Category-Aware Test-Time Training Domain Adaptation

Yangqin Feng, Xinxing Xu, Huazhu Fu, Yan Wang, Zizhou Wang, Liangli Zhen, Rick Siow Mong Goh, Yong Liu
Institute of High Performance Computing, Agency for Science, Technology and Research (A*STAR), Singapore

***Abstract*—Machine learning models often struggle to generalise to out-of-distribution (OOD) data. One promising solution for solving this problem is test-time training domain adaptation, which adjusts a trained model to the new test data without revisiting the source dataset, thus preserving the privacy of source data. However, existing test-time training methods have not considered the mining of category information of the test data in the training for domain adaptation, thus suffering in inaccurate domain alignment. In this paper, we propose a novel method called category-aware test-time training (CAT^3) to adapt the pre-trained model to test data on the fly. CAT^3 first trains a model with multiple diverse classifiers using the source datasets and conducts a source data summarisation. Then, it assigns the pseudo-labels for the test data that have consistent classification results from the trained classifiers and adjusts the pre-trained model by aligning these reliable test data to their corresponding source data categories iteratively. Unlike existing test-time training methods, CAT^3 reveals the category information of the test data and aligns these reliable test data to the source data at the category level instead of the dataset level. Empirical results demonstrate that CAT^3 can outperform the current state-of-the-art methods on several benchmarks, indicating our proposed method's effectiveness.**

***Index Terms*—Domain adaptation, test time training, category-aware learning**

I. Introduction

Machine learning usually assumes that the datasets used for training and testing share the same data distribution [1]. However, such an assumption does not hold in many real scenarios. For example, data distributions in changing environments for real-world applications like medical image analysis and autonomous driving. The distribution shift may cause a dramatic prediction accuracy drop. One popular approach to tackle this issue is domain adaptation (DA), which aligns the training and test data distributions in the latent space iteratively, as shown in Figure 1 (a). It has achieved promising results in various visual tasks [2, 3]. Unfortunately, domain adaptation requires the access to the training data when conducting prediction on test data, which faces difficulties due to privacy concerns, large sizes of source datasets, and other constraints. Another approach is domain generalisation (DG) — learning a universal model on the training domain and aiming to generalise well to unseen test domains (Figure 1 (b)). Domain generalisation does not require revisiting the source datasets or accessing the test data during the model building [4]. However, its performance is usually inferior to DA due to lacking the knowledge about test data distributions. The recent advance in solving this dilemma is the test-time training (TTT): adapting the trained model to the test data on the fly [5–8].

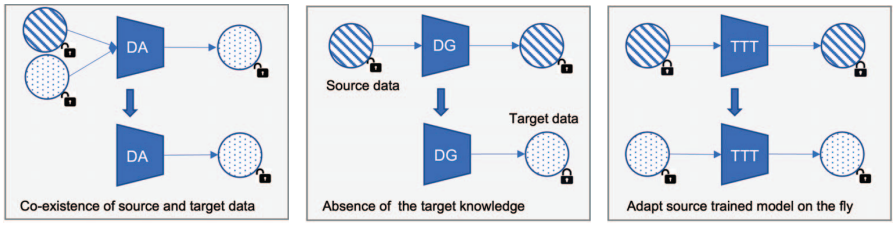
The existing TTT methods [7–9] achieve the adaptation by involving additional self-supervised tasks. The basic idea of methods in [7, 8] is ingenious and compelling: in the training phase, they train the model for the main task and the additional self-supervised learning (SSL) task on the source dataset simultaneously. During the test time, they fine-tuned the trained model using the SSL task. In addition, source hypothesis transfer (SHOT) [9] freezes the source hypothesis after pre-train the model on the source dataset, then adapts the feature generator module only. However, these existing TTT ones align the target to the source distribution at the dataset level, which lacks mining the category information implicit in the unlabelled target samples and suffers a sub-optimal performance.

In this paper, we propose a novel method called category-aware Test-Time Training (CAT^3) to align the test data to the source data on the fly via the summary information of the source dataset. The proposed method includes two stages: 1) we train the model with the source dataset and conduct the offline representation summarisation for the source domain. The trained model and the offline representation summarisation are stored and used in test time. 2) at the test time, we align the test data distribution to the training data distribution by enforcing the consistency between the online test data representation estimation and the offline source data representation summarisation. Specifically, we compute the mean and covariance matrix of each source class as the offline representation summarisation. Then, we match the offline representation summarisation from the training and test sets iteratively in the second stage. To obtain reliable pseudo-labels of the test data for the matching, we connect multiple classifiers (for simplicity, we will employ two classifiers in this work) on the top of the feature extractor, and the classifiers are encouraged to be diverse via negative correlation learning [10]. The test samples with consistent predictions from these classifiers are signed with the pseudo-labels and

This work was supported by the Agency for Science, Technology, and Research (A*STAR) through its AME Programmatic Funding Scheme Under Project A20H4b0141, and the Agency for Science, Technology, and Research (A*STAR) through its Biomedical Engineering Programme Project C221318005, the Agency for Science, Technology, and Research (A*STAR) through its RIE2020 Health and Biomedical Sciences (HBMS) Industry Alignment Fund Pre-Positioning (IAF-PP) (grant no. H20C6a0032).

Corresponding author: H. Fu (e-mail: hzfu@ieee.org)

979-8-3503-5409-6/24/$31.00 ©2024 IEEE
DOI 10.1109/CAI59869.2024.00065



(a) Conventional domain adaptation (b) Domain generalization (c) Test-time training

Fig. 1. Illustration of the different settings of DA (a), DG (b), and TTT (c). (a) conventional DA aligns the distribution of the training (source) and test (target) domains while needing the co-existence of source and target datasets which may lead to many concerns. (b) DG learns a generalised model from the training domain that can generalise well to unseen test domains without utilising the target knowledge. (c) TTT adapts the test domain to the training domain on the fly without access to the training data and test data annotations.

adopted for online representation summarisation. The novelty and main contributions are summarised as follows:

- We propose a test-time training domain adaptation framework (CAT^3) that reveals the semantics of the test samples to adapt a pre-trained model to test data on the fly without accessing the source data and target annotations. This learning paradigm leverages the knowledge of unlabelled test data and preserves the privacy of the source datasets to benefit the model's performance.
- We design a new distribution alignment strategy to efficiently align different data distributions via exploiting the latent semantics of the target test samples. Unlike the existing test-time training methods that ignore the category information of the data samples in domain alignment, our method proposes to identify test data with reliable pseudo-labels to align with its corresponding source data categories.
- Extensive experiments on various benchmarks have been conducted. The results demonstrate that our model can outperform current state-of-the-art (SOTA) methods.

II. Related Work

A. Domain adaptation

DA aims to mitigate the domain gap between the source and target domains as much as possible. Existing methods typically align the two domains by minimising the discrepancy losses [11–13] or employing the adversarial training strategy [14, 15]. For example, Saito et al. developed a strong-weak distribution alignment strategy to achieve distribution alignment on local and global levels [16]. Feng et al. [17] proposed a contrastive domain adaptation with consistency match. These methods have achieved promising results. However, they require the access to the source data for domain alignment, which is challenging to meet in many practical scenarios with privacy concerns and other constraints.

B. Domain generalisation

DG aims to learn a universal model from source domains and expects the trained model can be generalised well to an arbitrary unseen out-of-distribution (OOD) target domain [18]. The main advantage of DG is that it does not require revisiting

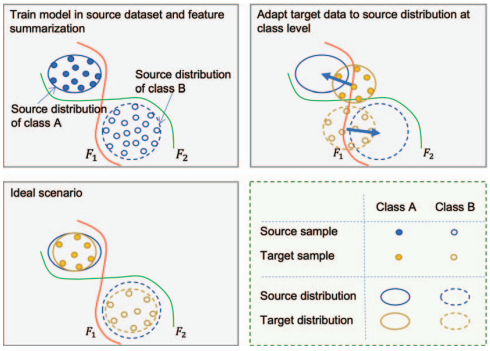


Fig. 2. Illustration of the proposed CAT^3. It first trains the model on the source dataset and summarises the distribution of each category. Then it aligns the unlabelled test samples to their source data category by mining the latent semantic information of the test data iteratively during the alignment process to adapt the trained model for the test data domain.

the source datasets nor accessing the test data during the model building [4]. One popular family of DG methods mainly tries to learn domain invariant features by kernel approach [19, 20] and domain-adversarial learning [21, 22]. For example, Shao et al. [22] proposed a multi-adversarial discriminative domain generalisation method. Another popular family of DG methods is domain augmentation. It simulates samples from fictitious domains with gradient-based adversarial perturbations [23, 24] and adversarially trained generators [25, 26]. Recently, a few studies have shown that data augmentation during training [27–30] and during testing [31–33] can improve model robustness and generalisation ability significantly. However, the performance of the DG approaches is usually inferior to domain adaptation due to lacking the knowledge of the test data distributions.

C. Test-time training domain adaptation

To improve the model's prediction accuracy without revisiting the source datasets, test-time training domain adaptation (T^{3}DA) updates the model on the fly according to test samples and the summary information of the source data. One key advantage of T^{3}DA is that there is no need to presuppose the test domain as in DG, and no need to rely on source training

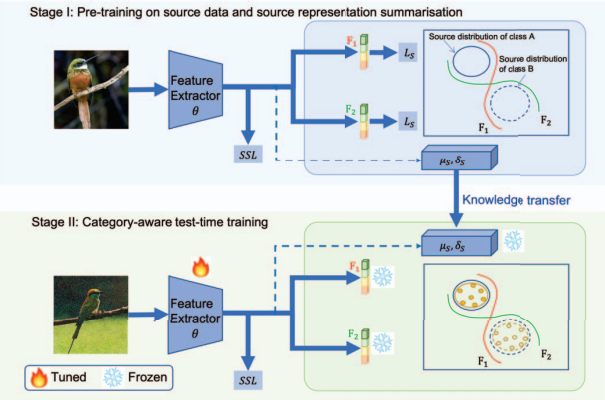


Fig. 3. An overview of the proposed CAT^3 method. It contains two stages. In the first stage (blue box), we propose a dual-classifier (C_1 and C_2 are both for the main task, and they are negative correlation to each other) model for source training, after training, we summarise the distribution for each class, then save it as a part of the model. In the second stage (green box), we fix the trained dual-classifier as it encodes the source data distribution, then update the feature extractor by aligning the target representation to the source distribution with the reliable pseudo label generated by the dual-classifier.

data as in DA. The pioneer test-time training methods create an additional self-supervised learning (SSL) task [6, 7] and train the model with such auxiliary SSL task together with the main task. Then it adapts the trained model through the auxiliary SSL task with the test data. The key limitation of this approache is that it may result in severe over-fitting of the SSL task [8]. Accordingly, one challenge is how to overcome this issue. We also note that all the existing methods align the distribution without considering the latent semantics of the target test data. Our proposed method belongs to the test-time training approach, and it presents a novel framework, CAT^3, that tackles these challenges.

III. Methodology

A. Problem formulation

We aim to tackle the unsupervised DA problem without revisiting the source dataset. Let us denote the source and target domains as $\mathfrak{D}_s$ and $\mathfrak{D}_t$, respectively. For an unsupervised DA problem, we observe a set of n_s labelled data $\mathcal{D}_s = \{\mathbf{x}_i^s, y_i^s\}_{i=1}^{n_s}$ sampled from $\mathfrak{D}_s$ and a set of n_t unlabelled data $\mathcal{D}_t = \{\mathbf{x}_i^t\}_{i=1}^{n_t}$ sampled from $\mathfrak{D}_t$. The target domain $\mathfrak{D}_t$ is relevant to $\mathfrak{D}_s$ but $\mathfrak{D}_t \neq \mathfrak{D}_s$. For test-time training domain adaptation, we are allowed to use the source data $\mathcal{D}_s$ for the model pre-training. After the deployment of this pre-trained model $\mathcal{M}_s$, we aim to adapt $\mathcal{M}_s$ to the test data $\mathcal{D}_t$ on the fly without revisiting $\mathcal{D}_s$ and obtain the new model $\mathcal{M}_t$ to achieve accurate classification of the test data.

B. Framework of CAT^3

The overview of the proposed CAT^3 method is shown in Figure 3. Our method includes two stages. In Stage 1, the source data are adopted to pre-train the feature extractor g and two classifiers f_1 and f_2 for classifying the source samples. Then, the source data samples are mapped to the feature space for the source data summarisation. In Stage 2, the feature extractor g is fine-tuned using the test data to adapt the pre-trained model $\mathcal{M}_s$ to $\mathcal{M}_t$ on the fly. Furthermore, by considering the performance improvement of using contrastive learning in test-time training as varied in [8], we also use contrastive learning as the SSL task.

C. Pre-training and source data summarisation

In the first stage, we train a model with two diverse classifiers on the source dataset. Specifically, we first map the source samples with the feature extractor $g(\theta, \cdot)$ into the feature space as $\mathbf{z}_i = g(\mathbf{x}_i)$. The feature representations of the source data $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_{n_s}$ will be classified by two different classifiers f_1 and f_2. To learn two diverse classifiers, we adopt the negative correlation learning strategy [10], which is originally designed for ensemble learning. Mathematically, we minimise the following function:

$$\mathcal{L}_1 = \mathcal{L}_{sc} + \lambda_1 \mathcal{L}_{nc} + \lambda_2 \mathcal{L}_{ss}, \tag{1}$$

where $\mathcal{L}_{sc}$ is the classification loss for the source data, $\mathcal{L}_{nc}$ is the loss for the negative correlation learning, and $\mathcal{L}_{ss}$ is the self-supervised learning loss for the source data (i.e., the loss for the SSL task). The two hyper-parameters of λ_1 and λ_2 are adopted to trade off the contributions of the three loss items.

In the main task, the classification loss for the source dataset $\mathcal{L}_{sc}$ is defined as follows:

$$\mathcal{L}_{sc} = \frac{1}{n_s} \sum_{i=1}^{n_s} \sigma(f_1(\mathbf{z}_i), \mathbf{y}_i) + \sigma(f_2(\mathbf{z}_i), \mathbf{y}_i), \tag{2}$$

where $\sigma(\cdot, \cdot)$ is the cross-entropy loss function for the two inputs.

In negative correlation learning, a correlation penalty term is introduced to the loss function of each classifier, thus enabling the joint training of both classifiers. The negative correlation learning loss for the two classifiers is defined as:

$$\mathcal{L}_{nc} = \frac{1}{n_s} \sum_{i=1}^{n_s} (f_1(\mathbf{z}_i) - \bar{f})^2 + (f_2(\mathbf{z}_i) - \bar{f})^2, \tag{3}$$

where $\bar{f} = \frac{f_1(\mathbf{z}_i) + f_2(\mathbf{z}_i)}{2}$ is the average value of the outputs from the two classifiers.

In the SSL,two enhanced perspectives derived from an identical source instance are employed as a positive duo, while all other combinations are considered negative for training purposes. It has a self-supervised learning head ϕ to map the feature representations of the two augmented views into a lower-dimensional space. For given a batch of n_b samples, we obtain the lower-dimensional representations as $\{h_k\}_{k=1}^{2n_b}$. Then, the negative pairs are encouraged to be far away and the positive pairs are pulled closer by minimising the loss function:

$$\mathcal{L}_{ss} = -\frac{1}{2n_b} \sum_{k=1}^{2n_b} \log \frac{1_{k==l} \exp(\frac{\cos(h_k, h_l)}{\tau})}{\sum_{m=1}^{2n_b} 1_{k \neq m} \exp(\frac{\cos(h_k, h_m)}{\tau})}, \tag{4}$$

where $\tau$is a temperature scaling parameter.

Upon the completion of the training for the model $\mathcal{M}_s$ via Equation (1), we conduct the offline source data summarisation to characterise the distribution of each source data category. For the c-th source category, we denote the feature vectors as $\mathbf{Z}_c = [\mathbf{z}_{\alpha_1}, \mathbf{z}_{\alpha_2}, \ldots, \mathbf{z}_{\alpha_{n_c}}]$, where $\alpha_1, \alpha_2, \ldots, \alpha_{n_c}$ are the indices of the c-th category samples and n_c is the number of samples in category c. Then, we compute its empirical mean and covariance matrix as:

$$u_c = \frac{1}{n_c}(\mathbf{z}_{\alpha_1} + \mathbf{z}_{\alpha_2} + \cdots + \mathbf{z}_{\alpha_{n_c}}) \tag{5}$$

and

$$\Sigma_c = \frac{1}{n_c - 1}(\mathbf{Z}_c^T \mathbf{Z}_c - (\mathbf{I}^T \mathbf{Z}_c)^T (\mathbf{I}^T \mathbf{Z}_c)), \tag{6}$$

where $c = 1, 2, \ldots, C$ and C is the total number of categories.

D. Time-time training strategy

In Stage 2, we conduct the test-time training with self-supervised learning and domain alignment. Note that we freeze the weights of the classifiers and fine-tune the feature extractors and the SSL head as shown in Figure 3. For the SSL task, we adjust the model weights on the test data by minimising the loss function, as same as the first stage, in Equation (4).

For the domain alignment, we first map the test samples with the pre-trained feature extractor g into the feature space as $\hat{\mathbf{z}}_1, \hat{\mathbf{z}}_2, \ldots, \hat{\mathbf{z}}_{n_t}$ and compute the prediction results from the two classifiers f_1 and f_2 for each test sample $\mathbf{x}_i^t$. Then, we assign a pseudo-label for the test sample with consistent prediction results from the classifiers and classify it into its corresponding category. For the c-th target category, we denote the feature vectors as $\hat{\mathbf{Z}}_c = [\hat{\mathbf{z}}_{\beta_1}, \hat{\mathbf{z}}_{\beta_2}, \ldots, \hat{\mathbf{z}}_{\beta_{n_c}}]$, where $\beta_1, \beta_2, \ldots, \beta_{n_c}$ are the indices of the c-th category test samples and n_c is the number of samples in category c. Next, we compute its empirical mean and covariance matrix as:

$$\hat{u}_c = \frac{1}{n_c}(\hat{\mathbf{z}}_{\beta_1} + \hat{\mathbf{z}}_{\beta_2} + \cdots + \hat{\mathbf{z}}_{\beta_{n_c}}) \tag{7}$$

and

$$\hat{\Sigma}_c = \frac{1}{n_c - 1}(\hat{\mathbf{Z}}_c^T \hat{\mathbf{Z}}_c - (\mathbf{I}^T \hat{\mathbf{Z}}_c)^T (\mathbf{I}^T \hat{\mathbf{Z}}_c)). \tag{8}$$

The domain alignment is based on minimising the distance between the distribution statistics estimated from a mini-batch of test data (i.e., $\hat{u}_c$ and $\hat{\Sigma}_c$) and the pre-stored source summarisation statistics (i.e., u_c and Σ_c):

$$\mathcal{L}_{da} = \sum_{c=1}^{C}(\|\hat{u}_c - u_c\|_2^2 + \|\hat{\Sigma}_c - \Sigma_c\|_F^2), \tag{9}$$

where $\|\cdot\|$ denotes the Euclidean norm and $\|\cdot\|_F$ denotes the Frobenius norm.

In summary, we minimise the following objective function in the second stage:

$$\mathcal{L}_2 = \mathcal{L}_{ss} + \eta \mathcal{L}_{da}, \tag{10}$$

where η is a hyper-parameter to trade off the two terms.

The objective functions for the two stages, i.e., Equation (1) and Equation (10), can be optimised using a stochastic gradient descent optimisation algorithm, such as ADAM [34]. After the training, we infer the test samples by averaging the outputs of the two trained classifiers.

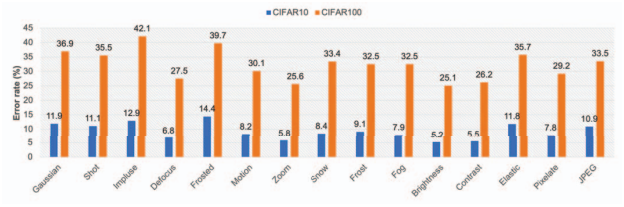


Fig. 4. The classification of CAT^3 under different image corruptions on CIFAR10 and CIFAR100.

TABLE I
THE AVERAGE CLASSIFICATION ERROR RATES (%) OF OUR PROPOSED CAT^3 AND OTHER PEER METHODS ON CIFAR10-C, CIFAR100-C [39], AND CIFAR10.1 [36].

Method	CIFAR10-C	CIFAR100-C	CIFAR10.1
Test	29.1	61.2	12.1
BN [37]	15.7	43.3	14.1
TTT-R [7]	14.3	40.4	11.0
SHOT [9]	14.7	38.1	11.1
TENT [5]	12.6	36.3	13.4
TTT++ [8]	10.2	34.4	10.4
CAT^3	9.2	32.4	9.0

IV. EXPERIMENTAL STUDY

We test our method under three scenarios: common image corruptions on CIFAR10 and CIFAR100 [35] and natural domain shifts (CIFAR10.1 [36]). We compare CAT^3 with the current SOTA, including test-time training domain adaptation methods, namely Batch Normalization (BN) [37], Test-Time Entropy Minimization (TENT) [5], Source Hypothesis Transfer (SHOT) [9], Test-Time Training (TTT-R) [7], and an improved test-time training (TTT++) [8]. Also, we report the performance of the source-trained model (without any test-time training adaptation) by directly evaluating it on the test data.

A. Experimental settings

We use the ResNet-50 [38] network as the backbone for the feature extraction by following the same setting with [8] for a fair comparison. The entire network is implemented in PyTorch and trained on an NVIDIA DGX Station A100, which includes four NVIDIA A100 Tensor Core GPUs. To optimise our method's objective functions, we adopt the ADAM optimiser with a learning rate of 0.001 and the maximal number of training epochs as 500.

B. Common image corruption

Firstly, we evaluate our CAT^3 method by using 15 kinds of common image corruptions. The evaluation process followed the protocol of TENT [5] and TTT++ [8]. The source domain is the original CIFAR10 and CIFAR100 [35], and the target domain is generated by involving 15 kinds of different corruptions (e.g., snow, frog, blur, and noise effects) in the

TABLE II
THE CLASSIFICATION ERROR RATES (%) OF OUR PROPOSED CAT^3 AND THE PEER METHODS ON THE VISDA-C DATASET.

Class	Test	BN	TENT	SHOT	TTT++	CAT^3
plane	56.52	44.38	13.43	5.73	4.13	4.01
bcycl	88.71	56.98	77.98	13.64	26.20	20.33
bus	62.77	33.24	20.17	23.33	21.60	19.5
car	30.56	55.28	48.15	42.69	31.70	29.42
horse	81.88	37.45	21.72	7.93	7.43	6.25
knife	99.03	66.60	82.45	86.99	83.30	75.44
mcycl	17.53	16.55	12.37	19.17	7.83	7.79
person	95.85	59.02	35.78	19.97	21.10	20.37
plant	51.66	43.55	21.06	11.63	7.03	6.53
sktbrd	77.86	60.72	76.41	11.09	7.73	6.77
train	20.44	31.07	34.11	15.06	6.91	5.93
truck	99.51	82.98	98.93	43.26	51.40	45.86
Avg.	58.72	48.12	42.73	25.04	22.46	20.68

original test set as the CIFAR10-C and CIFAR100-C [39]. Firstly, we employ a ResNet-50 [38] as the backbone and train it on the source dataset. Then, we do test-time training on the 15 generated target test sets and compute the average classification error based on all 15 target test sets. The batch size is set as 256 in the procedure of test-time training. For the online feature alignment, we use the dynamic queue by following [8], which contains 16 batches. The average classification error on the CIFAR10-C and CIFAR100-C [39] datasets are reported in Table I and Figure 4, from which we can find that:

- The model pre-trained on the original datasets of CIFAR10 and CIFAR100 cannot generalise well to the common image corruption scenarios. The test error rates on CIFAR10 and CIFAR100 reach 29.1% and 61.2%, respectively.
- All test-time training domain adaptation methods can significantly improve the performance of the pre-trained model. For example, the method of BN can reduce the error rate from 29.1% to 15.7% on CIFAR10 and from 61.2% to 43.3% on CIFAR100.
- CAT^3 outperforms all the peer methods on the two test datasets. It reduces the error rate of the current SOTA method (TTT++) from 10.2% to 9.2% on CIFAR10 and from 34.4% to 32.4% on CIFAR100. However, CAT^3 struggles to handle the corruptions of "impluse noise" and "frosted glass blur" and results in the error rates of 12.9% and 14.4% on CIFAR10 and 42.1% and 39.7% on CIFAR100. This result shows that we need to improve CAT^3 for handling the scenarios with bit errors and "frosted glass" windows or panels. For other common image corruptions, CAT^3 can have a lower error rate, especially for the corruptions of "zoom blur", "brightness", and "contrast". These results indicate that the category-aware strategy contributes to the accurate domain alignment and verifies the effectiveness of CAT^3 for handling common image corruptions.

C. Natural domain shift

In addition, we assess the performance of our proposed CAT^3 on a natural distribution shift scenario. The models are pre-trained on CIFAR10 and tested on CIFAR10.1 [36]. As demonstrated in [36], the natural distribution shift from CIFAR10 to CIFAR10.1 typically makes the model an accuracy drop around 4% to 10% for a wide range of deep neural networks. The comparison of the test-time training methods is reported in Table I, from which we can see that:

- The model pre-trained on the original datasets of CIFAR10 suffers an error rate of 12.1% on the CIFAR10.1 dataset. The natural distribution shift causes a much milder problem for the pre-trained model compared with some of the common image corruptions.
- Most test-time training domain adaptation methods can improve the performance of the pre-trained model. However, some of the domain adaptation methods make the case worse. For instance, the method of BN increases the error rate of the pre-trained model from 12.1% to 14.1%. The potential reason is that BN assumes different samples and spatial locations are shifted in a similar manner [37], which is not held in the natural distribution shit scenario.
- CAT^3 outperforms all the peer methods on CIFAR10.1. It reduces the error rate of current SOTA method (TTT++) from 10.4% to 9.0%. It verifies the effectiveness of CAT^3 for handling the natural distribution shift.

D. Ablation study

To study the effectiveness of different components of our method, we construct three variants of CAT^3, including the variant that only uses the SSL task to adapt the model, the variant that assigns the pseudo-label with one classifier only (i.e., w/o f_2) for domain alignment, the variant that minimises the distance between the mean of each category (i.e., w/o Σ_c). The results are shown in Table II, from which we see that:

- The incorporation of the SSL task can adopt the model well for the data with common image corruptions and natural distribution shit. The model only with the $\mathcal{L}_{ss}$ obtains the error rate of 11.1% on CIFAR10-C and 10.7% on CIFAR10.1.
- The category-level domain alignment can achieve promising results even with only one classifier (i.e., w/o f_2) to assign the pseudo-labels for the test samples. The involvement of the second classifier can reduce the error rate further.
- The minimisation of the distance between the two domains using the covariance of the data categories can reduce the error significantly. Specifically, it can reduce the error rate by 5%, 5.1%, and 4.8% on CIFAR10-C, CIFAR100-C, and CIFAR10.1, respectively.
- Full CAT^3 outperforms all of its three variants on all three datasets, which indicates that all the developed components are essential to CAT^3 and contribute to its final performance.

(a) Source test data representations obtained by the pre-trained model (b) Target test data representations obtained by the pre-trained model (c) Target test data representations obtained by the fine-tuned model with CAT^3

Fig. 5. The embeddings of the learnt representations from the pre-trained model and the fine-tuned model with CAT^3. The source data are from the CIFAR10 dataset, and the target test samples are from CIFAR10 with the corruption of "snow". The samples from different categories are denoted in different colours.

TABLE III
THE CLASSIFICATION ERROR RATES (%) OF CAT^3 AND ITS THREE VARIANTS ON CIFAR10-C, CIFAR100-C [39] AND CIFAR10.1 [36].

Method	CIFAR10-C	CIFAR100-C	CIFAR10.1
$\mathcal{L}_{ss}$ only	11.1	36.9	10.7
w/o f_2	9.7	34.1	9.6
w/o Σ_c	14.2	37.5	13.8
Full CAT^3	9.2	32.4	9.0

E. Visualisation of learnt representations

To visually understand the effectiveness of CAT^3 for representation learning on CIFAR10 with the corruption of "snow" on the target test data, we adopt the t-SNE algorithm [40] to project the learnt representations into a 2D plane, as shown in Figure 5. Specifically, we plot the embeddings of the representations obtained by the pre-trained model for the source data (Figure 5 (a)) and the test data (Figure 5 (b)) and the embeddings of the representations obtained CAT^3 for the test data (Figure 5 (c)).

From the results in Figure 5 (a), we can see that the test samples from different categories in the source data are well grouped into different clusters (denoted in different colours), except a few samples are mixed with other groups. These results are consistent with the fact that the pre-trained model can typically achieve high accuracy on CIFAR10.

As shown in Figure 5 (b), the representations from the pre-trained model for the target data from different categories are not well distinguished. Some of the target data categories are mixed. Since the learnt representations from the pre-trained model for the target test data are not distinguishable, its classification error rate can be as high as 29.1% for the CIFAR10-C dataset.

From Figure 5 (c), we can see that the samples from different categories are well grouped into different clusters. It means that by using the test-time training with CAT^3, the model can project the target test samples into different clusters in the feature space. Only a small portion of samples are mixed into samples from other categories, which is consistent with the result that the error rate is 9.2% on CIFAR10-C and the error rate on the test data with the corruption of "snow" is 8.37%.

V. CONCLUSION

In this paper, we proposed a novel domain adaptation method CAT^3 to adapt the pre-trained model for the test data on the fly. It does not require revisiting the source data during the domain adaptation process, thus preserving the source data privacy and extending its application to more scenarios with constraints to store the source data. We introduced the category-aware domain alignment strategy, which had yet to be explored in test-time training. We employed negative correlation learning to train two classifiers for the label prediction to obtain more accurate pseudo-labels for the test samples during the iterative domain alignment. By conducting extensive experiments, we verified the effectiveness of our proposed method on several benchmarks, which cover the common corruptions and natural distribution shift. CAT^3 can outperforms the current SOTA methods on all three test benchmarks with a significant margin. One limitation of our method is handling the scenario where some categories may have a very limited number of test samples. Our method may not be stable since the mean and covariance of one category can change significantly during different iterations of category-level alignment, which will be investigated in future work.

REFERENCES

[1] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do imagenet classifiers generalize to imagenet?" in *International Conference on Machine Learning*. PMLR, 2019, pp. 5389–5400.

[2] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE transactions on neural networks*, vol. 22, no. 2, pp. 199–210, 2010.

[3] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa, "Visual domain adaptation: A survey of recent advances," *IEEE signal processing magazine*, vol. 32, no. 3, pp. 53–69, 2015.

[4] K. Zhou, Z. Liu, Y. Qiao, T. Xiang, and C. C. Loy, "Domain generalization: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[5] D. Wang, E. Shelhamer, S. Liu, B. Olshausen, and T. Darrell, "Tent: Fully test-time adaptation by entropy minimization," *arXiv preprint arXiv:2006.10726*, 2020.

[6] Q. Wang, O. Fink, L. Van Gool, and D. Dai, "Continual test-time domain adaptation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 7201–7211.

[7] Y. Sun, X. Wang, Z. Liu, J. Miller, A. Efros, and M. Hardt, "Test-time training with self-supervision for generalization under distribution shifts," in *International conference on machine learning*. PMLR, 2020, pp. 9229–9248.

[8] Y. Liu, P. Kothari, B. van Delft, B. Bellot-Gurlet, T. Mordan, and A. Alahi, "Ttt++: When does self-supervised test-time training fail or thrive?" *Advances in Neural Information Processing Systems*, vol. 34, pp. 21 808–21 820, 2021.

[9] J. Liang, D. Hu, and J. Feng, "Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation," in *International Conference on Machine Learning*. PMLR, 2020, pp. 6028–6039.

[10] Y. Liu and X. Yao, "Ensemble learning via negative correlation," *Neural networks*, vol. 12, no. 10, pp. 1399–1404, 1999.

[11] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *International conference on machine learning*. PMLR, 2015, pp. 97–105.

[12] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, "Maximum classifier discrepancy for unsupervised domain adaptation," in *IEEE conference on computer vision and pattern recognition*, 2018, pp. 3723–3732.

[13] Y. Feng, X. Xu, Y. Wang, X. Lei, S. K. Teo, J. Z. T. Sim, Y. Ting, L. Zhen, J. T. Zhou, Y. Liu *et al.*, "Deep supervised domain adaptation for pneumonia diagnosis from chest x-ray images," *IEEE Journal of Biomedical and Health Informatics*, vol. 26, no. 3, pp. 1080–1090, 2021.

[14] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *International conference on machine learning*. PMLR, 2015, pp. 1180–1189.

[15] Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. Chandraker, "Learning to adapt structured output space for semantic segmentation," in *IEEE conference on computer vision and pattern recognition*, 2018, pp. 7472–7481.

[16] K. Saito, Y. Ushiku, T. Harada, and K. Saenko, "Strong-weak distribution alignment for adaptive object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 6956–6965.

[17] Y. Feng, Z. Wang, X. Xu, Y. Wang, H. Fu, S. Li, L. Zhen, X. Lei, Y. Cui, J. S. Z. Ting *et al.*, "Contrastive domain adaptation with consistency match for automated pneumonia diagnosis," *Medical Image Analysis*, p. 102664, 2022.

[18] J. Wang, C. Lan, C. Liu, Y. Ouyang, T. Qin, W. Lu, Y. Chen, W. Zeng, and P. Yu, "Generalizing to unseen domains: A survey on domain generalization," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[19] K. Muandet, D. Balduzzi, and B. Schölkopf, "Domain generalization via invariant feature representation," in *International Conference on Machine Learning*, 2013, pp. 10–18.

[20] M. Ghifary, D. Balduzzi, W. B. Kleijn, and M. Zhang, "Scatter component analysis: A unified framework for domain adaptation and domain generalization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 7, pp. 1414–1430, 2016.

[21] H. Li, S. J. Pan, S. Wang, and A. C. Kot, "Domain generalization with adversarial feature learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5400–5409.

[22] R. Shao, X. Lan, J. Li, and P. C. Yuen, "Multi-adversarial discriminative deep domain generalization for face presentation attack detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 023–10 031.

[23] R. Volpi, H. Namkoong, O. Sener, J. C. Duchi, V. Murino, and S. Savarese, "Generalizing to unseen domains via adversarial data augmentation," *Advances in neural information processing systems*, vol. 31, 2018.

[24] S. Shankar, V. Piratla, S. Chakrabarti, S. Chaudhuri, P. Jyothi, and S. Sarawagi, "Generalizing across domains via cross-gradient training," *arXiv preprint arXiv:1804.10745*, 2018.

[25] K. Zhou, Y. Yang, T. Hospedales, and T. Xiang, "Deep domain-adversarial image generation for domain generalisation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 13 025–13 032.

[26] ——, "Learning to generate novel domains for domain generalization," in *European conference on computer vision*. Springer, 2020, pp. 561–578.

[27] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo *et al.*, "The many faces of robustness: A critical analysis of out-of-distribution generalization," in *Proceedings of the International Conference on Computer Vision*, 2021, pp. 8340–8349.

[28] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan, "Augmix: A simple data processing method to improve robustness and uncertainty," *arXiv preprint arXiv:1912.02781*, 2019.

[29] B. Li, F. Wu, S.-N. Lim, S. Belongie, and K. Q. Weinberger, "On feature normalization and data augmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12 383–12 392.

[30] D. Yin, R. Gontijo Lopes, J. Shlens, E. D. Cubuk, and J. Gilmer, "A fourier perspective on model robustness in computer vision," in *Advances in Neural Information Processing Systems*, 2019, pp. 13 276–13 286.

[31] A. Ashukha, A. Lyzhov, D. Molchanov, and D. Vetrov, "Pitfalls of in-domain uncertainty estimation and ensembling in deep learning," *arXiv preprint arXiv:2002.06470*, 2020.

[32] M. Zhang, S. Levine, and C. Finn, "Memo: Test time robustness via adaptation and augmentation," *arXiv preprint arXiv:2110.09506*, 2021.

[33] D. Molchanov, A. Lyzhov, Y. Molchanova, A. Ashukha, and D. Vetrov, "Greedy policy search: A simple baseline for learnable test-time augmentation," *arXiv preprint arXiv:2002.09103*, 2020.

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

[35] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[36] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do cifar-10 classifiers generalize to cifar-10?" *arXiv preprint arXiv:1806.00451*, 2018.

[37] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[39] D. Hendrycks and T. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," in *International Conference on Learning Representations*, 2018.

[40] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of machine learning research*, vol. 9, no. 11, 2008.