Mint: A Simple Test-Time Adaptation of Vision-Language Models against Common Corruptions

Wenxuan Bao[1*], **Ruxi Deng**[1*], **Jingrui He**[1]
[1]University of Illinois Urbana-Champaign
{wbao4,ruxid2,jingrui}@illinois.edu

Abstract

Pretrained vision-language models such as CLIP achieve strong zero-shot generalization but remain vulnerable to distribution shifts caused by input corruptions. In this work, we investigate how corruptions affect CLIP's image embeddings and uncover a consistent phenomenon we term as embedding variance collapse, where both intra-class and inter-class variances shrink as corruption severity increases. We find that this collapse is closely tied to performance degradation, with inter-class variance strongly correlated with classification accuracy. To explain this phenomenon, we analyze how corruptions alter the structure of the embedding space. Our theoretical results suggest that the visual encoder tends to encode corruption-related signals, which dilute class-discriminative features and compress the representation geometry. We further show that maximizing inter-class variance, even when estimated from pseudo-labels, can provably enhance embedding quality. Based on this insight, we propose Mint, a simple test-time adaptation method that maximizes pseudo-label-based inter-class variance on the fly using cumulative prototypes and gradient estimates. Mint operates effectively with small batch sizes and consistently improves performance across multiple corruption benchmarks and CLIP architectures. Our code is available at https://github.com/baowenxuan/Mint.

1 Introduction

Pretrained vision-language models (VLMs) such as CLIP [32] have demonstrated strong zero-shot generalization across a wide range of vision tasks [47, 33, 29]. However, their performance can degrade significantly under distribution shifts, such as common image corruptions [16]. Test-time adaptation (TTA) has emerged as a promising strategy for improving model robustness under distribution shifts, by adapting the model during test-time without accessing source data or target labels [24, 40, 38]. This property makes TTA particularly suitable for the adaptation of pretrained VLMs, where the source training data is often large-scale, proprietary, or unavailable at deployment.

Most existing TTA methods for VLMs focus on modifying the text prompt or embedding to improve image-text alignment [35, 12, 1, 31, 25, 36], or leveraging similarities between different image embeddings to adapt the model prediction [45, 19, 42]. While these approaches achieve strong performance on standard benchmark datasets, they often overlook a key issue: the quality of image embeddings themselves can significantly degrade under corruption. Some recent methods [15, 28] attempt to address this by adjusting the image encoder's normalization layers to align image-to-image or text-to-text similarities. However, such techniques typically require large batches to perform effective adaptation, making them unsuitable for many online TTA scenarios where only a few test samples are available at a time. Furthermore, these methods offer limited insight into why common

[*]Equal contribution.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

corruptions cause accuracy degradation, and most lack theoretical analysis, making it difficult to understand when and why they succeed or fail.

In this work, we take a step back and ask: *how exactly does corruption affect CLIP's image embeddings?* To answer this, we evaluate the intra-class and inter-class variances of the embeddings using ground-truth labels, referred to as GT-intra and GT-inter, respectively. This analysis reveals a consistent and intriguing pattern: as corruption severity increases, both GT-intra and GT-inter variances consistently decrease. We term this phenomenon *variance collapse*, which implies that under corruptions, embeddings of images tend to become more similar, regardless of whether they belong to the same class or different classes. The phenomenon is illustrated in Figure 1. Moreover, we observe a strong correlation between inter-class variance and classification accuracy, suggesting that variance collapse is a key factor contributing to performance degradation.

To better understand and counteract variance collapse, we conduct a theoretical analysis of image embeddings' variances under distribution shifts. Our analysis shows that the simultaneous reduction of GT-intra and GT-inter variances can be attributed to the visual encoder projecting corruption-related patterns into the embedding space, which dilutes class-discriminative information. Our theoretical analysis further shows that even in the absence of ground-truth labels during adaptation, maximizing the inter-class variance computed from pseudo-labels (PL-inter) by updating the LayerNorm parameters can provably improve the quality of image embeddings and lead to more accurate classification.

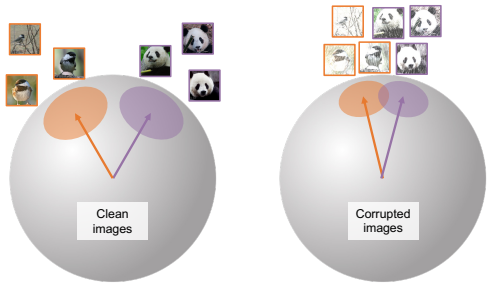

Figure 1: **Variance collapse.** Under corruptions, embeddings of images tend to become similar, regardless of whether they belong to the same class or different classes.

Motivated by this result, we design `Mint`, a simple test-time adaptation method that **M**aximizes the PL-**int**er variance on the fly. `Mint` is designed to operate reliably even when the batch size is extremely small, which is common in online adaptation settings. To enable stable adaptation under such constraints, `Mint` incorporates two key components: a mean accumulator and a gradient accumulator. The mean accumulator maintains cumulative averages of image embeddings for each pseudo-class and for the entire set of samples observed so far. This allows the estimation of PL-inter variance within each batch without requiring access to the full test set. In parallel, the gradient accumulator keeps track of the average update direction across batches, which reduces noise in parameter updates and improves adaptation stability. These two components together allow `Mint` to enhance class-discriminative signals and suppress corruption-related patterns in the image embedding space. We evaluate our method across a wide range of corruption benchmark datasets and CLIP architectures to demonstrate its robustness and generality. In all settings, `Mint` consistently outperforms existing TTA methods for VLMs, while also offering significant efficiency advantages. Our contributions are summarized as follows:

- We identify a phenomenon we refer to as variance collapse in CLIP image embeddings, where both intra-class and inter-class variances decrease as corruption severity increases.
- We provide a theoretical analysis that attributes this collapse to the visual encoder embedding corruption-related patterns, and show that maximizing PL-inter can improve embedding quality.
- We propose `Mint`, a simple TTA method that maximizes PL-inter on the fly using a mean accumulator and a gradient accumulator, enabling effective adaptation even with extremely small batch sizes.
- We demonstrate that `Mint` consistently improves the performance of CLIP models across multiple corruption benchmarks and architectures, outperforming existing TTA methods in both accuracy and efficiency.

2 Related works

Test-time adaptation (TTA) adapts a source model to an unlabeled target domain during testing, without access to source data, making it suitable for pre-trained VLMs like CLIP. Early TTA methods for CLIP focus on modifying the text encoder or embedding, via prompt tuning [35, 12, 34], prompt weighting [1], or ensembling [44, 31]. Memory-based approaches [19, 45, 42, 3] store embeddings of high-confidence samples and use image similarity to guide predictions. Other training-free methods [8, 11] apply augmentations and confidence selection to enhance robustness without updating any model parameters. Although effective, most methods overlook a key issue: the quality of image embeddings degrades under corruptions. Recent approaches [15, 28] attempt to mitigate this by adjusting normalization layers, aligning the image-image and text-text similarities. However, they are very sensitive to batch size and introduce high computational overhead, limiting their use in online TTA settings.

Inter-class separability has been widely explored in supervised learning, where a common goal is to increase the distance between classes while reducing the variance within each class. A classic example is the Fisher score [10], which measures this separation and has been used for feature selection [14] and to improve domain adaptation by applying the Fisher criterion on the labeled source domains [46]. However, computing such metrics typically requires access to ground-truth labels, which are unavailable in test-time adaptation. More recently, Matcha [5] extended similar ideas to graph-based TTA by leveraging soft pseudo-labels. While effective, this method assumes simultaneous access to all nodes in the test graph, making it unsuitable for online adaptation scenarios where only small batches are available.

We provide a broader discussion of related works in Appendix A.1.

3 Analysis

Preliminary CLIP [32] is a VLM consisting of an image encoder and a text encoder, which aligns images with their corresponding textual descriptions. Pretraining on a large-scale image-text dataset enables CLIP to perform zero-shot prediction. Specifically, for a classification task with C classes, the text encoder embeds class descriptions (e.g., "`a photo of a {class}`") into normalized text embeddings $\boldsymbol{T} = [\boldsymbol{t}_1, \cdots, \boldsymbol{t}_C]^\top \in \mathbb{R}^{C\times d}$, where d is the embedding dimension. Given a test image, the image encoder produces a normalize image embedding $\boldsymbol{z}_i \in \mathbb{R}^d$, and prediction is made via the maximum similarity score $\arg\max_y \boldsymbol{z}_i^\top \boldsymbol{t}_y$. However, CLIP's performance degrades noticeably under common image corruptions [15, 28], as its image encoder was not explicitly trained for robustness.

3.1 Variance collapse

In this subsection, we investigate how common corruptions affect the image embeddings extracted by CLIP's visual encoder, and how these changes influence classification accuracy. Motivated by Fisher score [10, 46, 14] and contrastive learning [32, 37] objectives, we posit that high-quality image embeddings should exhibit low intra-class variance (i.e., samples from the same class are close) and high inter-class variance (i.e., samples from different classes are well separated). To formalize this intuition, given a target dataset with C classes and N images, we define the following variances:

- GT-total variance: $\mathcal{V}_{\text{total}}^{\text{GT}} = \frac{1}{C}\sum_{c=1}^{C} \frac{\sum_{i=1}^{N} y_{ic}\|\boldsymbol{z}_i - \bar{\boldsymbol{z}}\|_2^2}{\sum_{i=1}^{N} y_{ic}}$,
- GT-inter variance: $\mathcal{V}_{\text{inter}}^{\text{GT}} = \frac{1}{C}\sum_{c=1}^{C} \|\bar{\boldsymbol{z}}_c - \bar{\boldsymbol{z}}\|_2^2$,
- GT-intra variance: $\mathcal{V}_{\text{intra}}^{\text{GT}} = \frac{1}{C}\sum_{c=1}^{C} \frac{\sum_{i=1}^{N} y_{ic}\|\boldsymbol{z}_i - \bar{\boldsymbol{z}}_c\|_2^2}{\sum_{i=1}^{N} y_{ic}}$,

where $\bar{\boldsymbol{z}} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{z}_i$ is the average embedding for all images, $\bar{\boldsymbol{z}}_c = \frac{\sum_{i=1}^{N} y_{ic}\boldsymbol{z}_i}{\sum_{i=1}^{N} y_{ic}}$ is the average embedding of class c, and $\boldsymbol{y}_i = [y_{i1}, \cdots, y_{iC}]^\top \in \{0,1\}^C$ is the one-hot ground-truth label of image i, where $y_{ic} = 1$ if image i corresponds to class c, and $y_{ic} = 0$ otherwise. Note that these variances are computed using the ground-truth labels. To distinguish them from the pseudo-label-based counterparts introduced later, we denote these metrics with a GT- prefix (e.g., GT-intra, GT-inter).

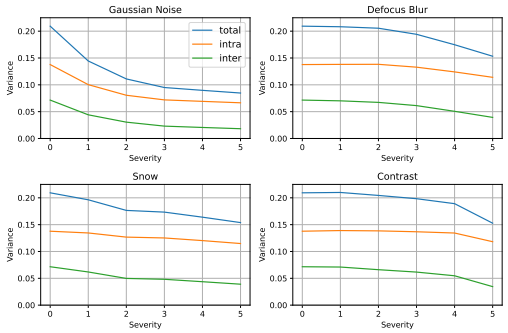

Figure 2: All types of variances decrease as the severity of corruption increases (severity=0 indicates original CIFAR-100 datasets without corruptions).

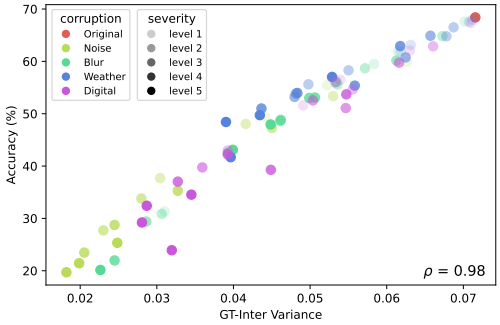

Figure 3: GT-inter variance is highly correlated with accuracy, with correlation 0.98. In comparison, the correlations between accuracy and GT-intra, GT-total are 0.86 and 0.94, respectively.

We compute the GT-total, GT-inter, and GT-intra variances on the corruption benchmark [16], which includes 15 types of common corruptions, each evaluated at 5 severity levels. Figure 2 presents the results on four representative corruptions. Additional results across all corruption types and datasets are included in Appendix C.1. We observe that for all types of corruptions, both GT-intra and GT-inter variances consistently decrease as the severity increases. This indicates that the pairwise similarity between image embeddings increases under corruptions, regardless of whether the images belong to the same class or not. We refer to this phenomenon as **variance collapse**, where the image embeddings become increasingly indistinguishable under stronger corruptions. Furthermore, we find that variance collapse is closely linked to the drop in accuracy. We computed all three variance metrics along with the corresponding classification accuracy under 76 corruption settings (15 corruption types $\times$ 5 severity levels, plus the clean setting) on CIFAR-100-C. As shown in Figure 3, the inter-class variance exhibits an extremely strong correlation with accuracy, indicating that this collapse could be a key factor driving the performance decline.

3.2 Theoretical explanation

In this subsection, we provide a theoretical explanation for the emergence of variance collapse. For clarity, we consider a balanced binary classification problem with $y_i \in \{0, 1\}$. Motivated by [39], we assume that each image can be mapped to a disentangled latent representation $\boldsymbol{v}_i = [\boldsymbol{v}_i^{\text{cls}}, \boldsymbol{v}_i^{\text{irr}}, \boldsymbol{v}_i^{\text{shift}}, \boldsymbol{v}_i^{\text{noise}}] \in \mathbb{R}^d$, composed of four components:

1. *Task-relevant features* $\boldsymbol{v}_i^{\text{cls}}$: Semantic features that are directly predictive of the class label, $\boldsymbol{v}_i^{\text{cls}} = \boldsymbol{\mu}$ if $y_i = 1$ and $\boldsymbol{v}_i^{\text{cls}} = -\boldsymbol{\mu}$ if $y_i = 0$.
2. *Task-irrelevant features* $\boldsymbol{v}_i^{\text{irr}}$: Features unrelated to classification, such as background. It is preserved during pretraining due to CLIP's general representation learning objective. We assume $\boldsymbol{v}_i^{\text{irr}} \sim \text{Rademacher}^{d_{\text{irr}}}$, i.e., uniformly distributed in $\{-1, 1\}^{d_{\text{irr}}}$.
3. *Structured distribution shift* $\boldsymbol{v}_i^{\text{shift}}$: Features representing systematic distribution changes in the target domain, such as weather conditions or digital transforms. We assume $\boldsymbol{v}_i^{\text{shift}} = s \cdot \boldsymbol{\delta}$, where s indicates the severity of corruptions or distribution shifts.
4. *Unstructured noise* $\boldsymbol{v}_i^{\text{noise}}$: Random noise introduced by the corruption process. We assume $\boldsymbol{v}_i^{\text{noise}} \sim s \cdot \text{Rademacher}^{d_{\text{noise}}}$, i.e., uniformly distributed in $\{-1, 1\}^{d_{\text{noise}}}$.

Notice that by controlling the ratio of s, $\|\boldsymbol{\mu}\|_2$, $\|\boldsymbol{\delta}\|_2$, we can freely adjust the ratio for each component. Following the structure of CLIP's visual encoder, we assume that the latent representation $\boldsymbol{v}$ first passes through a LayerNorm layer [2] with a linear transformation, followed by normalization to unit length. For analytical simplicity, we omit the demeaning step in LayerNorm and ignore the bias term in its parameters,[1] under which the image embedding can be formulated as

$$\boldsymbol{z}_i = \text{normalize}\left(\frac{\boldsymbol{v}_i}{\sqrt{\text{Var}[\boldsymbol{v}_i]}} \odot \boldsymbol{w}\right), \tag{1}$$

[1]This simplification is also known as RMSNorm [41].

where $\odot$ represents element-wise multiplication of vectors, $\boldsymbol{w} \in \mathbb{R}^d$ is the learnable weight of the LayerNorm layer, $\mathrm{normalize}(\cdot)$ denotes ℓ_2 normalization. For simplicity, we assume $\boldsymbol{w} = \mathbf{1}$ at initialization. $\boldsymbol{w}$ is updated during the adaptation.

Theorem 3.1 (Variance collapse)**.** *When the sample size $N \to +\infty$,*

$$\mathcal{V}_{inter}^{GT} \xrightarrow{p} \frac{\|\boldsymbol{\mu}\|_2^2}{\|\boldsymbol{\mu}\|_2^2 + d_{irr} + s^2 \cdot \|\boldsymbol{\delta}\|_2^2 + s^2 \cdot d_{noise}}, \ \mathcal{V}_{intra}^{GT} \xrightarrow{p} \frac{d_{irr} + s^2 \cdot d_{noise}}{\|\boldsymbol{\mu}\|_2^2 + d_{irr} + s^2 \cdot \|\boldsymbol{\delta}\|_2^2 + s^2 \cdot d_{noise}}, \quad (2)$$

where s denotes the corruption severity. As s increases, $\mathcal{V}_{inter}^{GT}$ strictly decreases. In addition, $\mathcal{V}_{intra}^{GT}$ also decreases when $\|\boldsymbol{\delta}\|_2 \geq \sqrt{d_{noise}/d_{irr}} \cdot \|\boldsymbol{\mu}\|_2$.

Theorem 3.1 characterizes how the GT-inter and GT-intra variances change with increasing corruption severity. Combined with the empirical trends observed in Figure 2, this suggests that common corruptions often induce significant structured distribution shifts, reflected as large $\boldsymbol{\delta}$ in the latent space. As a result, the image encoder tends to embed corruption-related patterns into the representation itself. This dilutes class-discriminative features and introduces bias into the resulting image embeddings.

3.3 Maximization of inter variance

Theorem 3.1 also supports that GT-inter variance has strong relevance to classification accuracy, as it reflects the proportion of task-relevant features within the overall feature representation. This insight motivates the idea that maximizing GT-inter variance should lead to improved classification accuracy under distribution shifts. However, several challenges arise in the context of TTA. First, the ground-truth labels are unavailable, so we must rely on pseudo-labels, i.e., the model's own prediction, which are noisy due to distribution shifts. Second, model updates in TTA are typically restricted to a small subset of parameters, such as LayerNorm weights, for better efficiency. In this part, we show that even under these constraints, using only pseudo-labels and updating only LayerNorm parameters, maximizing inter-class variance remains an effective and theoretically justified strategy for improving robustness to distribution shifts.

Theorem 3.2 (Maximization of PL-inter variance)**.** *When the sample size $N \to \infty$,*

$$\mathcal{V}_{inter}^{PL} \xrightarrow{p} \frac{C(\mathbb{E}\hat{y}_i)}{2} \cdot \frac{4\sigma_{\hat{y}y}^2 \cdot \|\boldsymbol{\mu} \odot \boldsymbol{w}^{cls}\|_2^2 + \|\boldsymbol{\sigma}_{irr} \odot \boldsymbol{w}^{irr}\|_2^2 + \|\boldsymbol{\sigma}_{noise} \odot \boldsymbol{w}^{noise}\|_2^2}{\|\boldsymbol{\mu} \odot \boldsymbol{w}^{cls}\|_2^2 + \|\boldsymbol{w}^{irr}\|_2^2 + s^2 \cdot \|\boldsymbol{\delta} \odot \boldsymbol{w}^{shift}\|_2^2 + s^2 \cdot \|\boldsymbol{w}^{noise}\|_2^2}, \quad (3)$$

where $C(\mathbb{E}\hat{y}_i) = \frac{1}{(\mathbb{E}\hat{y}_i)^2} + \frac{1}{(1-\mathbb{E}\hat{y}_i)^2}$, $\sigma_{\hat{y}y} = \mathrm{Cov}(y_i, \hat{y}_i)$, $\boldsymbol{\sigma}_{irr} = \mathrm{Cov}(\boldsymbol{v}^{irr}, \hat{y}_i)$, and $\boldsymbol{\sigma}_{noise} = \mathrm{Cov}(\boldsymbol{v}^{noise}, \hat{y}_i)$. Furthermore, when $\sigma_{\hat{y}y}^2 \geq \frac{\|\boldsymbol{\sigma}_{irr}\|_2^2}{4d_{irr}}$ and $\sigma_{\hat{y}y}^2 \geq \frac{\|\boldsymbol{\sigma}_{noise}\|_2^2}{4d_{noise}}$, we have

$$\nabla_{\boldsymbol{w}^{cls}} \mathcal{V}_{inter}^{PL} = C(\mathbb{E}\hat{y}_i) \cdot \frac{(4\sigma_{\hat{y}y}^2 d_{irr} - \|\boldsymbol{\sigma}_{irr}\|_2^2) + 4\sigma_{\hat{y}y}^2 s^2 \|\boldsymbol{\delta}\|_2^2 + (4\sigma_{\hat{y}y}^2 d_{noise} - \|\boldsymbol{\sigma}_{noise}\|_2^2)}{(\|\boldsymbol{\mu}\|_2^2 + d_{irr} + s^2 \cdot (\|\boldsymbol{\delta}\|_2^2 + d_{noise}))^2} \cdot \boldsymbol{\mu}^2 \geq \mathbf{0}, \quad (4)$$

$$\nabla_{\boldsymbol{w}^{shift}} \mathcal{V}_{inter}^{PL} = -C(\mathbb{E}\hat{y}_i) \cdot \frac{\mathcal{V}_{inter}^{PL}}{\|\boldsymbol{\mu}\|_2^2 + d_{irr} + s^2 \cdot (\|\boldsymbol{\delta}\|_2^2 + d_{noise})} \cdot s^2 \cdot \boldsymbol{\delta}^2 \leq \mathbf{0}. \quad (5)$$

This implies that when we perform a single gradient ascent step to maximize the PL-inter variance by updating the parameters of LayerNorm, the parameters associated with structured distribution shifts (i.e., w_{shift}) are necessarily suppressed. Furthermore, as long as the current prediction is reasonably accurate, meaning it depends more on task-relevant features than on task-irrelevant components or unstructured noise, maximizing PL-inter variance will increase the weights associated with task-relevant features (i.e., w_{cls}). As a result, this process reweighs the components in the final image embedding, enhancing the influence of task-relevant features while suppressing the effects of distribution shifts.

4 Proposed method: `Mint`

In this section, we introduce our proposed algorithm `Mint`, which maximizes the PL-inter variance on the fly. While the previous section provides theoretical justification that maximizing PL-inter variance can improve test-time robustness, directly computing PL-inter variance requires access to the entire

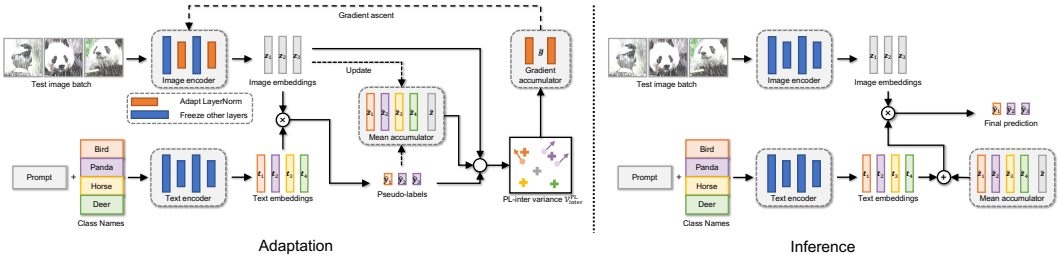

Figure 4: Overview of `Mint`. **Left: Adaptation phase.** Given a test image batch, we compute the PL-inter variance with the help of a mean accumulator and maximize it via gradient ascent. A gradient accumulator aggregates update directions across batches to robustly update the LayerNorm parameters. **Right: Inference phase.** The mean accumulator is used to adjust the text embeddings, and final predictions are made based on the similarity between image and text embeddings.

test dataset. However, in the online TTA setting, the model typically adapts using only a small batch or even a single sample at a time. This leads to noisy and potentially biased gradient directions that deviate from the true optimization target. To address this, we reparameterize the PL-inter variance and employ both a mean accumulator and a gradient accumulator to aggregate information across batches, enabling a more accurate approximation of the gradient in a streaming setting. Figure 4 gives an overview of our method.

4.1 Mean accumulator

The most straightforward way to estimate PL-inter variance is to assume that the current batch fully represents the test data distribution and compute PL-inter using only samples in that batch. Under this approach, objectives across batches are computed independently. However, this naive strategy introduces significant noise, and even bias, into the PL-inter estimate. For instance, ImageNet-C contains 1,000 classes, but due to deployment and memory constraints, the test-time batch size is typically limited to just a few dozen samples. As a result, most classes present in a batch are represented by only a single sample, causing their estimated class means to degenerate into the samples themselves. In such cases, the distance between a sample and its class mean becomes zero, preventing us from estimating PL-intra variance. As a consequence, the computed objective ends up approximating PL-total variance rather than true PL-inter variance, which degrades adaptation performance under small batch sizes.

To address this issue, we first reparameterize PL-inter variance as the difference between PL-total and PL-intra variance. With detailed proof in Appendix B.1, this decomposition can be written as:

$$\underbrace{\frac{1}{C}\sum_{c=1}^{C}\|\tilde{\boldsymbol{z}}_c-\tilde{\boldsymbol{z}}\|_2^2}_{\mathcal{V}_{\text{inter}}^{\text{PL}}}=\underbrace{\frac{1}{C}\sum_{c=1}^{C}\frac{\sum_{i=1}^{N}\hat{y}_{ic}\|\boldsymbol{z}_i-\tilde{\boldsymbol{z}}\|_2^2}{\sum_{i=1}^{N}\hat{y}_{ic}}}_{\mathcal{V}_{\text{total}}^{\text{PL}}}-\underbrace{\frac{1}{C}\sum_{c=1}^{C}\frac{\sum_{i=1}^{N}\hat{y}_{ic}\|\boldsymbol{z}_i-\tilde{\boldsymbol{z}}_c\|_2^2}{\sum_{i=1}^{N}\hat{y}_{ic}}}_{\mathcal{V}_{\text{intra}}^{\text{PL}}}, \tag{6}$$

where $\boldsymbol{z}_i$ is the embedding for i-th image, $\tilde{\boldsymbol{z}}=\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{z}_i$ is the global average embedding, and $\tilde{\boldsymbol{z}}_c=\frac{\sum_{i=1}^{N}\hat{y}_{ic}\boldsymbol{z}_i}{\sum_{i=1}^{N}\hat{y}_{ic}}$ is the average embedding of all images predicted as class c by CLIP. This reformulation reveals that maximizing PL-inter is equivalent to jointly maximizing PL-total variance and minimizing PL-intra variance, encouraging each embedding $\boldsymbol{z}_i$ to move away from the global mean $\tilde{\boldsymbol{z}}$ and toward its corresponding class mean $\tilde{\boldsymbol{z}}_c$, with the gradient direction approximately given by $\tilde{\boldsymbol{z}}_c-\tilde{\boldsymbol{z}}$ when the sample size is sufficiently large. This insight suggests that more accurate estimates of $\tilde{\boldsymbol{z}}$ and $\tilde{\boldsymbol{z}}_c$ can lead to better gradient directions. Therefore, instead of estimating these means using only the current batch, we use a mean accumulator to maintain cumulative averages $\tilde{\boldsymbol{z}}$ and $\{\tilde{\boldsymbol{z}}_c\}_{c=1}^{C}$. Every time when we observe a new image with embedding $\boldsymbol{z}_i$ and CLIP's prediction $\hat{y}_i$ as pseudo-label,

$$\tilde{\boldsymbol{z}}\leftarrow\frac{K}{K+1}\tilde{\boldsymbol{z}}+\frac{1}{K+1}\boldsymbol{z}_i,\quad \tilde{\boldsymbol{z}}_{\hat{y}_i}\leftarrow\frac{K_{\hat{y}_i}}{K_{\hat{y}_i}+1}\tilde{\boldsymbol{z}}_{\hat{y}_i}+\frac{1}{K_{\hat{y}_i}+1}\boldsymbol{z}_i, \tag{7}$$

where K is the total number of seen samples, and $K_{\hat{y}_i}$ is the number of seen samples with pseudo-label $\hat{y}_i$. After replacing the class and global means in Equation (6) with the cumulative averages, the

final objective for the b-th batch $\mathbb{B}_b$ becomes

$$\mathcal{V}_{\text{inter}}^{\text{PL}}(\mathbb{B}_b) = \frac{1}{C_b}\sum_{c=1}^{C_b}\frac{\sum_{i\in\mathbb{B}_b}\hat{y}_{ic}\,\|\boldsymbol{z}_i-\tilde{\boldsymbol{z}}\|_2^2}{\sum_{i\in\mathbb{B}_b}\hat{y}_{ic}} - \frac{1}{C_b}\sum_{c=1}^{C_b}\frac{\sum_{i\in\mathbb{B}_b}\hat{y}_{ic}\,\|\boldsymbol{z}_i-\tilde{\boldsymbol{z}}_c\|_2^2}{\sum_{i\in\mathbb{B}_b}\hat{y}_{ic}}, \tag{8}$$

where C_b is the number of unique classes present in batch $\mathbb{B}_b$.

4.2 Gradient accumulator

While the mean accumulator mitigates systematic bias in the objective by stabilizing the estimates of class and global means, it does not eliminate the noise in the individual gradient contributions from $\boldsymbol{z}_i$, which are still computed over the current batch. To further reduce gradient estimation error, we introduce a simple gradient accumulator that mimics adaptation with a larger effective batch size. Specifically, for the b-th batch, if the gradient computed on the current batch is $\boldsymbol{g}_b$, we maintain a cumulative average of gradients $\bar{\boldsymbol{g}}$ over the seen b batches:

$$\bar{\boldsymbol{g}} \leftarrow \frac{b-1}{b}\bar{\boldsymbol{g}} + \frac{1}{b}\boldsymbol{g}_b, \tag{9}$$

and update the LayerNorm parameters in the direction of $\bar{\boldsymbol{g}}$. We perform only a single step of update on each batch.

4.3 Adjust text embedding

In addition to estimating PL-inter variance, the cumulative class means can also be leveraged to adjust the text embeddings, thereby improving alignment between the image and text modalities. Motivated by prior works in TTA [18] and Bayesian estimation, we adopt a simple, training-free approach to refine the text embeddings using accumulative embedding means. Specifically, we maintain a separate mean accumulator to store the image embeddings produced by the adapted image encoder. The refinement of text embeddings is given by

$$\tilde{\boldsymbol{t}}_c \leftarrow \text{normalize}\left(\frac{K_{\text{prior}}}{K_{\text{prior}}+K}\cdot\boldsymbol{t}_c + \frac{K}{K_{\text{prior}}+K}\cdot\tilde{\boldsymbol{z}}_c\right), \quad c=1,\cdots,C, \tag{10}$$

where K_{prior} is a hyperparameter controlling the strength of prior. This design enables dynamic adjustment of the text embedding. In the early stage of adaptation, the image embedding means may be less reliable, so we assign more weight to the original text embedding $\boldsymbol{t}_c$. As adaptation progresses and the quality of the estimated class-wise means $\tilde{\boldsymbol{z}}_c$ improves, we gradually place more weight on $\tilde{\boldsymbol{z}}_c$.

The final prediction is given by $\arg\max_y \boldsymbol{z}_i^\top\tilde{\boldsymbol{t}}_y$. After making prediction on each batch, we reset both the image encoder and the optimizer state to their initial values. However, the mean accumulator and gradient accumulator are preserved and carried over to the next batch, allowing information aggregated from previous samples to guide the adaptation on subsequent inputs.

5 Experiments

In this section, we use experiments to answer the following research questions

- **RQ1**: Can `Mint` effectively improve the performance of CLIP models under common corruptions, especially in low batch size scenarios?
- **RQ2**: Does `Mint` effectively mitigate the variance collapse?
- **RQ3**: How efficient is `Mint` in terms of computational time?

Setup and baselines We test `Mint` with different combination of model architectures and corruption datasets [16]: ViT-B/32 [9] on CIFAR-10-C [21], ViT-B/16 on CIFAR-100-C, and ViT-L/14 on ImageNet-C [7], all with corruption severity of 5. We consider a standard TTA setting, where the model is adapted to each type of corruption independently. We compare `Mint` with a wide range of existing TTA methods designed for VLMs. VTE [8] and Zero [11] aggregate image embeddings from multiple augmentations. TPT [35] and TPS [36] minimize the marginal entropy to encourage

Table 1: Mean accuracy (%) on corruption benchmarks. Error bars are deferred to Appendix C.3.

Method	Venue	ViT-B/32 on CIFAR-10-C															
		Noise			Blur				Weather				Digital				Avg.
		Gauss.	Shot	Impul.	Defoc.	Glass	Motion	Zoom	Snow	Frost	Fog	Brit.	Contr.	Elastic	Pixel	JPEG	
CLIP [32]	ICML'21	35.5	40.0	43.2	70.0	41.4	64.5	70.2	70.8	72.3	66.7	81.4	64.5	59.6	48.2	56.7	59.0
Ensemble	-	38.8	42.7	42.8	72.6	43.9	66.8	71.7	73.9	75.8	68.9	83.7	67.2	61.9	51.8	58.6	61.4
TPT [35]	NeurIPS'22	42.9	46.2	47.1	71.5	46.4	68.1	72.7	73.7	75.9	68.9	83.7	73.9	62.5	50.3	58.2	62.8
TDA [19]	CVPR'24	41.2	44.1	43.3	73.9	45.1	68.1	73.6	74.0	76.7	69.6	84.0	66.6	62.3	54.7	58.4	62.4
DMN-ZS [45]	CVPR'24	37.6	41.5	42.5	69.4	43.8	65.9	70.5	70.2	71.2	64.0	80.7	58.6	59.4	54.9	58.1	59.2
VTE [8]	ECCV-W'24	47.6	50.5	49.8	70.4	49.8	70.2	73.4	74.4	77.3	71.4	83.6	**81.2**	65.5	55.3	58.8	65.3
Zero [11]	NeurIPS'24	47.9	50.5	50.0	70.3	50.3	69.7	73.6	74.5	77.1	71.5	83.5	80.6	66.0	55.2	58.9	65.3
WATT-S [28]	NeurIPS'24	53.2	54.9	50.7	75.0	55.4	71.1	74.8	75.4	77.0	72.7	84.2	73.1	65.4	61.1	62.3	67.1
TPS [36]	WACV'25	45.5	49.4	49.2	73.8	50.7	71.4	76.0	77.0	**79.2**	73.3	85.3	79.5	67.2	56.6	61.8	66.4
CLIPArTT [15]	WACV'25	45.2	48.7	47.1	73.4	49.9	69.0	73.0	74.1	76.2	70.1	84.3	71.4	64.1	58.5	60.5	64.4
`Mint`	-	**59.0**	**62.4**	**54.2**	**75.8**	**61.8**	**77.1**	**78.9**	**79.0**	78.9	**75.2**	**86.3**	76.9	**70.1**	**66.6**	**63.4**	**71.0**

Method	Venue	ViT-B/16 on CIFAR-100-C															
		Noise			Blur				Weather				Digital				Avg.
		Gauss.	Shot	Impul.	Defoc.	Glass	Motion	Zoom	Snow	Frost	Fog	Brit.	Contr.	Elastic	Pixel	JPEG	
CLIP [32]	ICML'21	19.7	21.4	25.3	42.5	20.2	43.1	48.0	48.4	49.7	41.7	57.0	34.5	29.2	23.9	32.4	35.8
Ensemble	-	22.9	24.3	29.6	43.6	20.1	43.7	48.7	48.9	50.3	41.8	58.1	35.2	29.2	26.3	33.6	37.1
TPT [35]	NeurIPS'22	17.3	19.2	25.6	42.4	20.0	42.2	47.9	49.0	50.0	42.7	57.5	38.0	30.3	25.5	32.5	36.0
TDA [19]	CVPR'24	23.8	26.0	32.5	45.7	21.5	44.4	50.5	49.6	51.5	42.8	59.2	36.8	29.7	28.1	34.3	38.4
DMN-ZS [45]	CVPR'24	23.9	25.6	31.7	45.5	21.6	45.0	51.1	49.6	52.0	43.0	60.3	36.0	30.5	27.5	34.7	38.5
VTE [8]	ECCV-W'24	20.2	21.2	28.4	39.9	18.5	39.0	44.7	47.6	48.8	43.2	55.7	49.9	30.4	30.3	30.6	36.6
Zero [11]	NeurIPS'24	19.9	21.5	29.6	40.4	18.5	39.6	44.8	47.8	48.3	43.3	55.8	**50.0**	30.6	30.4	30.7	36.8
WATT-S [28]	NeurIPS'24	27.5	29.8	36.4	47.5	26.8	46.8	51.6	51.6	52.3	46.6	61.0	43.5	34.3	**35.9**	37.3	41.9
TPS [36]	WACV'25	22.6	24.4	31.0	44.0	20.1	43.6	49.0	50.5	51.3	44.3	59.1	45.1	30.6	28.8	33.8	38.6
CLIPArTT [15]	WACV'25	24.9	27.1	32.5	47.4	23.4	47.2	52.0	51.6	**52.5**	46.5	61.2	41.2	33.7	32.6	37.0	40.7
`Mint`	-	**29.4**	**30.8**	**38.6**	**50.7**	**27.1**	**49.9**	**55.5**	**53.0**	51.8	**50.6**	**65.6**	48.1	**36.8**	34.4	**38.7**	**44.1**

Method	Venue	ViT-L/14 on ImageNet-C															
		Noise			Blur				Weather				Digital				Avg.
		Gauss.	Shot	Impul.	Defoc.	Glass	Motion	Zoom	Snow	Frost	Fog	Brit.	Contr.	Elastic	Pixel	JPEG	
CLIP [32]	ICML'21	27.4	29.4	28.7	34.6	25.3	41.0	36.7	49.8	44.1	49.7	65.4	35.1	30.3	53.5	42.2	39.6
Ensemble	-	29.1	30.4	30.1	37.5	27.2	44.2	39.2	52.4	46.4	52.7	67.8	34.5	32.4	56.2	44.3	41.6
TPT [35]	NeurIPS'22	27.2	29.1	29.3	35.7	26.6	41.1	38.1	51.4	46.3	51.6	67.7	39.4	32.1	55.9	45.5	41.1
TDA [19]	CVPR'24	29.1	30.5	31.0	37.7	28.0	44.5	39.5	53.4	47.8	53.6	68.3	36.8	33.3	56.7	44.4	42.3
DMN-ZS [45]	CVPR'24	29.0	30.4	30.4	37.5	27.3	44.3	39.3	52.5	46.7	52.7	67.8	34.9	32.4	56.2	44.3	41.7
VTE [8]	ECCV-W'24	23.2	26.5	24.9	34.5	25.8	39.7	38.2	49.0	45.7	49.8	67.0	44.4	32.1	55.8	46.5	40.2
Zero [11]	NeurIPS'24	24.1	26.9	25.8	35.8	26.9	40.3	39.4	49.5	46.2	50.7	66.8	44.9	32.6	56.4	47.4	40.9
WATT-S [28]	NeurIPS'24	31.7	33.5	34.6	38.7	31.3	45.2	41.2	52.7	47.8	54.5	67.5	42.9	34.8	56.3	45.9	43.9
TPS [36]	WACV'25	28.9	31.0	30.7	37.8	28.0	43.4	40.8	53.3	**47.9**	53.5	**69.2**	43.8	33.3	57.3	47.0	43.1
CLIPArTT [15]	WACV'25	29.2	31.0	30.8	34.5	28.1	41.9	38.0	49.9	44.7	50.1	64.5	39.2	32.4	53.0	42.4	40.7
`Mint`	-	**33.0**	**34.3**	**37.3**	**39.6**	**37.2**	**46.6**	**45.1**	**55.2**	46.6	**57.5**	67.7	**48.9**	**43.9**	**58.2**	**54.6**	**47.0**

Table 2: Accuracy (mean $\pm$ s.d. %) of `Mint` with various batch size.

Architecture	Dataset	CLIP	`Mint`							
			BS $= 1$	BS $= 2$	BS $= 5$	BS $= 10$	BS $= 20$	BS $= 50$	BS $= 100$	BS $= 200$
ViT-B/32	CIFAR-10-C	59.0	70.5 ± 0.1	70.5 ± 0.1	71.0 ± 0.0	71.0 ± 0.1	71.0 ± 0.1	71.0 ± 0.1	70.9 ± 0.1	70.6 ± 0.1
ViT-B/16	CIFAR-100-C	35.8	43.1 ± 0.1	43.1 ± 0.1	43.3 ± 0.1	43.6 ± 0.1	44.1 ± 0.1	44.5 ± 0.1	44.5 ± 0.1	44.6 ± 0.1
ViT-L/14	ImageNet-C	39.6	45.8 ± 0.1	46.2 ± 0.1	46.7 ± 0.1	46.8 ± 0.1	47.0 ± 0.2	47.1 ± 0.1	47.0 ± 0.2	46.8 ± 0.1

consistency across augmented views. TDA [19] and DMN-ZS [45] leverage sample-wise similarity to adjust predictions. WATT-S [28] and CLIPArTT [15] improve modality alignment by aligning image-to-image and text-to-text similarities. Unless otherwise specified, we use a default batch size of 20 during adaptation. `Mint` uses Adam [20] optimizer with learning rate 0.007 for ViT-B models and 0.015 for ViT-L/14, and $K_{\text{prior}} = 10{,}000$. Hyperparameter settings for baselines are provided in the Appendix C.2.

Main results (RQ1) The experimental results are summarized in Table 1. We observe that training-free methods generally perform worse, as they do not update the model during adaptation. Among them, TPS achieves relatively strong performance by adjusting the text embeddings. CLIPArTT and WATT-S, which allow updates to the image encoder, perform best among the baselines. However, these methods do not share information across batches, which limits their overall effectiveness. Across all settings, `Mint` consistently improves accuracy and achieves the best performance. Compared to the strongest baselines, `Mint` yields absolute gains of 3.9%, 2.2%, and 3.1%, respectively.

Robustness to batch size (RQ1) To evaluate the robustness of `Mint` under different test-time conditions, we run it with batch sizes ranging from 1 to 200, using the same set of hyperparameters

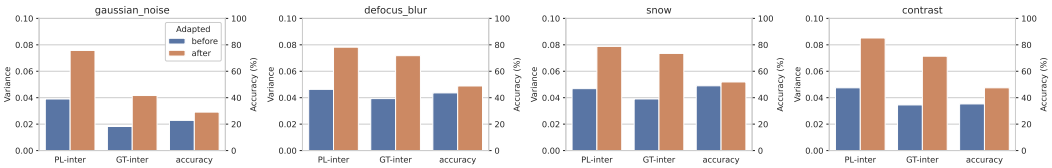

Figure 5: `Mint` alleviates variance collapse.

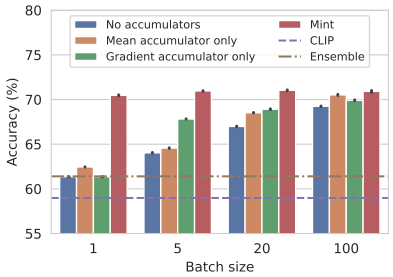

Figure 6: Ablation study.

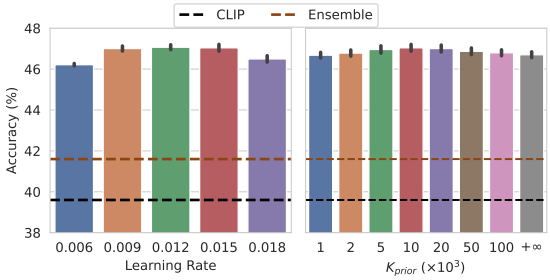

Figure 7: Hyperparameter sensitivity.

across all settings. As shown in Table 2, `Mint` consistently maintains strong performance across this range. Even in the extreme case of batch size 1, it achieves significant accuracy gains, demonstrating its effectiveness in highly constrained online adaptation scenarios.

Variance collapse (RQ2) We investigate the underlying mechanism of `Mint` by analyzing its effect on the image embeddings. Specifically, we evaluate the PL-inter variance, GT-inter variance, and classification accuracy on CIFAR-100-C before and after adaptation, under four representative types of corruption (same as Figure 2). As shown in Figure 5, `Mint` successfully increases PL-inter variance by design, and this also leads to a clear improvement in GT-inter variance. The increased GT-inter variance is accompanied by a rise in accuracy, indicating that `Mint` effectively mitigates variance collapse. Additional results across all corruption types and datasets are provided in Appendix C.4.

Efficiency (RQ3) We compare the testing time of `Mint` with baseline algorithms on CIFAR-100-C by measuring the time required to process one corruption type (10,000 images). As shown in Table 3, Mint runs substantially faster than other training-based TTA methods. This efficiency primarily stems from its simple design and the fact that it performs only a single model update per batch, unlike methods that require multiple iterative updates during adaptation. Notably, `Mint` is only slower than CLIP and other training-free and augmentation-free baselines.

Table 3: Comparison of testing time.

Method	Testing Time	Accuracy (%)	Gain (%)
CLIP	21s	35.8	–
TPT	23m21s	36.0	+0.2
VTE	9m45s	36.6	+0.8
Zero	9m50s	36.8	+1.0
TDA	33s	38.4	+2.6
DMN-ZS	30s	38.5	+2.7
TPS	9m58s	38.6	+2.8
CLIPArTT	7m40s	40.7	+4.9
WATT-S	50m20s	41.9	+6.1
`Mint`	1m07s	**44.1**	**+8.3**

Ablation study To understand the individual contributions of the two accumulators in `Mint`, we perform an ablation study on CIFAR-10-C comparing the full method with the following variants: (1) Mean accumulator only, which removes the gradient accumulator; (2) Gradient accumulator only, which removes the mean accumulator; and (3) No accumulators, which disables both components. We observe in Figure 6 that both accumulators contribute to the performance of `Mint`, especially under small batch sizes. The mean accumulator is essential for estimating PL-inter variance in extremely small batches, including the batch size of 1. Without it, gradients cannot be computed when the batch contains only a single class instance, rendering adaptation ineffective. Meanwhile, the gradient accumulator improves adaptation quality by reducing the noise in gradient estimates across batches. Overall, `Mint` exhibits the strongest robustness and performance when both accumulators are used, validating the necessity of their complementary roles in the online test-time adaptation setting. Additionally, we explore adapting different layers in the visual encoder and find that updating all LayerNorm layers yields the best performance (see Appendix C.5).

Hyperparameter sensitivity We study the sensitivity of `Mint` to its two hyperparameters: the learning rate and the prior strength K_{prior}, across three datasets. Results on ImageNet-C are shown in Figure 7, with results on CIFAR-10-C and CIFAR-100-C included in Appendix C.6. We observe that `Mint` remains stable across a broad range of hyperparameter values, without requiring precise tuning. In particular, we find that a learning rate of 0.009 and a prior size of $K_{\text{prior}} = 10{,}000$ consistently perform well across different datasets and architectures, demonstrating the robustness and generality of the method.

Additional experiments We further evaluate `Mint` on clean datasets (uncorrupted CIFAR-10, CIFAR-100, and ImageNet), ImageNet variants (ImageNet-A, -V2, -R, and -Sketch), and corruption benchmarks under the mixture-of-domain setting [27]. The corresponding results are provided in Appendix C.7, C.8, and C.9. `Mint` demonstrates consistently strong performance across these scenarios, confirming its broad applicability.

6 Conclusion

In this work, we identify variance collapse in image embeddings as a key factor behind CLIP's performance degradation under corruptions. Through theoretical analysis, we attribute this phenomenon to the image encoder encoding corruption-related patterns, which dilutes class-discriminative signals. We further show that maximizing inter-class variance, even when computed using pseudo labels, can provably enhance performance. Based on this insight, we propose Mint, a simple yet effective test-time adaptation method. Mint leverages cumulative mean and gradient accumulators to operate robustly in low-batch-size, online settings. Extensive experiments on corruption benchmarks demonstrate its strong performance and efficiency.

Acknowledgments and Disclosure of Funding

This work is supported by National Science Foundation under Award No. IIS-2416070, IIS-2117902. The views and conclusions are those of the authors and should not be interpreted as representing the official policies of the funding agencies or the government.

References

[1] James Urquhart Allingham, Jie Ren, Michael W. Dusenberry, Xiuye Gu, Yin Cui, Dustin Tran, Jeremiah Zhe Liu, and Balaji Lakshminarayanan. A simple zero-shot prompt weighting technique to improve prompt ensembling in text-image models. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 547–568. PMLR, 2023.

[2] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016.

[3] Wenxuan Bao, Ruxi Deng, Ruizhong Qiu, Tianxin Wei, Hanghang Tong, and Jingrui He. Latte: Collaborative test-time adaptation of vision-language models in federated learning. In *IEEE/CVF International Conference on Computer Vision, ICCV 2025, Honolulu, Hawaii, USA, October 19-23, 2025*. IEEE, 2025.

[4] Wenxuan Bao, Tianxin Wei, Haohan Wang, and Jingrui He. Adaptive test-time personalization for federated learning. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

[5] Wenxuan Bao, Zhichen Zeng, Zhining Liu, Hanghang Tong, and Jingrui He. Matcha: Mitigating graph structure shifts with test-time adaptation. In *The Thirteenth International Conference on Learning Representations*, 2025.

[6] Wei-Ting Chen, Yu-Jiet Vong, Sy-Yen Kuo, Sizhuo Ma, and Jian Wang. Robustsam: Segment anything robustly on degraded images. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 4081–4091. IEEE, 2024.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255. IEEE Computer Society, 2009.

[8] Mario Döbler, Robert A. Marsden, Tobias Raichle, and Bin Yang. A lost opportunity for vision-language models: A comparative study of online test-time adaptation for vision-language models. *CoRR*, abs/2405.14977, 2024.

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[10] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern classification, 2nd Edition*. Wiley, 2001.

[11] Matteo Farina, Gianni Franchi, Giovanni Iacca, Massimiliano Mancini, and Elisa Ricci. Frustratingly easy test-time adaptation of vision-language models. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[12] Chun-Mei Feng, Kai Yu, Yong Liu, Salman Khan, and Wangmeng Zuo. Diverse data augmentation with diffusions for effective test-time prompt tuning. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 2704–2714. IEEE, 2023.

[13] Jin Gao, Jialing Zhang, Xihui Liu, Trevor Darrell, Evan Shelhamer, and Dequan Wang. Back to the source: Diffusion-driven adaptation to test-time corruption. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 11786–11796. IEEE, 2023.

[14] Quanquan Gu, Zhenhui Li, and Jiawei Han. Generalized fisher score for feature selection. In *UAI 2011, Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence, Barcelona, Spain, July 14-17, 2011*, pages 266–273. AUAI Press, 2011.

[15] Gustavo Adolfo Vargas Hakim, David Osowiechi, Mehrdad Noori, Milad Cheraghalikhani, Ali Bahri, Moslem Yazdanpanah, Ismail Ben Ayed, and Christian Desrosiers. Clipartt: Adaptation of CLIP to new domains at test time. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2025, Tucson, AZ, USA, February 26 - March 6, 2025*, pages 7092–7101. IEEE, 2025.

[16] Dan Hendrycks and Thomas G. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[17] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 448–456. JMLR.org, 2015.

[18] Yusuke Iwasawa and Yutaka Matsuo. Test-time classifier adjustment module for model-agnostic domain generalization. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 2427–2440, 2021.

[19] Adilbek Karmanov, Dayan Guan, Shijian Lu, Abdulmotaleb El-Saddik, and Eric P. Xing. Efficient test-time adaptation of vision-language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 14162–14171. IEEE, 2024.

[20] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[21] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[22] Jonghyun Lee, Dahuin Jung, Saehyung Lee, Junsung Park, Juhyeon Shin, Uiwon Hwang, and Sungroh Yoon. Entropy is not enough for test-time adaptation: From the perspective of disentangled factors. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[23] Yoonho Lee, Annie S. Chen, Fahim Tajwar, Ananya Kumar, Huaxiu Yao, Percy Liang, and Chelsea Finn. Surgical fine-tuning improves adaptation to distribution shifts. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[24] Jian Liang, Ran He, and Tieniu Tan. A comprehensive survey on test-time adaptation under distribution shifts. *International Journal of Computer Vision*, pages 1–34, 2024.

[25] Xiaosong Ma, Jie Zhang, Song Guo, and Wenchao Xu. Swapprompt: Test-time prompt adaptation for vision-language models. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

[26] Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Yaofo Chen, Shijian Zheng, Peilin Zhao, and Mingkui Tan. Efficient test-time model adaptation without forgetting. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 16888–16905. PMLR, 2022.

[27] Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Zhiquan Wen, Yaofo Chen, Peilin Zhao, and Mingkui Tan. Towards stable test-time adaptation in dynamic wild world. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[28] David Osowiechi, Mehrdad Noori, Gustavo Adolfo Vargas Hakim, Moslem Yazdanpanah, Ali Bahri, Milad Cheraghalikhani, Sahar Dastani, Farzad Beizaee, Ismail Ben Ayed, and Christian Desrosiers. WATT: weight average test time adaptation of CLIP. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[29] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 2065–2074. IEEE, 2021.

[30] Priyank Pathak, Shyam Marjit, Shruti Vyas, and Yogesh S Rawat. LR0.FM: LOW-RESOLUTION ZERO-SHOT CLASSIFICATION BENCHMARK FOR FOUNDATION MODELS. In *The Thirteenth International Conference on Learning Representations*, 2025.

[31] Sarah M. Pratt, Ian Covert, Rosanne Liu, and Ali Farhadi. What does a platypus look like? generating customized prompts for zero-shot image classification. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 15645–15655. IEEE, 2023.

[32] 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 *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 2021.

[33] Yongming Rao, Wenliang Zhao, Guangyi Chen, Yansong Tang, Zheng Zhu, Guan Huang, Jie Zhou, and Jiwen Lu. Denseclip: Language-guided dense prediction with context-aware prompting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 18061–18070. IEEE, 2022.

[34] Jameel Abdul Samadh, Hanan Gani, Noor Hussein, Muhammad Uzair Khattak, Muzammal Naseer, Fahad Shahbaz Khan, and Salman H. Khan. Align your prompts: Test-time prompting with distribution alignment for zero-shot generalization. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

[35] Manli Shu, Weili Nie, De-An Huang, Zhiding Yu, Tom Goldstein, Anima Anandkumar, and Chaowei Xiao. Test-time prompt tuning for zero-shot generalization in vision-language models.

In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[36] Elaine Sui, Xiaohan Wang, and Serena Yeung-Levy. Just shift it: Test-time prototype shifting for zero-shot generalization with vision-language models. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2025, Tucson, AZ, USA, February 26 - March 6, 2025*, pages 825–835. IEEE, 2025.

[37] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018.

[38] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno A. Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[39] Olivia Wiles, Sven Gowal, Florian Stimberg, Sylvestre-Alvise Rebuffi, Ira Ktena, Krishnamurthy Dvijotham, and Ali Taylan Cemgil. A fine-grained analysis on distribution shift. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[40] Zehao Xiao and Cees G. M. Snoek. Beyond model adaptation at test time: A survey. *CoRR*, abs/2411.03687, 2024.

[41] Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 12360–12371, 2019.

[42] Ce Zhang, Simon Stepputtis, Katia P. Sycara, and Yaqi Xie. Dual prototype evolving for test-time generalization of vision-language models. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[43] Marvin Zhang, Sergey Levine, and Chelsea Finn. MEMO: test time robustness via adaptation and augmentation. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[44] Renrui Zhang, Wei Zhang, Rongyao Fang, Peng Gao, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-adapter: Training-free adaption of CLIP for few-shot classification. In *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXXV*, volume 13695 of *Lecture Notes in Computer Science*, pages 493–510. Springer, 2022.

[45] Yabin Zhang, Wenjie Zhu, Hui Tang, Zhiyuan Ma, Kaiyang Zhou, and Lei Zhang. Dual memory networks: A versatile adaptation approach for vision-language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 28718–28728. IEEE, 2024.

[46] Yinghua Zhang, Yu Zhang, Ying Wei, Kun Bai, Yangqiu Song, and Qiang Yang. Fisher deep domain adaptation. In *Proceedings of the 2020 SIAM International Conference on Data Mining, SDM 2020, Cincinnati, Ohio, USA, May 7-9, 2020*, pages 469–477. SIAM, 2020.

[47] Yiwu Zhong, Jianwei Yang, Pengchuan Zhang, Chunyuan Li, Noel Codella, Liunian Harold Li, Luowei Zhou, Xiyang Dai, Lu Yuan, Yin Li, and Jianfeng Gao. Regionclip: Region-based language-image pretraining. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 16772–16782. IEEE, 2022.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: In the abstract and introduction, we claim that this paper focuses on test-time adaptation of vision-language models. Contributions are clearly listed at the end of the introduction and in the abstract.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Please refer to Appendix A.2.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [Yes]

Justification: We clearly state the main assumptions in Section 3. Formal statements and proofs are given in Appendix B.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: In the paper, we clearly specified the methods we used to obtain the experimental results and all the hyperparameters used in the process, which can fully support the reproducibility of the experiment. More details are provided in Appendix C.2.

Guidelines:

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

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

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

   Answer: [Yes]

   Justification: We use open-source datasets, and provide the code in the supplemental material.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We provide key information in Section 5, and other details in Appendix C.2.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We report results with mean and standard deviation from five independent runs with different random seeds. Notice that for space limits, we provide the error bar of Table 1 in Appendix C.3 instead.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Please refer to Appendix C.2.2.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We strictly adhere to the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: Please refer to Appendix A.3.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

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

11. **Safeguards**

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

    Answer: [NA]

    Justification: Our paper does not release new data or models.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: All assets used in the paper are properly credited, and their licenses and terms of use have been explicitly mentioned and respected if provided in the original paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [NA]

    Justification: Our paper does not release new assets.

    Guidelines:

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

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

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

    Answer: [NA]

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

    Guidelines:

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

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

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

    Answer: [NA]

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

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [NA]

    Justification: LLM is used only for writing, editing, or formatting purposes in this paper.

    Guidelines:

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

Appendix

Contents

A Discussion

A.1 Additional related works

In this subsection, we discuss additional related work on general test-time adaptation. Many of these methods have inspired recent advances in TTA algorithms for VLMs.

Generic test-time adaptation (TTA) Most TTA methods aim to improve model accuracy by optimizing a carefully designed unsupervised loss on unlabeled test data. A prominent line of work minimizes the entropy of model predictions, based on the intuition that entropy quantifies prediction uncertainty. Pioneered by Tent [38], these methods typically update the running statistics and affine parameters of batch normalization [17] layers. However, entropy minimization is often unstable, and many subsequent works [26, 27, 22] focus on improving its robustness. One important variant is marginal entropy [43], which captures a model's uncertainty across different augmentations of the same input. This idea has inspired several follow-up TTA approaches [35, 36] for VLMs.

Another line of potential approaches focuses on restoring uncorrupted images from corrupted ones, using generative techniques such as diffusion models [13] or super-resolution [30, 6]. These methods do not require adapting the model at test time. However, as noted in [27], they often perform well on certain types of corruption but poorly on others, indicating limited generalization across corruption types.

A.2 Limitations

While our analysis reveals a consistent variance collapse pattern across multiple datasets and corruption types, it primarily focuses on natural distribution shifts and classification tasks. Extending our analysis and algorithm to broader types of distribution shifts (e.g., adversarial perturbations) and more diverse tasks (e.g., object detection, semantic segmentation) represents an important direction for future work.

A.3 Broader impacts

Our work focuses on understanding and mitigating the degradation of vision-language models under distribution shift, particularly in the context of image corruption. On the positive side, improving model robustness can enhance the reliability of real-world applications such as accessibility tools, autonomous systems, and content moderation, especially under suboptimal conditions. By providing theoretical insights and simple, efficient test-time adaptation methods, our work contributes toward safer and more dependable AI deployments.

We do not anticipate significant negative societal impacts. Our method is unsupervised, operates solely at test time, and does not require access to sensitive data or any form of user interaction. Nonetheless, as with all performance-enhancing techniques, there is potential for misuse in contexts where robustness could amplify existing biases or be deployed without appropriate oversight. We encourage future work to consider fairness and accountability as these methods are applied more broadly.

B Theoretical analysis

B.1 Variance decomposition

In this section we give formal proof of variance decomposition.

Lemma B.1.

$$\mathcal{V}_{total}^{GT} = \mathcal{V}_{intra}^{GT} + \mathcal{V}_{inter}^{GT}, \quad \mathcal{V}_{total}^{PL} = \mathcal{V}_{intra}^{PL} + \mathcal{V}_{inter}^{PL}. \tag{11}$$

Proof. We define "prior" for each class $c = 1, \cdots, C$:

$$\bar{y}_c = \frac{1}{N} \sum_{i=1}^{N} y_{ic}, \tag{12}$$

The class means are

$$\bar{\boldsymbol{z}}_c := \frac{\sum_{i=1}^{N} y_{ic} \cdot \boldsymbol{z}_i}{\sum_{i=1}^{N} y_{ic}} = \frac{1}{N} \sum_{i=1}^{N} \frac{y_{ic}}{\bar{y}_c} \cdot \boldsymbol{z}_i. \tag{13}$$

The global mean is

$$\bar{\boldsymbol{z}} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{z}_i. \tag{14}$$

Notice that for all class $c = 1, \cdots, C$, we have

$$\begin{aligned}
\frac{1}{N} \sum_{i=1}^{N} \frac{y_{ic}}{\bar{y}_c} (\boldsymbol{z}_i - \bar{\boldsymbol{z}}_c) &= \left(\frac{1}{N} \sum_{i=1}^{N} \frac{y_{ic}}{\bar{y}_c} \boldsymbol{z}_i \right) - \left(\frac{1}{N} \sum_{i=1}^{N} \frac{y_{ic}}{\bar{y}_c} \bar{\boldsymbol{z}}_c \right) \\
&= \bar{\boldsymbol{z}}_c - \left(\frac{1}{N} \sum_{i=1}^{N} \frac{y_{ic}}{\bar{y}_c} \bar{\boldsymbol{z}}_c \right) && \text{(definition of } \bar{\boldsymbol{z}}_c\text{)} \\
&= \bar{\boldsymbol{z}}_c - \bar{\boldsymbol{z}}_c && \text{(definition of } \bar{y}_c\text{)} \\
&= \mathbf{0}
\end{aligned}$$

Therefore,

$$\begin{aligned}
\mathcal{V}_{\text{total}}^{\text{GT}} &= \frac{1}{N \cdot C} \sum_{i=1}^{N} \sum_{c=1}^{C} \frac{y_{ic}}{\bar{y}_c} \left\| \boldsymbol{z}_i - \bar{\boldsymbol{z}} \right\|_2^2 \\
&= \frac{1}{N \cdot C} \sum_{i=1}^{N} \sum_{c=1}^{C} \frac{y_{ic}}{\bar{y}_c} \left\| \boldsymbol{z}_i - \bar{\boldsymbol{z}}_c + \bar{\boldsymbol{z}}_c - \bar{\boldsymbol{z}} \right\|_2^2 \\
&= \frac{1}{N \cdot C} \sum_{i=1}^{N} \sum_{c=1}^{C} \frac{y_{ic}}{\bar{y}_c} \left(\left\| \boldsymbol{z}_i - \bar{\boldsymbol{z}}_c \right\|_2^2 + \left\| \bar{\boldsymbol{z}}_c - \bar{\boldsymbol{z}} \right\|_2^2 + 2 \left(\boldsymbol{z}_i - \bar{\boldsymbol{z}}_c \right)^\top \left(\bar{\boldsymbol{z}}_c - \bar{\boldsymbol{z}} \right) \right) \\
&= \frac{1}{N \cdot C} \sum_{i=1}^{N} \sum_{c=1}^{C} \frac{y_{ic}}{\bar{y}_c} \left\| \boldsymbol{z}_i - \bar{\boldsymbol{z}}_c \right\|_2^2 + \frac{1}{N \cdot C} \sum_{i=1}^{N} \sum_{c=1}^{C} \frac{y_{ic}}{\bar{y}_c} \left\| \bar{\boldsymbol{z}}_c - \bar{\boldsymbol{z}} \right\|_2^2 \\
&= \frac{1}{N \cdot C} \sum_{i=1}^{N} \sum_{c=1}^{C} \frac{y_{ic}}{\bar{y}_c} \left\| \boldsymbol{z}_i - \bar{\boldsymbol{z}}_c \right\|_2^2 + \frac{1}{C} \sum_{c=1}^{C} \left\| \bar{\boldsymbol{z}}_c - \bar{\boldsymbol{z}} \right\|_2^2 && \text{(definition of } \bar{y}_c\text{)} \\
&= \mathcal{V}_{\text{intra}}^{\text{GT}} + \mathcal{V}_{\text{inter}}^{\text{GT}}
\end{aligned}$$

By replacing each y_{ic} with $\hat{y}_{ic}$, $\bar{\boldsymbol{z}}_c$ with $\tilde{\boldsymbol{z}}_c$, and $\bar{\boldsymbol{z}}$ with $\tilde{\boldsymbol{z}}$, and repeating the above steps, it is straightforward to prove $\mathcal{V}_{\text{total}}^{\text{PL}} = \mathcal{V}_{\text{intra}}^{\text{PL}} + \mathcal{V}_{\text{inter}}^{\text{PL}}$.

□

B.2 Theoretical setup

This section introduces the setup and assumptions of our theoretical analysis. For simplicity, we focus on a binary classification setting where $C = 2$. While the standard notation of label for image i is $\boldsymbol{y}_i = [y_{i0}, y_{i1}]^\top \in \mathbb{R}^2$, we write $y_i = y_{i1}$ for brevity, with a mild abuse of notation. We also assume there is no label imbalance, i.e., $\Pr(y_i = 0) = \Pr(y_1 = 1) = \frac{1}{2}$.

Image latent representation Motivated by [39], we assume that each image can be mapped to a disentangled latent representation $\boldsymbol{v}_i = [\boldsymbol{v}_i^{\text{cls}}; \boldsymbol{v}_i^{\text{irr}}; \boldsymbol{v}_i^{\text{shift}}; \boldsymbol{v}_i^{\text{noise}}] \in \mathbb{R}^d$, composed of four components:

1. *Class-relevant feature* $\boldsymbol{v}_i^{\text{cls}} \in \mathbb{R}^{d_{\text{cls}}}$: Semantic feature that are directly predictive of the class label, $\boldsymbol{v}_i^{\text{cls}} = \boldsymbol{\mu}$ for $y_i = 1$ and $\boldsymbol{v}_i^{\text{cls}} = -\boldsymbol{\mu}$ for $y_i = 0$.
2. *Class-irrelevant feature* $\boldsymbol{v}_i^{\text{irr}} \in \mathbb{R}^{d_{\text{irr}}}$: Features that are unrelated to the classification task, such as background information. It is preserved during pretraining due to CLIP's general representation learning objective. We assume $\boldsymbol{v}_i^{\text{irr}} \sim \text{Rademacher}^{d_{\text{irr}}}$, i.e., uniformly distributed in $\{-1, 1\}^{d_{\text{irr}}}$.
3. *Structured distribution shift* $\boldsymbol{v}_i^{\text{shift}} \in \mathbb{R}^{d_{\text{shift}}}$: Features representing systematic distribution changes in the target domain, such as weather conditions or digital transforms. We assume $\boldsymbol{v}_i^{\text{shift}} = s \cdot \boldsymbol{\delta}$, where s indicates the severity of corruption or distribution shift.
4. *Unstructured noise* $\boldsymbol{v}_i^{\text{noise}}$: Random noise introduced by the corruption process. We assume $\boldsymbol{v}^{\text{noise}} \sim s \cdot \text{Rademacher}^{d_{\text{noise}}}$, i.e., uniformly distributed in $\{-1, 1\}^{d_{\text{noise}}}$.

Notice that by controlling the ratio of $s, \|\boldsymbol{\mu}\|_2, \|\boldsymbol{\delta}\|_2$, we can freely adjust the ratio for four components.

LayerNorm and image embedding Following the structure of CLIP's visual encoder, we assume that the latent representation $\boldsymbol{v}_i$ first passes through a LayerNorm layer [2] with linear transformation, and then normalized to unit length. For analytical simplicity, we omit the demeaning step LayerNorm and ignore the bias term in its parameters. This simplification is also known as RMSNorm [41]. Under this simplification, the image embedding can be expressed as

$$\boldsymbol{z}_i = \text{normalize}\left(\frac{\boldsymbol{v}_i}{\sqrt{\text{Var}[\boldsymbol{v}_i]}} \odot \boldsymbol{w}\right), \tag{15}$$

where $\odot$ represents element-wise multiplication of vectors, $\boldsymbol{w} = [\boldsymbol{w}^{\text{cls}}; \boldsymbol{w}^{\text{irr}}; \boldsymbol{w}^{\text{shift}}; \boldsymbol{w}^{\text{noise}}] \in \mathbb{R}^d$ is the LayerNorm weights, and $\text{normalize}(\cdot)$ denotes ℓ_2 normalization. For simplicity, we assume $\boldsymbol{w} = \mathbf{1}$ at initialization. $\boldsymbol{w}$ is updated during TTA. Since $\sqrt{\text{Var}[\boldsymbol{v}_i]}$ is just a scalar, the equation above can be further reduced to

$$\boldsymbol{z}_i = \text{normalize}\left(\boldsymbol{v}_i \odot \boldsymbol{w}\right) = \frac{\boldsymbol{v}_i \odot \boldsymbol{w}}{\|\boldsymbol{v}_i \odot \boldsymbol{w}\|_2}. \tag{16}$$

Text embedding and prediction Let $\boldsymbol{t}_0, \boldsymbol{t}_1$ denotes the text embedding for class 0 and 1, respectively. The model prediction is given by

$$y_i = \begin{cases} 0, & \text{when } \boldsymbol{z}_i^\top \boldsymbol{t}_0 \geq \boldsymbol{z}_i^\top \boldsymbol{t}_1 \\ 1, & \text{when } \boldsymbol{z}_i^\top \boldsymbol{t}_0 < \boldsymbol{z}_i^\top \boldsymbol{t}_1 \end{cases} \tag{17}$$

B.3 Change of variances under corruption

This section studies the behavior of various types of variance with increasing corruption severity s.

Theorem 3.1 (Variance collapse)**.** *When the sample size $N \to +\infty$,*

$$\mathcal{V}_{inter}^{GT} \xrightarrow{p} \frac{\|\boldsymbol{\mu}\|_2^2}{\|\boldsymbol{\mu}\|_2^2 + d_{irr} + s^2 \cdot \|\boldsymbol{\delta}\|_2^2 + s^2 \cdot d_{noise}},\ \mathcal{V}_{intra}^{GT} \xrightarrow{p} \frac{d_{irr} + s^2 \cdot d_{noise}}{\|\boldsymbol{\mu}\|_2^2 + d_{irr} + s^2 \cdot \|\boldsymbol{\delta}\|_2^2 + s^2 \cdot d_{noise}}, \quad (2)$$

where s denotes the corruption severity. As s increases, $\mathcal{V}_{inter}^{GT}$ strictly decreases. In addition, $\mathcal{V}_{intra}^{GT}$ also decreases when $\|\boldsymbol{\delta}\|_2 \geq \sqrt{d_{noise}/d_{irr}} \cdot \|\boldsymbol{\mu}\|_2$.

Proof. We first compute the normalizing factor for each image:

$$\begin{aligned}
\|\boldsymbol{v}_i \odot \boldsymbol{w}\|_2^2 &= \|\boldsymbol{v}_i^{\text{cls}} \odot \boldsymbol{w}^{\text{cls}}\|_2^2 + \|\boldsymbol{v}_i^{\text{irr}} \odot \boldsymbol{w}^{\text{irr}}\|_2^2 + \|\boldsymbol{v}_i^{\text{shift}} \odot \boldsymbol{w}^{\text{shift}}\|_2^2 + \|\boldsymbol{v}_i^{\text{noise}} \odot \boldsymbol{w}^{\text{noise}}\|_2^2 \\
&= \|\boldsymbol{\mu} \odot \boldsymbol{w}^{\text{cls}}\|_2^2 + \|\boldsymbol{w}^{\text{irr}}\|_2^2 + s^2 \cdot \|\boldsymbol{\delta} \odot \boldsymbol{w}^{\text{shift}}\|_2^2 + s^2 \cdot \|\boldsymbol{w}^{\text{noise}}\|_2^2 \\
&= \|\boldsymbol{\mu}\|_2^2 + d_{\text{irr}} + s^2 \cdot (\|\boldsymbol{\delta}\|_2^2 + d_{\text{noise}}) \qquad \text{(at initialization } \boldsymbol{w} = \mathbf{1}\text{)}
\end{aligned}$$

Notice that this normalizing factor is the same for each image, and is a function of $\boldsymbol{w}$ and severity s. Let

$$Z(\boldsymbol{w}, s) = \sqrt{\|\boldsymbol{\mu} \odot \boldsymbol{w}^{\text{cls}}\|_2^2 + \|\boldsymbol{w}^{\text{irr}}\|_2^2 + s^2 \cdot \|\boldsymbol{\delta} \odot \boldsymbol{w}^{\text{shift}}\|_2^2 + s^2 \cdot \|\boldsymbol{w}^{\text{noise}}\|_2^2}$$

denote the normalizing factor. Under infinite sample size, the total mean $\bar{\boldsymbol{z}}$ and class means $\bar{\boldsymbol{z}}_0, \bar{\boldsymbol{z}}_1$ can be expressed as:

$$\begin{aligned}
\bar{\boldsymbol{z}} &\xrightarrow{p} \mathbb{E}\boldsymbol{z}_i = \frac{1}{Z(\boldsymbol{w}, s)} \cdot [\mathbf{0}; \mathbf{0}; s \cdot \boldsymbol{\delta} \odot \boldsymbol{w}^{\text{shift}}; \mathbf{0}] \\
\bar{\boldsymbol{z}}_0 &\xrightarrow{p} \mathbb{E}[\boldsymbol{z}_i | y_i = 0] = \frac{1}{Z(\boldsymbol{w}, s)} \cdot [\boldsymbol{\mu} \odot \boldsymbol{w}^{\text{cls}}; \mathbf{0}; s \cdot \boldsymbol{\delta} \odot \boldsymbol{w}^{\text{shift}}; \mathbf{0}] \\
\bar{\boldsymbol{z}}_1 &\xrightarrow{p} \mathbb{E}[\boldsymbol{z}_i | y_i = 1] = \frac{1}{Z(\boldsymbol{w}, s)} \cdot [-\boldsymbol{\mu} \odot \boldsymbol{w}^{\text{cls}}; \mathbf{0}; s \cdot \boldsymbol{\delta} \odot \boldsymbol{w}^{\text{shift}}; \mathbf{0}]
\end{aligned}$$

The GT-total variance:

$$\begin{aligned}
\mathcal{V}_{\text{total}}^{\text{GT}} &\xrightarrow{p} \mathbb{E}\|\boldsymbol{z}_i - \mathbb{E}\boldsymbol{z}_i\|_2^2 \\
&= \frac{1}{Z(\boldsymbol{w}, s)^2} \cdot \left(\|\boldsymbol{\mu} \odot \boldsymbol{w}^{\text{cls}}\|_2^2 + \|\boldsymbol{w}^{\text{irr}}\|_2^2 + 0 + \|\boldsymbol{w}^{\text{noise}}\|_2^2\right) \\
&= \frac{\|\boldsymbol{\mu} \odot \boldsymbol{w}^{\text{cls}}\|_2^2 + \|\boldsymbol{w}^{\text{irr}}\|_2^2 + s^2 \cdot \|\boldsymbol{w}^{\text{noise}}\|_2^2}{\|\boldsymbol{\mu} \odot \boldsymbol{w}^{\text{cls}}\|_2^2 + \|\boldsymbol{w}^{\text{irr}}\|_2^2 + s^2 \cdot \|\boldsymbol{\delta} \odot \boldsymbol{w}^{\text{shift}}\|_2^2 + s^2 \cdot \|\boldsymbol{w}^{\text{noise}}\|_2^2} \\
&= \frac{\|\boldsymbol{\mu}\|_2^2 + d_{\text{irr}} + s^2 \cdot d_{\text{noise}}}{\|\boldsymbol{\mu}\|_2^2 + d_{\text{irr}} + s^2 \cdot (\|\boldsymbol{\delta}\|_2^2 + d_{\text{noise}})} \qquad \text{(at initialization } \boldsymbol{w} = \mathbf{1}\text{)}
\end{aligned}$$

The GT-inter variance:

$$\begin{aligned}
\mathcal{V}_{\text{inter}}^{\text{GT}} &\xrightarrow{p} \frac{1}{2} \sum_{c=1}^{2} \|\mathbb{E}[\boldsymbol{z}_i | y_i = c] - \mathbb{E}\boldsymbol{z}_i\|_2^2 \\
&= \frac{1}{Z(\boldsymbol{w}, s)^2} \cdot \|\boldsymbol{\mu}\|_2^2 \\
&= \frac{\|\boldsymbol{\mu} \odot \boldsymbol{w}^{\text{cls}}\|_2^2}{\|\boldsymbol{\mu} \odot \boldsymbol{w}^{\text{cls}}\|_2^2 + \|\boldsymbol{w}^{\text{irr}}\|_2^2 + s^2 \cdot \|\boldsymbol{\delta} \odot \boldsymbol{w}^{\text{shift}}\|_2^2 + s^2 \cdot \|\boldsymbol{w}^{\text{noise}}\|_2^2} \\
&= \frac{\|\boldsymbol{\mu}\|_2^2}{\|\boldsymbol{\mu}\|_2^2 + d_{\text{irr}} + s^2 \cdot (\|\boldsymbol{\delta}\|_2^2 + d_{\text{noise}})} \qquad \text{(at initialization } \boldsymbol{w} = \mathbf{1}\text{)}
\end{aligned}$$

And the GT-intra variance:

$$\begin{aligned}
\mathcal{V}_{\text{intra}}^{\text{GT}} &= \mathcal{V}_{\text{total}}^{\text{GT}} - \mathcal{V}_{\text{inter}}^{\text{GT}} \\
&\xrightarrow{p} \frac{\|\boldsymbol{w}^{\text{irr}}\|_2^2 + s^2 \cdot \|\boldsymbol{w}^{\text{noise}}\|_2^2}{\|\boldsymbol{\mu} \odot \boldsymbol{w}^{\text{cls}}\|_2^2 + \|\boldsymbol{w}^{\text{irr}}\|_2^2 + s^2 \cdot \|\boldsymbol{\delta} \odot \boldsymbol{w}^{\text{shift}}\|_2^2 + s^2 \cdot \|\boldsymbol{w}^{\text{noise}}\|_2^2} \\
&= \frac{d_{\text{irr}} + s^2 \cdot d_{\text{noise}}}{\|\boldsymbol{\mu}\|_2^2 + d_{\text{irr}} + s^2 \cdot (\|\boldsymbol{\delta}\|_2^2 + d_{\text{noise}})} \qquad \text{(at initialization } \boldsymbol{w} = \mathbf{1}\text{)}
\end{aligned}$$

□

B.4 Adaptation

In this section, we derive how maximizing the pseudo-label inter-class (PL-inter) variance influences the learned representation, when only the LayerNorm parameters are updated during test-time adaptation.

Lemma B.2. *When the sample size $N \to +\infty$,*

$$\mathcal{V}_{inter}^{PL} \xrightarrow{p} \frac{1}{2}\left(\frac{1}{(\mathbb{E}\hat{y}_i)^2} + \frac{1}{(1-\mathbb{E}\hat{y}_i)^2}\right) \cdot \|\mathrm{Cov}(\boldsymbol{z}_i, \hat{y}_i)\|_2^2.$$

Proof.

$$\mathbb{E}[\boldsymbol{z}_i|\hat{y}_i = 1] = \frac{\mathbb{E}[\hat{y}_i \cdot \boldsymbol{z}_i]}{\mathbb{E}\hat{y}_i} = \frac{\mathbb{E}\hat{y}_i \cdot \mathbb{E}\boldsymbol{z}_i + \mathrm{Cov}(\boldsymbol{z}_i, \hat{y}_i)}{\mathbb{E}\hat{y}_i} = \mathbb{E}\boldsymbol{z}_i + \frac{\mathrm{Cov}(\boldsymbol{z}_i, \hat{y}_i)}{\mathbb{E}\hat{y}_i}$$

$$\mathbb{E}[\boldsymbol{z}_i|\hat{y}_i = 0] = \mathbb{E}\boldsymbol{z}_i - \frac{\mathrm{Cov}(\boldsymbol{z}_i, \hat{y}_i)}{1-\mathbb{E}\hat{y}_i}$$

$$\mathcal{V}_{\text{inter}}^{\text{PL}} \xrightarrow{p} \frac{1}{2}\sum_{c=1}^{2} \|\mathbb{E}[\boldsymbol{z}_i|\hat{y}_i = c] - \mathbb{E}\boldsymbol{z}_i\|_2^2 = \frac{1}{2}\left(\frac{1}{(\mathbb{E}\hat{y}_i)^2} + \frac{1}{(1-\mathbb{E}\hat{y}_i)^2}\right) \cdot \|\mathrm{Cov}(\boldsymbol{z}_i, \hat{y}_i)\|_2^2$$

□

Remark B.3.

$$\mathrm{Cov}(\boldsymbol{z}_i, \hat{y}_i) \sim \mathrm{Cov}(\boldsymbol{z}_i, \boldsymbol{z}_i^\top(\boldsymbol{t}_1 - \boldsymbol{t}_0)) = \Sigma_{\boldsymbol{z}_i}(\boldsymbol{t}_1 - \boldsymbol{t}_0)$$

where $\Sigma_{\boldsymbol{z}_i}$ is the covariance matrix of $\boldsymbol{z}_i$ and $\boldsymbol{t}_0, \boldsymbol{t}_1$ are the text embedding of class 0 and 1. This implies that maximizing PL-inter will enhance those features that (1) have high variance, and (2) are more relevant to the classification task described by the text embedding.

Theorem 3.2 (Maximization of PL-inter variance)**.** *When the sample size $N \to \infty$,*

$$\mathcal{V}_{inter}^{PL} \xrightarrow{p} \frac{C(\mathbb{E}\hat{y}_i)}{2} \cdot \frac{4\sigma_{\hat{y}y}^2 \cdot \|\boldsymbol{\mu} \odot \boldsymbol{w}^{cls}\|_2^2 + \|\boldsymbol{\sigma}_{irr} \odot \boldsymbol{w}^{irr}\|_2^2 + \|\boldsymbol{\sigma}_{noise} \odot \boldsymbol{w}^{noise}\|_2^2}{\|\boldsymbol{\mu} \odot \boldsymbol{w}^{cls}\|_2^2 + \|\boldsymbol{w}^{irr}\|_2^2 + s^2 \cdot \|\boldsymbol{\delta} \odot \boldsymbol{w}^{shift}\|_2^2 + s^2 \cdot \|\boldsymbol{w}^{noise}\|_2^2}, \tag{3}$$

where $C(\mathbb{E}\hat{y}_i) = \frac{1}{(\mathbb{E}\hat{y}_i)^2} + \frac{1}{(1-\mathbb{E}\hat{y}_i)^2}$, $\sigma_{\hat{y}y} = \mathrm{Cov}(y_i, \hat{y}_i)$, $\boldsymbol{\sigma}_{irr} = \mathrm{Cov}(\boldsymbol{v}^{irr}, \hat{y}_i)$, and $\boldsymbol{\sigma}_{noise} = \mathrm{Cov}(\boldsymbol{v}^{noise}, \hat{y}_i)$. Furthermore, when $\sigma_{\hat{y}y}^2 \geq \frac{\|\boldsymbol{\sigma}_{irr}\|_2^2}{4d_{irr}}$ and $\sigma_{\hat{y}y}^2 \geq \frac{\|\boldsymbol{\sigma}_{noise}\|_2^2}{4d_{noise}}$, we have

$$\nabla_{\boldsymbol{w}^{cls}} \mathcal{V}_{inter}^{PL} = C(\mathbb{E}\hat{y}_i) \cdot \frac{(4\sigma_{\hat{y}y}^2 d_{irr} - \|\boldsymbol{\sigma}_{irr}\|_2^2) + 4\sigma_{\hat{y}y}^2 s^2 \|\boldsymbol{\delta}\|_2^2 + (4\sigma_{\hat{y}y}^2 d_{noise} - \|\boldsymbol{\sigma}_{noise}\|_2^2)}{(\|\boldsymbol{\mu}\|_2^2 + d_{irr} + s^2 \cdot (\|\boldsymbol{\delta}\|_2^2 + d_{noise}))^2} \cdot \boldsymbol{\mu}^2 \geq \mathbf{0}, \tag{4}$$

$$\nabla_{\boldsymbol{w}^{shift}} \mathcal{V}_{inter}^{PL} = -C(\mathbb{E}\hat{y}_i) \cdot \frac{\mathcal{V}_{inter}^{PL}}{\|\boldsymbol{\mu}\|_2^2 + d_{irr} + s^2 \cdot (\|\boldsymbol{\delta}\|_2^2 + d_{noise})} \cdot s^2 \cdot \boldsymbol{\delta}^2 \leq \mathbf{0}. \tag{5}$$

Proof. Similar to the procedure of deriving GT-inter, we start by computing the total mean $\tilde{\boldsymbol{z}}$ and pseudo-class means $\tilde{\boldsymbol{z}}_0, \tilde{\boldsymbol{z}}_1$.

$$\tilde{\boldsymbol{z}} \xrightarrow{p} \mathbb{E}\boldsymbol{z}_i = \frac{1}{Z(\boldsymbol{w}, s)} \cdot [\mathbf{0}; \mathbf{0}; s \cdot \boldsymbol{\delta} \odot \boldsymbol{w}^{\text{shift}}; \mathbf{0}]$$

$$\tilde{\boldsymbol{z}}_1 \xrightarrow{p} \mathbb{E}[\boldsymbol{z}_i|\hat{y}_i = 1] = \frac{1}{\mathbb{E}\hat{y}_i} \cdot \frac{1}{Z(\boldsymbol{w}, s)} \cdot \mathbb{E}[\hat{y}_i \cdot \boldsymbol{v}_i \odot \boldsymbol{w}]$$

$$\tilde{\boldsymbol{z}}_0 \xrightarrow{p} \mathbb{E}[\boldsymbol{z}_i|\hat{y}_i = 0] = \frac{1}{1-\mathbb{E}\hat{y}_i} \cdot \frac{1}{Z(\boldsymbol{w}, s)} \cdot \mathbb{E}[(1-\hat{y}_i) \cdot \boldsymbol{v}_i \odot \boldsymbol{w}]$$

where $Z(\boldsymbol{w}, s) = \sqrt{\|\boldsymbol{\mu} \odot \boldsymbol{w}^{\text{cls}}\|_2^2 + \|\boldsymbol{w}^{\text{irr}}\|_2^2 + s^2 \cdot \|\boldsymbol{\delta} \odot \boldsymbol{w}^{\text{shift}}\|_2^2 + s^2 \cdot \|\boldsymbol{w}^{\text{noise}}\|_2^2}$ is the normalizing factor we defined in the proof of Theorem 3.1. For four components of the feature:

$$\mathbb{E}[\hat{y}_i \cdot \boldsymbol{v}_i^{\text{cls}} \odot \boldsymbol{w}^{\text{cls}}] = \mathbb{E}[\hat{y}_i \cdot (2y_i - 1) \cdot \boldsymbol{\mu} \odot \boldsymbol{w}^{\text{cls}}] = 2\mathrm{Cov}(\hat{y}_i, y_i) \cdot \boldsymbol{\mu} \odot \boldsymbol{w}^{\text{cls}}$$

$$\mathbb{E}[\hat{y}_i \cdot \boldsymbol{v}_i^{\text{irr}} \odot \boldsymbol{w}^{\text{irr}}] = \mathrm{Cov}(\hat{y}_i, \boldsymbol{v}^{\text{irr}}) \odot \boldsymbol{w}^{\text{irr}}$$

$$\mathbb{E}[\hat{y}_i \cdot \boldsymbol{v}_i^{\text{shift}} \odot \boldsymbol{w}^{\text{shift}}] = \mathbb{E}\hat{y}_i \cdot s \cdot \boldsymbol{\delta} \odot \boldsymbol{w}^{\text{shift}}$$

$$\mathbb{E}[\hat{y}_i \cdot \boldsymbol{v}_i^{\text{noise}} \odot \boldsymbol{w}^{\text{noise}}] = \mathrm{Cov}(\hat{y}_i, \boldsymbol{v}^{\text{noise}}) \odot \boldsymbol{w}^{\text{noise}}$$

And similarly,

$$\begin{aligned}
\mathbb{E}[(1-\hat{y}_i)\cdot \boldsymbol{v}_i^{\text{cls}} \odot \boldsymbol{w}^{\text{cls}}] &= \mathbb{E}[(1-\hat{y}_i)\cdot(2y_i-1)\cdot \boldsymbol{\mu} \odot \boldsymbol{w}^{\text{cls}}] = -2\text{Cov}(\hat{y}_i, y_i)\cdot \boldsymbol{\mu} \odot \boldsymbol{w}^{\text{cls}} \\
\mathbb{E}[(1-\hat{y}_i)\cdot \boldsymbol{v}_i^{\text{irr}} \odot \boldsymbol{w}^{\text{irr}}] &= -\text{Cov}(\hat{y}_i, \boldsymbol{v}^{\text{irr}}) \odot \boldsymbol{w}^{\text{irr}} \\
\mathbb{E}[(1-\hat{y}_i)\cdot \boldsymbol{v}_i^{\text{shift}} \odot \boldsymbol{w}^{\text{shift}}] &= \mathbb{E}(1-\hat{y}_i)\cdot s\cdot \boldsymbol{\delta} \odot \boldsymbol{w}^{\text{shift}} \\
\mathbb{E}[(1-\hat{y}_i)\cdot \boldsymbol{v}_i^{\text{noise}} \odot \boldsymbol{w}^{\text{noise}}] &= -\text{Cov}(\hat{y}_i, \boldsymbol{v}^{\text{noise}}) \odot \boldsymbol{w}^{\text{noise}}
\end{aligned}$$

Therefore, we have

$$\begin{aligned}
\mathcal{V}_{\text{inter}}^{\text{PL}} &\xrightarrow{p} \frac{1}{2}\sum_{c=1}^{2} \|\mathbb{E}[\boldsymbol{z}_i|\hat{y}_i=c] - \mathbb{E}\boldsymbol{z}_i\|_2^2 \\
&= \frac{1}{2}\cdot\left(\frac{1}{(\mathbb{E}\hat{y}_i)^2} + \frac{1}{(1-\mathbb{E}\hat{y}_i)^2}\right)\cdot \frac{1}{Z(\boldsymbol{w},s)^2}\cdot \\
&\quad \left(4\text{Cov}(\hat{y}_i, y)^2\cdot\|\boldsymbol{\mu}\odot\boldsymbol{w}^{\text{cls}}\|_2^2 + \|\text{Cov}(\hat{y}_i,\boldsymbol{v}^{\text{irr}})\odot\boldsymbol{w}^{\text{irr}}\|_2^2 + \|\text{Cov}(\hat{y}_i,\boldsymbol{v}^{\text{noise}})\odot\boldsymbol{w}^{\text{noise}}\|_2^2\right) \\
&= \frac{1}{2}\cdot\left(\frac{1}{(\mathbb{E}\hat{y}_i)^2} + \frac{1}{(1-\mathbb{E}\hat{y}_i)^2}\right)\cdot \frac{4\sigma_{\hat{y}y}^2\cdot\|\boldsymbol{\mu}\odot\boldsymbol{w}^{\text{cls}}\|_2^2 + \|\boldsymbol{\sigma}_{\text{irr}}\odot\boldsymbol{w}^{\text{irr}}\|_2^2 + \|\boldsymbol{\sigma}_{\text{noise}}\odot\boldsymbol{w}^{\text{noise}}\|_2^2}{\|\boldsymbol{\mu}\odot\boldsymbol{w}^{\text{cls}}\|_2^2 + \|\boldsymbol{w}^{\text{irr}}\|_2^2 + s^2\cdot\|\boldsymbol{\delta}\odot\boldsymbol{w}^{\text{shift}}\|_2^2 + s^2\cdot\|\boldsymbol{w}^{\text{noise}}\|_2^2}
\end{aligned}$$

As a simple correctness check, when the pseudo-label $\hat{y}_i = y_i, \forall i$, substituting $\mathbb{E}\hat{y}_i = \mathbb{E}y_i = \frac{1}{2}$, $\sigma_{\hat{y}y} = \text{Var}(y_i) = \frac{1}{4}$, $\boldsymbol{\sigma}_{\text{irr}} = \mathbf{0}$, and $\boldsymbol{\sigma}_{\text{noise}} = \mathbf{0}$ recovers the result of GT-inter variance $\mathcal{V}_{\text{inter}}^{\text{GT}}$ in Theorem 3.1.

Finally, we compute the gradients w.r.t. four components of $\boldsymbol{w}$ at initialization. Note that although the pseudo-labels $\hat{y}_i$ depend on the model parameters, this dependence involves an $\arg\max$ operation and is thus non-differentiable. Therefore, during optimization, we treat the pseudo-labels as fixed constants and do not backpropagate through them. Let $C(\mathbb{E}\hat{y}_i) = \frac{1}{(\mathbb{E}\hat{y}_i)^2} + \frac{1}{(1-\mathbb{E}\hat{y}_i)^2}$,

$$\begin{aligned}
\nabla_{\boldsymbol{w}^{\text{cls}}}\mathcal{V}_{\text{inter}}^{\text{PL}} &= \frac{C(\mathbb{E}\hat{y}_i)}{2}\cdot\frac{(4\sigma_{\hat{y}y}^2\|\boldsymbol{w}^{\text{irr}}\|_2^2 - \|\boldsymbol{\sigma}_{\text{irr}}\odot\boldsymbol{w}^{\text{irr}}\|_2^2) + (4\sigma_{\hat{y}y}^2 s^2\|\boldsymbol{\delta}\odot\boldsymbol{w}^{\text{shift}}\|_2^2) + (4\sigma_{\hat{y}y}^2\|\boldsymbol{w}^{\text{noise}}\|_2^2 - \|\boldsymbol{\sigma}_{\text{noise}}\odot\boldsymbol{w}^{\text{noise}}\|_2^2)}{Z(\boldsymbol{w},s)^4}\cdot \\
&\quad 2\boldsymbol{\mu}^2\odot\boldsymbol{w}^{\text{cls}} \\
&= C(\mathbb{E}\hat{y}_i)\cdot\frac{(4\sigma_{\hat{y}y}^2 d_{\text{irr}} - \|\boldsymbol{\sigma}_{\text{irr}}\|_2^2) + 4\sigma_{\hat{y}y}^2 s^2\|\boldsymbol{\delta}\|_2^2 + (4\sigma_{\hat{y}y}^2 d_{\text{noise}} - \|\boldsymbol{\sigma}_{\text{noise}}\|_2^2)}{(\|\boldsymbol{\mu}\|_2^2 + d_{\text{irr}} + s^2\cdot(\|\boldsymbol{\delta}\|_2^2 + d_{\text{noise}}))^2}\cdot\boldsymbol{\mu}^2 \\
\nabla_{\boldsymbol{w}^{\text{shift}}}\mathcal{V}_{\text{inter}}^{\text{PL}} &= \frac{C(\mathbb{E}\hat{y}_i)}{2}\cdot -\frac{4\sigma_{\hat{y}y}^2\cdot\|\boldsymbol{\mu}\odot\boldsymbol{w}^{\text{cls}}\|_2^2 + \|\boldsymbol{\sigma}_{\text{irr}}\odot\boldsymbol{w}^{\text{irr}}\|_2^2 + \|\boldsymbol{\sigma}_{\text{noise}}\odot\boldsymbol{w}^{\text{noise}}\|_2^2}{Z(\boldsymbol{w},s)^4}\cdot s^2\cdot 2\boldsymbol{\delta}^2\odot\boldsymbol{w}^{\text{shift}} \\
&= -C(\mathbb{E}\hat{y}_i)\cdot\frac{\mathcal{V}_{\text{inter}}^{\text{PL}}}{\|\boldsymbol{\mu}\|_2^2 + d_{\text{irr}} + s^2\cdot(\|\boldsymbol{\delta}\|_2^2 + d_{\text{noise}})}\cdot s^2\cdot\boldsymbol{\delta}^2
\end{aligned}$$

□

C Experiments

C.1 Effect of corruptions

C.1.1 CIFAR-10-C

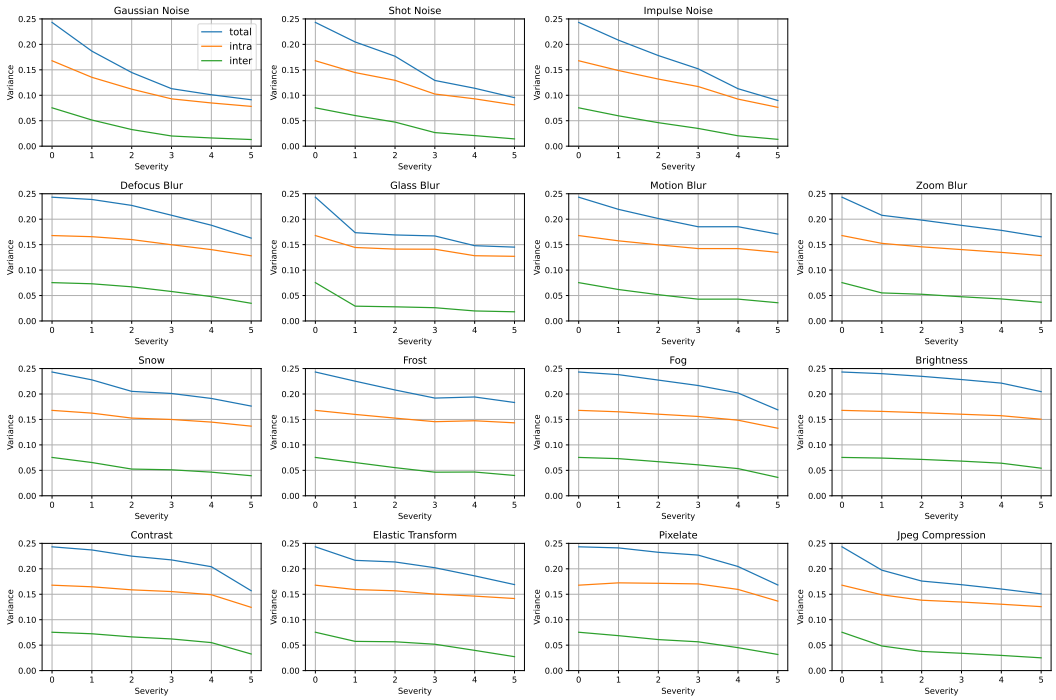

Figure 8: Effect of different levels of corruptions on ViT-B/32 on CIFAR-10-C.

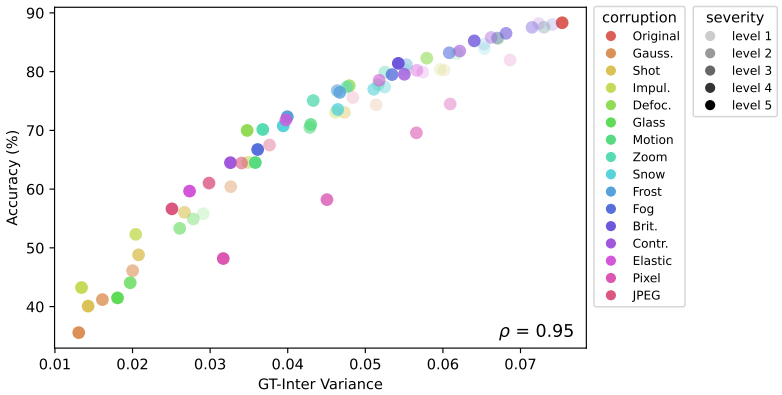

Figure 9: Correlation of GT-inter variance and classification accuracy of ViT-B/32 on CIFAR-10-C.

Table 4: Pearson correlation coefficients between accuracy and variances on ViT-B/32 on CIFAR-10-C.

	$\mathcal{V}_{\text{total}}^{\text{GT}}$	$\mathcal{V}_{\text{intra}}^{\text{GT}}$	$\mathcal{V}_{\text{inter}}^{\text{GT}}$
Accuracy	0.9104	0.8286	**0.9483**

C.1.2 CIFAR-100-C

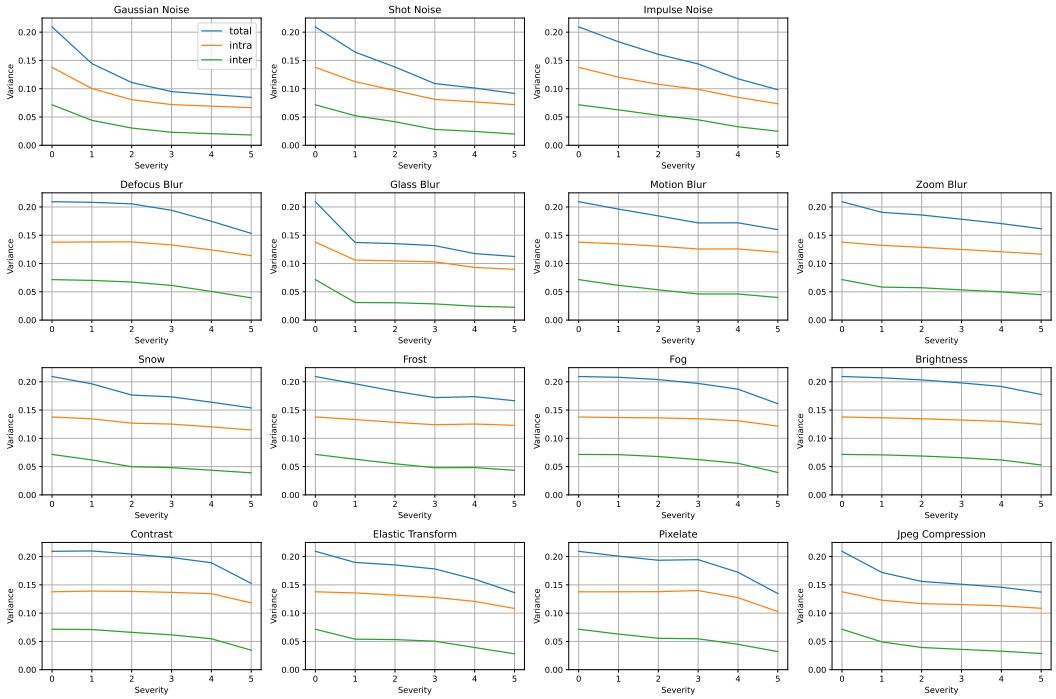

Figure 10: Effect of different levels of corruptions on ViT-B/16 on CIFAR-100-C.

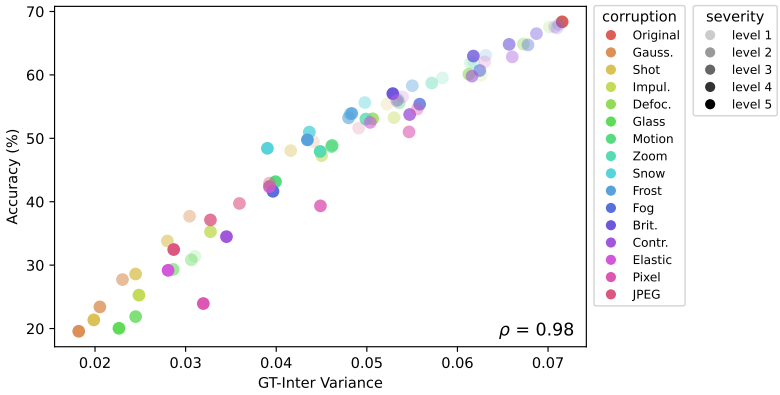

Figure 11: Correlation of GT-inter variance and classification accuracy of ViT-B/16 on CIFAR-100-C.

Table 5: Pearson correlation coefficients between accuracy and variances on ViT-B/16 on CIFAR-100-C.

	$\mathcal{V}_{\text{total}}^{\text{GT}}$	$\mathcal{V}_{\text{intra}}^{\text{GT}}$	$\mathcal{V}_{\text{inter}}^{\text{GT}}$
Accuracy	0.9364	0.8560	**0.9752**

C.1.3 ImageNet-C

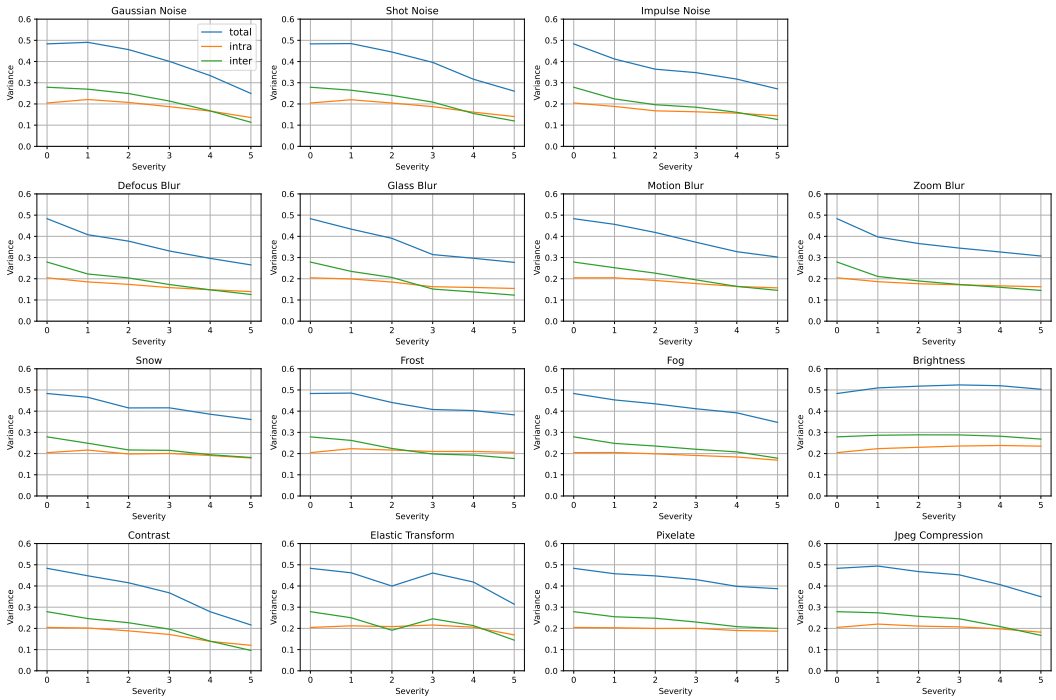

Figure 12: Effect of different levels of corruptions on ViT-L/14 on ImageNet-C.

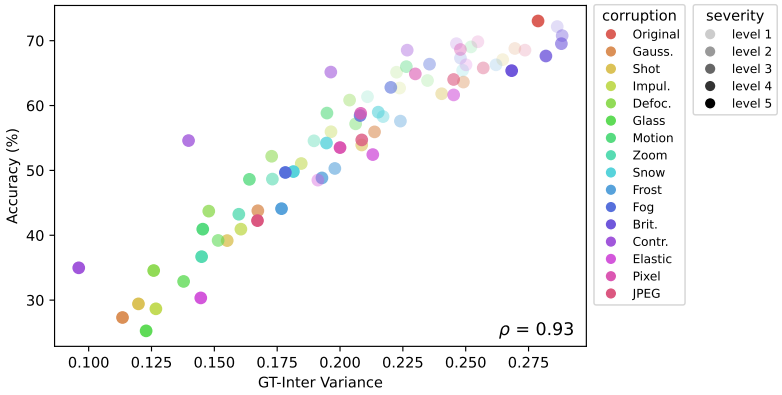

Figure 13: Correlation of GT-inter variance and classification accuracy of ViT-L/14 on ImageNet-C.

Table 6: Pearson correlation coefficients between accuracy and variances on ViT-L/14 on ImageNet-C.

	$\mathcal{V}_{\text{total}}^{\text{GT}}$	$\mathcal{V}_{\text{intra}}^{\text{GT}}$	$\mathcal{V}_{\text{inter}}^{\text{GT}}$
Accuracy	0.8937	0.7753	**0.9332**

C.2 Experiment details

C.2.1 Baselines

For all methods, expect from TPT [35] and CLIPArTT [15] which modifies the prompts, we use the 7 template in [44]:

- "`itap of a {class}`"
- "`a bad photo of the {class}`"
- "`a origami {class}`"
- "`a photo of the large {class}`"
- "`a {class} in a video game`"
- "`art of the {class}`"
- "`a photo of the small {class}`"

The text embedding for each class y is computed by

$$\boldsymbol{t}_y = \text{normalize}\left(\sum_{\kappa=1}^{k} \boldsymbol{t}_{y,\kappa}\right), \text{where } \boldsymbol{t}_{y,\kappa} = \text{normalize}(\text{text\_encoder}(\{\text{template}_\kappa, \text{classname}_y\}))$$

The following is our detailed handling method for other baselines and the usage of hyperparameters. The above hyperparameters are derived from those used in experiments reported in previous papers.

- For all augmentation-based baselines (TPT [35], TPS [36], Zero [11], VTE [8]), we use AugMix to augment each test image 63 times to obtain a batch of 64 images, which includes the original image. We select 10% of samples in the batch with lowest entropy to aggregate.
- In TPT [35], the number of prompt tokens is 4, the prompt is initialized with "a photo of a", and class-specific contexts are disabled. We use the AdamW optimizer and adopt a learning rate of 0.005, consistent with the setting used for ImageNet in the original papers.
- In TDA [19], positive cache is enabled with a shot capacity of 3, an adaptation strength (α) of 2.0, and a sharpness ratio (β) of 5.0. The negative cache is enabled with a shot capacity of 2, an adaptation strength (α) of 0.117, and a sharpness ratio (β) of 1.0, an entropy threshold between 0.2 and 0.5, and a mask threshold between 0.03 and 1.0.
- In DMN-ZS [45], the positive cache is enabled with a shot capacity of 50, an adaptation strength (α) of 0.3, and a sharpness ratio (β) of 5.5.
- In TPS [36], we also use the AdamW optimizer and adopt a learning rate of 0.005, consistent with the setting used for ImageNet in the original papers.
- In WATT-S [28], the learning rate is 0.001, the weight averaging is performed in a sequential manner, with 2 iterations per template and 5 total rounds of averaging.
- In CLIPArTT [15], the learning rate is 0.001, the adaptation process runs for 10 steps, and the top 3 predicted classes are used to construct the pseudo-label prompt.

C.2.2 Compute resources

All of our experiments are conducted on single NVIDIA Tesla V100 with 32GB memory, except for experiments on large batch size are conducted on single NVIDIA Tesla A100 with 80GB memory.

C.2.3 Licenses

The corruption benchmark is licensed under the Apache-2.0 License, as indicated at `https://github.com/hendrycks/robustness`. CLIP is licensed under the MIT License, as stated at `https://github.com/openai/CLIP`.

C.3 Full results for RQ1

Table 7: Accuracy (mean (s.d.) %) on corruption benchmarks with different batch sizes.

Batch Size	Noise			Blur				Weather				Digital				Avg.
	Gauss.	Shot	Impul.	Defoc.	Glass	Motion	Zoom	Snow	Frost	Fog	Brit.	Contr.	Elastic	Pixel	JPEG	
	ViT-B/32 on CIFAR-10-C															
1	57.9 (0.5)	61.3 (0.4)	52.3 (0.3)	76.2 (0.2)	60.0 (0.2)	76.9 (0.1)	79.2 (0.2)	79.1 (0.1)	78.9 (0.3)	75.2 (0.2)	86.4 (0.1)	76.6 (0.1)	70.3 (0.2)	63.4 (0.8)	63.3 (0.3)	70.5 (0.1)
2	58.0 (0.5)	61.3 (0.3)	52.3 (0.4)	76.2 (0.3)	60.0 (0.2)	76.9 (0.1)	79.3 (0.2)	79.1 (0.1)	78.9 (0.3)	75.2 (0.2)	86.4 (0.1)	76.6 (0.1)	70.3 (0.2)	63.4 (0.8)	63.2 (0.2)	70.5 (0.1)
5	60.6 (0.1)	62.9 (0.2)	53.3 (0.3)	76.2 (0.2)	60.8 (0.2)	77.0 (0.2)	79.2 (0.2)	79.1 (0.1)	78.9 (0.3)	75.1 (0.2)	86.4 (0.1)	76.8 (0.1)	70.4 (0.3)	64.6 (0.5)	63.2 (0.1)	71.0 (0.0)
10	59.8 (0.4)	62.8 (0.3)	54.0 (0.3)	76.0 (0.3)	61.3 (0.3)	77.1 (0.2)	79.1 (0.3)	79.0 (0.2)	78.8 (0.3)	75.1 (0.1)	86.4 (0.2)	76.8 (0.1)	70.3 (0.2)	65.7 (0.5)	63.2 (0.1)	71.0 (0.1)
50	59.0 (0.5)	62.4 (0.4)	54.2 (0.3)	75.8 (0.2)	61.8 (0.3)	77.1 (0.2)	78.9 (0.2)	79.0 (0.1)	78.9 (0.2)	75.2 (0.1)	86.3 (0.1)	76.9 (0.1)	70.1 (0.3)	66.6 (0.3)	63.4 (0.2)	71.0 (0.1)
100	58.8 (0.7)	62.0 (0.4)	54.2 (0.3)	75.6 (0.3)	62.4 (0.4)	77.0 (0.2)	78.7 (0.3)	78.9 (0.1)	78.7 (0.3)	75.1 (0.2)	86.4 (0.1)	77.1 (0.1)	69.7 (0.2)	67.2 (0.2)	63.4 (0.2)	71.0 (0.1)
200	58.0 (1.0)	61.3 (0.5)	54.7 (0.5)	75.0 (0.2)	62.2 (0.6)	77.0 (0.2)	78.4 (0.3)	78.5 (0.2)	78.6 (0.3)	74.9 (0.1)	86.2 (0.1)	77.0 (0.1)	68.0 (0.3)	67.0 (0.5)	62.0 (0.2)	70.6 (0.1)
	ViT-B/16 on CIFAR-100-C															
1	23.9 (0.4)	26.1 (0.4)	37.2 (0.2)	50.5 (0.1)	27.0 (0.2)	49.6 (0.3)	55.3 (0.2)	53.0 (0.2)	51.6 (0.2)	50.1 (0.2)	65.5 (0.1)	46.6 (0.3)	36.7 (0.2)	34.4 (0.5)	38.7 (0.6)	43.1 (0.1)
2	23.9 (0.4)	26.2 (0.4)	37.2 (0.2)	50.5 (0.1)	27.0 (0.2)	49.7 (0.3)	55.3 (0.2)	53.0 (0.2)	51.6 (0.1)	50.1 (0.2)	65.5 (0.2)	46.7 (0.3)	36.7 (0.2)	34.5 (0.5)	38.7 (0.6)	43.1 (0.1)
5	24.8 (0.3)	27.1 (0.6)	37.5 (0.2)	50.6 (0.1)	27.1 (0.2)	49.7 (0.3)	55.4 (0.1)	53.0 (0.2)	51.7 (0.2)	50.3 (0.2)	65.5 (0.2)	47.0 (0.3)	36.7 (0.2)	34.4 (0.6)	38.7 (0.5)	43.3 (0.1)
10	26.4 (0.7)	28.9 (0.8)	38.0 (0.2)	50.6 (0.1)	27.1 (0.2)	49.7 (0.2)	55.4 (0.1)	53.0 (0.2)	51.7 (0.2)	50.4 (0.1)	65.5 (0.1)	47.5 (0.3)	36.8 (0.2)	34.4 (0.6)	38.7 (0.6)	43.6 (0.1)
20	29.4 (0.5)	30.8 (0.7)	38.6 (0.2)	50.7 (0.2)	27.1 (0.2)	49.9 (0.2)	55.5 (0.1)	53.0 (0.2)	51.8 (0.1)	50.6 (0.2)	65.6 (0.1)	48.1 (0.2)	36.8 (0.2)	34.4 (0.7)	38.7 (0.5)	44.1 (0.1)
50	31.4 (0.3)	33.0 (0.5)	39.4 (0.3)	50.9 (0.1)	26.8 (0.2)	50.0 (0.3)	55.5 (0.2)	53.1 (0.2)	51.9 (0.1)	50.8 (0.1)	65.8 (0.2)	49.2 (0.1)	36.9 (0.2)	34.6 (1.0)	38.3 (0.5)	44.5 (0.1)
100	31.1 (0.2)	33.4 (0.4)	40.0 (0.3)	51.0 (0.2)	27.1 (0.3)	50.1 (0.3)	55.5 (0.1)	53.1 (0.2)	51.9 (0.1)	51.0 (0.1)	65.8 (0.1)	49.5 (0.2)	36.7 (0.2)	34.6 (0.9)	37.4 (0.4)	44.5 (0.1)
200	30.8 (0.4)	33.5 (0.5)	40.2 (0.2)	51.2 (0.2)	27.5 (0.5)	50.3 (0.2)	55.6 (0.2)	53.3 (0.2)	52.0 (0.2)	51.2 (0.1)	65.9 (0.3)	49.6 (0.2)	36.8 (0.1)	34.9 (0.7)	36.8 (0.3)	44.6 (0.1)
	ViT-L/14 on ImageNet-C															
1	31.6 (0.5)	33.2 (0.1)	36.3 (0.2)	41.0 (0.2)	37.6 (0.3)	46.5 (0.4)	41.3 (0.4)	54.0 (0.1)	43.4 (0.1)	57.4 (0.3)	67.7 (0.1)	45.7 (0.3)	41.5 (0.1)	55.7 (0.5)	54.8 (0.2)	45.8 (0.1)
2	31.9 (0.4)	33.0 (0.3)	37.2 (0.2)	40.1 (0.2)	37.6 (0.2)	46.7 (0.3)	42.1 (0.4)	54.5 (0.3)	43.8 (0.2)	57.7 (0.2)	67.7 (0.1)	47.7 (0.5)	41.5 (0.3)	57.1 (0.3)	55.1 (0.1)	46.2 (0.1)
5	32.4 (0.4)	33.4 (0.3)	37.1 (0.3)	39.7 (0.4)	37.6 (0.3)	46.7 (0.5)	43.4 (0.3)	55.6 (0.4)	44.4 (0.4)	57.7 (0.2)	67.8 (0.1)	49.4 (0.7)	42.0 (0.4)	57.8 (0.2)	55.3 (0.1)	46.7 (0.1)
10	32.6 (0.3)	33.6 (0.2)	36.8 (0.1)	39.7 (0.4)	37.6 (0.5)	46.8 (0.4)	44.1 (0.4)	55.3 (0.2)	45.2 (0.7)	57.5 (0.2)	67.7 (0.1)	49.4 (0.8)	42.8 (0.3)	58.3 (0.1)	55.2 (0.3)	46.8 (0.1)
20	33.0 (0.3)	34.3 (0.3)	37.3 (0.2)	39.6 (0.4)	37.2 (0.4)	46.6 (0.3)	45.1 (0.5)	55.2 (0.1)	46.6 (0.7)	57.5 (0.1)	67.7 (0.2)	48.9 (0.8)	43.9 (0.4)	58.2 (0.2)	54.6 (0.7)	47.0 (0.2)
50	33.2 (0.4)	35.1 (0.2)	37.6 (0.5)	38.8 (0.3)	36.9 (0.4)	47.1 (0.2)	45.4 (0.6)	55.0 (0.3)	48.3 (0.7)	57.5 (0.2)	67.4 (0.2)	47.0 (1.1)	45.1 (0.1)	57.9 (0.3)	54.8 (0.6)	47.1 (0.1)
100	33.8 (0.5)	34.9 (0.3)	37.6 (0.2)	38.6 (0.4)	37.2 (0.5)	47.4 (0.5)	45.7 (0.4)	55.0 (0.3)	48.9 (0.2)	57.5 (0.4)	67.2 (0.2)	44.2 (1.4)	46.0 (0.3)	57.4 (0.2)	54.0 (0.4)	47.0 (0.2)
200	33.9 (0.2)	34.7 (0.5)	37.7 (0.1)	38.7 (0.1)	37.2 (0.5)	47.2 (0.4)	45.3 (0.5)	54.7 (0.3)	49.1 (0.2)	57.5 (0.3)	67.3 (0.2)	42.0 (1.9)	46.3 (0.5)	57.0 (0.3)	53.8 (0.7)	46.8 (0.1)

C.4 Full results for RQ2

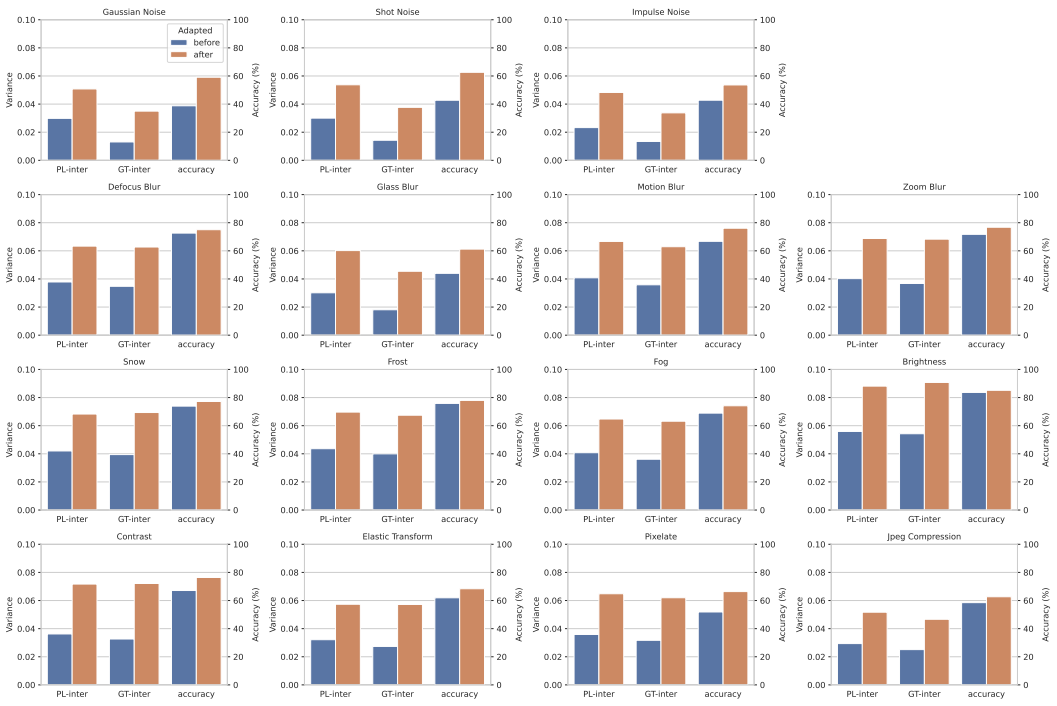

Figure 14: Variance collapse mitigation on CIFAR-10-C.

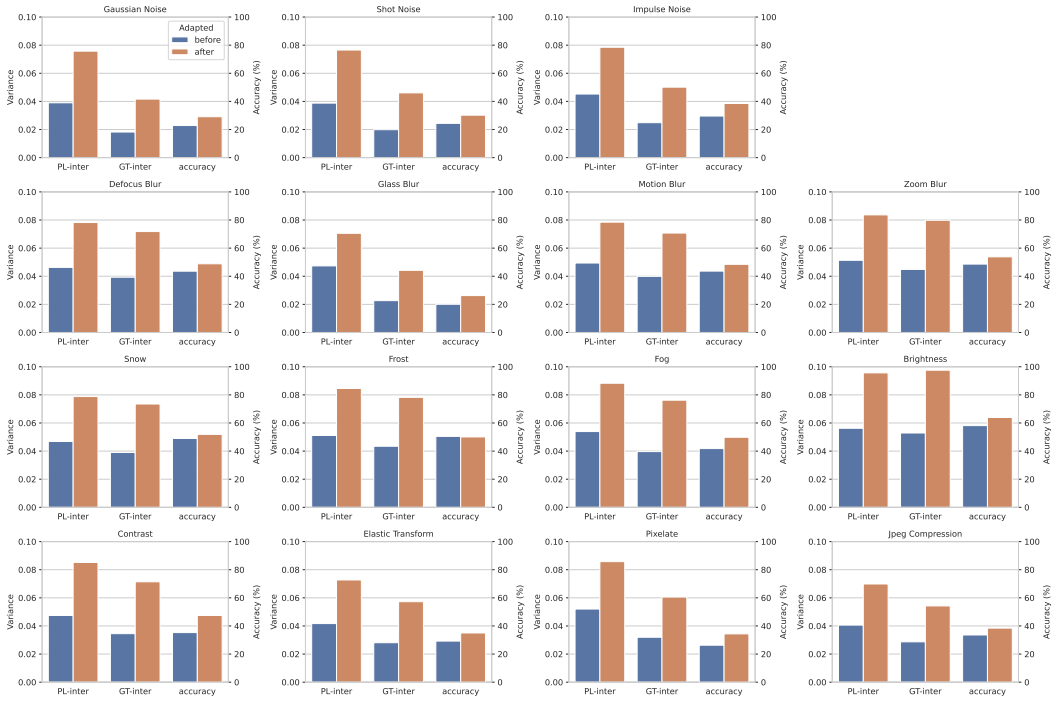

Figure 15: Variance collapse mitigation on CIFAR-100-C.

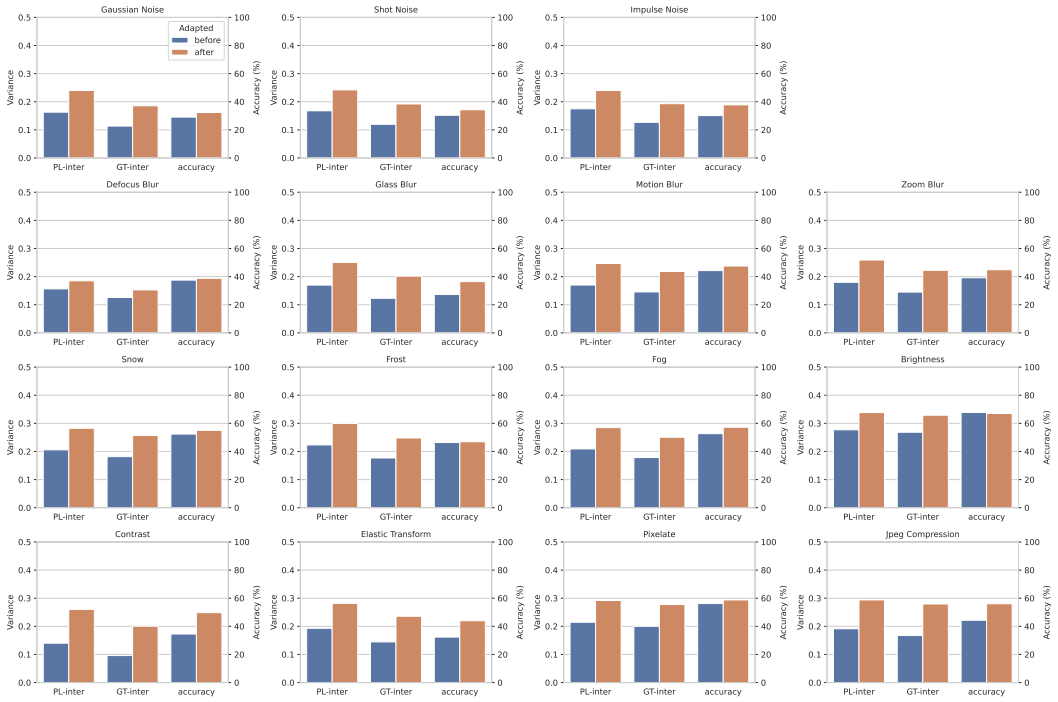

Figure 16: Variance collapse mitigation on ImageNet-C.

C.5 Ablation on layers to adapt

`Mint` adapts all *LayerNorm* layers in the vision encoder. Alternative choices include updating only the last block, the last MLP layer, or the first patching layer. Our comparison shows that updating *all LayerNorm* layers yields the best performance. For further discussions on which layers are most effective to adapt, we refer readers to related studies [23, 4].

Table 8: Comparison of different parts of the image encoder to update at test-time on CIFAR-10-C with ViT-B/32.

Layers to adapt	Accuracy (%)
Last block	63.4
Last MLP	62.9
Patching layer	63.6
All LayerNorm (`Mint`)	**71.0**

C.6 Hyperparameter sensitivity

In this subsection, we provide the results of hyperparameter sensitivity experiments on CIFAR-10-C and CIFAR-100-C.

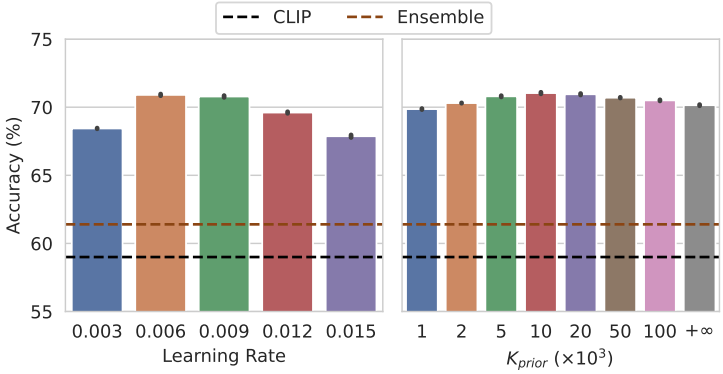

Figure 17: Hyperparameter sensitivity on CIFAR-10-C.

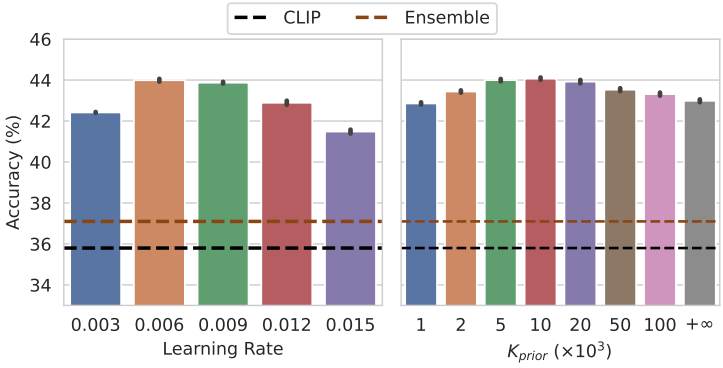

Figure 18: Hyperparameter sensitivity on CIFAR-100-C.

C.7 Experiments on clean datasets

In this subsection, we offer the comparison between `Mint` and currently outstanding baselines, CLIPArTT [15], WATT-S [28], TDA [19], DMN-ZS [45], Tent [38], and ETA [26], on clean (non-corruption) datasets with ViT-B/32 on CIFAR-10, ViT-B/16 on CIFAR-100, and ViT-L/14 on ImageNet.

Table 9: Accuracy (%) on clean datasets.

Method	ViT-B/32 on CIFAR-10	ViT-B/16 on CIFAR-100	ViT-L/14 on ImageNet
CLIP	88.3	68.4	73.0
CLIPArTT	89.1	70.2	72.1
WATT-S	89.8	72.3	74.5
TDA	89.6	70.1	73.4
DMN-ZS	90.2	69.4	73.1
Tent	91.1	72.2	73.4
ETA	91.4	73.0	73.6
`Mint`	**91.6**	**74.1**	**75.6**

C.8 Experiments on ImageNet variants

In this subsection, we provide the comparison between `Mint` and currently outstanding baselines, CLIPArTT [15], WATT-S [28], TDA [19], DMN-ZS [45], Tent [38], and ETA [26] in ImageNet variants (-A, -V2, -R, -Sketch) datasets with ViT-B/16.

Table 10: Accuracy (%) on ImageNet variants with ViT-B/16.

Method	ImageNet-A	ImageNet-V2	ImageNet-R	ImageNet-Sketch
CLIP	49.2	60.4	72.7	44.9
CLIPArTT	49.6	60.5	72.8	45.0
WATT-S	51.7	61.2	75.7	47.0
TDA	51.0	61.2	73.9	46.4
DMN-ZS	49.7	60.5	73.0	45.4
Tent	51.9	61.0	77.0	45.4
ETA	52.0	61.0	77.4	46.8
`Mint`	**54.7**	**62.6**	**78.1**	**48.4**

C.9 Mixture corruption datasets

In this subsection, we compare between `Mint` and currently outstanding baselines, CLIPArTT [15], WATT-S [28], TDA [19], and DMN-ZS [45], on mixture of 15 types of corruption datasets on CIFAR-10-C with ViT-B/32, CIFAR-100-C with ViT-B/16, and ImageNet-C with ViT-L/14. While the results in the main text are obtained by testing on each corruption type separately, here we first mix the data from all 15 corruptions together and then perform evaluation on this mixed-domain setting, following the setup in [27].

Table 11: Accuracy on Mixture of 15 Types of Corruptions.

Method	CIFAR-10-C	CIFAR-100-C	ImageNet-C
CLIP	59.0	35.8	39.6
TDA	62.1	38.3	42.3
DMN-ZS	60.2	36.0	39.9
WATT-S	63.6	39.0	43.9
CLIPArTT	56.9	38.7	40.5
`Mint`	**65.9**	**39.8**	**45.2**