Think Twice: Test-Time Reasoning for Robust CLIP Zero-Shot Classification

Supplementary Material

7. Experiment details for section 3

In this section, we present the experimental details for Section 3. We perform zero-shot classification on three variants of the Waterbirds dataset: original variant (OV), core features on a black background (Core Blk), and core features on a white background (Core Wht). We use the segmented bird images from the Caltech-UCSD Birds-200-2011 (CUB) dataset [48]. The images of water birds were resampled so that the number of water bird images equals the number of land bird images, resulting in 9410 images in total. We utilize the segmented bird images and use different backgrounds to fill in the images: (1) water/land backgrounds from the Places dataset [63]; (2) backgrounds with pure color (black or white). We randomly sample from these three dataset variants, as illustrated in Figure 9.

- Original variant of Waterbirds testset (OV): This variant retains both the bird species as the core attribute and the background habitat as a spurious attribute. While the individual attributes (e.g., water birds vs. land birds, water backgrounds vs. land backgrounds) are balanced, their combinations exhibit spurious correlations. Specifically, images of water birds with water backgrounds and land birds with land backgrounds each constitute 40% of the dataset, whereas water birds with land backgrounds and land birds with water backgrounds each account for 10%.
- Core Blk & Core Wht: To isolate the effect of background information, we introduce two additional dataset variants: Core Blk and Core Wht. In these variants, we took the segmented bird images, the same foreground as that of OV, and filled in the background with solid black and white backgrounds, respectively.

The model used in this experiment is sourced from OpenCLIP [7] and employs ViT-L/14 [12], pretrained on the LAION-2B dataset [42]. The text prompts follow CLIP's standard guidelines [38]: `["A photo of a landbird.", "A photo of a waterbird."]`.

8. Experiment details for section 4.1

In this section, we provide details on the experiments conducted for Section 4.1. We evaluate the approaches on the original Waterbirds test set. The model and text prompts are identical to those used in Section 3, with further details available in Section 7.

We introduce the methods utilized in our study, for those requiring auxiliary models:

- Grounding DINO [27]: Grounding DINO (GDINO) [27]: GDINO is a zero-shot object detection model. Given the text prompt "a bird," it returns the bounding box of the bird object in the Waterbirds test set. We crop the test image based on the bounding box and resize it to 224 $\times$ 224. The cropped image is then processed using the CLIP model for zero-shot classification.
- GDINO + SAM [21]: SAM is a prompt-based segmentation model that accepts various types of prompts, such as points or bounding boxes. To achieve precise segmentation, we use the bounding box generated by GDINO as input to SAM, refining the segmentation of bird objects. We set the unmasked regions to zero and pass the segmented bird object to the CLIP image encoder for zero-shot classification.

The information for pure CLIP model methods is listed as follows:

- Dimension reduction (PCA, TSNE, UMAP): We extract the image's latent representation in the form of $\mathbf{z}_{imag} \in \mathbb{R}^{N\times d}$, where N denotes the number of patches. In ViT/L-14, $N = 256$. To reduce the dimensionality from $d = 768$ to 3, we apply dimensionality reduction techniques, including principal component analysis (PCA), t-distributed stochastic neighbor embedding (t-SNE), and uniform manifold approximation and projection (UMAP). Each patch is assigned a cluster label based on $c = \arg\max d_{reduced}$ forming three distinct clusters. We then generate segmentation masks corresponding to these clusters and apply each mask to the original image, setting the unmasked regions to zero. This process results in three separate masked images. Next, we extract the latent representations of these masked images using the `[CLS]` token and perform zero-shot classification with the text prompt ``` ``A photo of a bird'' ``` to identify the mask corresponding to the core object. Once the core segment is determined, we conduct zero-shot classification on the segmented core object to obtain the final prediction.
- K-Means Clustering: We generate masks directly from the high-dimensional image representations. Specifically, we apply K-Means clustering to the image's latent representation $\mathbf{z}_{imag} \in \mathbb{R}^{N\times d}$, partitioning the representation into three clusters, resulting in a shape of $N \times 3$. Each cluster is treated as a segmentation mask. We then follow the same procedure described in the Dimensionality Reduction approach to identify the core object. Once the core segment is determined, we perform zero-shot classification on the segmented core object to obtain the final prediction.
- Dimension reduction + K-Means Clustering: Sohoni et al. [45] empirically demonstrate that clustering in high-

Figure 9. Sample images from our Waterbirds dataset variants, presented in the order: OV, Core Blk, and Core Wht.

dimensional spaces often performs poorly compared to clustering in lower-dimensional representations. Motivated by this observation, we first reduce the dimensionality of the latent space to $d = 3$ before applying K-Means clustering to generate segmentation masks. The remaining procedure follows the method introduced above.

We evaluate our methods based on three key aspects: performance, latency, and GPU usage.

Performance We assess zero-shot classification accuracy, including both worst-group accuracy (WG) and average accuracy (Avg). For methods that generate core feature masks, we compute the DICE score to evaluate how well the predicted mask covers the core object. The DICE score is a standard metric for assessing segmentation performance, defined as:

$$DICE = \frac{2 \times |A \cap B|}{|A| + |B|}, \tag{1}$$

where A represents the predicted segmentation region, and B is the reference segmentation. Since the Waterbirds test set does not provide ground-truth segmentation masks, we use GDINO+SAM's segmentation results as the reference, as its performance closely aligns with the Core Object Only set.

Latency We measure the inference time for each method, running all experiments on a server equipped with four RTX A5000 GPUs, an AMD EPYC 7313 16-Core Processor, and 256 GB of memory. The inference time is reported in minutes.

GPU usage Each method is executed on a single GPU with a batch size of 1. We report the peak GPU memory usage during inference.

9. Experiment details

In this section, we present the detailed experimental setup for our main experiment in Section 5.1. We use y to represent the target label and a to denote the spurious feature. The datasets used in our study include:

Dataset introduction

- Waterbirds: The Waterbirds dataset is a synthetic benchmark designed to study spurious correlations in image classification. It consists of images with two bird categories: waterbirds and landbirds, each placed against either water or land backgrounds, i.e. $y \in \{\text{Waterbird, Landbirds}\}$, $a \in \{\text{Water background, Land background}\}$. There is a spurious correlation between bird type and background, as waterbirds are predominantly shown with water backgrounds and landbirds with land backgrounds.
- CelebA: CelebA is a large-scale face attribution dataset. In this study, we utilize the hair color annotations, $y = \{\text{blond, non-blond}\}$, and use gender $a = \{\text{female, male}\}$ as a spurious feature.
- ISIC: ISIC is a large-scale resource for skin cancer diagnostics. Following the study by Wu et al. [54], we classify skin lesions with target labels $y = \{\text{benign, malignant}\}$. The spurious feature is the presence of color patches $a = \{\text{colored patches present, no colored patches}\}$.
- CXR: CXR is a collection of chest radiographs used for diagnosing various lung diseases. In this study, we follow the setting from [1] to classify images into two categories: y ={non-pneumothorax and pneumothorax}. We consider the presence of support devices, such as catheters or tubes, as spurious features that could bias the model's predictions.
- Metashift: Metashift is a dataset for classifying y ={cat, dog}. Following the study by Wu et al. [54], we consider the background as the spurious feature $a = \{indoors\}$, Since all test samples are indoors, this feature is negatively correlated with dogs, as dog images are more frequently found in outdoor settings.
- UrbanCars: The task in UrbanCars is to classify y={Urban Cars, Country Cars}. The dataset includes two additional components: background and co-occurring objects. Following Yang et al. [57], we define the spurious feature as the background a ={Urban background, Country background}.

Baselines introduction The baseline models considered in our comparison are categorized into two groups: (1) Ro-

bustifying Vision-Language Models (VLMs), and (2) Test-Time Adaptation methods. We provide detailed descriptions of state-of-the-art models.

- Orth-Cali [8]: Orth-Cali is a debiasing method for vision-language models that focuses on text embeddings. It requires a set of text prompts representing spurious features to identify biased directions within the embedding space. By constructing a projection matrix, Orth-Cal projects the text embeddings onto the null space orthogonal to these spurious directions.
- Perception CLIP [2]: Perception CLIP is a zero-shot classification method that enhances the generalizability of vision-language models like CLIP by providing more context in text prompts. Given an image, the model first infers contextual attributes such as background, orientation, or other relevant features. The model then performs object classification conditioned on the inferred contextual attributes by incorporating descriptions of these attributes into the text prompts.
- ROBOSHOT [1]: ROBOSHOT is a method designed to reduce spurious correlations in VLMs. It utilizes large language models (LLMs) to identify potential spurious correlations by generating text prompts that represent these biases. It involves projecting image embeddings onto a subspace orthogonal to the embeddings of these spurious text prompts, mitigating the influence of unwanted correlations.
- TIE*: TIE* is a method designed to mitigate spurious correlations in zero-shot classification tasks. It operates by adjusting image embeddings to reduce the influence of spurious features. The process involves identifying spurious text embeddings, which are derived from text prompts representing the spurious features. TIE* then translates the latent representations of image embeddings along the negative direction of these spurious text embeddings. This translation aims to reduce the impact of spurious features in the image representation.
- Bias Elimination with Nonlinear Debiasing of Vision Language Models (BEND-VLM) [14]: BEND-VLM is a debiasing technique for text embeddings, consisting of two key steps. The first step follows the Orth-Cal approach, projecting text embeddings onto a subspace orthogonal to the spurious feature to mitigate bias. In the second step, a reference image set is used to identify the most relevant images, and the text embedding is adjusted to maintain equal distance from image embeddings with different spurious labels.
- MeanShift for Test-time Augmentation (MTA) [59]: MTA is a test-sample ensemble method designed to enhance robustness. It generates multiple augmented views of a test sample and computes a robust mean of the augmented representations. The updated embedding is then used for zero-shot testing.
- Test entropy minimization (TENT) [49]: TENT is a test-time adaptation method that enhances model confidence by minimizing the entropy of its predictions. It updates the model's parameters, specifically the channel-wise affine transformations, and estimates normalization statistics to adapt to new test data distributions.
- Test-Time Low-rank adaptation (TTLRA) [17]: TTLRA applies low-rank adaptation (LoRA) to the vision encoder, updating attention weights by maximizing prediction confidence. This is achieved by minimizing a weighted entropy loss across various augmented views of the test sample, enabling the model to adapt during testing.
- Test-Time Label-Shift Adaptation (TTLSA) [46]: TTLSA is a method designed to address label distribution shifts between training and test domains by capturing discrepancies in the joint distribution $p(y, a)$ The approach estimates the test-set priors and reweights the model's predictions using the ratio of these estimated priors to the training-set priors, mitigating the adverse effects of label shift.

We implement all baseline methods using their official source code from GitHub, ensuring that all methods are evaluated with the same backbone models for a fair comparison. For text prompts, we use those provided by the authors if the methods have been tested on the respective dataset. For methods not originally evaluated in the corresponding papers, we apply the same text prompts as used in the zero-shot setting and our approach. For methods requiring spurious text prompts, we follow the authors' guidelines to generate them appropriately.

10. Implementation details and text prompts

To ensure reproducibility, we provide the details of our experimental setup.

Model and Preprocessing Experiments in general datasets are conducted using the OpenCLIP pre-trained model ViT-L-14 [7] with the pre-trained weights "`laion2b_s32b_b82k`". Each image is resized to 224×224 to be compatible with the ViT-L-14 configuration. By default, CLIP returns the representation of the `[CLS]` token. To obtain the latent representation for each patch, we set

```
model.visual.pool_type = None
```

The output image embedding has a shape $[1, 257, 768]$, we discard the CLS token, resulting in a final shape of $[1, 256, 768]$. We apply this image representation to perform semantic identification, and core feature zero-shot classification.

Text embedding configuration For general text embedding, we set: `model.text_pool_type = 'last'` This configuration achieves the best results. For task-specific text embedding, we use the default setting of pool type.

Dimensionality Reduction and Clustering

- PCA: We apply `torch.svd_lowrank` with a rank of 16 and retain 3 principal components to ensure consistency across all datasets.
- K-Means Clustering: Implemented using scikit-learn [36], with initialization set to "auto".

Zero-shot Inference We perform two rounds of zero-shot inference:

- Core Segment Identification: We use the [CLS] token as the pooling method to represent all patches. Segmentation normalization is not applied, as it performs best across all datasets.
- Zero-Shot Classification on Core Segments: We use the [CLS] token as the pooling method. We normalize the core segmentation before classification.

The aforementioned settings are consistent across all datasets. Dataset-specific hyperparameter configurations are detailed in Table 5, and all text prompts are provided in Table 6.

Table 5. Hyperparameter details

Dataset	Queue length	Number of clusters
Waterbirds	28	3
CelebA	12	3
Metashift	24	3
UrbanCars	50	3
ISIC	32	3
CXR	42	3

10.1. Latency analysis

We further analyze the test-time latency associated with different computational steps. Compared to the standard zero-shot classification task, the increase in latency can be attributed to the following components:

- Join queue: Extracting image latent representations from the image encoder and storing them in a queue.
- PCA: Applying PCA to patches based on the queued representations.
- K-means: Segmenting the image using features transformed by PCA.
- Segmentation identification: Performing an additional zero-shot classification to identify the segment containing the core feature.

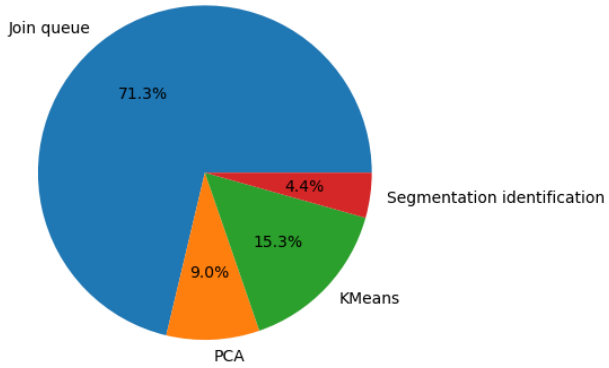

Figure 10. Contribution of each component to the increase in test-time latency.

Figure 10 reports the proportion of time consumed by each procedure. Join Queue accounts for approximately three-quarters of the total increase in latency. Since joining the queue requires an additional feedforward pass during testing, it significantly contributes to the overall latency. We note that the reasoning step (PCA + K-means) only accounts for one-fourth of the latency increase, indicating its efficiency. Although total test-time latency increases, this trade-off provides substantial improvements in performance and robustness.

11. Visualization and numerical results for Section 5.4

In this section, we provide additional details on the ablation study. In Section 5.4.2, we analyze the impact of different mask types on performance. For the Gaussian blur, we apply a Gaussian blur filter with a kernel size of 51 and a sigma of 30. To ensure the feature is sufficiently blurred, we apply this filter three times. A visual illustration of the different masking types is presented in Figure 11.

We also present numerical results corresponding to Figure 8, offering a more precise evaluation. These results are detailed in Tables 7 and 8.

Beyond the results presented in Section 5.2, we further evaluate the effectiveness of `TTR` across different backbone models. We provide additional results on ViT/B-32 and ViT/H-14, as illustrated in Figure 12. Our observations indicate that across various backbone models, `TTR` consistently demonstrates strong performance in identifying core features.

Using a larger-scale backbone model leads to more refined object segmentation. For instance, in the UrbanCars dataset, the ViT/B-32 model struggles to accurately distinguish cars from distracting features, such as stop signs. In

Table 6. Text prompts details

Dataset	Label prompts	General prompts
Waterbirds	[a photo of a landbird., a photo of a waterbird.]	[a photo of a bird.]
CelebA	[a photo of a celebrity with dark hair., a photo of a celebrity with blonde hair.]	[A photo of visible hair.]
Metashift	[a photo of a cat., a photo of a dog.]	[a photo of a pet.]
Urbancars	[a photo of an urban car., a photo of a country car.]	[a photo of a car.]
ISIC	[A photo of benign melanoma., A photo of malignant melanoma.]	[A photo of melanoma.]
CXR	[a photo of no pneumothorax., a photo of a pneumothorax.]	[An X-ray of a lung.]

contrast, the core features are less affected by these concurrent objects with ViT/H-14. We attribute this to the difference in semantic representation capability between smaller and larger models. Additionally, this performance gap can be partially explained by differences in patch size, as smaller models typically use larger patches, leading to reduced granularity in feature extraction.

Table 7. Zero-Shot classification results with ViT/Base-32 model

	ViT/B			TTR (ViT/B)		
	WG↑	Avg↑	Gap↓	WG↑	Avg↑	Gap↓
Waterbirds	43.68	69.85	26.17	61.84	77.88	16.04
CelebA	78.89	84.27	5.38	81.11	85.02	3.91
Metashifts	88.56	89.83	1.27	90.21	91.50	1.29
Urbancars	16.80	51.70	34.90	22.80	56.90	34.10

Table 8. Zero-Shot classification results with ViT/Huge-14 model

	ViT/H			TTR (ViT/H)		
	WG↑	Avg↑	Gap↓	WG↑	Avg↑	Gap↓
Waterbirds	49.69	85.40	35.71	62.46	92.60	30.14
CelebA	82.15	84.74	2.59	83.00	85.00	2.00
Metashifts	81.70	89.09	7.39	89.54	92.24	2.70
Urbancars	6.80	55.90	49.10	58.80	77.60	18.80

Figure 11. Applying different types of masks to regions unrelated to the core feature.

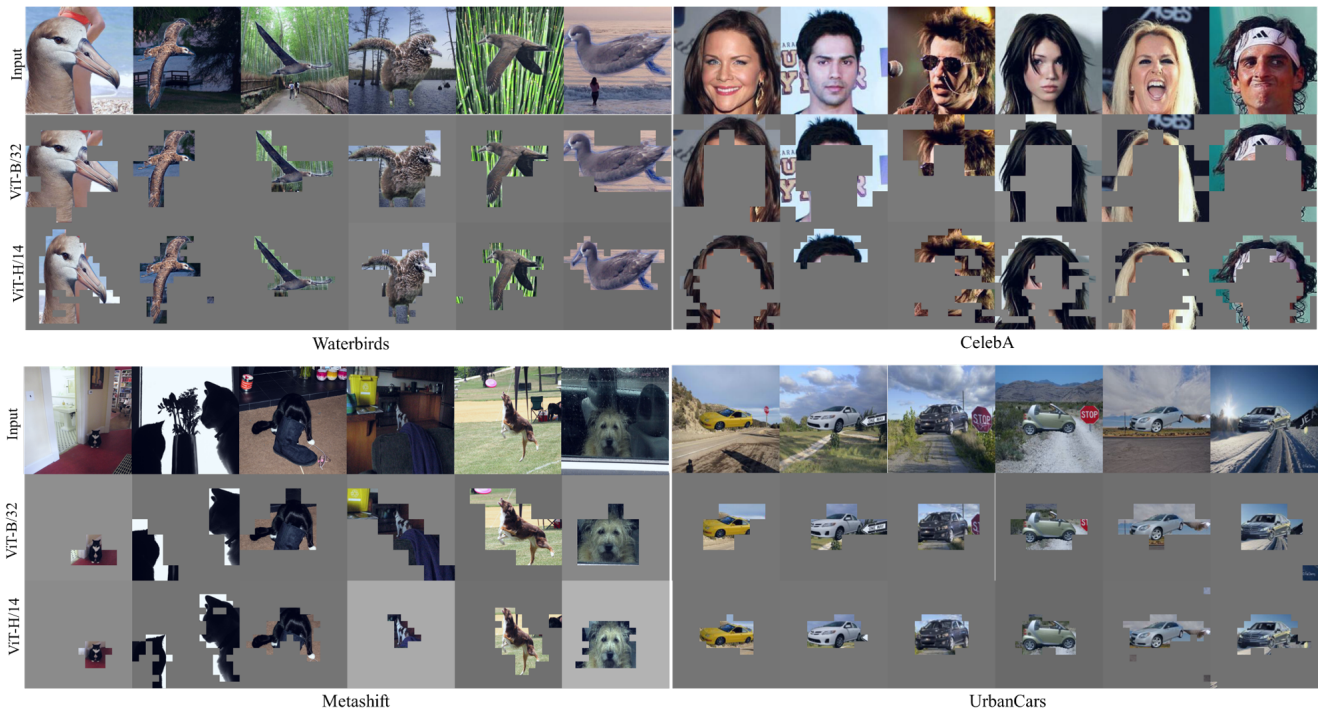

Figure 12. Ablation Study: TTR identified core features using different backbone models.