned by CoOp or CoCoOp, we can further improve the accuracy of their in-domain ImageNet data, as well as generalization ability to OOD data. We also compare to the ensembles of baseline models in Appendix A.3, where we find that applying TPT to baseline methods can bring more substantial improvement then model ensembles. In addition, among the five datasets, few-shot prompt tuning methods bring the most accuracy gain on the ImageNet validation set and ImageNet-V2. However, on datasets with more significant distribution shifts, few-shot prompt tuning methods trained on ImageNet show no advantage over the ensemble of hand-crafted prompts.

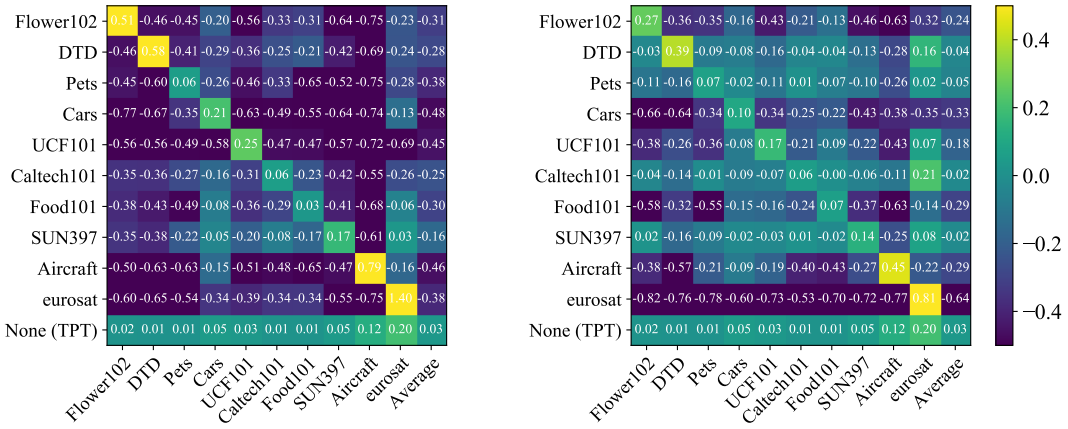
Table 1: **Robustness to Natural Distribution Shifts**. CoOp and CoCoOp are tuned on ImageNet using 16-shot training data per category. Baseline CLIP, prompt ensemble and TPT do not require training data.

Method	ImageNet Top1 acc. ↑	ImageNet-A Top1 acc. ↑	ImageNet-V2. Top1 acc. ↑	ImageNet-R. Top1 acc. ↑	ImageNet-Sketch Top1 acc. ↑	Average	OOD Average
CLIP-RN50	58.16	21.83	51.41	56.15	33.37	44.18	40.69
Ensemble	59.81	23.24	52.91	**60.72**	35.48	46.43	43.09
CoOp	63.33	23.06	55.40	56.60	34.67	46.61	42.43
CoCoOp	62.81	23.32	55.72	57.74	34.48	46.81	42.82
TPT	60.74	26.67	54.7	59.11	35.09	47.26	43.89
TPT + CoOp	**64.73**	**30.32**	**57.83**	58.99	**35.86**	**49.55**	**45.75**
TPT + CoCoOp	62.93	27.4	56.6	59.88	35.43	48.45	44.83
CLIP-ViT-B/16	66.73	47.87	60.86	73.98	46.09	59.11	57.2
Ensemble	68.34	49.89	61.88	**77.65**	48.24	61.20	59.42
CoOp	71.51	49.71	64.20	75.21	47.99	61.72	59.28
CoCoOp	71.02	50.63	64.07	76.18	48.75	62.13	59.91
TPT	68.98	54.77	63.45	77.06	47.94	62.44	60.81
TPT + CoOp	**73.61**	**57.95**	**66.83**	77.27	**49.29**	**64.99**	**62.83**
TPT + CoCoOp	71.07	58.47	64.85	78.65	48.47	64.30	62.61

4.2 Cross-Datasets Generalization

Pre-trained vision-language models like CLIP are ideal for "open-world" problems. For example, we can apply CLIP to classify arbitrary categories in a zero-shot manner in image classification.. However, a prompt tuned on a specific downstream dataset can be less generalizable to categories outside its training set. In this section, we evaluate the cross-dataset generalization of existing few-shot prompt tuning methods (same as in section 4.1), and compare them with TPT, which is not dataset-specific.

Setup. We conduct cross-dataset evaluation on the task of image classification. We consider 10 datasets, covering fine-grained classifications including species of plants or animals (Flower102 [55],



(a) CoOp with CLIP-RN50.

(b) CoCoOp with CLIP-RN50.

Figure 2: **Cross-dataset improvement normalized by the zero-shot baseline performance.** In each matrix A, $A_{i,j}$ is the *normalized relative improvement* on the j_{th} dataset of using the prompt tuned on the i-th dataset. The value $A_{i,j}$ stands for how well a method trained on a source dataset i performs on a target dataset j, in comparison with a zero-shot CLIP baseline (using a hand-crafted prompt). Thus, the higher, the better. The last row is the performance of TPT, which is not tuned on any source dataset. The last column summarize the average improvement over 10 datasets, measuring the overall generalization ability across the 10 datasets.

OxfordPets [56]), scenes (SUN397 [57]), textures (DTD [58]), food (Food101 [59]), transportation (StanfordCars [60], Aircraft [61]), human actions (UCF101 [62]), satellite images (EuroSAT [63]), and general objects (Caltech101 [64]). We consider two different settings of cross-dataset generalization. In the first setting, we consider ImageNet with 1000 categories as a comprehensive source dataset, and use other fine-grained datasets as target datasets for evaluation. We implement CoOp and CoCoOp using the same setting as in section 4.1, and evaluate their generalization performance to the 10 datasets. In the second setting, we consider a more challenging scenario, where the source data for few-shot prompt tuning also comes from the specialized fine-grained datasets, and there is no overlapping in categories between a source-target pair.

Implementation details. We implement CoOp and CoCoOp on each source dataset following their original configurations. For TPT, we use the same initialization of "a photo of a" on each of the fine-grained classification datasets. We adopt the same hyper-parameter setting as in section 4.1, by only optimizing the prompt for 1 step at test time. We use AugMix [65] as a stronger data augmentation for this task.

Results. In Table 2, we compare TPT with few-shot prompt tuning methods on generalization from ImageNet to fine-grained datasets. Note that TPT works in a zero-shot manner; thus it is not trained on

Table 2: **Cross-dataset generalization from ImageNet to fine-grained classification datasets**. CoOp and CoCoOp are tuned on ImageNet using 16-shot training data per category. Baseline CLIP, prompt ensemble and TPT methods does not require training data or annotations. We report the top-1 classification accuracy on each dataset.

Method	Flower102	DTD	Pets	Cars	UCF101	Caltech101	Food101	SUN397	Aircraft	EuroSAT	Average
CLIP-RN50	61.75	40.37	83.57	55.70	58.84	85.88	73.97	58.80	15.66	23.69	55.82
Ensemble	62.77	40.37	82.97	55.89	59.48	**87.26**	74.82	60.85	16.11	25.79	56.63
CoOp	61.55	37.29	87.00	55.32	59.05	86.53	75.59	58.15	15.12	26.20	56.18
CoCoOp	**65.57**	38.53	**88.39**	56.22	57.10	87.38	**76.2**	59.61	14.61	**28.73**	57.23
TPT	62.69	**40.84**	84.49	**58.46**	**60.82**	87.02	74.88	**61.46**	**17.58**	28.33	**57.66**
CLIP-ViT-B/16	67.44	44.27	88.25	65.48	65.13	93.35	83.65	62.59	23.67	42.01	63.58
Ensemble	66.99	45.04	86.92	66.11	65.16	93.55	82.86	65.63	23.22	**50.42**	64.59
CoOp	68.71	41.92	89.14	64.51	66.55	93.70	**85.30**	64.15	18.47	**46.39**	63.88
CoCoOp	**70.85**	45.45	**90.46**	64.90	**68.44**	**93.79**	83.97	**66.89**	22.29	39.23	64.63
TPT	68.98	**47.75**	87.79	**66.87**	68.04	94.16	84.67	65.5	**24.78**	42.44	**65.10**

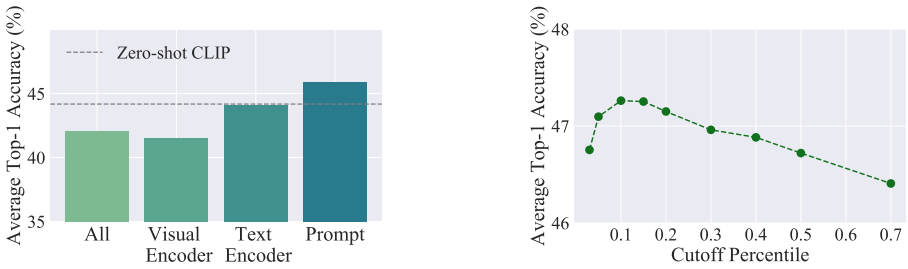
ImageNet. Nonetheless, we find TPT to achieve on par generalization as ImageNet trained CoCoOp. In Figure 2, we present the results of the more challenging setting of cross-dataset generalization, where there is no overlap between the source and target dataset. For better visualization, we plot the relative accuracy improvement $acc' = (acc - acc_{base})/acc_{base}$, normalized by the zero-shot baseline accuracy acc_{base} of a CLIP-RN50. For example, baseline CLIP with a hand-crafted prompt achieves 61.75% accuracy on Flower102, while CoOp trained on DTD only has 33.41% on Flower102. In this case, we calculate acc' as $(33.41 - 61.75)/61.75 = -0.46$. From Figure 2, we can see that the averaged accuracy improvement (in the last column of each matrix) of few-shot prompt tuning methods is always negative, meaning that they do worse than the zero-shot baseline. TPT, on the other hand, shows consistent improvement in each of the 10 datasets.

5 Ablation Study

In this section, we provide an empirical analysis of our design choices and ablating the effects of different components of TPT. For simplicity, if not otherwise specified, analyses in this section are evaluated on the natural distribution shifts benchmarks. We first conduct test-time optimization on different parameter groups of CLIP, showing that prompt tuning achieves the most accuracy gain for CLIP. Next, we show the improvement brought by confidence selection, and analyze how the confidence threshold affects the performance. Lastly, we provide a quantitative analysis of TPT on the trade-off between efficiency and performance. Additional ablation study on data augmentation can be found in Appendix A.3. We also provide a qualitative study on the effect of TPT to model prediction probability distributions in Appendix A.8.

Test-time optimization on different parameter groups of CLIP. Existing test-time optimization methods have worked on different parameter groups of a model. Although there is a strong intuition for tuning prompt on CLIP, it is unclear whether it is the most effective choice. In Figure 3 (a), we evaluate different design choices of test-time optimization on CLIP. Inspired by MEMO [44], a single-point test-time optimization method that optimizes the entire network, we consider four different parameter groups: 1) the entire model, 2) the text encoder, 3) the visual encoder, and 4) the text prompt. For a fair comparison, we adopt the same setup as MEMO, using AugMix [65] as the data augmentation. Confidence selection is not used in this ablation study. For each design choice, we run a grid-search for hyper-parameter tuning (on the learning rate and the number of optimization steps) and report the best result.

From the results, we see that optimizing text prompt achieves the most performance gain compared to other parameter groups. In addition, we find optimizing the visual encoder to have the worst result. This observation is in alignment with previous work that suggests fine-tuning the image encoder can distort pre-trained features [66, 49].



(a) Test-time optimization on different modules. (b) Different cutoff percentile in confidence selection.

Figure 3: **Ablating the effects of different components of TPT.** We evaluate the top-1 accuracy on the distribution shifts benchmarks in section 4.1. Methods are implemented based on a CLIP-RN50.

The effect of confidence selection. We present confidence selection as a major component of our method, which filters out "noisy" augmented views that provide little information. In Table 8, we provide the performance of TPT without confidence selection, in comparison with the full method. Confidence selection brings non-trivial performance improvement to our baseline TPT. We further show the effect of confidence threshold ρ in Figure 3 (b). The result suggests that using the top-10% confident sample leads to the highest average accuracy. In addition, we find that the effect of

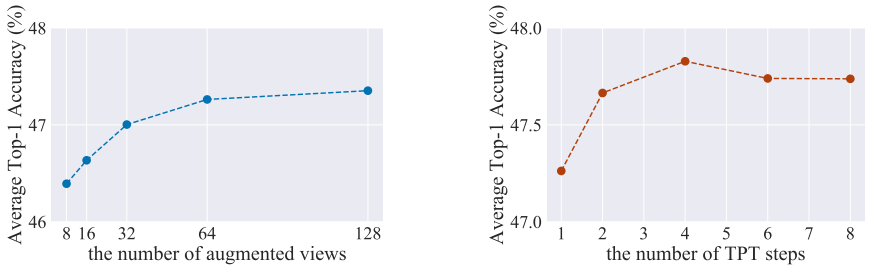
confidence selection is generalizable to other entropy-based test-time optimization methods. More details about this analysis are included in appendix A.4

Table 3: **The effect of confidence selection**. The last row is the performance of our full method.

Method	ImageNet Top1 acc. ↑	ImageNet-A Top1 acc. ↑	ImageNet-V2. Top1 acc. ↑	ImageNet-R. Top1 acc. ↑	ImageNet-Sketch Top1 acc. ↑	Average	OOD Average
CLIP-RN50	58.16	21.83	51.41	56.15	33.37	44.18	40.69
baseline TPT	60.31	23.65	53.66	57.48	34.31	45.88	42.28
+ confidence selection	60.74 (+0.43)	26.67 (+3.02)	54.70 (+1.04)	59.11 (+1.63)	35.09 (+0.78)	47.26 (+1.38)	43.89 (+1.61)

The trade-off between inference efficiency and accuracy. We analyze two factors that affect TPT's efficiency: 1) The number of augmented views N that increases the actual number of test samples; 2) The number of optimization steps that increases the runtime and memory usage mainly induced by backpropagation. Figure 4 shows the relationship between the two factors and the average accuracy of TPT on natural distribution shifts.

In Figure 4 (a), the accuracy increases as the number of augmented views grows until reaching a plateau at around $N = 64$. Even when $N = 8$, TPT still brings about over 2% average accuracy gain to the zero-shot CLIP, suggesting that TPT can be adapted for more efficient applications. In Figure 4 (b), we find that increasing the number of optimization steps from 1 to 2 can slightly increase the accuracy (by 0.4%), while there is no significant performance gain from taking more than 2 steps. Considering that the performance gain comes at the expense of linearly increasing the inference time, we use 1-step TPT as our default setting, which is already capable of boosting the average accuracy of zero-shot CLIP by more than 3%.



(a) Different number of augmented views. (b) Different number of optimization steps.

Figure 4: **Analysis on the trade-off between efficiency and accuracy.** We evaluate the top-1 accuracy on the distribution shifts benchmarks in section 4.1. Results are based on a CLIP-RN50.

6 Conclusion

In this work, we investigated how to fully exploit the potential of pre-trained vision-language foundation models as better zero-shot learners. We developed Test-time Prompt Tuning (TPT), a new prompt tuning method that can learn adaptive prompts on the fly with a single test sample. We demonstrated the effectiveness of our method on the robustness to natural distribution shifts and cross-dataset generalization, by using CLIP as the foundation model. Without the need for any training data or annotations, TPT improves the zero-shot generalization ability of CLIP.

Limitations. While TPT does not require training data or annotations, our method requires a one-step backpropagation when optimizing the prompt at test time. Since TPT generates multiple augmented views of a single test sample, it increases the memory cost during inference. We include an analysis of time complexity in Appendix A.6.

Future directions. The idea of TPT can be applied to other foundation models for various downstream tasks, including other vision-language models [5, 67] and foundation models of other modalities (*e.g.*, pre-trained large-scale language models [18, 17]) to further boost their zero-shot generalization. The most interesting part in this direction is to design a test-time objective that fits the nature of the model and the downstream task. Besides, it is also interesting to explore how to reduce the memory cost of TPT and make it more computationally efficient.

References

[1] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 1, 2, 3, 5, 6, 19

[2] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V. Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*, 2021. 1, 3

[3] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021. 1

[4] Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. MDETR - modulated detection for end-to-end multi-modal understanding. In *ICCV*, 2021. 1

[5] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, Kai-Wei Chang, and Jianfeng Gao. Grounded language-image pre-training. In *CVPR*, 2022. 9

[6] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *ICCV*, 2021.

[7] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with CLIP latents. *CoRR*, abs/2204.06125, 2022.

[8] Zhaowei Cai, Gukyeong Kwon, Avinash Ravichandran, Erhan Bas, Zhuowen Tu, Rahul Bhotika, and Stefano Soatto. X-DETR: A versatile architecture for instance-wise vision-language tasks. *CoRR*, abs/2204.05626, 2022.

[9] Tianyi Liu, Zuxuan Wu, Wenhan Xiong, Jingjing Chen, and Yu-Gang Jiang. Unified multimodal pre-training and prompt-based tuning for vision-language understanding and generation. *CoRR*, abs/2112.05587, 2021.

[10] Mengde Xu, Zheng Zhang, Fangyun Wei, Yutong Lin, Yue Cao, Han Hu, and Xiang Bai. A simple baseline for zero-shot semantic segmentation with pre-trained vision-language model. *CoRR*, abs/2112.14757, 2021. 1

[11] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *EMNLP*, 2021. 1, 3

[12] Mitchell Wortsman, Gabriel Ilharco, Mike Li, Jong Wook Kim, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, and Ludwig Schmidt. Robust fine-tuning of zero-shot models. *CoRR*, abs/2109.01903, 2021. 2, 4

[13] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *ICCV*, 2021. 2, 5, 19

[14] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *CVPR*, 2022. 2, 3, 6, 19

[15] Cheng Zou, Bohan Wang, Yue Hu, Junqi Liu, Qian Wu, Yu Zhao, Boxun Li, Chenguang Zhang, Chi Zhang, Yichen Wei, and Jian Sun. End-to-end human object interaction detection with HOI transformer. In *CVPR*, 2021. 2, 15

[16] Huaizu Jiang, Xiaojian Ma, Weili Nie, Zhiding Yu, Yuke Zhu, and Anima Anandkumar. Bongard-hoi: Benchmarking few-shot visual reasoning for human-object interactions. In *CVPR*, 2022. 2, 15, 16

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*. Association for Computational Linguistics, 2019. 3, 9

[18] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, 2020. 3, 9

[19] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020. 3

[20] Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *ACL*, 2022. 3

[21] Renrui Zhang, Rongyao Fang, Wei Zhang, Peng Gao, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-adapter: Training-free clip-adapter for better vision-language modeling. *CoRR*, abs/2111.03930, 2021.

[22] Renrui Zhang, Longtian Qiu, Wei Zhang, and Ziyao Zeng. VT-CLIP: enhancing vision-language models with visual-guided texts. *CoRR*, abs/2112.02399, 2021.

[23] Zhecan Wang, Noel Codella, Yen-Chun Chen, Luowei Zhou, Jianwei Yang, Xiyang Dai, Bin Xiao, Haoxuan You, Shih-Fu Chang, and Lu Yuan. CLIP-TD: CLIP targeted distillation for vision-language tasks. *CoRR*, abs/2201.05729, 2022. 3

[24] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL/IJCNLP*. Association for Computational Linguistics, 2021. 3

[25] Zhuofeng Wu, Sinong Wang, Jiatao Gu, Rui Hou, Yuxiao Dong, V. G. Vinod Vydiswaran, and Hao Ma. IDPG: an instance-dependent prompt generation method. In *NAACL*, 2022. 3

[26] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *CoRR*, abs/2109.01134, 2021. 3, 6, 19

[27] Yu Du, Fangyun Wei, Zihe Zhang, Miaojing Shi, Yue Gao, and Guoqi Li. Learning to prompt for open-vocabulary object detection with vision-language model. In *CVPR*, 2022. 3

[28] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer G. Dy, and Tomas Pfister. Learning to prompt for continual learning. *CoRR*, abs/2112.08654, 2021. 3

[29] Tony Huang, Jack Chu, and Fangyun Wei. Unsupervised prompt learning for vision-language models. *CoRR*, abs/2204.03649, 2022. 3

[30] Chunting Zhou, Junxian He, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Prompt consistency for zero-shot task generalization. *CoRR*, abs/2205.00049, 2022. 3

[31] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton Earnshaw, Imran Haque, Sara M. Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. WILDS: A benchmark of in-the-wild distribution shifts. In *ICML*, 2021. 3

[32] Akshay Raj Dhamija, Manuel Günther, and Terrance E. Boult. Reducing network agnostophobia. In *NeurIPS 2018*, 2018. 3

[33] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In Doina Precup and Yee Whye Teh, editors, *ICML*, 2017. 3

[34] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *ICLR*, 2019. 3

[35] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *ICML*, 2019. 3

[36] Olivier Chapelle and Alexander Zien. Semi-supervised classification by low density separation. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, AISTATS*, 2005. 3

[37] Assaf Shocher, Nadav Cohen, and Michal Irani. "zero-shot" super-resolution using deep internal learning. In *CVPR*, 2018. 3

[38] David Bau, Hendrik Strobelt, William S. Peebles, Jonas Wulff, Bolei Zhou, Jun-Yan Zhu, and Antonio Torralba. Semantic photo manipulation with a generative image prior. *ACM Trans. Graph.*, 2019.

[39] Jogendra Nath Kundu, Naveen Venkat, Rahul M. V., and R. Venkatesh Babu. Universal source-free domain adaptation. In *CVPR*, 2020.

[40] Yujia Huang, James Gornet, Sihui Dai, Zhiding Yu, Tan M. Nguyen, Doris Y. Tsao, and Anima Anandkumar. Neural networks with recurrent generative feedback. In *NeurIPS*, 2020. 3

[41] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei A. Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *ICML*, 2020. 3

[42] Yuejiang Liu, Parth Kothari, Bastien van Delft, Baptiste Bellot-Gurlet, Taylor Mordan, and Alexandre Alahi. Ttt++: When does self-supervised test-time training fail or thrive? In *NeurIPS*, 2021. 3

[43] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno A. Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *ICLR*, 2021. 3, 19

[44] Marvin Zhang, Sergey Levine, and Chelsea Finn. MEMO: test time robustness via adaptation and augmentation. *CoRR*, abs/2110.09506, 2021. 3, 4, 8, 18

[45] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. In *ICLR Workshop*, 2017. 3

[46] Manli Shu, Zuxuan Wu, Micah Goldblum, and Tom Goldstein. Encoding robustness to image style via adversarial feature perturbations. In *NeurIPS*, 2021. 3

[47] Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. In *NeurIPS*, 2020. 3

[48] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *ICML*, 2020. 3

[49] Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. *CoRR*, abs/2202.10054, 2022. 4, 8

[50] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009. 5, 19

[51] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *ICML*, 2019. 5, 19

[52] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *CVPR*, pages 15262–15271, 2021. 5, 19

[53] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5

[54] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *NeurIPS*, 2019. 5, 19

[55] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*, Dec 2008. 6

[56] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *CVPR*, 2012. 7, 19

[57] 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*, June 2010. 7, 19

[58] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014. 7, 19

[59] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014. 7

[60] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *ICCV Workshops*, 2013. 7, 19

[61] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. Technical report, 2013. 7, 19

[62] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *CoRR*, abs/1212.0402, 2012. 7

[63] 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 J. Sel. Top. Appl. Earth Obs. Remote. Sens.*, 2019. 7

[64] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *CVPR Workshops*, 2004. 7

[65] Dan Hendrycks, Norman Mu, Ekin Dogus Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. In *ICLR*, 2020. 7, 8

[66] 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 *CVPR*, 2022. 8

[67] Junnan Li, Dongxu Li, Caiming Xiong, and Steven C. H. Hoi. BLIP: bootstrapping language-image pre-training for unified vision-language understanding and generation. *CoRR*, abs/2201.12086, 2022. 9

[68] Weili Nie, Zhiding Yu, Lei Mao, Ankit B. Patel, Yuke Zhu, and Anima Anandkumar. Bongard-logo: A new benchmark for human-level concept learning and reasoning. In *NeurIPS*, 2020. 15

[69] Yinbo Chen, Xiaolong Wang, Zhuang Liu, Huijuan Xu, and Trevor Darrell. A new meta-baseline for few-shot learning. *CoRR*, abs/2003.04390, 2020. 15

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.

- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] See our main results and discussion in Section 4
   (b) Did you describe the limitations of your work? [Yes] See section 6
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] We discuss the social impacts in Appendix A.1
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See implementation details in section 4 and the supplemental materials.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See implementation details in section 4.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We include an analysis in Appendix A.3.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See the supplemental materials.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes] We provide the license information in Appendix A.5
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Appendix

A.1 Broader Impact

We introduce TPT to promote the generalization of vision-language foundation models. We believe it is worth exploring the zero-shot robustness and generalization of large-scale foundation models as they present a promising direction toward more reliable machine learning systems. In this work, to better leverage the knowledge of pre-trained foundation models, we explore the idea of prompt tuning because it does not alter the inner representations of pre-trained models. We hope this work inspires future studies in reaching the full potential of foundation models, especially in their robustness and generalization.

A.2 Context-Dependent Visual Reasoning on Bongard-HOI

Background. For the task of context-dependent visual reasoning, such as Bongard-HOI [16], a test sample contains two sets of support images and a query image for evaluation. The two sets of support images exemplify the presence and the absence of a human-object interaction (HOI) concept (*e.g.,* "ride bike"). The model is then asked to infer whether the query image contains the underlying concept.

Specifically, each concept in this task is a visual relationship $c = \langle s, a, o \rangle$, with s being the subject ($s =$"human" for HOI tasks), a denoting the action and o for the object. Each test sample X_{test} captures a concept by presenting $c = \langle s, a, o \rangle$ in one set of support images (positive examples), while having the other set (negative examples) to demonstrate $c' = \langle s, a', o \rangle$, where $a' \neq a$. Note that neither o nor a is given explicitly, and it relies on model's reasoning ability to predict whether the query image contains the featured concept c of the test sample. Existing methods [68, 69] approach the Bongard-HOI problem by training the model on a collection of similar tasks (using the Bongard-HOI training split), so that it can make similar inferences on test samples at test time. When applying CLIP to this task, we do not use additional training data because CLIP has learned abundant visual concepts and thus is a natural fit for such visual reasoning tasks.

TPT for context-dependent visual reasoning. Different from image classification, where every image has one and only ground-truth label, the correctness of the prediction in Bongard-HOI depends on the context (*i.e.*, example images), which is binary (containing the concept c or not). In the case of binary labels, a straightforward prompting strategy is to hand-craft "labels" for positive and negative examples, such as "True/False" or "Yes/No". With our proposed TPT, on the other hand, we can directly learn an optimal label token $\boldsymbol{cls}$ on the example images in the test sample. More importantly, for visual reasoning, TPT can explicitly learn the context (*i.e.*, visual concept) in the form of text prompts, and assists visual reasoning of vision-language models with language context. Formally, given M support images in each test sample, the TPT objective for context-dependent reasoning can be written as:

$$\boldsymbol{p}^* = \arg\min_{\boldsymbol{p}, \boldsymbol{cls}} \frac{1}{M} \sum_{X \in \{X_{\mathcal{P}}, X_{\mathcal{N}}\}} \mathcal{L}(\mathcal{F}_{\boldsymbol{c}, \boldsymbol{cls}}(X), y), \tag{7}$$

where we assign $y \in \{0, 1\}$ to negative and positive example images respectively for computing the cross-entropy loss $\mathcal{L}$. Unlike in image classification, we tune the binary label tokens $\boldsymbol{cls} = \{\boldsymbol{cls}^1, \boldsymbol{cls}^2\}, \boldsymbol{cls}^i \in \mathbb{R}^{1,D}$ and prompt $\boldsymbol{p} \in \mathbb{R}^{L,D}$ simultaneously. For each image, we assemble the text input to CLIP as $T = \{T_1, T_2 \,|\, T_i = \{\boldsymbol{p}, \boldsymbol{cls}^i\}\}$. Note that the support set is an essential part of a Bongard-HOI sample, which provides the context for this context-dependent task. Therefore, our approach is still considered to work purely at test time, without training data or annotations (*i.e.*, TPT has not been trained on a collection of similar tasks from the Bongard-HOI training split).

Baselines. We include three previous methods for comparison: (1) The CNN-baseline [68] is a simple classifier trained on Bongard-HOI training data, for which the model is trained to map a training sample as a whole (including the support and query images) to a binary output, indicating whether the query image contains the corresponding concept; (2) The Meta-baseline [69] regards each sample in Bongard-HOI as a few-shot task and the model is trained on the Bongard-HOI training data with a meta-objective that aims to quickly adapt the model to new tasks; (3) HOITrans [15] is the previous best method on Bongard-HOI. It is a transformer-based HOI detection model that

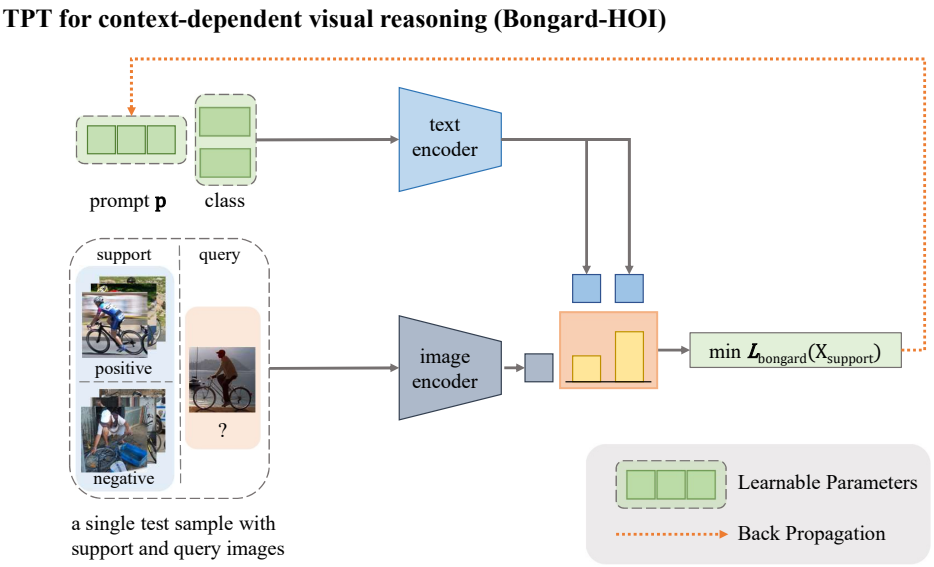


Figure 5: **Test-time Prompt Tuning (TPT) for context-dependent visual reasoning on Bongard-HOI benchmark.** A test sample in Bonagrd-HOI consists of several support images that exemplify a visual concept, and the model need to predict whether the query image contains the concept. TPT tune the prompt and class tokens simultaneously on the support images using the cross-entropy loss.

achieves state-of-the-art accuracy on various HOI detection benchmarks. It solves Bongard-HOI by comparing the detected HOIs of the query images to those of the support images.

Implementation details. For each Bongard-HOI test sample, TPT tunes the prompt prefix and class tokens simultaneously from scratch. All learnable tokens are initialized in the text embedding space from a Gaussian distribution with $\sigma = 0.02$. We optimize the prompt on all support images of a test sample for 64 steps, using the AdamW optimizer with a learning rate of 0.005, and then infer the query image with the updated prompt. For other baselines, we directly report the results from Jiang et al. [16], and we refer interested readers to this paper for more details. Note that the HOITrans model is trained on all possible HOI concepts, including the ones in the testing splits.

Results. In Table 4, we follow the setup in Jiang et al. [16], and compare TPT with previous methods on 4 test splits of Bongard-HOI respectively. In Bongard-HOI, test images are split into four sets by their overlap in the HOI concept with the training data: whether the action a or the object o has appeared in the training data. Note that our CLIP-based TPT is not trained on the training split of Bongard-HOI, and thus the definition of the four splits is not strictly applicable to TPT.

Table 4: **Evaluation on the Bongard-HOI benchmark**. CNN and Meta baselines are implemented based on a ResNet-50 (RN50). ($*$ denotes that the method uses ground truth bounding boxes to assist the inference.)

Method	Test Splits				Average
	seen act., seen obj.,	unseen act., seen obj.,	seen act., unseen obj.,	unseen act., unseen obj.,	
CNN-baseline	50.03	49.89	49.77	50.01	49.92
Meta-baseline*	58.82	58.75	58.56	57.04	58.30
HOITrans	59.50	64.38	63.10	62.87	62.46
TPT (w/ CLIP-RN50)	**66.39**	**68.50**	**65.98**	**65.48**	**66.59**

A.3 More Experiments.

Analysis on error bars. We run TPT multiple times using 3 different random seeds and report the average accuracy with standard deviation. The randomness of TPT mainly comes from random data augmentation and the one-step optimization. In addition, we report error bars for few-shot prompt tuning methods (CoOp and CoCoOp). The randomness of these methods mainly comes from the random few-shot sampling of the training data. We provide error bar analysis for the results in Table 1 and Table 2 in the main paper.

Table 5: **Robustness to natural distribution shifts.** We report the accuracy with an error bar (standard deviation) obtained from three runs with different random seeds.

Method	ImageNet Top1 acc. ↑	ImageNet-A Top1 acc. ↑	ImageNet-V2. Top1 acc. ↑	ImageNet-R. Top1 acc. ↑	ImageNet-Sketch Top1 acc. ↑	Average	OOD Average
CLIP-RN50	58.16	21.83	51.41	56.15	33.37	44.18	40.69
CoOp	**63.27** (± .07)	23.23 (± .19)	55.50 (± .09)	57.08 (± .42)	34.68 (± .03)	46.75(± .12)	42.62 (± .17)
CoCoOp	62.86 (± .11)	23.38 (± .50)	**55.59** (± .14)	57.55 (± .23)	34.74 (± .29)	46.82 (± .21)	42.82 (± .23)
TPT	60.77 (± .03)	**26.60**(± .13)	54.70 (± .11)	**59.08** (± .03)	**35.17** (± .08)	**47.27** (± .004)	**43.89** (± .003)
CLIP-ViT-B/16	66.73	47.87	60.86	73.98	46.09	59.11	57.2
CoOp	**71.71** (± .19)	49.99 (± .29)	**64.49** (± .39)	75.51 (± .26)	48.10 (± .14)	61.96 (± .25)	59.52 (± .26)
CoCoOp	70.70 (± .32)	50.76 (± .13)	63.93 (± .19)	76.09 (± .29)	**48.60** (± .38)	62.02 (± .20)	59.85 (± .19)
TPT	68.96 (± .03)	**54.47** (± .26)	63.46 (± .07)	**77.10** (± .04)	47.93 (± .03)	**62.38** (± .05)	**60.74** (± .06)

Table 6: **Cross-dataset generalization from ImageNet to fine-grained classification datasets**. CoOp and CoCoOp are tuned on ImageNet using 16-shot training data per category. Baseline CLIP and TPT methods does not require training data or annotations. We report the averaged top-1 accuracy on each dataset, with standard deviations obtained from three runs using different random seeds.

Method	Flower102	DTD	Pets	Cars	UCF101	Caltech101	Food101	SUN397	Aircraft	EuroSAT	Average
CLIP-RN50	61.75	40.37	83.57	55.70	58.84	85.88	73.97	58.80	15.66	23.69	55.82
CoOp	61.62 (± .2)	37.77 (± .9)	87.24 (± .2)	55.72 (± .8)	59.89 (± .8)	87.23 (± .6)	75.86 (± .2)	59.28 (± .9)	15.20 (± .4)	25.43 (± 4)	56.52 (± .7)
CoCoOp	**65.11** (± 1)	39.14 (± .7)	**87.83** (± .6)	56.40 (± .3)	58.57 (± 1)	86.95 (± .5)	**76.18** (± .5)	60.62 (± .9)	15.13 (± .5	**28.79** (± .9)	57.47 (± .2)
TPT	62.80 (± .3)	**41.43** (± .5)	84.42 (± .1)	**58.53** (± .1)	**60.64** (± .3)	**87.23** (± .2)	75.02 (± .1)	**61.46** (± .0)	**17.60** (± .4)	28.46 (± .1)	**57.76** (± .1)
CLIP-ViT-B/16	67.44	44.27	88.25	65.48	65.13	93.35	83.65	62.59	23.67	42.01	63.58
CoOp	68.25 (± 0.5)	42.34 (± 2)	89.38 (± .2)	63.35 (± 1)	67.17 (± 1)	92.82 (± .5)	83.74 (± .4)	64.51 (± .6)	19.99 (± 2)	40.22 (± 4)	63.18 (± .7)
CoCoOp	**71.59** (± .6)	45.48 (± .2)	**90.20** (± .2)	65.17 (± .2)	**68.77** (± .8)	**94.15** (± .3)	**84.83** (± 1)	**67.07** (± .3)	22.95 (± .7)	42.13 (± 3)	**65.23** (± .6)
TPT	68.79 (± .1)	**46.79** (± .1)	87.09 (± .1)	**66.38** (± .2)	67.86 (± .1)	94.13 (± .1)	84.67 (± .1)	65.41 (± .1)	**23.44** (± .3)	**42.78** (± .3)	64.73 (± .1)

From Table 5, our conclusion for remains the same as in Table 1, that TPT achieves higher accuracy than few-shot prompt tuning methods on OOD benchmarks. In addition, we find that on average TPT has smaller standard deviation than other methods.

In Table 6, we provide the same error bar analysis for the cross-dataset generalization experiment, corresponding to Table 2 in Section 4. Similar to the observation above, our conclusion for cross-dataset evaluation remains the same. On average, we find TPT to have smaller standard deviation than few-shot prompt tuning methods.

Comparison with model ensemble. We include more comparisons with conventional model ensembles of the baselines in Table 1. Note that there exists a fundamental difference between our "ensemble" (e.g., TPT + CoOp) and conventional model ensemble (e.g., naive "hand-crafted" ensemble, CoCoOp with different seeds, etc.). As our method (TPT) works solely at test time and uses a pre-defined prompt (which may come from a hand-crafted prompt, CoOp, or CoCoOp) as the initialization, it is complementary to CoOp, CoCoOp, and hand-crafted prompts.

Specifically, taking "TPT + CoCoOp" as an example, it is done in the following steps: (1). Using the prompt output by CoCoOp as the prompt initialization; (2). Running TPT to tune this prompt at test time to get the final result. However, conventional model ensemble aggregates the predictions obtained from different prompts.

As shown in the table above, ensembles of existing baselines (e.g., CoOp + CoCoOp) fail to bring improvement as substantial as their combination with TPT (e.g. CoOp + TPT or CoCoOp + TPT).

Other baselines using data augmentation. To ablate the contribution of data augmentation in our method, we include two additional baselines that are based on data augmentations without any optimization (and without TPT). The *averaged prediction* baseline is to run inference on the

Table 7: **Robustness to Natural Distribution Shifts.**. CoOp and CoCoOp are tuned on ImageNet using 16-shot training data per category. Baseline CLIP, prompt ensemble and TPT do not require training data.

Method	ImageNet Top1 acc. ↑	ImageNet-A Top1 acc. ↑	ImageNet-V2. Top1 acc. ↑	ImageNet-R. Top1 acc. ↑	ImageNet-Sketch Top1 acc. ↑	Average	OOD Average
CLIP-RN50	58.16	21.83	51.41	56.15	33.37	44.18	40.69
Hand-crafted ensemble	59.81	23.24	52.91	**60.72**	35.48	46.43	43.09
CoOp	63.33	23.06	55.40	56.60	34.67	46.61	42.43
CoOp (ensemble 3 seeds)	61.66	22.96	54.14	57.89	34.94	46.32	42.48
CoOp + hand-crafted ensemble	63.60	23.23	55.63	57.07	34.84	46.87	42.69
CoCoOp	62.81	23.32	55.72	57.74	34.48	46.81	42.82
CoCoOp (ensemble 3 seeds)	63.34	24.27	56.12	58.24	35.46	47.49	43.52
CoCoOp + hand-crafted ensemble	63.03	24.16	55.73	57.88	35.22	47.20	43.25
CoCoOp + CoOp	63.86	23.69	56.45	57.7	35.5	47.44	43.34
TPT (ours)	60.74	26.67	54.7	59.11	35.09	47.26	43.89
TPT + CoOp	**64.73**	**30.32**	**57.83**	58.99	**35.86**	**49.55**	**45.75**
TPT + CoCoOp	62.93	27.4	56.6	59.88	35.43	48.45	44.83

augmented images and then average the probability. The *majority vote* baseline runs majority vote in the predictions of augmented images, and takes the results as the final prediction.

We find that both methods fail to achieve the comparable improvement as TPT. It uggests that it is non-trivial to design an algorithm for using the augmented images.

Table 8: **Out-of-distribution evaluation of data augmentation baselines**. We include two additional baselines that are based on data augmentations without any optimization (and without TPT). The limited performance of these baselines suggests that it is non-trivial to design an algorithm for using the augmented images.

Method	ImageNet Top1 acc. ↑	ImageNet-A Top1 acc. ↑	ImageNet-V2. Top1 acc. ↑	ImageNet-R. Top1 acc. ↑	ImageNet-Sketch Top1 acc. ↑	Average	OOD Average
CLIP-RN50	58.16	21.83	51.41	56.15	33.37	44.18	40.69
averaged prediction	57.94	22.71	52.02	51.25	29.51	42.69	38.87
majority vote	59.53	25.67	53.56	54.77	32.27	45.16	41.57
TPT (Ours)	**60.74**	**26.67**	**54.70**	**59.11**	**35.09**	**47.26**	**43.89**

A.4 Apply Confidence Selection to Other Methods

In section 5, we show that confidence selection is an important part of TPT that improves the baseline entropy minimization. In this section, we further show that confidence selection can also benefit other entropy-based test-time optimization methods, which work on different model architectures and optimize different parameter groups. MEMO [44] minimizes the marginal entropy of model's predictions across augmented views at test time, by adapting all parameters of a network model.

In Table 9, we implement MEMO based on a standard ResNet-50 from PyTorch, following their original hyper-parameter configurations. To apply confidence selection to a method, we try different cutoff percentile ρ that controls the threshold for confidence selection (smaller ρ means a higher confidence threshold). We find MEMO can also benefit from confidence selection, and the improvement increases as the threshold increases (*i.e.* ρ becomes smaller).

Table 9: **Apply confidence selection to other test-time methods**. All methods are based on a standard ResNet-50. ρ denotes applying confidence selection to a method with a cutoff percentile ρ.

Method	ImageNet Top1 acc. ↑	ImageNet-A Top1 acc. ↑	ImageNet-V2. Top1 acc. ↑	ImageNet-R. Top1 acc. ↑	ImageNet-Sketch Top1 acc. ↑	Average	OOD Average
ResNet-50	76.13	0.00	63.20	36.17	24.09	39.92	30.87
MEMO	77.23	0.75	65.03	41.34	27.72	42.41	33.71
MEMO ($\rho = 0.7$)	77.56	0.92	65.51	41.93	28.20	42.82	34.14
MEMO ($\rho = 0.5$)	**77.72**	1.15	65.77	42.29	**28.55**	43.10	34.44
MEMO ($\rho = 0.3$)	77.57	1.43	**65.85**	42.64	28.33	43.16	34.56
MEMO ($\rho = 0.1$)	77.38	**2.59**	65.37	**42.90**	28.04	**43.26**	**34.72**

A.5 License information of the assets used in this work.

Datasets. Below are the datasets used int his paper that have known license information:
The following datasets used in this paper are under the MIT License: ImageNet-A [52], ImageNetV2 [51], ImageNet-R [13], ImageNet-Sketch [54].
The following datasets used in this paper are under the CC BY-SA 4.0 License: Oxford-IIIT Pets [56].
The following datasets used in this paper are for research purposes only (term of access): ImageNet [50], DTD [58], StanfordCars [60], SUN397 [57], FGVC-Aircraft [61].

Source code. We use the implementation of existing baseline methods for reporting their results in this paper. Below are their license information: Source code used in this paper that are under the MIT License: CLIP [1], CoOp [26], CoCoOp [14], TENT [43].

A.6 An overview of different prompting strategies for CLIP.

In addition to our discussion on related work in Section 2, we summarize the differences between existing prompting strategies for CLIP in Table 10. We focus on three preferred properties of a prompting strategy, and use them to categorize the methods.

Table 10: **An overview of different prompting strategies for CLIP**. "learnable" means the prompt is optimized based on certain objective functions. "no training data" means that no additional data are needed for tuning the prompt. "input-dependent" means the prompt is adaptive to each input instance.

Prompt Type	learnable	no training data	input-dependent
Hand-crafted [1]	✗	✓	✗
CoOp [26]	✓	✗	✗
CoCoOp [14]	✓	✗	✓
TPT (ours)	✓	✓	✓

Comparison of training and inference budgets. We include a comparison of training and inference budgets for prompting strategies of CLIP.

Table 11: **Comparison of training and testing budgets of prompting strategies for CLIP**.

	Hand-crafted [1]	CoOp [26]	CoCoOp [14]	TPT (Ours)
Inference speed (seconds/iter)	0.10	0.10	0.11	0.25
Number of training samples	0	16K	16K	0
Number of training iterations	0	12.5K	800K	0
Number of additional parameters	0	0	65.8K	0

The major computation overhead of TPT comes from the 1-step optimization, which involves back propagations through the text encoder of CLIP. Another overhead comes from the data augmentation, which can be parallelized with little memory increase, as the CLIP's image branch does not require back propagation. However, note that TPT works solely at test time, and therefore it does not have any training budget. In addition, as shown by our empirical results, prompt tuning without training data can generalize better to unseen distributions.

A.7 Reproducibility

The code for experiments in this paper can be found here: `https://tinyurl.com/yr3zmhma`. (The code is also included in the supplemental materials.)

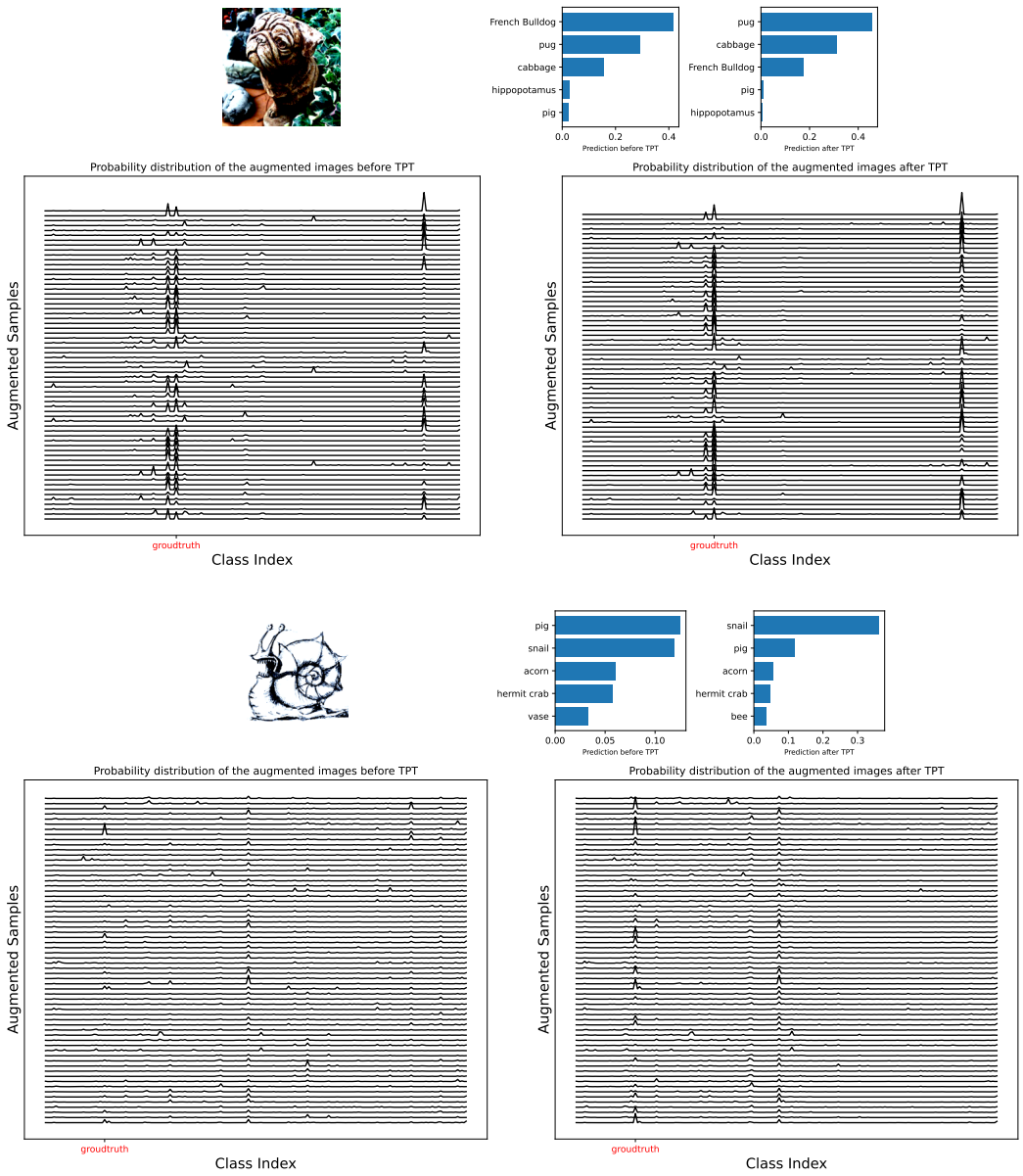
For evaluation on natural distribution shifts, we use a single V100 GPU with 32GB memory. For all other experiments, we use a single V100 GPU with 16GB memory.

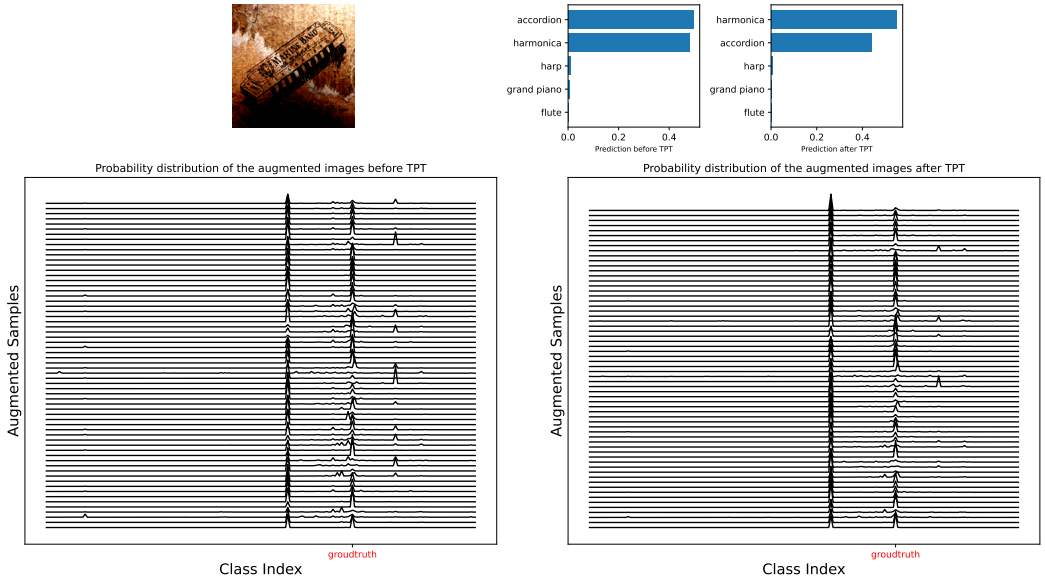
Details of hyper-parameter tuning. For the OOD experiments on ImageNet variants, we follow the setting from [14]. We find the optimal set of hyper-parameters on the original ImageNet validation set (not including any of the out-of-distribution data). For the fine-grained image classification, each

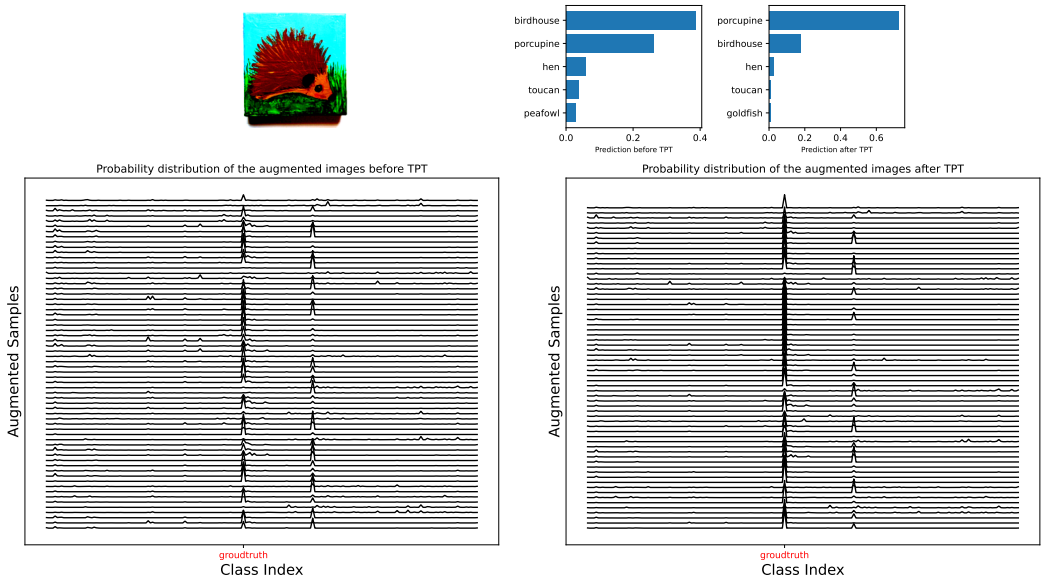
dataset comes with a validation split and a testing split, and we use the former for hyper-parameter tuning. Among the 10 fine-grained datasets, we choose a set of hyper-parameter that achieves the best average validation accuracy.

A.8 Qualitative Analysis

In this section, we provide qualitative analysis on the effect of TPT to the probability distribution of the augmented views of a test sample. In each figure, the top left panel shows the test image, and the bar plots on top right shows CLIP's prediction on the test image before (on the left) and after (on the right) we apply TPT. The bottom two panels shows the probability distribution among 200 classes (of ImageNet-R) of 64 augmented views. Each peak of a prediction curve indicates high probability in the corresponding class. The prediction probability of the original test sample is in the bottom, and we mark the index of the groundtruth class on x-axis. From the example images, we find that TPT can effectively make the predictions more consistent across different augmented views.




accordion
harmonica
harp
grand piano
flute
0.0
0.2
0.4
Prediction before TPT
harmonica
accordion
harp
grand piano
flute
0.0
0.2
0.4
Prediction after TPT
Probability distribution of the augmented images before TPT
Augmented Samples
groudtruth
Class Index
Probability distribution of the augmented images after TPT
Augmented Samples
groudtruth
Class Index


birdhouse
porcupine
hen
toucan
peafowl
0.0
0.2
0.4
Prediction before TPT
porcupine
birdhouse
hen
toucan
goldfish
0.0
0.2
0.4
0.6
Prediction after TPT
Probability distribution of the augmented images before TPT
Augmented Samples
groudtruth
Class Index
Probability distribution of the augmented images after TPT
Augmented Samples
groudtruth
Class Index