New Generation Computing (2023) 41:5–24
https://doi.org/10.1007/s00354-022-00197-9

NEW GENERATION COMPUTING

Robustifying Vision Transformer Without Retraining from Scratch Using Attention-Based Test-Time Adaptation

Takeshi Kojima[1] · **Yusuke Iwasawa**[1] · **Yutaka Matsuo**[1]

Received: 2 September 2022 / Accepted: 24 November 2022 / Published online: 27 December 2022

Abstract
Vision Transformer (ViT) is becoming more and more popular in the field of image processing. This study aims to improve the robustness against the unknown perturbations without retraining the ViT model from scratch. Since our approach does not alter the training phase, it does not need to repeat computationally heavy pretraining of ViT. Specifically, we use test-time adaptation (TTA) for this purpose, which corrects its prediction during test-time by itself. The representative test-time adaptation method, Tent, is recently found to be applicable to ViT by modulating parameters and gradient clipping. However, we observed that Tent sometimes catastrophically fails, especially under severe perturbations. To stabilize the adaptation, we propose a new loss function called Attent, which minimizes the distributional differences of the attention entropy between the source and target. Experiments of image classification task on CIFAR-10-C, CIFAR-100-C, and ImageNet-C show that both Tent and Attent are effective on a wide variety of corruptions. The results also show that by combining Attent and Tent, the classification accuracy on corrupted data is further improved.

Keywords Vision transformer (ViT) · Test-time adaptation (TTA) · CIFAR-10-C · CIFAR-100-C · ImageNet-C · Attent

1 Introduction

Inspired by the success in natural language processing, e.g., BERT [1] and GPT [2, 3], transformer is becoming more and more popular in various image processing tasks including recognition [4, 5], object detection [6], and video processing [7, 8].

Notably, Dosovitskiy et al. [4] propose Vision Transformer (ViT), which adapts a pure transformer for image classification, shows that it achieves comparable or

✉ Takeshi Kojima
t.kojima@weblab.t.u-tokyo.ac.jp

[1] Graduate School of Engineering, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, 113-8656 Tokyo, Japan

superior performance to the conventional convolutional neural networks (CNN). Follow-up studies also show that ViT is more robust against the common corruptions and perturbations than convolution-based model (e.g., ResNet) [9, 10], which is an important property for safety-critical applications.

This paper seeks to answer the following question: *can we further improve the robustness of ViT without retraining from scratch*? Using heavy data augmentation during training is a natural way to improve the robustness, and prior works demonstrate that several data augmentation can indeed improve robustness of CNN [11, 12]. Another study empirically shows that sharpness-aware optimizer improves the robustness of ViT [13]. However, retraining ViT from scratch is not desirable, since it requires huge computational burden. Moreover, the dataset for the pretraining is sometimes not publicly available, making it impossible to retrain the model.

Test-time adaptation is a recently proposed approach for improving the robustness of the model [14–16]. In test-time adaptation, a model corrects its prediction of test data without looking at the label during test-time by modulating the small portion of model's parameters [typically parameters of Batch Normalization (BN) and/or its statistics]. For example, Wang et al. [16] proposes Tent, which modulates the parameters of Batch Normalization (BN) by minimizing prediction entropy, and shows that it can significantly improve the robustness of ResNet. This approach is favorable for our case as it does not alter the training phase and thus does not need to repeat the computationally heavy training of ViT. Recently, Kojima et al. [17] have shown that the existing representative TTA methods, including Tent, are also effective on ViT by modulating parameters in ViT. Specifically, the study demonstrated that, for each TTA method, updating only the affine transformation parameters within layer-normalization layers in ViT boosts the performance on target datasets including common corruptions and domain shift. Besides the parameter choice, Kojima et al. [17] found gradient clipping [18], which is not used in the original Tent paper [16], is essential for applying Tent to ViT to mitigate catastrophic failure. However, we observed that just minimizing prediction entropy of ViT, as with Tent, often causes catastrophic failure especially under severe distribution shifts.

In this paper, we design a new loss function, called Attent, to stabilize the test-time adaptation of ViT. Attent adapts to target data by minimizing distributional difference of attention entropy between source data and target data. Optimization is performed for each layer, head, and token on the target data to make their distributions go back to the source data. The distributional statistics of attention entropy on the source dataset are calculated and stored in memory beforehand. Therefore, we can use this approach without source dataset during adaptation.

In summary, our main contributions are as follows.

- This paper proposes a new test-time adaptation method for ViT. To mitigate the catastrophic failure of existing TTA method (Tent) on ViT, we introduce a new loss function called Attent. Attent minimizes the distributional differences of the attention entropy between the source and target in an online manner.
- Using multiple standard datasets to benchmark the robustness against common corruption and perturbations, namely CIFAR-10-C, CIFAR-100-C, and Ima-

geNet-C, we validate that robustness of ViT (ViT-B16 and ViT-L16) is improved by Attent without retraining the model from scratch.
- We show that the robustness is further improved by combining Attent and other TTA methods that do not require source information, e.g., Tent or SHOT-IM. Especially, the improvement is significant for more severe corruption that Tent alone cannot recover. In addition, Attent is less sensitive to hyperparameters, which is a favorable property in practical setting.

2 Related Works

2.1 Robustness of ViT

2.1.1 Transformer Architecture

Models based on Transformer architecture [19] achieve great performance not only in NLP but also in image processing as Vision Transformer [4]. Self-attention is one of the building blocks in Transformer. Let $z^l \in \mathbb{R}^{T \times D}$ be the input to l_{th} self-attention layer, where T is the number of tokens, and D is the number of features in each token. For each layer l and head h, the attention block takes z^l as input and compute attention weight matrix $A^{lh} \in \mathbb{R}^{T \times T}$. Specifically, the attention weights between position i and j, A_{ij}^{lh} are calculated based on the inner dot product between their respective query Q_i^{lh} and key K_j^{lh} representations as follows:

$$[Q^{lh}, K^{lh}, V^{lh}] = z_l W^{lh},\ W^{lh} \in \mathbb{R}^{D \times 3D_h} \tag{1}$$

$$A^{lh} = softmax\left(\frac{Q^{lh} K^{lh^T}}{\sqrt{D_h}}\right), \tag{2}$$

where W^{lh} is a learnable parameters. D_h is typically set to D/H to keep the number of parameters constant even when the number of head H is changed.

2.1.2 Inherent Robustness of ViT

Recent studies verify by experiments that ViT already has robustness without any adaptation or any additional data augmentation. Several studies empirically show that ViT is inherently more robust than CNNs using several benchmark datasets [9, 10]. The datasets include ImageNet-C (corruption), ImageNet-P (perturbations), ImageNet-R (semantic shifts), ImageNet-O (out-of-domain distribution), and ImageNet-9 (background dependence) [12, 20–22]. Other studies show that the robustness of ViT is further improved by changing the training strategy, such as using larger data set for pretraining phase [9, 23] or sharpness-aware optimizer for training phase [13].

2.2 Improving Robustness of CNN

2.2.1 Test-Time Adaptation

Test-time adaptation (TTA) is an online adaptation approach. Our proposal belongs to this category. Test-time adaptation does not require altering training process, so that this approach is generally applicable to wide variety of training method. This approach can adapt the model to the target data in a real-time mini-batch level to boost the accuracy in an online manner. Test-time batch normalization [14, 15] reestimates the statistics of the batch normalization [24] on the target dataset. Test-time entropy minimization (Tent) [16] adapts to the target unlabelled online data by minimizing the Shannon entropy [25] of its prediction by updating only the affine transformation parameters of batch normalization. Tent was proven to be effective when the model is CNN that has batch normalization, e.g., ResNet50 [26], while recent study [17] has demonstrated that Tent is also applicable to ViT (see Sect. 3.1 for the details). One can also use different loss functions for updating parameters, such as pseudo-label (PL) [27], diversity regularization (SHOT-IM) [28], feature alignment (TFA [29] and CFA [17]), or contrastive learning [30]. A recent study of Iwasawa et al. [31] has proposed gradient-free procedures to update only the classifier parameter of model (T3A). Our approach is categorized as a feature alignment approach, which minimizes the statistical distance between source and target dataset. It is worth noting that feature alignment approaches for test-time adaptation assume that one can access to the statistics on the source dataset during the test phase but does not need to access to the source dataset itself and to repeat the computationally heavy training [17]. Test-time adaptation generally assumes that the model would be distributed without source data due to bandwidth, privacy, or profit reasons [16]. We argue that the statistics of source data would be distributed even in such a situation, since it could drastically compress data size and eliminate sensitive information. In fact, some layers often used in typical neural networks contain statistics of source data (e.g., batch normalization) [17].

2.2.2 Data Augmentation

Several studies show that adding data augmentation naturally increases the robustness of model. For example, Hendrycks et al. [11] show that randomly selected noise operations and their compositions can improve the robustness against corruptions (AugMix). Hendrycks et al. [12] propose to make a variety of noises by passing training images into image-to-image network and introducing several perturbations during the forward pass, the output of which can be used as noisy training dataset (DeepAug). However, changing data augmentation usually needs to retrain the model from scratch, which is computationally heavy for the ViT.

2.2.3 Unsupervised Domain Adaptation

Several recent studies use unsupervised domain adaptation (UDA) [32–34] to improve the robustness of the CNN. For example, Xie et al. [35] propose

Noisy-Student, which trained the model with pseudo-labeled data [27] that come from the target distribution. Similarly, Rusak et al. [36] propose to use pseudo-label technique with robust classification loss, which is called Generalized Cross Entropy (GCE) [37], as an objective function. These works prove the usefulness of UDA to robustify the model; however, this approach needs to preassume the type of corruption, so that the unlabeled data from the target domain need to be available at the training time. Unlike UDA, our approach does not need to preassume the corruption type, which is a more practical setting.

2.3 Central Moment Discrepancy

To stabilize the adaptation, our method uses central mean discrepancy (CMD) [38] to measure the discrepancy between source and target. Let X and Y be bounded random samples with respective probability distributions p and q on the interval $[a;b]^N$. Formally, CMD is defined by

$$\begin{aligned} CMD_k = & \frac{1}{|b-a|}\|\mathbb{E}(X)-\mathbb{E}(Y)\|_2 \\ & + \frac{1}{|b-a|^k}\sum_{k=2}^{K}\|\mathbb{M}_k(X)-\mathbb{M}_k(Y)\|_2, \end{aligned} \tag{3}$$

where $\mathbb{E}(X) = \frac{1}{|X|}\sum_{x\in X} x$ is the empirical expectation vector computed on the sample X. Here, $|X|$ is denoted as the total number of samples following the annotation in [38]. $\mathbb{M}_k(X) = \mathbb{E}((x-\mathbb{E}(X))^k)$ is the vector that consists of kth-order central moments for each element of X. Y follows the same idea. Previous works focus on using CMD in the field of UDA to reduce the distributional gaps between source representations and target ones. However, CMD can also be potentially used for test-time adaptation, because CMD does not need to store the source dataset itself; instead, we store central moment statistics of source data in memory, and use it during online adaptation for moment matching between source and target.

3 Methodology

3.1 Tent for ViT

Recently, Kojima et al. [17] study applying existing TTA methods including Tent to ViT. Assume that we have pretrained model whose parameters are denoted by θ. The model is trained on clean source data, and need to predict the data with unknown corruption. Given the input image x, the model outputs the conditional probability $P(y \mid x;\theta)$. During test-time, Tent [16] minimizes the following prediction entropy using the stochastic gradient decent (SGD):

$$L_{\text{tent}} = \mathbb{E}\left[\sum_{c=1}^{C} -P_c(x;\theta)\log P_c(x;\theta)\right], \tag{4}$$

where C is the number of class, and P_c is the estimated probability that x belongs to y confidence of class c.

While updating entire θ is technically possible, it is known to be ineffective in the test-time adaptation. A key feature of Tent is that it does not alter the entire parameters θ, but it alters a small portion of the parameters $\psi \in \theta$. Specifically, Tent [16] updates a set of parameters related to affine transformation of batch normalization. However, recent large models including ViT do not have batch normalization. Therefore, we do not have trivial answer to the question: which parameters should we update when applying Tent to ViT models? According to Wang et al. [16], updating only feature modulations that are linear and low-dimensional leads to stability and efficiency for adaptation.

[17] has found that, to boost the performance of ViT on target datasets, it is stably effective to update only the affine transformation parameters within layer-normalization layers in ViT for each TTA method. The layer normalization in ViT reestimates the mean and standard deviation of the input across the dimensions of inputs itself, followed by the affine transformation for each dimension. The notable difference between the modulation of layer normalization and batch normalization is that LayerNorm does not need to calculate the mean and standard deviation across multiple samples for normalization, i.e., layer normalization automatically reestimates the mean and standard deviation across dimensions on every single target data itself. Therefore, we only need to update the affine transformation parameters in layer normalization for adaptation. In addition, Kojima et al. [17] demonstrated that all the parameters can be updated with the best performance improvement when we use appropriate loss function, such as SHOT-IM or CFA. We experimentally find the best set of parameters for our case (Attent) in Sect. 4.5.

Besides the parameter choice, Kojima et al. [17] found that gradient clipping [18], which is not used in the original paper [16], is essential for applying Tent into ViT. Specifically, Kojima et al. [17] clip the gradients whose norm is greater than 1.0 in the entire experiments. Whether these techniques are unique to ViT (or huge models) needs further investigation, but without these techniques, Tent often gave catastrophic failure (significant drop of classification accuracy). See Sect. 4.7 for details.

3.2 Attent

Tent is sometimes unstable during adaptation even when we implement several techniques described in Sect. 3.1 (see Table 4 for details). The reason why Tent is unstable is that the objective function is just minimizing the entropy of classification. One of the extreme solution to this function is always assigning 100% probability to only one class, which indicates catastrophic failure of adaptation.

To alleviate this problem, we propose a new approach for test-time adaptation called Attent. In this approach, we focus on attention mechanism in ViT, which is one of the critical architectures for correct prediction [9, 39]. We hypothesize that the inner dot

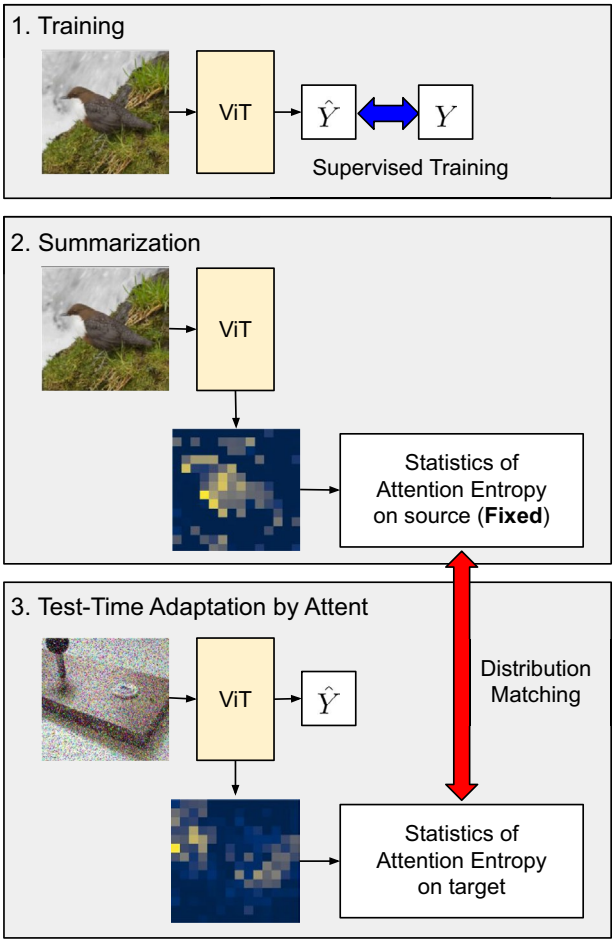

Fig. 1 Overview of our method (Attent). Similar to Liu et al. [29], our method consists of three stages: model training, offline statistics summarization, and online test-time adaptation. (1) ViT is trained in a supervised manner on a labeled source dataset. (2) After training, statistics (mean and higher order moments) of attention entropy on source dataset is calculated and stored in memory as fixed value. (3) During test-time, label is predicted, while partial parameters in ViT are updated by distribution matching of attention entropy between source and target dataset

product between Q and K for the attention (Eq. 2) is shifted if ViT takes target data as input whose distribution is different from that of source data. Consequently, the attention weight distribution is shifted in an unexpected way. Our concept idea is to make the anomalous distribution goes back to normal by distribution matching of attention entropy between source and target data (see Fig. 1 for method overview). Simply minimizing attention entropy would fail, which may lead to paying attention to extremely narrow areas (see Sect. 4.5 for the experiment result). Similar to the most prior works, our method uses stochastic gradient decent to adapt the model during test-time. Unlike the prior methods, such as Tent [16], PL [27], and T3A [31] that modulate the

parameters only using the data available at test-time, our method aligns the statistics of features between source and target. In other words, we leverage the source statistics as an auxiliary information regarding the source distribution to avoid catastrophic failure.

Let L, H, T be the number of layer, the number of head, and the number of tokens in transformer. Given a sample data from Source dataset $x_n^{\mathcal{S}} \in X^{\mathcal{S}}$, attention entropy can be calculated for each layer $l \in L$, head $h \in H$, and tokens $i \in T$. Specifically, following some annotations from Sect. 2.1.1, we define $A^{lh}(x_n^{\mathcal{S}}; \bar{\theta}) \in \mathbb{R}^{T \times T}$ as the attention weight matrix parameterized by parameters $\bar{\theta}$. $\bar{\theta}$ are the parameters just after the training on source dataset, i.e., before adaptation. The attention entropy on source data sample is defined as follows:

$$\mathcal{H}_{lhi}^{\mathcal{S}n} = \sum_{j=1}^{T} -A_{ij}^{lh}(x_n^{\mathcal{S}}; \bar{\theta}) \log A_{ij}^{lh}(x_n^{\mathcal{S}}; \bar{\theta}). \tag{5}$$

The mean and kth-order central moments of the entropy for source data are calculated by the following form and stored in memory as fixed values:

$$\mathbb{E}(\mathcal{H}_{lhi}^{\mathcal{S}}) = \frac{1}{|X^{\mathcal{S}}|} \sum_{x_n^{\mathcal{S}} \in X^{\mathcal{S}}} \mathcal{H}_{lhi}^{\mathcal{S}n}, \tag{6}$$

$$\mathbb{M}_k(\mathcal{H}_{lhi}^{\mathcal{S}}) = \frac{1}{|X^{\mathcal{S}}|} \sum_{x_n^{\mathcal{S}} \in X^{\mathcal{S}}} (\mathcal{H}_{lhi}^{\mathcal{S}n} - \mathbb{E}(\mathcal{H}_{lhi}^{\mathcal{S}}))^k, \tag{7}$$

where $k = 2, \ldots, K$. Attent uses these statistics to adapt the model during test phase.

During test-time phase, assume that a sequence of test data drawn from target distribution arrives at our model one after another. A set of test samples in mth batch is denoted as $X_m^{\mathcal{T}} \subset X^{\mathcal{T}}, m = 1, \ldots, M$. For each sample in each batch $x_{mn}^{\mathcal{T}} \in X_m^{\mathcal{T}}$, the attention entropy on target data is calculated like the first phase

$$\mathcal{H}_{lhi}^{\mathcal{T}mn} = \sum_{j=1}^{T} -A_{ij}^{lh}(x_{mn}^{\mathcal{T}}; \theta) \log A_{ij}^{lh}(x_{mn}^{\mathcal{T}}; \theta). \tag{8}$$

The mean and higher order central moments of the entropy are calculated for each target batch data

$$\mathbb{E}(\mathcal{H}_{lhi}^{\mathcal{T}m}) = \frac{1}{|X_m^{\mathcal{T}}|} \sum_{x_{mn}^{\mathcal{T}} \in X_m^{\mathcal{T}}} \mathcal{H}_{lhi}^{\mathcal{T}mn}, \tag{9}$$

$$\mathbb{M}_k(\mathcal{H}_{lhi}^{\mathcal{T}m}) = \frac{1}{|X_m^{\mathcal{T}}|} \sum_{x_{mn}^{\mathcal{T}} \in X_m^{\mathcal{T}}} (\mathcal{H}_{lhi}^{\mathcal{T}mn} - \mathbb{E}(\mathcal{H}_{lhi}^{\mathcal{T}m}))^k. \tag{10}$$

Note that for online adaptation, we can only use real-time batch data at hand. Let $\mathbb{E}(\mathcal{H}_{lh})$ be the concatenation of $\{\mathbb{E}(\mathcal{H}_{lhi}) \mid i \in T\}$ alongside the last dimension. Similarly, Let $\mathbb{M}_k(\mathcal{H}_{lh})$ be the concatenation of $\{\mathbb{M}_k(\mathcal{H}_{lhi}) \mid i \in T\}$ alongside the last

dimension. A loss function for test-time adaptation is defined by applying the aforementioned central moments to CMD formula (Eq. 3)

$$\mathcal{L}_{attn}^{lh} = \frac{1}{\log T}\|\mathbb{E}(\mathcal{H}_{lh}^{\mathcal{T}m}) - \mathbb{E}(\mathcal{H}_{lh}^{\mathcal{S}})\|_2 + \frac{1}{(\log T)^k}\sum_{k=2}^{K}\|\mathbb{M}_k(\mathcal{H}_{lh}^{\mathcal{T}m}) - \mathbb{M}_k(\mathcal{H}_{lh}^{\mathcal{S}})\|_2, \tag{11}$$

$$\mathcal{L}_{attn} = \frac{1}{LH}\sum_{l=1}^{L}\sum_{h=1}^{H}\mathcal{L}_{attn}^{lh}. \tag{12}$$

The maximum and minimum values of the attention entropy with T tokens are calculated as $\log T$ and 0. Therefore, $|b - a| = \log T$ in Eq. 11. Like Tent, we update only few parameters in ViT by SGD to stabilize adaptation, which is detailed in Sect. 4.5.

Finally, this objective function (Eq. 12) can be used alone or in combination with Tent (Eq. 4). We can optionally combine Attent and Tent loss functions as follows:

$$\mathcal{L}_{\text{mix}} = \mathcal{L}_{\text{tent}} + \lambda\mathcal{L}_{\text{attn}}, \tag{13}$$

where λ is a balancing hyperparameter. Following Wang et al. [16], the parameter update follows the prediction for the current batch. Therefore, the parameter update only affects the next batch. The adaptation procedure is summarized in Algorithm 1.

Algorithm 1 Online Adaptation using Attent

Input: Fine-tuned DNN model with parameters θ, and partial parameters to be updated during adaptation $\psi \subset \theta$, Target test dataset $X^{\mathcal{T}}$, m-th ordered batch data $X^{\mathcal{T}m} \subset X^{\mathcal{T}}$, Statistics of Eq. 6 and Eq. 7 calculated from the source training dataset.

Output:

1: **for** $m = 1$ to M **do**
2: Predict labels $\hat{Y}^{\mathcal{T},m}$ for $X^{\mathcal{T},m}$
3: Calculate statistics of Eq. 9 and Eq. 10 for $X^{\mathcal{T},m}$
4: Update $\psi \subset \theta$ based on Eq. 12
5: **end for**
6: **return** $(\hat{Y}^{\mathcal{T},1}, .., \hat{Y}^{\mathcal{T},M})$

4 Experiment

We show that our approach is effective for corruption using CIFAR-10-C, CIFAR-100-C, and ImageNet-C datasets, respectively. We run all the experiments three times with different seeds for different data order shuffling. A mean and unbiased

standard deviation of the metric is reported. Our implementation is in PyTorch [40], and every experiment is run on cloud A100 × 1GPU instance except for ViT-L16 on A100 × 2GPU instance.

4.1 Dataset and Preprocess

CIFAR-10/CIFAR-100 [41] and ImageNet [42] are used as source datasets. CIFAR-10/CIFAR-100 are, respectively, 10-class/100-class color image datasets including 50,000 training data and 10,000 test data with 32×32 resolution. ImageNet is a 1000-class image dataset with more than 1.2 million training data and 50,000 validation data with various resolution.

CIFAR-10-C/CIFAR-100-C and ImageNet-C are used as target datasets for test-time adaptation [20]. These datasets, respectively, contain data with 15 types of corruption with 5-level severity. Therefore each dataset has 75 varieties of corruptions in total. Each corrupted image is created based on image from original CIFAR-10 and CIFAR-100 test images, and ImageNet validation images. Therefore, the CIFAR-10-C/CIFAR-100-C consists 10,000 images for each corruption/type and ImageNet-C dataset consists 50,000 images for each corruption/type.

For this experiment, images of all the datasets are preprocessed, so that the images are resized uniformly to 224×224. As for ImageNet, some images are rectangle, so all the images are resized to 256 and center-cropped with 224×224 size. ImageNet-C data have been already preprocessed the same way and publicly available. For ViT models, the pixels are first rescaled from [0, 255] to [0, 1]. Then, they are further rescaled to $[-1, 1]$ by normalization with mean and std specified as [0.5, 0.5, 0.5] and [0.5, 0.5, 0.5].

4.2 Model and Training Setting

Vision Transformer (ViT) is used as a model for this experiment. For CIFAR-10 and CIFAR-100, we use ViT-B16 as initial parameters for fine-tuning. ViT-B16 is already pretrained on ImageNet-21K [43], which is a large dataset with 21k classes and 14M images. For fine-tuning hyperparameters, following [4], we use batch size of 512, set optimizer as SGD with momentum 0.9 and gradient clipping at global norm 1.0. We choose a learning rate of 0.03. We apply cosine schedule of 200 warmup steps and the total number of iteration as 1000 for CIFAR-10. We apply a cosine schedule of 500 warmup steps and the total number of iterations as 2000 for CIFAR-100. The fine-tuning result is 1.1% Top-1 error for CIFAR-10, 6.8% for CIFAR-100, respectively. For ImageNet, we use parameters of ViT-B16 and ViT-L16 that are already pretrained for ImageNet-21K and also fine-tuned for ImageNet (-2012). Therefore, fine-tuning on ImageNet is not needed. The top-1 error on ImageNet is 18.6% by ViT-B16 and 17.1% by ViT-L16, respectively, in our setting. ViT-B16 has 12 layers and 12 heads. ViT-L16 has 24 layers and 16 heads [4]. Both models have input patch size of 16, so that the number of tokens for ViT is defined as 196 ($= 14 \times 14$) in this experiment.

Table 1 Modulation parameter study

	Tent	Attent	Attent + Tent
LN	**50.6 ± 0.5**	51.7 ± 0.0	**47.6 ± 0.8**
CLS	59.4 ± 0.0	61.4 ± 0.0	59.2 ± 0.0
ALL	59.1 ± 1.0	**50.1 ± 0.0**	51.2 ± 1.5

Evaluation metric is top-1 error on ImageNet-C for Gaussian Noise with the highest severity. ViT-B16 is used as a model. As a reference, the performance without any adaptation ("Source") was 61.9 ± 0.0

CLS CLS token (CLS token is a parameterized vector and proven to be efficient for fine-tuning large models for downstream tasks in NLP [45]), *LN* LayerNorm, *ALL* all the parameters of ViT

4.3 Adaptation Setting

Before adaptation, we need to calculate the central moments of attention entropies for source data (Eqs. 6 and 7) to store them in memory. For this purpose, we use all the training data in source and set the dropout [44] off during the calculation.

As default hyperparameters for adaptation on target data, batch size is set to 64, optimization is SGD with constant learning rate 0.001 and momentum 0.9 with gradient clipping 1.0, and the maximum central moments' order K is set to 3 across all the experiments. Dropout is set off during both forward and backward pass. We set $\lambda = 1.0$ in Eq. 13 to balance Tent and Attent losses. The detailed hyperparameter sweeping results are found at Sect. 4.7.

Top-1 error of classification is used as evaluation metric across all the experiments.

4.4 Baseline Methods

We compare Attent with some existing baseline test-time adaptation methods that do not need to alter training phase: *Tent* [16], *PL* [27], *TFA(-)* [29][1] , *T3A* [31], *SHOT-IM* [28], and *CFA* [17]. In addition, we report the performance of the model on target datasets without any adaptation as *Source*. T-BN [14, 15] is excluded from the baseline, because ViT does not have a batch normalization layer. For a fair comparison, we use the same hyperparameter values across all the methods as described in Sect. 4.3. The detailed setting for each method is described in Appendix 1.

[1] Original TFA [29] needs to alter training phase (add contrastive learning), while this study focuses on robustifying large-scale models without retraining them from scratch. Therefore, we have changed some of the settings from the original TFA, so that the model does not need to alter training phase. The modified version of TFA is denoted as TFA(-) in our experiments. See Appendix 1 for details.

Table 2 Method comparison

Method	C10→C10-C	C100→C100-C	IN→IN-C
Source	14.6 ± 0.0	35.1 ± 0.0	61.9 ± 0.0
T3A	13.7 ± 0.0	34.0 ± 0.0	61.2 ± 0.0
TFA(-) *1	8.8 ± 0.0	32.2 ± 0.2	57.8 ± 0.1
PL	11.9 ± 0.0	30.1 ± 0.5	55.7 ± 1.4
Tent	10.9 ± 0.2	27.4 ± 0.5	50.6 ± 0.5
Attent	**10.3 ± 0.0**	**28.8 ± 0.0**	**51.7 ± 0.0**
Attent + Tent	**9.6 ± 0.1**	**26.5 ± 0.0**	**47.6 ± 0.8**
Attent – CMD	25.7 ± 0.2	51.5 ± 0.2	88.7 ± 0.1
Tent + CMD	11.4 ± 0.3	29.8 ± 0.3	51.9 ± 0.6
SHOT-IM	8.9 ± 0.0	25.6 ± 0.0	45.7 ± 0.0
Attent + SHOT-IM	**8.9 ± 0.0**	**25.5 ± 0.0**	**45.1 ± 0.0**
TFA(-) *2	8.7 ± 0.0	25.4 ± 0.0	46.7 ± 0.0
Attent + TFA(-) *2	**8.6 ± 0.0**	**25.3 ± 0.0**	**46.5 ± 0.0**
CFA	8.4 ± 0.0	24.6 ± 0.1	43.9 ± 0.0
Attent + CFA	**8.4 ± 0.0**	**24.5 ± 0.1**	**43.4 ± 0.0**

Top-1 error on CIFAR-10-C/CIFAR-100-C/ImageNet-C averaged over 15 corruption types for the highest severity. ViT-B16 is used as a model. TFA(-) *1 uses hyperparameters of $\beta_1 = 1, \beta_2 = 1$, while TFA(-) *2 uses $\beta_1 = 1, \beta_2 = 0$. See Appendix 1 for the details of the hyperparameters

Bold values indicate points to be highlighted in the table

4.5 Quantitative Result

Table 1 answers the question about which modulation parameters are the most suitable for improving the performance of Attent. The results indicate that layer-normalization parameters can stably reduce top-1 error for both Attent and Tent. Therefore, in all the subsequent experiments, layer-normalization parameters are updated for adaptation across all the methods for fair comparison.

Table 2 summarizes the adaptation result (top-1 error) on CIFAR-10-C, CIFAR-100-C, and ImageNet-C for our method and other existing test-time adaptation baselines. We measure the top-1 error on each dataset with the highest severity (= 5). It is verified that Attent can stably reduce the error across all the datasets. Tent alone also has the stronger improvement gain than Attent. However, it is found that Tent is sometimes unstable on some of the corruptions with higher severity, while Attent is not. See Table 4 for details. For example, in Table 4, Tent on ViT-B16 fails to adapt to "snow" corruption on ImageNet-C, causing the significant performance drop from 60.9 to 75.7% top-1 error rate. Interestingly, combining Tent and Attent improves the performance more while preventing the aforementioned significant performance drop (See "Attent + Tent" at Table 2). We observe similar performance gains when combining Attent with other existing high-performance methods, such as SHOT-IM (whose objective is classifier entropy minimization + diversity regularizer), TFA(-) and CFA (whose objectives are minimizing distribution differences of the hidden

Table 3 Model and corruption severity study

Model	*S*	Source	Tent	Attent	Attent +Tent
ResNet50	1	39.4 ± 0.0	**31.6 ± 0.1**	–	–
	2	50.2 ± 0.0	**37.1 ± 0.1**	–	–
	3	60.2 ± 0.0	**41.7 ± 0.0**	–	–
	4	72.3 ± 0.0	**49.3 ± 0.1**	–	–
	5	82.0 ± 0.0	**59.0 ± 0.0**	–	–
ViT-B16	1	26.0 ± 0.0	22.8 ± 0.0	25.1 ± 0.0	**22.6 ± 0.0**
	2	32.3 ± 0.0	27.1 ± 0.0	30.1 ± 0.0	**26.8 ± 0.0**
	3	37.6 ± 0.0	30.5 ± 0.1	33.9 ± 0.0	**30.0 ± 0.0**
	4	47.9 ± 0.0	38.4 ± 0.2	41.3 ± 0.0	**37.2 ± 0.5**
	5	61.9 ± 0.0	50.6 ± 0.5	51.7 ± 0.0	**47.6 ± 0.8**
ViT-L16	1	23.5 ± 0.0	**21.2 ± 0.0**	23.2 ± 0.0	**21.2 ± .0**
	2	28.8 ± 0.0	25.3 ± 0.0	28.1 ± 0.0	**25.2 ± 0.0**
	3	32.8 ± 0.0	27.9 ± 0.0	31.6 ± 0.0	**27.8 ± 0.0**
	4	41.0 ± 0.0	33.6 ± 0.1	38.5 ± 0.0	**33.5 ± 0.0**
	5	53.4 ± 0.0	42.3 ± 0.0	48.8 ± 0.0	**42.1 ± 0.0**

Top-1 error based on ImageNet-C averaged over 15 corruption types for each severity

S severity

Bold values indicate points to be highlighted in the table

representation just before the classifier of a model between the source and target). See "Attent + SHOT-IM", "Attent + TFA(-)", and "Attent + CFA" in Table 2.

To investigate the effectiveness of our approach, we introduce the following two analyses focusing on CMD. As a first analysis, instead of optimizing attention entropy in Attent by CMD, we simply minimize the attention entropy in the same way that Tent minimizes prediction entropy. This approach is described as "Attent - CMD" in Table 2. The result shows that this approach significantly deteriorates the performance. It indicates that matching distribution is necessary for attention entropy control instead of minimization. As a second analysis, we optimize the classification entropy distribution in Tent using CMD to make it go back to that of source dataset. This approach is described as "Tent + CMD" in Table 2. The result shows that the performance slightly deteriorates compared to the original Tent. It indicates that CMD is not appropriate for the classifier entropy.

Table 3 summarizes the adaptation results on ImageNet-C with severity from 1 to 5 based on various backbone networks: ResNet50, ViT-B16, and ViT-L16. The ResNet50 result is included as a reference to see the performance difference between a regular ResNet model and ViT models. Following [16], Tent for ResNet50 updates batch normalization parameters and reestimates the statistics. It is verified that both Tent and Attent consistently improve the performance across all the models for various degree of corruption severity. This indicates that Tent and Attent are model agnostic. The experiment also demonstrated that Attent + Tent further boosts the performance regardless of network backbones and corruption severity.

Table 4 Detail experiment results of method comparison

Method	Gaussian	Shot	Impulse	Defocus
Source	77.7 ± 0.0	75.1 ± 0.0	77.0 ± 0.0	66.9 ± 0.0
Tent	66.3 ± 8.8	77.8 ± 1.6	59.3 ± 0.2	50.9 ± 0.2
Attent	62.7 ± 0.1	**60.4 ± 0.1**	61.5 ± 0.1	56.1 ± 0.0
Attent + Tent	**59.0 ± 0.1**	63.8 ± 6.9	**58.1 ± 0.0**	**50.2 ± 0.1**
Method	Glass	Motion	Zoom	Snow
Source	69.1 ± 0.0	58.5 ± 0.0	62.8 ± 0.0	60.9 ± 0.0
Tent	49.8 ± 0.2	46.7 ± 0.0	50.9 ± 0.2	75.7 ± 2.0
Attent	57.0 ± 0.1	51.2 ± 0.1	53.3 ± 0.0	**53.9 ± 0.1**
Attent + Tent	**48.8 ± 0.1**	**45.9 ± 0.0**	**49.1 ± 0.2**	62.5 ± 5.5
Method	Frost	Fog	Brightness	Contrast
Source	57.6 ± 0.0	62.9 ± 0.0	31.6 ± 0.0	88.9 ± 0.0
Tent	48.2 ± 0.4	44.3 ± 0.3	26.1 ± 0.1	58.5 ± 0.3
Attent	50.8 ± 0.0	51.2 ± 0.1	29.7 ± 0.0	62.8 ± 0.2
Attent + Tent	**46.7 ± 0.0**	**43.2 ± 0.2**	**25.8 ± 0.1**	**57.8 ± 0.2**
Method	Elastic	Pixelate	Jpeg	Average
Source	51.9 ± 0.0	45.3 ± 0.0	42.9 ± 0.0	61.9 ± 0.0
Tent	37.6 ± 0.3	32.7 ± 0.1	34.7 ± 0.1	50.6 ± 0.5
Attent	44.8 ± 0.1	39.7 ± 0.1	40.1 ± 0.1	51.7 ± 0.0
Attent + Tent	**36.8 ± 0.2**	**32.1 ± 0.1**	**34.5 ± 0.1**	**47.6 ± 0.8**

Evaluation metric is top-1 error based on ImageNet-C for each corruption type with the highest severity. ViT-B16 is used as a model

Bold values indicate points to be highlighted in the table

4.6 Attention Map Reconstruction

We quantitatively analyze how close the attention map gets after adaptation compared to the before. Specifically, for each sample, we measure cross entropy of attention maps between corrupted image $x^{\mathcal{T}}$ in ImageNet-C and the corresponding clean image $x^{\mathcal{S}\leftarrow\mathcal{T}}$ in ImageNet by the following formulas:

$$\mathcal{H}^{mn}_{\text{Forward}} = \frac{1}{LHT}\sum_{l,h,i,j} -A^{lh}_{ij}(x^{\mathcal{S}\leftarrow\mathcal{T}}_{mn};\bar{\theta}) \log A^{lh}_{ij}(x^{\mathcal{T}}_{mn};\theta_m), \tag{14}$$

$$\mathcal{H}^{mn}_{\text{Reverse}} = \frac{1}{LHT}\sum_{l,h,i,j} -A^{lh}_{ij}(x^{\mathcal{T}}_{mn};\theta_m) \log A^{lh}_{ij}(x^{\mathcal{S}\leftarrow\mathcal{T}}_{mn};\bar{\theta}), \tag{15}$$

where θ_m is a set of parameters at the time of mth batch during test-time adaptation. Note that $\theta_m = \bar{\theta}$ if without adaptation (Source). Intuitively, Eq. 14 gives us a higher penalty if the attention does not focus on correct locations, while Eq. 15 gives us a higher penalty if the attention focuses on wrong locations. We combine the two

cross entropies and take the average across all the test data, making it the evaluation metric

$$\mathcal{H} = \frac{1}{|X^{\mathcal{T}}|} \sum_{x^{\mathcal{T}}_{mn} \in X^{\mathcal{T}}} \left(\mathcal{H}^{mn}_{\text{Forward}} + H^{mn}_{\text{Reverse}} \right). \tag{16}$$

The lower score of Eq. 16 indicates the better attention map reconstruction.

Table 5 summarizes the result of measuring the metric using images from ImageNet and ImageNet-C of 15 corruptions with highest severity. The result demonstrates that Attent has the most tendency of reconstructing the attention map to the original one. This tendency may cause the improvement of image classification accuracy. Tent also has a tendency of reconstructing attention map, but not as much as Attent, which implies that the performance improvement by Tent is related to other latent variables as well as attention map. The score of Attent + Tent is between Tent and Attent. It can be assumed that attention map and other latent variables are optimized at the same time. In the case of Snow corruption, Tent fails in Adaptation and performance deteriorates substantially (see Table 4), but at the same time, the score of attention map reconstruction also deteriorates significantly (see Table 5), indicating the importance of attention map.

4.7 Hyperparameter Sensitivity

For online adaptation, hyperparameter tuning is a challenging issue. Figure 2 shows the results for each hyperparameter sensitivity on ImageNet-C with highest severity averaged over 15 corruption types. We check the following hyperparameters by changing one of the values from the default described at Sect. 4.3: (a) learning rate, (b) batch size, (c) maximum number of central moments K in Eq. 11, and (d) whether to enable gradient clipping for SGD optimization. $K = 1$ denotes first-order moment (mean) matching only, which ignores higher order moment matching, i.e., Eqs. 7 and 10.

The important finding is that Tent is more sensitive to some hyperparameters described above than Attent. Especially, enabling gradient clipping is essential for applying Tent into ViT models for avoiding catastrophic failure of adaptation. Furthermore, large learning rate also leads to catastrophic failure of Tent. In contrast, Attent is quite insensitive to each hyperparameters. This indicates that we can use Attent safely in the unknown environment with rough hyperparameter tuning. It is also shown that the order of central moments K improves the performance of Attent, but the gain decreases as K gets larger. This is consistent with the original CMD study [38], which states that the performance is similar when $K \geq 3$.

Table 5 The result of measuring attention map reconstruction metric based on Eq. 16

Method	Gaussian	Shot	Impulse	Defocus
Source	10.92 ± 0.00	10.79 ± 0.00	10.93 ± 0.00	10.22 ± 0.00
Tent	10.69 ± 0.00	10.68 ± 0.02	10.73 ± 0.03	10.08 ± 0.00
Attent	**10.64 ± 0.00**	**10.54 ± 0.00**	**10.64 ± 0.00**	**9.92 ± 0.00**
Attent + Tent	10.66 ± 0.00	10.56 ± 0.00	10.66 ± 0.00	9.94 ± 0.00
Method	Glass	Motion	Zoom	Snow
Source	10.05 ± 0.00	10.20 ± 0.00	10.20 ± 0.00	10.30 ± 0.00
Tent	9.92 ± 0.00	10.15 ± 0.00	10.25 ± 0.00	10.43 ± 0.01
Attent	**9.79 ± 0.00**	**10.03 ± 0.00**	**10.10 ± 0.00**	**10.16 ± 0.00**
Attent + Tent	9.81 ± 0.01	10.05 ± 0.00	10.14 ± 0.00	10.2 ± 0.01
Method	Frost	Fog	Brightness	Contrast
Source	10.09 ± 0.00	10.28 ± 0.00	9.22 ± 0.00	11.14 ± 0.00
Tent	10.14 ± 0.02	10.06 ± 0.00	9.20 ± 0.01	10.58 ± 0.00
Attent	**10.02 ± 0.00**	10.11 ± 0.00	9.13 ± 0.00	**10.39 ± 0.00**
Attent + Tent	10.03 ± 0.00	**10.00 ± 0.00**	**9.12 ± 0.00**	10.43 ± 0.00
Method	Elastic	Pixelate	Jpeg	Average
Source	9.55 ± 0.00	9.20 ± 0.00	9.54 ± 0.00	10.18 ± 0.00
Tent	9.59 ± 0.00	9.05 ± 0.00	**9.44 ± 0.00**	10.06 ± 0.00
Attent	**9.54 ± 0.00**	9.06 ± 0.00	9.46 ± 0.00	**9.97 ± 0.00**
Attent + Tent	9.55 ± 0.00	**9.01 ± 0.00**	**9.44 ± 0.00**	**9.97 ± 0.00**

The value indicates how close the attention map is between clean and corresponding corrupted image. Images are used from ImageNet and ImageNet-C of 15 corruptions with highest severity. ViT-B16 is used as a model

Bold values indicate points to be highlighted in the table

5 Conclusion and Future Work

This study proposed a novel method of test-time adaptation for ViT, called Attent, which adapts ViT by minimizing the distributional differences of the attention entropy between the source and target during test-time. Experiments on CIFAR-10-C, CIFAR-100-C, and ImageNet-C show that Attent is effective on various ViT models. By combining Attent and other TTA methods, the robustness is further improved. As a limitation, Attent is not effective for some of the domain adaptation benchmarks, such as digits style shift; e.g., from SVHN to MNIST/MNIST-M. Future work includes improving our method to adapt well on these tasks. We hope that research of test-time adaptation on ViT will be further encouraged by this study.

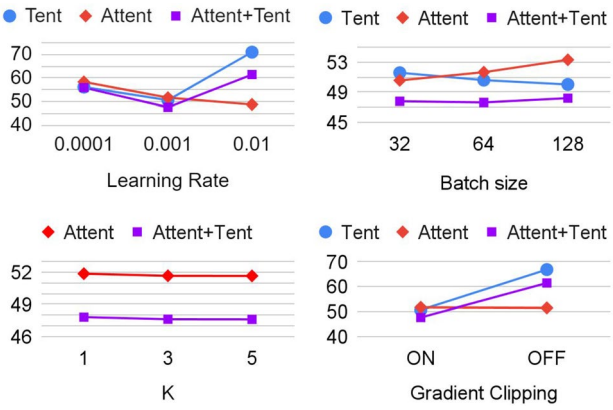

Fig. 2 The effect of sweeping hyperparameters on each method. The evaluation metric is top-1 error on ImageNet-C averaged over 15 corruption types with highest severity. ViT-B16 is used as model. Either one of the hyperparameter values is changed from the default described in Sect. 4.3

Appendix A: Detail Settings of Baseline Methods

A.1. TFA(-)

Test-time feature alignment (TFA) [29] aligns the hidden representation on target data by minimizing the distance of the mean vector $\mu^s, \mu^t \in \mathbb{R}^D$ and covariance matrix $\Sigma^s, \Sigma^t \in \mathbb{R}^{D \times D}$ between source and target. D is the dimension size of the hidden representation. We focus only on the "Online Feature Alignment" part in TTT++ [29]. Original TFA [29] aligns the distributions at both the hidden representation and the output of the self-supervised head. However, in our experiment, TFA(-) does not employ self-supervised learning, so we only focus on distribution matching of the hidden representation. Specifically, in this experiment, the hidden representation to align is defined as the one before the classifier head $h(x) = f(x;\phi)$. The loss function is $\mathcal{L} = \beta_1 \|\mu^s - \mu^t\|_2^2 + \beta_2 \|\Sigma^s - \Sigma^t\|_F^2$, where $\|\cdot\|_2$ is the Euclidean norm and $\|\cdot\|_F$ is the Frobenius norm. μ and Σ are, respectively, mean vector and covariance matrix. β_1 and β_2 are balancing hyperparameters. Like CFA [17], TFA(-) calculates the statistics on source dataset and store them in memory before adaptation. Note that "Online Dynamic Queue" [29] is not used in TFA(-) in our experiment. Table 6 describes the preliminary experiment results of TFA(-) on ImageNet-C datasets with severity = 5 by changing the balancing hyperparameters β_1, β_2. For the main experiment in Table 2, we use default hyperparameter $\beta_1 = 1, \beta_2 = 1$ based on [29], as well as the best performance hyperparameter $\beta_1 = 1, \beta_2 = 0$.

Table 6 Preliminary experiment results of TFA(-) on ImageNet-C by changing the balancing hyperparameters β_1, β_2. Evaluation metric is Top-1 error on ImageNet-C averaged over 15 corruption types with severity level=5. ViT-B16 is used as a model

Method	ImageNet-C
TFA(-) ($\beta_1 = 1, \beta_2 = 1$)	57.7 ± 0.1
TFA(-) ($\beta_1 = 1, \beta_2 = 1/D$)	48.8 ± 0.0
TFA(-) ($\beta_1 = 1, \beta_2 = 0$)	46.7 ± 0.0
TFA(-) ($\beta_1 = 0, \beta_2 = 1/D$)	65.5 ± 0.4
TFA(-) ($\beta_1 = 1/D, \beta_2 = 1/D$)	61.0 ± 0.1
TFA(-) ($\beta_1 = 1/D, \beta_2 = 1/D^2$)	51.8 ± 0.0
TFA(-) ($\beta_1 = 1/D, \beta_2 = 0$)	51.8 ± 0.0
TFA(-) ($\beta_1 = 0, \beta_2 = 1/D^2$)	62.0 ± 0.0

A.2. T3A

T3A [31] updates only the classifier module by the centroid of each class averaged over the pseudo-labeled samples' feature vectors in an online manner. This is a gradient-free approach and there is no loss function. The hyperparameter filter size K is set to 100 in our experiment.

A.3. CFA

CFA [17] minimizes both the class-conditional distribution differences and the whole distribution differences of the hidden representation just before the classifier of a model between the source and target in an online manner. The hyperparameters of CFA are the same as the default described in [17]. Specifically, the balancing hyperparameter between the whole distribution loss and class-conditional loss is set as 1. The maximum central moments' order K is set as 3.

Data Availability Statement The experiment code for this study is not publicly available. The datasets used for the experiments in this study are publicly available through the Internet.

Declarations

Conflict of interest The authors declare no conflicts of interest associated with this manuscript.

References

1. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL, pp. 4171–4186 (2019). https://doi.org/10.18653/v1/N19-1423
2. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al.: Language models are unsupervised multitask learners. OpenAI Blog **1**(8), 9 (2019)
3. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D.: Language models are few-shot learners. In: Advances in NeurIPS, vol. 33, pp. 1877–1901 (2020). https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf
4. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: transformers for image recognition at scale. In: ICLR (2020)
5. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H.: Training data-efficient image transformers & distillation through attention. In: ICML, pp. 10347–10357, PMLR (2021)
6. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European Conference on Computer Vision, pp. 213–229, Springer (2020)
7. Zhou, L., Zhou, Y., Corso, J.J., Socher, R., Xiong, C.: End-to-end dense video captioning with masked transformer. In: Proceedings of the IEEE Conference on CVPR, pp. 8739–8748 (2018)
8. Zeng, Y., Fu, J., Chao, H.: Learning joint spatial-temporal transformations for video inpainting. In: European Conference on Computer Vision, pp. 528–543, Springer (2020).
9. Paul, S., Chen, P.-Y.: Vision transformers are robust learners (2021)
10. Morrison, K., Gilby, B., Lipchak, C., Mattioli, A., Kovashka, A.: Exploring corruption robustness: inductive biases in vision transformers and MLP-mixers (2021)
11. Hendrycks, D., Mu, N., Cubuk, E.D., Zoph, B., Gilmer, J., Lakshminarayanan, B.: Augmix: a simple data processing method to improve robustness and uncertainty. arXiv preprint arXiv:1912.02781 (2019)
12. Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., et al.: The many faces of robustness: a critical analysis of out-of-distribution generalization. arXiv preprint arXiv:2006.16241 (2020)
13. Chen, X., Hsieh, C.-J., Gong, B.: When vision transformers outperform ResNets without pretraining or strong data augmentations (2021)
14. Schneider, S., Rusak, E., Eck, L., Bringmann, O., Brendel, W., Bethge, M.: Improving robustness against common corruptions by covariate shift adaptation. In: Advances in NeurIPS (2020)
15. Nado, Z., Padhy, S., Sculley, D., D'Amour, A., Lakshminarayanan, B., Snoek, J.: Evaluating prediction-time batch normalization for robustness under covariate shift (2021)
16. Wang, D., Shelhamer, E., Liu, S., Olshausen, B., Darrell, T.: Tent: fully test-time adaptation by entropy minimization. In: ICLR (2020)
17. Kojima, T., Matsuo, Y., Iwasawa, Y.: Robustifying vision transformer without retraining from scratch by test-time class-conditional feature alignment. In: Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22, pp. 1009–1016 (2022). https://doi.org/10.24963/ijcai.2022/141
18. Zhang, J., He, T., Sra, S., Jadbabaie, A.: Why gradient clipping accelerates training: A theoretical justification for adaptivity. In: ICLR (2020). https://openreview.net/forum?id=BJgnXpVYwS
19. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in NeurIPS, pp. 5998–6008 (2017)
20. Hendrycks, D., Dietterich, T.: Benchmarking neural network robustness to common corruptions and perturbations. In: Proceedings of the ICLR (2019)
21. Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., Song, D.: Natural adversarial examples. In: Proceedings of the IEEE/CVF Conference on CVPR, pp. 15262–15271 (2021)
22. Xiao, K., Engstrom, L., Ilyas, A., Madry, A.: Noise or signal: the role of image backgrounds in object recognition (2020)

23. Bhojanapalli, S., Chakrabarti, A., Glasner, D., Li, D., Unterthiner, T., Veit, A.: Understanding robustness of transformers for image classification (2021)
24. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: ICML, pp. 448–456, PMLR (2015).
25. Shannon, C.E.: A mathematical theory of communication. Bell Syst. Tech. J. **27**(3), 379–423 (1948)
26. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on CVPR, pp. 770–778 (2016)
27. Lee, D.-H., et al.: Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In: Workshop on Challenges in Representation Learning, ICML, p. 896 (2013)
28. Liang, J., Hu, D., Feng, J.: Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In: ICML, pp. 6028–6039, PMLR (2020)
29. Liu, Y., Kothari, P., van Delft, B.G., Bellot-Gurlet, B., Mordan, T., Alahi, A.: TTT++: when does self-supervised test-time training fail or thrive? In: Advances in NeurIPS (2021). https://openreview.net/forum?id=86NHK__yFDl
30. Chen, D., Wang, D., Darrell, T., Ebrahimi, S.: Contrastive test-time adaptation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 295–305 (2022)
31. Iwasawa, Y., Matsuo, Y.: Test-time classifier adjustment module for model-agnostic domain generalization. In: Advances in NeurIPS (2021). https://openreview.net/forum?id=e_yvNqkJKAW
32. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Trans. Knowl. Data Eng. **22**(10), 1345–1359 (2010). https://doi.org/10.1109/TKDE.2009.191
33. Patel, V.M., Gopalan, R., Li, R., Chellappa, R.: Visual domain adaptation: a survey of recent advances. IEEE Signal Process. Mag. **32**(3), 53–69 (2015)
34. Wilson, G., Cook, D.J.: A survey of unsupervised deep domain adaptation. ACM Trans. Intell. Syst. Technol. (TIST) **11**(5), 1–46 (2020)
35. Xie, Q., Luong, M.-T., Hovy, E., Le, Q.V.: Self-training with noisy student improves imagenet classification. In: Proceedings of the IEEE/CVF Conference on CVPR (CVPR) (2020)
36. Rusak, E., Schneider, S., Gehler, P., Bringmann, O., Brendel, W., Bethge, M.: Adapting imagenet-scale models to complex distribution shifts with self-learning. arXiv preprint arXiv:2104.12928 (2021)
37. Zhang, Z., Sabuncu, M.R.: Generalized cross entropy loss for training deep neural networks with noisy labels. In: NeurIPS (2018)
38. Zellinger, W., Grubinger, T., Lughofer, E., Natschläger, T., Saminger-Platz, S.: Central moment discrepancy (CMD) for domain-invariant representation learning. In: 5th ICLR (2017)
39. Zhou, D., Yu, Z., Xie, E., Xiao, C., Anandkumar, A., Feng, J., Alvarez, J.M.: Understanding the robustness in vision transformers. In: International Conference on Machine Learning, pp. 27378–27394, PMLR (2022)
40. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: an imperative style, high-performance deep learning library. Adv. NeurIPS **32**, 8026–8037 (2019)
41. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario (2009)
42. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. IJCV **115**(3), 211–252 (2015)
43. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L.: Imagenet: a large-scale hierarchical image database. In: 2009 IEEE Conference on CVPR, pp. 248–255, IEEE (2009)
44. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. JMLR **15**(1), 1929–1958 (2014)
45. Lester, B., Al-Rfou, R., Constant, N.: The power of scale for parameter-efficient prompt tuning (2021)