ClusT3: Information Invariant Test-Time Training

Gustavo A. Vargas Hakim* , David Osowiechi*,
Mehrdad Noori , Milad Cheraghalikhani , Ali Bahri , Ismail Ben Ayed , and Christian Desrosiers

ÉTS Montreal, Canada

Abstract

Deep Learning models have shown remarkable performance in a broad range of vision tasks. However, they are often vulnerable to domain shifts at test-time. Test-time training (TTT) methods have been developed in an attempt to mitigate these vulnerabilities, where a secondary task is solved at training time, simultaneously with the main task, to be later used as an self-supervised proxy task at test-time. In this work, we propose a novel unsupervised TTT technique based on the maximization of Mutual Information between multi-scale feature maps and a discrete latent representation, which can be integrated to the standard training as an auxiliary clustering task. Experimental results demonstrate competitive classification performance on different popular test-time adaptation benchmarks. The code can be found at: `https://github.com/dosowiechi/ClusT3.git`

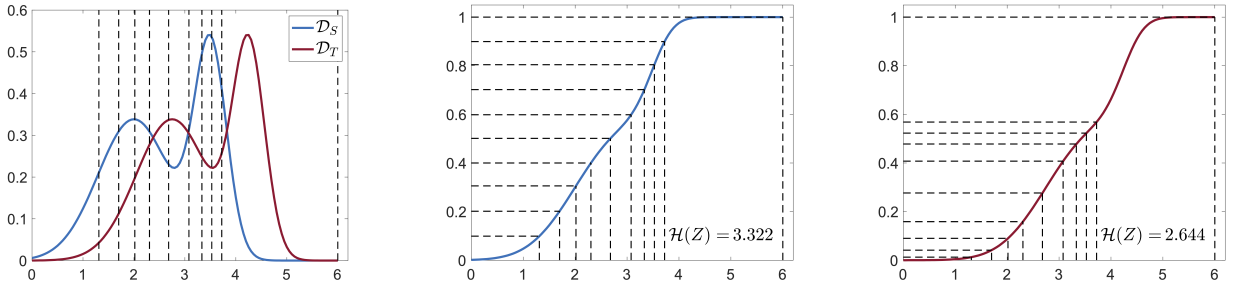
1. Introduction

The domain invariance hypothesis has been key to the success of deep learning methods for computer vision. In this hypothesis, the training and testing data are both assumed to be drawn from the same distribution, which rarely holds in practical settings. Moreover, it has been shown in numerous studies that the performance in classification and segmentation can drop significantly when domain shifts are present [26, 24]. In response, Domain Adaptation (DA) studies the adaptation of learning algorithms to new domains, when different types of domain shifts are present in the test data. From this field, two promising directions have emerged: Domain Generalization and Test-Time Adaptation. On the one hand, Domain Generalization (DG) [30, 25, 33, 14, 32] assumes a model is trained on a large source dataset composed of different domains, and evaluates the performances on new domains at test-time. On the other hand, Test-Time Adaptation (TTA) [31, 18, 13, 1] adapts the model to test data *on the fly*, typically adjusting to subsets of the new domain (e.g., mini-batches) each time. In TTA, there is no supervision from the testing samples nor access to the source domain, which makes it a challenging, yet realistic problem. The main limitation of DG is the requirement of a large amount of training data from different domains, without the guarantee that the model generalizes well to the (virtually unlimited) possible new domains it may encounter. TTA methods do not have this issue. However, they are highly sensitive to the choice of the unsupervised loss functions deployed at test-time, which may severely hurt the performances.

Test-Time Training (TTT) [28, 19, 7, 23] is an attractive variant of TTA, where an auxiliary task is learned from the training data (source domain) and later used at test-time to update a model. Typically, unsupervised and self-supervised tasks are chosen, as they allow for an adaptation process that does not require any label. Moreover, the joint, two-task training protocol for the source domain provides *momentum* at test-time, enabling the use of a loss function that is not completely foreign to the model.

Inspired by the recent success of Mutual-Information (MI) maximization in several learning tasks, such as representation learning [12, 10, 22, 29], deep clustering [11] and few-shot learning [2], we propose an information invariant TTT method called ClusT3. Our method maximizes the MI between the feature maps at different scales and discrete latent representations related to clustering. The main idea is that the amount of information between the features and their corresponding discrete encoding should remain constant in both the source and target domains (see Fig. 1). Toward this goal, we introduce an auxiliary task that performs information-maximization clustering while training on the source examples. At test time, we use the MI between the features and cluster assignments as a measure of representation quality, and maximize the MI as objective for test-time adaptation. Unlike previous TTT approaches, which rely

*Equal contribution. Correspondence to gustavo-adolfo.vargas-hakim.1@ens.etsmtl.ca, david.osowiechi.1@ens.etsmtl.ca



(a) Source $\mathcal{D}_S$ and Target $\mathcal{D}_T$ distributions (b) Cumulative density function of $\mathcal{D}_S$ (c) Cumulative density function of $\mathcal{D}_T$

Figure 1. **Illustration of our Information Invariant TTT method on a 1D feature space**. (*a*) The clustering of source features $\mathbf{x}$ (*blue*) into $K = 10$ regions, maximizing the entropy of the cluster marginal distribution $\mathcal{H}(Z)$, is such that regions have the same probability mass in the source distribution. At test-time, the probability density function of the target domain (*red*) is shifted, which results in a different clustering of features. (*b*) The optimal clustering corresponds to dividing the cumulative density function (CDF) in even steps, giving a cluster marginal entropy of $\mathcal{H}(Z) = \log_2(K) \approx 3.332$. (*c*) Since the CDF of the target is not divided in even steps, the mutual information between features $\mathbf{x}$ and clusters $\mathbf{z}$ is no longer maximized. Note: we assume that cluster assignments are confident, i.e., $\mathcal{H}(Z|X) \approx 0$ and thus $\mathcal{I}(Z;X) = \mathcal{H}(Z) - \mathcal{H}(Z|X) \approx \mathcal{H}(Z)$.

on problem-specific, self-supervised learning strategies, our auxiliary clustering task is problem-agnostic and could be added on top of any model via a low-dimensional linear projection. Test-time adaptation could also be done using only the test samples, without any type of distilled information from the source domain. On the technical side, minimal architectural changes are needed, and the joint training approach is more efficient than proceeding with multiple, complex and time-consuming steps.

Our contributions could be summarized as follows:

- We propose a novel Test-Time Training approach based on maximizing the MI between feature maps and discrete representations learned in training. At test time, adaptation is achieved based on the principle that information between the features and their discrete representation should remain constant across domains.
- ClusT3 is evaluated across a series of challenging TTA scenarios, with different types of domain shifts, obtaining competitive performance compared to previous methods.
- To the best of our knowledge, this is the first Unsupervised Test-Time Training approach using a joint training based on the MI and linear projectors. Our approach is lightweight and more general than its previous self-supervised counterparts.

The rest of this paper is organized as follows. Section 2 presents previous work in both TTA and TTT. Section 3 introduces the ClusT3 method with the experimental setting to evaluate it in Section 4. Experimental results and discussions are provided in Section 5, and the closing conclusions are given in Section 6.

2. Related Work

Test-Time Adaptation. The goal of TTA is to adapt a pre-trained model to a target dataset *on the fly*, i.e., as batches of data appear. Additional challenges include (1) the inaccessibility of source samples, which makes direct domain alignment impossible, (2) the lack of label supervision, which makes using unsupervised losses necessary, and (3) the fact that there is no access to all the target distribution, as the data come in the form of batches and not as a whole dataset. Adaptation can then be performed on different components of a network, such as the feature extractor, the classifier, or even the whole network.

Prediction Time Batch Normalization (PTBN) [21] proposes to use the feature mean and variance from the batch of test samples as statistics in the batch norm layers. TENT [31] instead focuses its adaptation on the affine parameters of the batch normalization layers only, based on the conditional entropy loss of the predictions. By updating linear parameters, the model can be more easily optimized and the source knowledge is preserved. SHOT [18] also freezes the classifier, but adapts the entire feature encoder by minimizing the uncertainty of predictions (low conditional entropy) while making them class-balanced (high entropy of class marginals). To circumvent the problem of erroneous predictions, the model also uses a pseudo-labeling mechanism coupled with cross-entropy as part of the final loss. LAME [1] reduces the adaptation focus even more, by only refining the classifier's predictions on the target batches. Given the fixed, pre-trained features obtained from the source domain, LAME performs a graph clustering based on optimizing a pairwise Laplacian term, which encourages nearby samples in the feature space to have the same cluster assignments.

Test-Time Training. In line with TTA methods, TTT seeks to update a model at test-time using an auxiliary task that

has been trained along the main classification objective during source training. TTT [28], which is among the first of such techniques, uses a Y-shaped architecture where a self-supervised rotation prediction network is attached to an arbitrary layer in the feature extractor of a CNN. A standard supervised cross-entropy loss ($\mathcal{L}_{\text{CE}}$) is optimized jointly with the auxiliary self-supervised loss $\mathcal{L}_{\text{aux}}$ of the secondary branch, as follows:

$$\mathcal{L}_{\text{TTT}} = \mathcal{L}_{\text{CE}} + \lambda \mathcal{L}_{\text{aux}} \tag{1}$$

At test-time, only the layers connected to the secondary branch are updated. The loss in Eq. (1) served as basis for subsequent TTT methods. TTT++ [19] introduced contrastive learning as the secondary task, similarly to TTT. However, to further improve performance at test-time, the statistics of source data are computed from a preserved queue of source feature maps. These statistics are then used for alignment with target data, thus regularizing the contrastive loss. TTT-MAE [7] proposes using Masked Autoencoders (MAE) [8] as the second branch for test-time training. This approach also introduced Vision Transformers [6] in the context of TTA and TTT. Different from standard TTT methods, TTTFlow [23] first pre-trains the model with a standard cross-entropy loss and then adds a Normalizing Flow (NF) [5, 15] as a secondary task on top of early encoder layers. The NF is trained on the source data independently of the classification task, by maximizing the log likelihood of source examples mapped to a simple distribution (Gaussian). The same loss function is later used to adapt the feature extractor to the target data.

3. Method

In this section, we present a formal definition of Test-Time Training, followed by the description of our ClusT3 method.

3.1. Problem formulation

Let $P(\mathcal{X}_s, \mathcal{Y}_s)$ be the joint distribution that represents the source domain, where $\mathcal{X}_s$ and $\mathcal{Y}_s$ are the input and label spaces, respectively. Similarly, $P(\mathcal{X}_t, \mathcal{Y}_t)$ corresponds to the target domain distribution, with inputs and labels $\mathcal{X}_t$ and $\mathcal{Y}_t$. In this work, we consider a likelihood shift [1] between the source and target datasets, i.e., $P(\mathcal{X}_s|\mathcal{Y}_s) \neq P(\mathcal{X}_t|\mathcal{Y}_t)$, with both domain sharing the same label space ($\mathcal{Y}_s = \mathcal{Y}_t$).

A standard TTT-based model is composed of a feature extractor f_θ, a classifier h_φ, and an auxiliary module g_ϕ, all collected inside the functional $F(f_\theta, h_\varphi, g_\phi)$. During training, the goal is to learn $F_s : \mathcal{X}_s \rightarrow \mathcal{Y}_s$ using Eq. (1), where the unsupervised loss $\mathcal{L}_{\text{aux}}$ is chosen to be related to the auxiliary task g_ϕ. At test-time, only the unsupervised loss is used to adapt the model, such that we learn an adapted function $F_t : \mathcal{X}_t \rightarrow \mathcal{Y}_t$.

3.2. Proposed method

ClusT3 is built on the formulation of previous work on TTT, following Eq. (1) and using modules plugged to the feature extractor. As shown in Fig. 2, we learn a discretized encoding of feature maps in the encoder using a clustering strategy based on MI maximization. Denote as $f_\theta(\mathbf{x}) \in \mathbb{R}^{N \times C}$ the combined features of examples in a batch of size B, where the first dimension $N = B \cdot W \cdot H$ is obtained by flattening along the batch index and feature map dimensions. We use a shallow projector g_ϕ to map $f_\theta(\mathbf{x})$ into a set of K-cluster probability distributions $\mathbf{z} = g_\phi(f_\theta(\mathbf{x})) \in [0,1]^{N \times K}$. In its simplest form, this projector is implemented by a single linear mapping followed by a softmax. A more complex projector, comprised of additional linear layers with ReLU activation can also be employed. We train the projector by maximizing the MI between $\mathbf{x}$ and its discrete representation $\mathbf{z}$:

$$\begin{aligned} \mathcal{L}_{\text{IM}} &= -\mathcal{I}(X;Z) = \mathcal{H}(Z|X) - \mathcal{H}(Z) \\ &= -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} z_{ik}\log z_{ik} + \sum_{k=1}^{K} \overline{z}_k \log \overline{z}_k \end{aligned} \tag{2}$$

where $\overline{z}_k = \frac{1}{N}\sum_i z_{ik}$ is the marginal probability of cluster K. The first term, $\mathcal{H}(\mathbf{z}|\mathbf{x})$, is the conditional entropy of $\mathbf{z}$ given $\mathbf{x}$. Minimizing this term enforces the model to make confident assignments of examples to clusters. The second term, $\mathcal{H}(\mathbf{z})$, evaluates the entropy of the cluster marginal distribution. Maximizing this term encourages the clusters to be balanced, and avoids the trivial solution of mapping all the examples to a single cluster.

In connection to information theory, our approach seeks a compressed encoding Z of features $U = f(X)$, modeled by a Markov chain $X \rightarrow U \rightarrow Z$, which best preserves information. Following the data processing inequality, we necessarily have that $\mathcal{I}(X;U) \geq \mathcal{I}(X;Z)$. The clustering defined by random variable Z divides the feature space in K regions. To maximize MI, it is known that the clustering must satisfy two conditions. First, it should divide the feature space in regions $\{\mathcal{R}_k\}_{k=1}^K$ of equal probability mass, i.e., $\int_{\mathcal{R}_k} p(u)du = \int_{\mathcal{R}_{k'}} p(u)du$, for any k, k' [20]. Second, the features falling into each region $\mathcal{R}_k$ should be similar, i.e., the entropy of U given Z should be low. Accordingly, increasing the number K of clusters leads to a higher MI. Assuming that the clustering in Z is a good representation of the distribution of features U, a shift in this distribution at test-time is likely to decrease MI since the shifted distribution is not well represented by Z.

Multi-scale clustering. In ClusT3, different projectors can be independently placed on top of different layer blocks of a CNN (e.g., ResNet). In such case, the output of the ℓ-th layer is now written as $\mathbf{z}^\ell = g_\phi(f_\theta^\ell(\mathbf{x}))$. At training time,

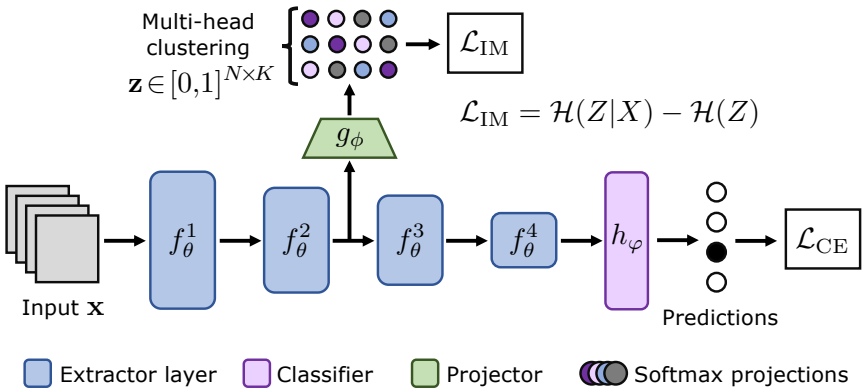


Figure 2. The configuration of ClusT3. A projector g_ϕ is plugged to the output of a feature extractor layer block to compute a set of N, $K-$dimensional latent points $\mathbf{z}$ that are clustered through Information Maximization ($\mathcal{L}_{\text{IM}}$). The cross-entropy loss ($\mathcal{L}_{\text{CE}}$) is used for the classification component of training.

the model learns with a combined loss

$$\mathcal{L}_{\text{CT3}} = \mathcal{L}_{\text{CE}} + \sum_{\ell=j}^{J} \mathcal{L}_{\text{IM}}^{\ell} \tag{3}$$

where j the index of the layer from which the first projector is connected. At test-time, the classifier h_φ and the projectors $\{g_\phi^\ell\}_{\ell=j}^J$ are frozen, and only the feature extractor f_θ up to layer J is updated based on the IM loss of Eq. (2). It is worth noting that the gradient flow is going to affect only the layer blocks connected to the projectors and the ones before. The hypothesis is that the latent space of the feature maps should be information invariant across domains, thus updating the encoder to maintain a high mutual information should also improve classification accuracy.

Multi-head clustering. As mentioned above, an encoding that better preserves information can be achieved by using a larger number of clusters. In practice, doing so might give poor results since the constraint of having balanced clusters (low entropy of the marginal) then becomes too restrictive. As better alternative, we propose a multi-head clustering strategy where multiple projectors $\{g_\phi^{\ell,c}\}_{c=1}^C$ are trained for a given layer ℓ and the loss $\mathcal{L}_{\text{IM}}^\ell$ for that layer is the sum of MI losses for all its projectors. The following lemma relates this strategy to our previous information theory analysis.

Lemma 3.1. *Let $\mathcal{Z} = \{Z_1, \dots, Z_C\}$ be a set of random discrete variables representing C cluster assignments of features X. The MI between X and $\mathcal{Z}$ is bounded as follows*

$$\max_c \mathcal{H}(Z_c) - \sum_c \mathcal{H}(Z_c|X) \leq \mathcal{I}(X;\mathcal{Z}) \leq \sum_c \mathcal{I}(X;Z_c)$$

Proof. We start by writing the MI between X and $\mathcal{Z}$ as

$$\mathcal{I}(X;\mathcal{Z}) = \mathcal{H}(Z_1,\dots,Z_C) - \mathcal{H}(Z_1,\dots,Z_C|X).$$

The second term on the right simplifies as

$$\begin{aligned}\mathcal{H}(Z_1,\dots,Z_C|X) &= -\mathbb{E}\big[\log p(Z_1,\dots,Z_C|X)\big] \\ &= -\mathbb{E}\big[\sum_c \log p(Z_c|X)\big] \\ &= \sum_c \mathcal{H}(Z_c|X)\end{aligned}$$

where we used the fact that the Z_c variables are conditionally independent given X. To complete the proof, we use the following two properties of entropy: $\mathcal{H}(Z_1,\dots,Z_C) \leq \sum_c \mathcal{H}(Z_c)$ and $\mathcal{H}(Z_1,\dots,Z_C) \geq \max_c \mathcal{H}(Z_c)$. □

Note that the upper bound on $\mathcal{I}(X;\mathcal{Z})$, which corresponds to our multi-head clustering objective, is tight if the clustering variables Z_c are statistically independent. Although we do not enforce this constraint, since our objective maximizes $\mathcal{H}(Z_c)$ for *each* cluster, the lower bound of the lemma tells us that we can indirectly maximize mutual information with the same objective.

4. Experimental Setup

ClusT3 is evaluated on four popular TTA/TTT benchmarks, comprehending different types of domain shifts. The first two benchmarks are based on the CIFAR-10 dataset [16] as the source domain. It contains 50,000 images from 10 different categories.

Common image corruptions. First, we study adaptation on the CIFAR-10-C [9] dataset, which consists of 15 different corruption types (e.g., Gaussian noise, frost, etc.) with 10,000 images, 10 classes, and 5 different severity levels for each type. This results in 75 evaluation scenarios. We then extend the evaluation to CIFAR-100-C, scaling the number of classes to 100.

Natural domain shift. We also evaluate the performance of our method in a natural domain shift setting, i.e., classifying images that were manually selected to diverge from

those seen in training. The CIFAR-10.1 dataset [27] is used for this experiment, consisting of 2,000 images strategically sampled from CIFAR-10 to highly differ from training data.

Sim-to-real domain shift. ClusT3 is finally assessed in the context of large-scale adaptation from simulation to real images. The VisDa-C dataset [24] offers a benchmark with a source dataset based on 3D renderings of 12 different object categories, accumulating a total of 152,397 images. The test set comprises 72,372 video frames, corresponding to real images of the same classes.

4.1. Joint training

For the joint training on the CIFAR-10 dataset [16], we followed previous research and trained our model for 350 epochs with SGD, using a batch size of 128 images and an initial learning rate of 0.1 which is reduced by a factor of 10 at epochs 150 and 250. For VisDA-C, the model is warm-started with pre-trained weights from ImageNet [4], according to the protocol in [31, 19, 18], and then trained for 100 epochs with a batch size of 100, using SGD with a learning rate of 0.001. The training was executed on four 16 GB NVIDIA V100 GPUs.

4.2. Test-time adaptation

At test-time, projectors are used to detect distribution shift with the IM loss. For all the experiments with CIFAR-10-C and CIFAR-10.1, we keep a batch size of 128, and use the ADAM optimizer with 10^{-5} as learning rate. For VisDA-C, we used a batch size of 32 images with the same aforementioned learning rate. We update the extractor and the statistics of all the BatchNorm layers. To avoid the error accumulation associated to optimization, we reset our weights to the initial source ones after adapting to each batch. This way, each batch can have different corruptions as assumed by [28] in their offline mode.

5. Results and discussion

First, we perform a series of ablation experiments on the CIFAR-10-C dataset, and then compare ClusT3 against state-of-art approaches. Afterward, we extend our evaluation to natural domain shift using the CIFAR-10.1 dataset and sim-to-real domain shift with the VisDA-C dataset. For all methods, we compute the accuracy for 1, 3, 5, 10, 20, 50 and 100 iterations and report the maximum accuracy when we experiment on CIFAR-10-C and CIFAR-10.1 and do the same for VisDAC by adapting for 1, 3, 10, 15, and 20 iterations. For all experiments, we report the mean and standard deviation accuracy obtained over 3 runs with different random seeds.

	Gaussian Noise	Shot Noise	Snow
Layer 1	70.72 ±0.22	**73.57 ±0.11**	80.29 ±0.04
Layer 2	67.48 ±0.09	68.96 ±0.02	78.46 ±0.10
Layer 3	66.57 ±0.06	67.97 ±0.22	78.84 ±0.17
Layer 4	65.75 ±0.12	68.10 ±0.31	79.37 ±0.11
Layers 1-2	**71.36 ±0.03**	72.93 ±0.34	**80.94 ±0.13**
Layers 2-3	66.74 ±0.24	68.76 ±0.07	78.21 ±0.12
Layers 3-4	65.21 ±0.32	67.09 ±0.15	78.34 ±0.18
Layers 1-2-3	67.44 ±0.11	68.59 ±0.14	79.27 ±0.05
Layers 1-2-3-4	68.71 ±0.18	71.39 ±0.12	78.38 ±0.14

Table 1. Accuracy (%) with different combinations of projectors on 3 corruptions of CIFAR-10-C dataset. Layer l means that we only use the projector after layer l, and Layer l-l means that we use the sum of the two projectors' losses of these layers as total IM loss. The extractor ends at the last named layer.

5.1. Object recognition on corrupted images

First, we evaluate ClusT3 on the CIFAR-10-C dataset across the 15 different corruptions. For the following experiments, we focus solely on the Level 5, as it is the most challenging adaptation scenario. Extensive results on all the severity levels can be found in the supplementary material.

On which layers should projectors be placed? We compare the accuracy of ClusT3 on different combinations of projectors. The goal is to determine which layers are the most useful to adapt at test-time. In Table 1, the results show that only taking the first two encoder layers provides more effective results. Indeed, as assumed in [28, 19, 23], the first layers seem to contain the most important domain-related information. This finding also aligns with empirical evidence demonstrating that different layers are sensitive to different types of domain shifts [17]. Hence, in subsequent experiments, we keep projectors on Layer 1 and Layer 2.

On the number of clusters. As explained in Section 3.2, the proxy task consists of a projector-based clustering head made by a linear mapping (implemented with a 1×1 convolution) followed by a K-way softmax that projects features to a cluster probability map $\mathbf{z} \in [0,1]^{BWH\times K}$. In Table 2, we experiment with different number of clusters. Results show that having a greater number of clusters, e.g., $K = 100$, can provide a better accuracy. We also notice that having $K=10$ (corresponding to the number of classes in CIFAR-10-C) results in a competitive performance compared to other larger values, such as $K=20$ or $K=50$. This becomes a sensible approach, as projectors can help learn better class boundaries inside features. In the next experiments, we keep $K = 10$ for an efficient trade-off between performance and computational cost.

On the number of projectors per layer. In the previous experiments, only one projector per layer was used. Here, we evaluate whether having more projectors per layer can

	Gaussian	Shot	Snow	Avg*
K=2	71.58 ±0.12	73.41 ±0.09	82.98 ±0.10	80.39
K=5	71.10 ±0.09	72.89 ±0.15	**83.76 ±0.09**	80.40
K=10	**72.96 ±0.13**	**74.55 ±0.12**	83.61 ±0.09	**80.94**
K=20	70.13 ±0.12	72.35 ±0.10	83.29 ±0.09	80.10
K=50	71.54 ±0.18	74.15 ±0.07	83.39 ±0.12	80.70
K=100	68.47 ±0.11	70.82 ±0.11	82.51 ±0.08	79.77

*: Average over the 15 corruption types

Table 2. Accuracy (%) with different number of clusters on 3 corruptions of CIFAR-10-C dataset.

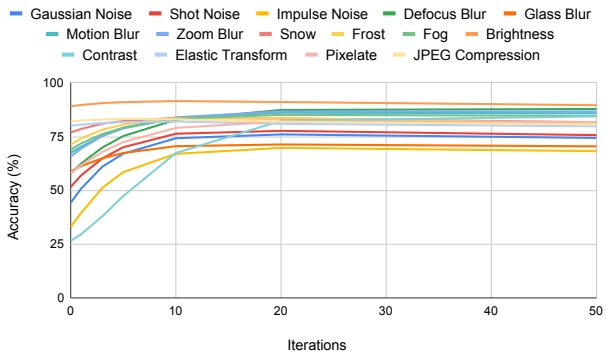


Figure 3. Evolution of accuracy for all corruptions in CIFAR-10-C.

further improve performance. It has been found that increasing the number of projectors per layer increases accuracy compared to using a single projector per layer (Table 3). However, each corruption in CIFAR-10-C can be benefited differently from different configurations. On the average, using 15 projectors on layers 1 and 2 results corresponds to the best option. In the following experiments, we compare this architecture (called ClusT3-H15) to the leading Test-Time Adaptation methods.

Comparison of the number of iterations. As shown in Fig 3, in most cases, the best accuracy is obtained after 10 or 20 iterations, depending on the corruption. Most importantly, accuracy remains constant even after 20 iterations. Furthermore, we observe that adaptation to strong corruptions (e.g., contrast) can also be done at a fast rate.

Comparison with main TTA methods. Several *state-of-the-art* TTA/TTT techniques were chosen for comparison: TTA methods include TENT[31], LAME[1], and PTBN[21]. TTT[28] and TTT++[19] are chosen for Test-Time Training. As shown in Table 4, the overall performance of ClusT3-H15 on all the corruptions outperforms ResNet50 with a gain of 28.26% as well as all the different TTA methods. Moreover, there is a considerably large improvement on all the individual corruptions with respect to the same baseline. A significant increase in accuracy can also be observed in most corruptions compared to previous methods, with some exceptions (e.g., Defocus blur against TTT++[19] or Contrast against TTT[28]). It is however important to mention that ClusT3 differs from previous TTT methods whose self-supervised secondary task requires a higher computational overhead. TTT++, which improves considerably with respect to its predecessor TTT on Level 5, also requires preserving a queue of source feature maps to compare statistics at test-time. In comparison, ClusT3 is self-sufficient and less costly in both computation and memory. A more detailed comparison on all the corruption levels of CIFAR-10-C can be found in the supplementary material.

Table 5 shows the overall performance of ClusT3 on CIFAR-100-C, in an effort to demonstrate the scalable capabilities of the method on a larger set of classes. ClusT3 mitigates the natural degradation of the ResNet50 baseline, while also outperforming *state-of-the-art* methods by an important margin.

Visualization of adaptation. To visualize the effect of ClusT3 during adaptation, Figure 4 displays the t-SNE plots of the target feature maps before and after the adaptation with the corresponding model prediction. The projector induces the model to make better predictions by improving the clustering of the different samples' classes in the target dataset.

5.2. Object recognition on natural domain shift

The best configuration of ClusT3 (i.e., with 5 in that case or 15 projectors on Layer 1 and 2) is evaluated on CIFAR-10.1, which contains a more natural domain shift. A comparison is made against previous TTA methods, and as reported in Table 6, ClusT3 achieves a competitive accuracy despite the baseline (ResNet50) being the most accurate in this scenario. The gain of TTT++ [19] comes from a better pre-trained encoder thanks to the influence of contrastive learning [3]. This limitation can be explained by the fact that CIFAR-10 and CIFAR-10.1 are similar, thus having a smaller domain shift [23].

5.3. Object recognition on sim-to-real domain shift

We use the VisDA-C dataset to test ClusT3 on the sim-to-real domain shift. To account for the challenge of this scenario, a slightly different projector is proposed: using two linear (1×1 convolutional) layers with a ReLU activation in between. The output number of channels of the first layer is set to half the input feature maps' number of channels. This setting is named *Large* projector. The best configuration (i.e., type of projector, number of projectors, and combination of layers) was found based on a hyperparameter study that can be found in the supplementary material. The resulting best approach consisted in using one large projector on Layer 2 (ClusT3-H1*).

Comparison with other methods. Our method is compared against the previously presented, popular TTT/TTA

	Head = 1	Heads = 5	Heads = 10	Heads = 15	Heads = 20
Gaussian Noise	71.40 ±0.26	72.72 ±0.08	75.24 ±0.02	76.01 ±0.19	**76.04 ±0.20**
Shot noise	72.79 ±0.04	74.84 ±0.14	76.77 ±0.04	77.67 ±0.17	**78.00 ±0.05**
Impulse Noise	65.96 ±0.12	67.78 ±0.06	68.62 ±0.07	**69.76 ±0.15**	68.80 ±0.23
Defocus blur	82.77 ±0.09	87.83 ±0.09	**87.91 ±0.14**	87.85 ±0.11	87.86 ±0.19
Glass blur	69.65 ±0.14	65.85 ±0.04	**71.70 ±0.12**	71.34 ±0.15	67.26 ±0.07
Motion blur	82.03 ±0.17	86.58 ±0.07	86.44 ±0.03	86.10 ±0.11	**86.91 ±0.06**
Zoom blur	83.88 ±0.09	86.83 ±0.06	**87.21 ±0.09**	86.68 ±0.05	87.57 ±0.06
Snow	80.87 ±0.04	82.68 ±0.13	83.41 ±0.06	**83.71 ±0.09**	83.17 ±0.06
Frost	79.04 ±0.07	81.38 ±0.14	83.39 ±0.03	**83.69 ±0.03**	82.45 ±0.11
Fog	76.32 ±0.09	84.40 ±0.05	84.47 ±0.14	**85.12 ±0.13**	83.98 ±0.04
Brightness	89.16 ±0.10	**92.29 ±0.11**	91.91 ±0.03	91.52 ±0.02	91.81 ±0.02
Contrast	74.57 ±0.25	85.28 ±0.09	84.37 ±0.07	84.40 ±0.11	**85.67 ±0.08**
Elastic transform	80.16 ±0.16	80.07 ±0.13	**82.33 ±0.04**	82.04 ±0.17	82.02 ±0.09
Pixelate	80.09 ±0.02	79.94 ±0.04	**82.75 ±0.06**	82.03 ±0.09	82.00 ±0.07
JPEG compression	80.90 ±0.01	79.86 ±0.08	83.01 ±0.08	**83.24 ±0.10**	82.38 ±0.07
Average	77.97	80.56	81.97	**82.08**	81.73

Table 3. Accuracy (%) with different number of projectors per layer on Layer 1 and 2 with $K = 10$ on the CIFAR-10-C dataset.

	ResNet50	LAME [1]	PTBN [21]	TENT [31]	TTT [28]	TTT++ [19]	ClusT3-H15 (Ours)
Gaussian Noise	21.01	22.90	57.23 ±0.13	57.15 ±0.19	66.14 ±0.12	75.87 ±5.05	**76.01 ±0.19**
Shot noise	25.77	27.24	61.18 ±0.03	61.08 ±0.18	68.93 ±0.06	77.18 ±1.36	**77.67 ±0.17**
Impulse Noise	14.02	30.99	54.74 ±0.13	54.63 ±0.15	56.65 ±0.03	**70.47 ±2.18**	69.76 ±0.15
Defocus blur	51.59	45.38	81.61 ±0.07	81.39 ±0.22	**88.11 ±0.08**	86.02 ±1.35	87.85 ±0.11
Glass blur	47.96	36.66	53.43 ±0.11	53.36 ±0.14	60.67 ±0.06	69.98 ±1.62	**71.34 ±0.15**
Motion blur	62.30	55.29	78.20 ±0.28	78.04 ±0.17	83.52 ±0.03	85.93 ±0.24	**86.10 ±0.11**
Zoom blur	59.49	51.40	80.29 ±0.13	80.26 ±0.22	87.25 ±0.03	**88.88 ±0.95**	86.68 ±0.05
Snow	75.41	66.17	71.59 ±0.21	71.59 ±0.04	79.29 ±0.05	82.24 ±1.69	**83.71 ±0.09**
Frost	63.14	49.98	68.77 ±0.25	68.52 ±0.20	79.84 ±0.11	82.74 ±1.63	**83.69 ±0.03**
Fog	69.63	64.49	75.79 ±0.05	75.73 ±0.10	84.46 ±0.09	84.16 ±0.28	**85.12 ±0.13**
Brightness	90.53	84.26	84.97 ±0.05	84.77 ±0.13	91.23 ±0.08	89.07 ±1.20	**91.52 ±0.02**
Contrast	33.88	31.50	80.81 ±0.15	80.70 ±0.15	**88.58 ±0.09**	86.60 ±1.39	84.40 ±0.11
Elastic transform	74.51	64.16	67.14 ±0.17	67.13 ±0.10	75.69 ±0.10	78.46 ±1.83	**82.04 ±0.17**
Pixelate	44.43	39.34	69.17 ±0.31	68.70 ±0.29	76.35 ±0.19	**82.53 ±2.01**	82.03 ±0.09
JPEG compression	73.61	66.05	65.86 ±0.05	65.83 ±0.07	73.10 ±0.19	81.76 ±1.58	**83.24 ±0.10**
Average	53.82	49.05	70.05	69.93	77.32	81.46	**82.08**

Table 4. Accuracy (%) on CIFAR-10-C dataset with Level 5 corruption for ClusT3-15 compared to ResNet50, LAME, PTBN, TENT, TTT, and TTT++.

Method	Acc. (%)
ResNet50	31.37
LAME	29.63
PTBN	54.53
TENT	54.48
TTT	51.43
Ours	**56.70**

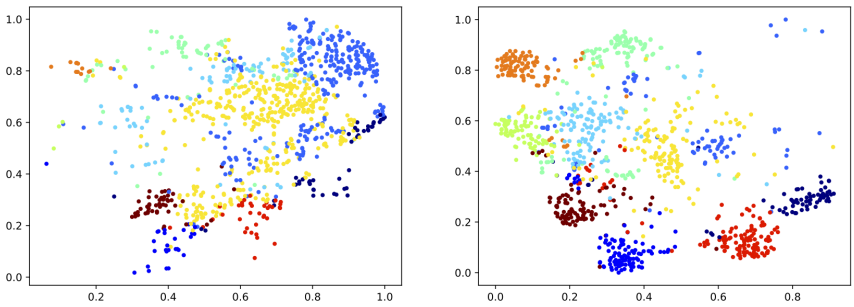
Table 5. Results on the CIFAR-100-C dataset.

Method	Accuracy (%)
ResNet50	**88.45**
LAME [1]	82.68
PTBN [21]	79.57 ±0.47
TENT [31]	79.69 ±0.21
TTT [28]	86.30 ±0.20
TTT++ [19]	88.03 ±0.17
ClusT3-H5 (Ours)	87.43 ±0.02
ClusT3-H15 (Ours)	85.57 ±0.11

Table 6. Accuracy of compared methods on the CIFAR-10.1 dataset containing natural domain shift.

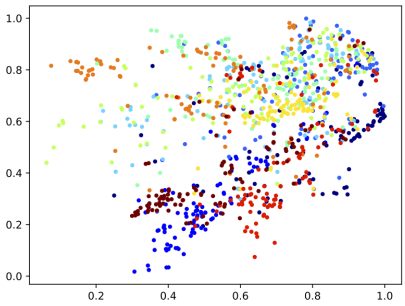
methods. For fairness, we evaluate LAME [1] using the three proposed affinity matrices in its original publication: LAME-L (linear affinity), LAME-K (K-NN affinity with 5 neighbors), and LAME-R (RBF affinity with 5 neighbors). As shown in Table 7, ClusT3 achieves a higher performance than its competitors in the reproduced experiments. With respect to the baseline, ClusT3 obtains a gain of around 15.6%.

Computational cost. The nature of the auxiliary tasks in Test-Time Training methods can importantly impact the training efficiency. For instance, methods based on self-supervised learning might require additional forward passes, or a higher memory input, which ultimately increases the computation time. ClusT3 does not depend on additional data transformations, hence reducing execution times. We evaluate the time of one epoch of joint train-

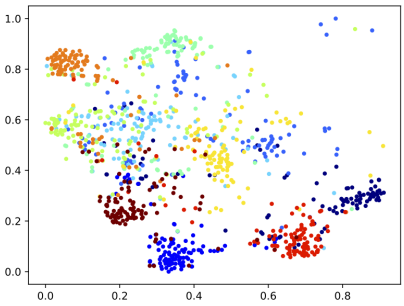


(a) Prediction (before adaptation)

(b) Prediction (after adaptation)



(c) Ground truth (before adaptation)



(d) Ground truth (after adaptation)

Figure 4. t-SNE plots of gaussian noise for the features at the output of the extractor from ClusT3 with one projector on Layer 1 and 2 each. (a) prediction of the model without adaptation. (b) prediction of the model after 20 iterations of adaptation. (c) ground truth labels without adaptation. (d) ground truth labels of adapted representations.

ing (without evaluation steps) of ClusT3, utilizing 1 Large projector on top of all layers, one of the heaviest configurations. The average execution time of one epoch was 2.7947 ± 0.0294 minutes, compared to 12.4941 ± 1.3994 minutes for TTT.

Method	Accuracy (%)
ResNet50	46.31
LAME-L [1]	22.02 $\pm$0.23
LAME-K [1]	42.89 $\pm$0.14
LAME-R [1]	19.33 $\pm$0.11
PTBN [21]	60.33 $\pm$0.04
TENT [31]	60.34 $\pm$0.05
TTT [28]	40.57 $\pm$0.02
TTT++ [19]	60.42†
ClusT3-H1* (Ours)	**61.91** $\pm$**0.02**

Table 7. Accuracy values of ClusT3 and the *state-of-the-art* TTT/TTA methods on the VisDA-C dataset. †: Result of TTT++ obtained from the original paper, were not reproducible.

6. Conclusion

In this work, we proposed ClusT3, a new unsupervised Test-Time Training framework based on Information Maximization of feature latent spaces across domains. This method allows adapting the model at test-time when there is a distribution shift between the source and the target datesets. By using simple linear projectors and Mutual Information in our proxy task, we update the feature extractor to improve the accuracy at test-time.

A complete ablation study helped determine the best hyperparameters and to better understand the different possible configurations of the model. As shown in our experimental results, ClusT3 obtains a highly-competitive performance against previous TTT and TTA models. Thus, on the CIFAR-10-C dataset, ClusT3 outperforms *state-of-the-art*. Surprisingly, the baseline defeats all previous methods on CIFAR-10.1, as the domain shift with respect to the source dataset is smaller and adaptation causes performance degradation. Nonetheless, ClusT3 remains competitive and robust to this scenario.

Future work includes further investigation on different architectures for the projector. As it has been shown, adding layers and nonlinearity can further improve performance in some cases. This could be due to the fact that having more complex and thus flexible projectors relaxes constraints on the feature space (e.g., balanced clusters) which can hurt the learning of a good representation for classification if too strong. Additionally, a uniform distribution has been assumed for the cluster marginal distribution. Diverging from this premise and exploring other distribution priors also constitutes an interesting line of future research. This can turn particularly useful in the scenario where adaptation to a single data sample is required.

References

[1] Malik Boudiaf, Romain Mueller, Ismail Ben Ayed, and Luca Bertinetto. Parameter-free online test-time adaptation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8344–8353, 2022.

[2] Malik Boudiaf, Imtiaz Masud Ziko, Jérôme Rony, Jose Dolz, Pablo Piantanida, and Ismail Ben Ayed. Transductive information maximization for few-shot learning. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR, 13–18 Jul 2020.

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[5] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[7] Yossi Gandelsman, Yu Sun, Xinlei Chen, and Alexei A Efros. Test-time training with masked autoencoders. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[8] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

[9] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. 2019.

[10] Weihua Hu, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto, and Masashi Sugiyama. Learning discrete representations via information maximizing self-augmented training. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1558–1567. PMLR, 06–11 Aug 2017.

[11] Mohammed Jabi, Marco Pedersoli, Amar Mitiche, and Ismail Ben Ayed. Deep clustering: On the link between discriminative models and k-means. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(6):1887–1896, 2021.

[12] Xu Ji, Joao F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9865–9874, 2019.

[13] Ansh Khurana, Sujoy Paul, Piyush Rai, Soma Biswas, and Gaurav Aggarwal. Sita: single image test-time adaptation. *arXiv:2112.02355 [cs]*, Dec. 2021. arXiv: 2112.02355.

[14] Donghyun Kim, Kaihong Wang, Stan Sclaroff, and Kate Saenko. A broad study of pre-training for domain generalization and adaptation. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, pages 621–638, Cham, 2022. Springer Nature Switzerland.

[15] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[16] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[17] Yoonho Lee, Annie S Chen, Fahim Tajwar, Ananya Kumar, Huaxiu Yao, Percy Liang, and Chelsea Finn. Surgical fine-tuning improves adaptation to distribution shifts. In *The Eleventh International Conference on Learning Representations*, 2023.

[18] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. 2020.

[19] Yuejiang Liu, Parth Kothari, Bastien van Delft, Baptiste Bellot-Gurlet, Taylor Mordan, and Alexandre Alahi. Ttt++: When does self-supervised test-time training fail or thrive? *Neural Information Processing Systems (NeurIPS)*, 2021.

[20] David JC MacKay and David JC Mac Kay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[21] Zachary Nado, Shreyas Padhy, D. Sculley, Alexander D'Amour, Balaji Lakshminarayanan, and Jasper Snoek. Evaluating prediction-time batch normalization for robustness under covariate shift. *arXiv:2006.10963 [cs, stat]*, Jan. 2021. arXiv: 2006.10963.

[22] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[23] D. Osowiechi, G. A. Vargas Hakim, M. Noori, M. Cheraghalikhani, I. Ayed, and C. Desrosiers. Tttflow: Unsupervised test-time training with normalizing flow. In *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2125–2126, Los Alamitos, CA, USA, jan 2023. IEEE Computer Society.

[24] Xingchao Peng, Ben Usman, Neela Kaushik, Dequan Wang, Judy Hoffman, and Kate Saenko. Visda: A synthetic-to-real benchmark for visual domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2018.

[25] Aayush Prakash, Shaad Boochoon, Mark Brophy, David Acuna, Eric Cameracci, Gavriel State, Omer Shapira, and Stan Birchfield. Structured domain randomization: Bridging the reality gap by context-aware synthetic data. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7249–7255. IEEE, 2019.

[26] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do CIFAR-10 classifiers generalize to cifar-10? *CoRR*, abs/1806.00451, 2018.

[27] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5389–5400. PMLR, 09–15 Jun 2019.

[28] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei A. Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International Conference on Machine Learning (ICML)*, 2020.

[29] Michael Tschannen, Josip Djolonga, Paul K Rubenstein, Sylvain Gelly, and Mario Lucic. On mutual information maximization for representation learning. In *International Conference on Learning Representations*, 2020.

[30] Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John C Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to unseen domains via adversarial data augmentation. *Advances in neural information processing systems*, 31, 2018.

[31] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. 2021.

[32] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip Yu. Generalizing to unseen domains: A survey on domain generalization. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[33] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. In *International Conference on Learning Representations*, 2020.