# *Just Shift It*: Test-Time Prototype Shifting for Zero-Shot Generalization with Vision-Language Models

## Supplementary Material

This document provides more details of our approach and additional experimental results, organized as follows:

- § A Implementation Details.
- § B Additional Quantitative Results with Different Random Seeds.
- § C Additional Ablation Studies.
- § D Comparison to Training-Free Methods.
- § E Research Impact and Limitations.

## A. Implementation Details of TPS

Algorithm 2 shows more detailed pseudocode in PyTorch-like style for Test-Time Prototype Shifting over an entire dataset. We will release the models and source code to ensure reproducibility.

## B. Main Results With More Random Seeds

In Sec B.1 and B.2, we run Test-Time Prototype Shifting (TPS) over 3 random seeds on both the natural distribution shifts (Table 1) and cross-dataset generalization (Table 3), respectively. The randomness comes from the image augmentation in creating a diverse minibatch for the entropy minimization objective.

### B.1. Natural Distribution Shifts

From Table 7, we observe that our conclusion from Sec 4.1.1 still holds. That is, our method outperforms SoTA TPT [47] by $> 3.4\%$ on average. We also observe that augmenting the TPT-tuned class prototypes with more advanced off-the-shelf prototypes only boosts performance by a mere $0.5\%$ on average over vanilla TPT, demonstrating TPT's limitation in maximally leveraging these advanced prototypes.

### B.2. Cross-Dataset Generalization

From Table 8, we see that our conclusion from Sec 4.1.2 remains valid. Specifically, TPS outperforms TPT [47] by $> 2\%$ on average. Similarly to Sec B.1, we observe that taking the mean of the TPT-tuned and advanced off-the-shelf prototypes increases performance by only $0.5\%$ on average over TPT, demonstrating TPT's inflexibility in utilizing these more robust class representations.

### B.3. Context-Dependent Visual Reasoning

From Table 9, we see that our conclusion from Sec 4.2 remains valid. Specifically, TPS outperforms TPT [47] by $> 0.5\%$ on average.

## C. Full Ablations

In Sec C.1, we report full ablations on TPS on the effectiveness of feature-space shift on various prototypes. These results are comparable to those reported in Sec 4.4. In Sec C.2, we include additional ablations to observe the effect of learning a class-specific shift over a universal shift for all classes. In Sec C.3, we explore variants on prototype generation using the class-agnostic CLIP ImageNet context prompt templates [42] and the class-specific descriptors generated using GPT-4 [38]. All these ablations are run over 3 random seeds.

### C.1. Effect of Shift on Different Prototypes

Full comparisons between zero-shot and feature-shifted performance on all natural distribution shift and cross-domain generalization benchmark datasets over 3 random seeds are in Tables 11 and 12, respectively. We demonstrate that our conclusion from Sec 4.4.1 stills holds – that learning a small perturbation in the feature space results in performance gains of $> 4\%$ and up to $1\%$ on average across natural distribution shift and cross-domain generalization tasks regardless of what prototypes are used.

### C.2. Effect of Per-Class vs. Shared Shift

Test-time prompt tuning methods involve tuning a prompt that is shared across all classes in a dataset. Given that the tuneable prompt tokens form a portion of the text encoder input, these full prompts are then mapped to the embedding space with the encoder's learned complex feature-space mapping. This results in non-linear perturbations from the original class prototypes. However, for our method, tuning shift parameters that are shared for all class prototypes in the feature-space means that the relative distance between class prototypes will remain constant before and after test-time shift tuning, limiting the expressive capability of the learned shift. Rather, we believe that each class prototype should be modulated by slightly different magnitudes and/or directions to provide more degrees of freedom in capturing the class-level distribution shifts in addition to the dataset-level shifts present in a domain gap.

Table 13 shows that, on average, learning a per-class shift increases performance by $> 1.2\%$ regardless of which prototypes are used. Moreover, we see that Table 14 demonstrates that, on average, learning a per-class shift increases performance by around $0.5\%$ on average over different prototype settings. This demonstrates that learning per-class

**Algorithm 2** Test-Time Prototype Shifting Pseudocode in PyTorch-like style

```
# Define frozen parameters
image_encoder = CLIPImageEncoder()
prototypes = load_class_prototypes()

predictions = []
for img, label in data_loader:
    # Test-Time Shifting
    shift_params = nn.Parameter(torch.zeros(num_classes, embed_dim), requires_grad=True)
    aug_imgs = [aug(img) for i in range(batch_size - 1)]
    imgs = torch.stack([img] + aug_imgs, dim=0)
    image_features = image_encoder(imgs)

    text_features = prototypes + shift_params
    text_features = F.normalize(text_features, dim=-1)

    logits = (logit_scale * text_features @ image_features.T)

    # Confidence selection
    entropies = compute_batch_entropies(logits)
    top_k_idx = torch.argsort(batch_entropy, descending=False)[:k]

    loss = compute_average_entropy(logits[top_k_idx])
    optimizer.zero_grad()
    loss.backward()
    optimizer.step()

    # Test-Time Inference
    new_prototypes = prototypes + shift_params
    new_prototypes = F.normalize(new_prototypes, dim=-1)

    logits = (logit_scale * new_prototypes @ image_features[0].unsqueeze(0).T)
    pred = torch.argmax(logits)

    predictions.append(pred)

return predictions
```

| Method | ImageNet | ImageNet-A | ImageNet-V2 | ImageNet-R | ImageNet-Sketch | Average | OOD Average |
|---|---|---|---|---|---|---|---|
| | | | *Test-Time Adaptation Baselines* | | | | |
| TPT [47] | 68.96 (±.03) | 54.47 (±.26) | 63.46 (±.07) | 77.10 (±.04) | 47.93 (±.03) | 62.38 (±.05) | 60.74 (±.06) |
| TPT + (templates + descriptors)* | 69.51 (±.05) | 54.94 (±.17) | 63.86 (±.11) | 77.57 (±.11) | 48.38 (±.04) | 62.85 (±.03) | 61.19 (±.04) |
| **Ours** | **71.43** (±.06) | **60.78** (±.21) | **65.00** (±.09) | **80.06** (±.13) | **50.97** (±.09) | **65.65** (±.06) | **64.20** (±.08) |

Table 7. Acc@1 of zero-shot image classification with CLIP-ViT-B/16 backbone on ImageNet and its OOD variants over 3 random seeds. Best performances are in **bold**.

shifts allows the model to capture both dataset-level and class-level distribution shifts in a domain gap.

### C.3. Prototype Variants

We explore different methods for creating class prototypes. Specifically, we experiment with different forms of aggregating the text encoded with the 80 ImageNet context prompts from CLIP [42] and our LLM-generated descriptors. The CLIP ImageNet templates are class-agnostic and add image-level characteristics whereas the descriptors are class-specific and add class-level semantic information.

Tables 15 and 16 compare three variants of pooling these CLIP templated embeddings and descriptor embeddings to obtain a single class prototype. Similarly to the conclusion

| Method | Flower102 | DTD | Pets | Cars | UCF101 | CalTech101 | Food101 | SUN397 | Aircraft | EuroSAT | Average |
|---|---|---|---|---|---|---|---|---|---|---|---|
| TPT [47] | 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) |
| TPT + (templates + descriptors)* | 69.67 (±.11) | 47.56 (±.55) | 87.88 (±.02) | 66.91 (±.17) | 68.35 (±.21) | 94.17 (±.13) | 84.89 (±.07) | 66.23 (±.12) | 23.55 (±.31) | 43.12 (±.18) | 65.23 (±.06) |
| **Ours** | **71.47** (±.12) | **51.00** (±.47) | **87.45** (±.09) | **68.99** (±.10) | **70.98** (±.24) | **94.90** (±.16) | **85.15** (±.08) | **68.85** (±.16) | **25.82** (±.45) | **44.61** (±.11) | **66.92** (±.04) |

Table 8. Acc@1 of zero-shot image classification with CLIP-ViT-B/16 backbone on cross-dataset generalization over 3 random seeds. Best performances are in **bold**.

| Method | seen act., seen obj., | unseen act., seen obj., | seen act., unseen obj., | unseen act., unseen obj., | Average |
|---|---|---|---|---|---|
| TPT (reprod.) | 65.81 (±.12) | 69.15 (±.10) | 65.69 (±.01) | **66.87** (±.03) | 66.88 (±.04) |
| **Ours** (Shift) | **66.67** (±0.68) | **70.31** (±1.67) | **66.00** (±1.38) | 66.67 (±0.40) | **67.41** (±.98) |

Table 9. Acc on the Bongard-HOI benchmark with CLIP-ResNet-50 backbone over 3 random seeds. Best performances are in **bold**.

| Method | ImageNet | ImageNet-R | ImageNet-Sketch | Cross-Dataset Average | Needs Support Set |
|---|---|---|---|---|---|
| SuS-X-SD-C [51] | 61.65 | 61.69 | 35.88 | 60.49 | ✓ |
| SuS-X-LC-P [51] | 61.80 | 61.62 | 36.25 | 60.64 | ✓ |
| CALIP [12] | 60.57 | - | - | 59.34 | ✗ |
| **Ours** (Shift + SuS-X descriptors) | **63.52** | **63.66** | **37.66** | **61.47** | ✗ |

Table 10. Acc@1 of zero-shot image classification with CLIP-ResNet-50 backbone on ImageNet and its OOD variants. Best performances are in **bold**.

of Sec 4.4.1, we observe that in general, the gains observed using more advanced prototypes in the zero-shot setting almost directly translate to the test-time adaptation setting with shifting. In Sec 4, we present the results of our method using prototypes that are a micro average of the CLIP templates and LLM-generated descriptors.

## D. Comparison to Training-Free Methods

In a similar spirit to zero-shot test-time adaptation, training-free methods perform domain adaptation without tuning any parameters. We compare our method to state-of-the-art training-free methods in Table 10. We show that our method, TPS, when using the same GPT-3-generated [2] text prompts from the official SuS-X [51] code, out-performs both CALIP [12] and SuS-X [51] without any additional image support set constructed with Stable Diffusion [44] or LAION-5B [46]. This demonstrates how a simple feature-space shift is more effective than complex training-free methods. For fair comparison, we compare TPS with SuS-X results with fixed hyperparameter settings as ours are not tuned per dataset and use the same CLIP-ResNet50 backbone.

## E. Research Impact and Limitations

We propose TPS, a framework that can be used to easily and effectively improve zero-shot generalization of VLMs. Given the large-scale training of foundation VLMs, we believe it is important to understand different ways to better leverage the resulting rich multi-modal contrastive representation spaces in parameter- and runtime-constrained settings. We propose to learn a slight perturbation to the class prototypes to maintain the overall representation quality of the pre-trained embedding space while learning a better alignment to the OOD target dataset. We hope that this framework can inspire future work to explore other tasks where learning directly in the feature space can be an efficient alternative to more complex tuning approaches.

Our work builds on the CLIP [42] representation space and uses GPT-4 [38] to generate class descriptors to create more advanced class prototypes. Thus, our model has the potential to magnify the biases of both these models. Future studies may explore how to best leverage these models' capabilities without promoting its biases.

| Prompt Type | Setting | ImageNet | ImageNet-A | ImageNet-V2 | ImageNet-R | ImageNet-Sketch | Average | OOD Average |
|---|---|---|---|---|---|---|---|---|
| Vanilla | Zero-Shot | 66.74 | 47.79 | 60.89 | 73.99 | 46.12 | 59.10 | 57.20 |
| | + shift | 68.81 (±.03) | 58.11 (±.16) | 63.51 (±.17) | 76.98 (±.05) | 48.11 (±.09) | 63.10 (±.08) | 61.68 (±.09) |
| | Δ | **+ 2.07** | **+ 10.32** | **+ 2.62** | **+ 2.99** | **+ 1.99** | **+ 4.00** | **+ 4.48** |
| CoOp [60] | Zero-Shot | 71.51 | 49.71 | 64.20 | 75.21 | 47.99 | 61.72 | 59.28 |
| | + shift | 73.76 (±.04) | 60.43 (±.12) | 66.84 (±.10) | 77.39 (±.05) | 49.08 (±.06) | 65.50 (±.02) | 63.44 (±.03) |
| | Δ | **+ 2.25** | **+ 10.72** | **+ 2.64** | **+ 2.18** | **+ 1.09** | **+ 3.78** | **+ 4.16** |
| CLIP templates | Zero-Shot | 68.35 | 49.95 | 61.97 | 77.59 | 48.21 | 61.21 | 59.43 |
| | + shift | 70.39 (±.06) | 60.47 (±.07) | 64.66 (±.04) | 80.70 (±.04) | 50.38 (±.14) | 65.32 (±.03) | 64.05 (±.02) |
| | Δ | **+ 2.04** | **+ 10.52** | **+ 2.69** | **+ 3.11** | **+ 2.17** | **+ 4.11** | **+ 4.62** |
| Descriptors | Zero-Shot | 68.52 | 48.91 | 61.78 | 74.81 | 47.68 | 60.34 | 58.29 |
| | + shift | 70.38 (±.03) | 59.21 (±.09) | 63.80 (±.07) | 77.49 (±.12) | 49.57 (±.06) | 64.09 (±.02) | 62.52 (±.03) |
| | Δ | **+ 1.86** | **+ 10.30** | **+ 2.02** | **+ 2.68** | **+ 1.89** | **+ 3.75** | **+ 4.23** |
| CLIP templates + Descriptors | Zero-Shot | 69.54 | 50.51 | 63.01 | 77.18 | 48.84 | 61.82 | 59.88 |
| | + shift | 71.43 (±.06) | 60.78 (±.21) | 65.00 (±.09) | 80.06 (±.13) | 50.97 (±.09) | 65.65 (±.06) | 64.20 (±.08) |
| | Δ | **+ 1.89** | **+ 10.27** | **+ 1.99** | **+ 2.88** | **+ 2.13** | **+ 3.83** | **+ 4.32** |

Table 11. Acc@1 for zero-shot and with feature-space shift with features initialized using different prototype generation techniques on ImageNet and its out-of-distribution variants. Results are over 3 random seeds.

| Prompt Type | Setting | Flower102 | DTD | Pets | Cars | UCF101 | CalTech101 | Food101 | SUN397 | Aircraft | EuroSAT | Average |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Vanilla | Zero-Shot | 67.28 | 44.44 | 87.98 | 65.24 | 65.08 | 92.98 | 83.80 | 62.55 | 23.70 | 41.42 | 63.45 |
| | + shift | 67.75 (±.10) | 45.69 (±.10) | 87.57 (±.10) | 67.60 (±.23) | 66.79 (±.21) | 93.79 (±.08) | 84.62 (±.03) | 64.58 (±.03) | 24.75 (±.39) | 41.35 (±.03) | 64.45 (±.04) |
| | Δ | **+ 0.47** | **+ 1.25** | - 0.41 | **+ 2.36** | **+ 1.71** | **+ 0.81** | **+ 0.82** | **+ 2.03** | **+ 1.05** | - 0.07 | **+ 1.00** |
| CLIP templates | Zero-Shot | 65.57 | 44.86 | 88.25 | 66.19 | 67.46 | 93.67 | 83.77 | 65.78 | 23.64 | 47.74 | 64.69 |
| | + shift | 66.41 (±.05) | 45.61 (±.19) | 87.99 (±.10) | 68.66 (±.31) | 68.02 (±.11) | 93.85 (±.14) | 84.54 (±.08) | 67.19 (±.05) | 24.66 (±.13) | 48.28 (±.20) | 65.52 (±.05) |
| | Δ | **+ 0.84** | **+ 0.75** | - 0.26 | **+ 2.47** | **+ 0.56** | **+ 0.18** | **+ 0.77** | **+ 1.41** | **+ 1.02** | **+ 0.54** | **+ 0.83** |
| Descriptors | Zero-Shot | 71.13 | 52.72 | 86.75 | 65.15 | 70.53 | 94.08 | 84.12 | 67.10 | 25.26 | 43.31 | 66.02 |
| | + shift | 71.69 (±.15) | 53.80 (±.21) | 87.82 (±.19) | 67.00 (±.14) | 71.18 (±.15) | 94.56 (±.08) | 84.78 (±.05) | 68.25 (±.18) | 26.27 (±.09) | 42.11 (±.18) | 66.75 (±.06) |
| | Δ | **+ 0.56** | **+ 1.08** | **+ 1.07** | **+ 1.85** | **+ 0.65** | **+ 0.48** | **+ 0.66** | **+ 1.15** | **+ 1.01** | - 1.20 | **+ 0.73** |
| CLIP templates + Descriptors | Zero-Shot | 70.52 | 49.94 | 87.22 | 66.48 | 70.24 | 94.12 | 84.47 | 67.55 | 24.69 | 44.14 | 65.94 |
| | + shift | 71.47 (±.12) | 51.00 (±.47) | 87.45 (±.09) | 68.99 (±.10) | 70.98 (±.24) | 94.90 (±.16) | 85.15 (±.08) | 68.85 (±.16) | 25.82 (±.45) | 44.61 (±.11) | 66.92 (±.04) |
| | Δ | **+ 0.95** | **+ 1.06** | **+ 0.23** | **+ 2.51** | **+ 0.74** | **+ 0.78** | **+ 0.68** | **+ 1.30** | **+ 1.13** | **+ 0.47** | **+ 0.98** |

Table 12. Acc@1 for zero-shot and with feature-space shift with features initialized using different prototype generation techniques on cross-domain generalization datasets. Results are over 3 random seeds.

| Method | ImageNet | ImageNet-A | ImageNet-V2 | ImageNet-R | ImageNet-Sketch | Average | OOD Average |
|---|---|---|---|---|---|---|---|
| Shared | 71.23 (±.02) | 56.57 (±.19) | 64.98 (±.03) | 79.31 (±.03) | 50.80 (±.06) | 64.58 (±.04) | 62.92 (±.06) |
| Per class | **71.43** (±.06) | **60.78** (±.21) | **65.00** (±.09) | **80.06** (±.13) | **50.97** (±.09) | **65.65** (±.06) | **64.20** (±.08) |

Table 13. Acc@1 for learning a shared vs. per-class shift on top of different prototypes over 3 random seeds. Best performances are in **bold**.

| Method | Flower102 | DTD | Pets | Cars | UCF101 | CalTech101 | Food101 | SUN397 | Aircraft | EuroSAT | Average |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Shared | 71.36 (±.12) | 50.49 (±.12) | **87.46** (±.12) | 67.33 (±.06) | 70.77 (±.12) | 94.35 (±.06) | 84.82 (±.01) | 68.12 (±.04) | 25.27 (±.02) | **44.67** (±.06) | 66.47 (±.03) |
| Per-class | **71.47** (±.12) | **51.00** (±.47) | 87.45 (±.09) | **68.99** (±.10) | **70.98** (±.24) | **94.90** (±.16) | **85.15** (±.08) | **68.85** (±.16) | **25.82** (±.45) | 44.61 (±.11) | **66.92** (±.04) |

Table 14. Acc@1 for learning a shared vs. per-class shift on top of different prototypes over 3 random seeds. Best performances are in **bold**.

| Prompt Type(s) | Pooling Method | ImageNet | ImageNet-A | ImageNet-V2 | ImageNet-R | ImageNet-Sketch | Average | OOD Average |
|---|---|---|---|---|---|---|---|---|
| | | | | | *Zero-Shot* | | | |
| Vanilla prompt | N/A | 66.74 | 47.79 | 60.89 | 73.99 | 46.12 | 59.10 | 57.20 |
| CLIP templates + Descriptors | Macro | 68.73 | 50.32 | 62.31 | **77.67** | 48.56 | 61.52 | 59.72 |
| CLIP templates + Descriptors | Micro | **69.54** | 50.51 | **63.01** | 77.18 | 48.84 | **61.82** | **59.88** |
| CLIP templates × Descriptors | Macro | 69.03 | **50.73** | 62.22 | 76.91 | **49.07** | 61.59 | 59.73 |
| | | | | | *With Shift* | | | |
| Vanilla prompt | N/A | 68.81 (±.03) | 58.11 (±.16) | 63.51 (±.17) | 76.98 (±.05) | 48.11 (±.09) | 63.10 (±.08) | 61.68 (±.09) |
| CLIP templates + Descriptors | Macro | 70.75 (±.08) | **60.86** (±.09) | 64.95 (±.11) | **80.84** (±.03) | 50.70 (±.11) | 65.62 (±.02) | **64.34** (±.02) |
| CLIP templates + Descriptors | Micro | **71.43** (±.06) | 60.78 (±.21) | **65.00** (±.09) | 80.06 (±.13) | 50.97 (±.09) | **65.65** (±.06) | 64.20 (±.08) |
| CLIP templates × Descriptors | Macro | 70.82 (±.02) | 60.42 (±.06) | 64.50 (±.05) | 79.53 (±.09) | **51.13** (±.02) | 65.28 (±.01) | 63.89 (±.02) |

Table 15. Acc@1 for different variants of prototype generation, i.e. ways of combining templates and descriptors, on natural distribution shifts, over 3 random seeds. Best performances for each setting are in **bold**.

| Prompt Type(s) | Pooling Method | Flower102 | DTD | Pets | Cars | UCF101 | CalTech101 | Food101 | SUN397 | Aircraft | EuroSAT | Average |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | *Zero-Shot* | | | | | |
| Vanilla prompt | N/A | 67.28 | 44.44 | 87.98 | 65.24 | 65.08 | 92.98 | 83.80 | 62.55 | 23.70 | 41.42 | 63.45 |
| CLIP templates + Descriptors | Macro | 66.91 | 45.86 | **88.33** | 66.46 | 68.12 | 93.83 | 83.97 | 66.34 | 24.03 | 46.62 | 65.05 |
| CLIP templates + Descriptors | Micro | 70.52 | 49.94 | 87.22 | **66.48** | 70.24 | 94.12 | **84.47** | 67.55 | 24.69 | 44.14 | 65.94 |
| CLIP templates $\times$ Descriptors | Macro | **72.03** | **50.83** | 86.21 | 66.12 | **70.90** | **94.16** | 83.73 | **67.98** | **25.53** | **47.19** | **66.47** |
| | | | | | | | *With Shift* | | | | | |
| Vanilla prompt | N/A | 67.75 ($\pm$.10) | 45.69 ($\pm$.10) | 87.57 ($\pm$.10) | 67.60 ($\pm$.23) | 66.79 ($\pm$.21) | 93.79 ($\pm$.08) | 84.62 ($\pm$.03) | 64.58 ($\pm$.03) | 24.75 ($\pm$.39) | 41.35 ($\pm$.03) | 64.45 ($\pm$.04) |
| CLIP templates + Descriptors | Macro | 67.52 ($\pm$.27) | 46.43 ($\pm$.28) | **88.00** ($\pm$.13) | **69.04** ($\pm$.16) | 68.67 ($\pm$.18) | 94.16 ($\pm$.18) | 84.77 ($\pm$.04) | 67.70 ($\pm$.08) | 24.79 ($\pm$.30) | **47.09** ($\pm$.19) | 65.82 ($\pm$.06) |
| CLIP templates + Descriptors | Micro | 71.47 ($\pm$.12) | 51.00 ($\pm$.47) | 87.45 ($\pm$.09) | 68.99 ($\pm$.10) | 70.98 ($\pm$.24) | **94.90** ($\pm$.16) | **85.15** ($\pm$.08) | 68.85 ($\pm$.16) | 25.82 ($\pm$.45) | 44.61 ($\pm$.11) | 66.92 ($\pm$.04) |
| CLIP templates $\times$ Descriptors | Macro | **72.53** ($\pm$.12) | **52.56** ($\pm$.09) | 86.15 ($\pm$.05) | 68.89 ($\pm$.07) | **71.44** ($\pm$.20) | 94.43 ($\pm$.06) | 84.44 ($\pm$.08) | **69.04** ($\pm$.02) | **26.51** ($\pm$.26) | 45.65 ($\pm$.15) | **67.16** ($\pm$.03) |

Table 16. Acc@1 for different variants of knowledge injection, i.e. ways of combining templates and descriptors, over 3 random seeds on cross-dataset generalization tasks. Best performances in each setting are in **bold**.