Effective Restoration of Source Knowledge in Continual Test Time Adaptation

Fahim Faisal Niloy[1], Sk Miraj Ahmed[1], Dripta S. Raychaudhuri[2,*], Samet Oymak[3], Amit K. Roy-Chowdhury[1]
[1]University of California, Riverside [2]AWS AI Labs [3]University of Michigan, Ann Arbor
{fnilo001@, sahme047@, drayc001@, amitrc@ece.}ucr.edu, oymak@umich.edu

Abstract

Traditional test-time adaptation (TTA) methods face significant challenges in adapting to dynamic environments characterized by continuously changing long-term target distributions. These challenges primarily stem from two factors: catastrophic forgetting of previously learned valuable source knowledge and gradual error accumulation caused by miscalibrated pseudo labels. To address these issues, this paper introduces an unsupervised domain change detection method that is capable of identifying domain shifts in dynamic environments and subsequently resets the model parameters to the original source pre-trained values. By restoring the knowledge from the source, it effectively corrects the negative consequences arising from the gradual deterioration of model parameters caused by ongoing shifts in the domain. Our method involves progressive estimation of global batch-norm statistics specific to each domain, while keeping track of changes in the statistics triggered by domain shifts. Importantly, our method is agnostic to the specific adaptation technique employed and thus, can be incorporated to existing TTA methods to enhance their performance in dynamic environments. We perform extensive experiments on benchmark datasets to demonstrate the superior performance of our method compared to state-of-the-art adaptation methods.

1. Introduction

Deep neural networks (DNNs) have demonstrated remarkable success in numerous applications. However, they are known to suffer from performance degradation when faced with distributional shifts between the training and test data. This poses a significant risk in deploying DNNs in domains such as autonomous driving or medical imaging, where encountering unseen types of test data during deployment could result in undesirable consequences. To overcome this challenge, test-time adaptation (TTA) [20,31] has emerged as a promising approach.

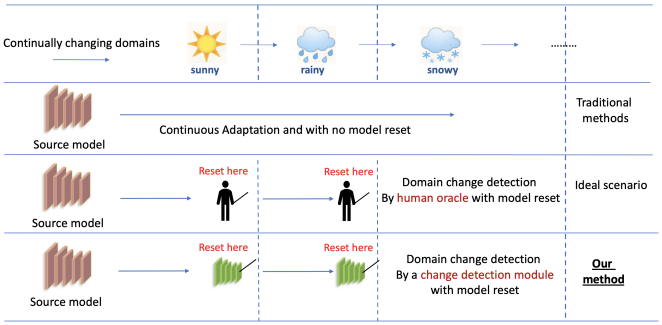

Figure 1. **Problem setup.** Traditional test-time adaptation methods suffer from forgetting and error accumulation over time as they continuously adapt to incoming target distributions. Previous research has demonstrated the benefits of resetting the model to its original parameters when a domain change occurs, but these approaches rely on an oracle with additional domain knowledge for detecting such changes. In this paper, we propose an automated strategy to detect domain changes, allowing for efficient and effective model resets in practical scenarios.

TTA aims to adapt DNNs to the unseen target domain using only unlabeled test data streams, without the need for a substantial portion of the test data to be available as in traditional domain adaptation settings [31], and without accessing the source data used for training the model. This enables efficient and on-the-fly adaptation to distribution shifts, making it computationally efficient and highly effective in real-world scenarios.

Existing TTA approaches typically focus on adapting to distribution shifts between a fixed source domain and a target domain. However, in real-world scenarios, the target domain may not remain static and can continuously evolve. For instance, a self-driving car model trained on data from clear weather conditions may encounter test data from diverse weather conditions such as rain, snow, fog, etc. In such cases, it is crucial for the model to adapt its parameters to these new domains accordingly in a continual fashion to ensure optimal performance. This particular setting is often referred to as continual test time adaptation in the literature [25, 32]. Current TTA methods mostly fail to account for such dynamic domain changes due to two primary reasons:

*Currently at AWS AI Labs. Work done while the author was at UCR.

- **Catastrophic Forgetting**: Over extended periods of continuous adaptation to new distributions, the model may experience catastrophic forgetting, wherein the knowledge learned from the source domain is gradually lost [32]. This can potentially erase valuable information learned from the source domain and can have a negative impact on the model's performance when adapting to subsequent new domains.
- **Error Accumulation**: Several TTA methods [31, 36] leverage pseudo-labels for unsupervised adaptation. In a continually changing environment, the dynamic distribution shift can cause the pseudo-labels to become progressively noisier and miscalibrated. Thus, early prediction errors are more prone to propagate and accumulate over time, potentially leading to error accumulation.

To effectively tackle the challenges of catastrophic forgetting and error accumulation in dynamic environments, we propose to *detect the underlying domain changes* and *restore the source knowledge accordingly*. Specifically, we observe that the KL divergence between pre-trained batch-norm statistics and the batch-norm statistics of incoming test batches can effectively quantify domain shifts. Based on this insight, we develop an approach to detect domain changes in dynamic environments. We maintain an exponential running average of the batch-norm statistics of incoming test batches to progressively estimate the global batch-norm statistics specific to each domain, using an adaptive momentum value determined by the KL divergence between the running batch-norm statistics and the incoming test batch statistics. When a sudden distribution shift occurs in the incoming test batches, the momentum value undergoes a sharp change. By monitoring this, we can identify outliers caused by domain shifts.

Using this approach, our method detects changes in the distribution of a dynamic environment and appropriately restores the model's source knowledge. The primary objective is to prevent the forgetting of previously acquired source knowledge and address the cumulative negative impact of noisy and miscalibrated pseudo-labels. As a result, our method enables more effective model adaptation. This makes it highly suitable for dynamic scenarios where the target domain may experience shifts over time.

Also, our domain change detection module is independent of the adaptation method used. Therefore, our proposed module can be seamlessly integrated into existing TTA methods, enhancing their robustness to distribution changes in dynamic environments. It is important to note that the test time adaptation literature [20,32] often assumes an 'online' version of the models, where an oracle with additional ground-truth domain knowledge is available to restore the source knowledge when a domain change occurs. Such online models are inherently resilient to the forgetting issue. However, the requirement of ground-truth knowledge renders the models impractical for application in real-world scenarios. This paper is the first attempt to develop an approach that effectively serves as this oracle, yet without requiring any ground-truth domain knowledge, enabling the practical implementation of such online models.

Main contributions. To summarize, our primary contributions are as follows:

- We address a novel problem of detecting domain changes in the context of continual test time adaptation, which focuses on dynamic environments where the target distribution undergoes continual domain shifts.
- We demonstrate that detecting such domain changes and thereby restoring the source knowledge has the ability to mitigate catastrophic forgetting and error accumulation.
- We extensively evaluate our proposed method on real-world datasets, encompassing a wide range of tasks. Through these experiments, we provide substantial empirical evidence that demonstrates the effectiveness and applicability of our approach.

2. Related Work

Unsupervised Domain Adaptation. Unsupervised domain adaptation (UDA) aims to enhance the performance of a pre-trained source model when there is a distribution shift between the labeled source domain and the unlabeled target domain. UDA has been extensively applied in various computer vision tasks, including image classification [27], semantic segmentation [26], object detection [9], and reinforcement learning [21]. Existing approaches typically focus on aligning the distributions of the source and target domains using techniques like maximum mean discrepancy [18], adversarial learning [7, 27], and more. Recently, there has been a growing interest in adaptation using only a pre-trained model without the need for source data. This shift is motivated by privacy and memory storage concerns associated with the source data. These approaches leverage techniques such as information maximization [1, 2, 15], pseudo-labeling [13, 35], and self-supervision [33].

Test Time Adaptation. UDA methods typically require a significant amount of target domain data to adapt a model, regardless of whether they utilize source data. Thus, the adaptation process is performed offline, meaning that it happens before the model is deployed or used for inference on the target domain. In contrast, test time adaptation (TTA) approaches adapt a model to the target data after deployment, i.e., during inference or testing phase. It involves updating the model's parameters or internal representations during inference based on the characteristics of the current test batch from the target domain to improve performance on the subsequent test batches. TTA approaches also do not require the source data to be available during adaptation.

In an early work [14], authors leverage the batch-norm

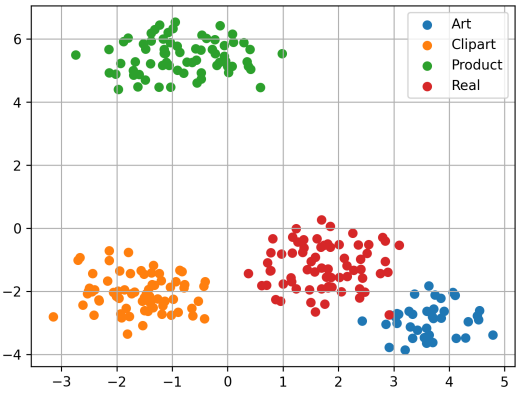

Figure 2. **BN statistics for domain separation.** t-SNE diagram of the batch-norm means extracted from the last CNN layer of ResNet-18 model. Clearly, the extracted batch-norm statistics allow for a clear separation between domains.

statistics of incoming test batches to adapt the model to the target distribution instead of relying on pre-trained batch-norm statistics. TENT [31] adapts a pre-trained source model on incoming target data by minimizing entropy and updating the batch-norm parameters of the source model. DUA [19] continuously updates the batch-norm statistics of the pre-trained source model with the incoming test batches in order to align to the target distribution. TTA methods have also been extended to the segmentation task [10, 17, 24, 28].

TTA methods can also be used to adapt during inference to a dynamically varying target distribution, that is, where the target distribution changes after different time intervals. In this case, TTA methods usually suffer from the problem of error accumulation and catastrophic forgetting - continually drifting away from source knowledge. Few approaches have been proposed to address these issues. CoTTA [32] applies stochastic restoration of source knowledge such that the model does not drift much away from source knowledge. EATA [20] introduces a regularization loss to ensure that important model weights are preserved during adaptation, thereby alleviating forgetting. In contrast to these methods, our approach provides a more structured and effective solution to mitigate forgetting and error accumulation, resulting in better performance.

3. Method

3.1. Problem Setting

Consider a model f_{θ_0} pre-trained on the source data $\mathcal{X}_s \sim \mathcal{D}_s$, where $\mathcal{D}_s$ denotes the source distribution. During deployment, the model encounters a sequence of test data $\mathcal{X}_1 \rightarrow \mathcal{X}_2 \rightarrow \ldots \rightarrow \mathcal{X}_t \rightarrow \ldots$, where $\mathcal{X}_t$ represents a batch of test samples from the test distribution $\mathcal{D}^t_{test}$. Following the TTA setting, the model needs to adapt to each incoming test batch $\mathcal{X}_t$ and update its parameters from $f_{\theta_{t-1}} \rightarrow f_{\theta_t}$ in order to improve its predictions on the subsequent test batch $\mathcal{X}_{t+1}$. Since $\mathcal{D}^t_{test}$ changes continually over time, our objective is to determine the specific time instance t when a change in the domain occurs, $\mathcal{D}^t_{test} \neq \mathcal{D}^{t-1}_{test}$. Detecting these domain changes allows us to mitigate forgetting and error accumulation by reverting the model parameters to f_{θ_0}, preserving source knowledge and enabling effective adaptation to new domains.

3.2. Overall Framework

In this work, we hypothesize that leveraging the batch-norm (BN) statistics of incoming test batches can provide valuable insights into domain differentiation. To illustrate this, we conduct an experiment using a pre-trained ResNet-18 model on ImageNet. By extracting the batch-norm mean from the last CNN layer for images from various domains in the Office-Home [30] dataset, we create a t-SNE [29] visualization (Figure 2). The plot clearly shows distinct separations between domains, indicating that BN statistics exhibit unique patterns for different domains. Tracking changes in these statistics allows us to detect domain shifts. However, in dynamic environments with limited data availability, accurately estimating BN statistics becomes challenging.

Towards solving the problem, we aim to capture the global feature statistics of the current domain from the sequential incoming test batches, each containing limited data. When a domain shift occurs, the underlying distribution of the data changes significantly, causing a noticeable deviation in the feature statistics. By detecting these abrupt changes, we can identify the occurrence of a domain change. For this purpose, we employ a running average strategy to progressively correct and update the BN statistics to the incoming domain. Specifically, we first make a copy of the pre-trained source model that essentially serves as the oracle. During testing on the current batch t, let $\tilde{\mu}_{l_c}$ and $\tilde{\sigma}^2_{l_c}$ represent the BN statistics (mean and variance) extracted from the l-th layer's c-th channel of the oracle-model. We maintain two running averages as follows:

$$\mu^t_{l_c} = (1 - \bar{\alpha}^t) \times \mu^{t-1}_{l_c} + \bar{\alpha}^t \times \tilde{\mu}_{l_c} \quad (1)$$

$$(\sigma^t_{l_c})^2 = (1 - \bar{\alpha}^t) \times (\sigma^{t-1}_{l_c})^2 + \bar{\alpha}^t \times \tilde{\sigma}^2_{l_c} \, . \quad (2)$$

Here, $\mu^t_{l_c}$ and $(\sigma^t_{l_c})^2$ denote the running BN statistics that get updated with each incoming test batch. We initialize $\mu^0_{l_c}$ and $(\sigma^0_{l_c})^2$ with the BN statistics of the pre-trained source model. $\bar{\alpha}^t$ is the adaptive momentum value that controls the influence of the current batch on the running average. We want this value to gradually decrease over time when encountering test batches from the same domain. This decrease reflects the alignment between the running statistics and the global feature statistics of that specific domain. As the running statistics gradually align through incoming test

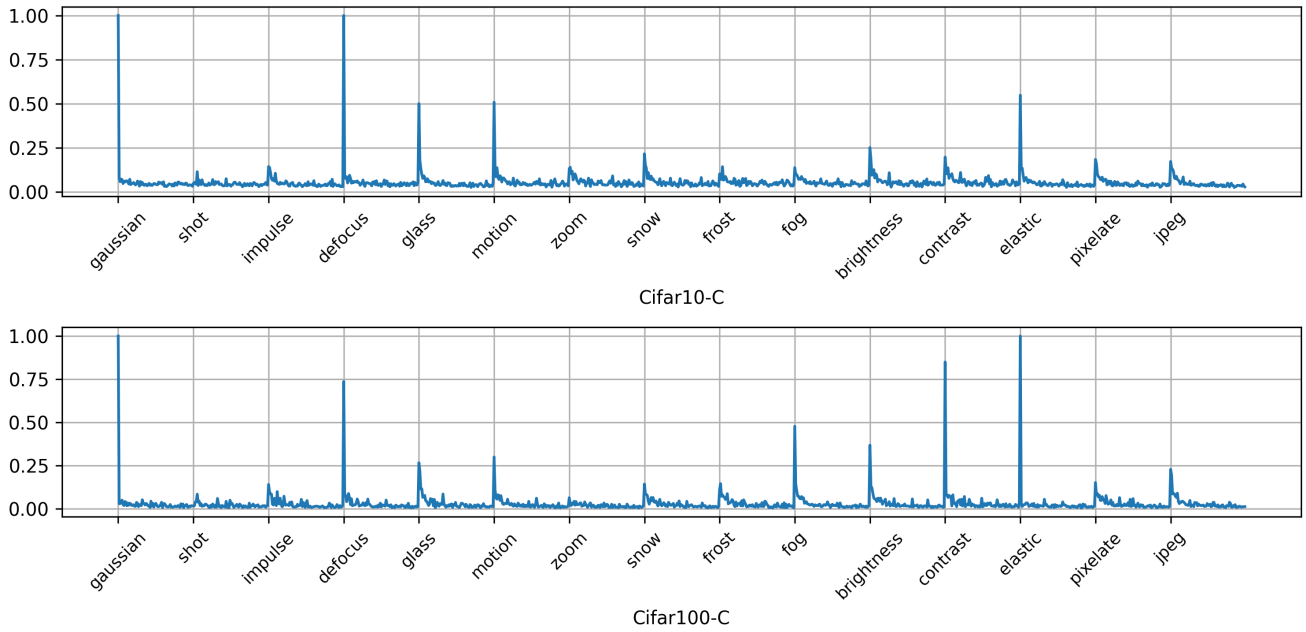

Figure 3. Plot of $\bar{\alpha}^t$ with respect to incoming target batches from 15 sequential target domains as the running statistics get aligned. The target datasets are from CIFAR10-C and CIFAR100-C. As the oracle-models (WideResNet-28 and ResNeXt-29) encounter test batches from a new domain, the running statistics gradually align with the global statistics of that domain. Consequently, during a domain change, $\bar{\alpha}^t$ exhibits a peak in its neighborhood due to the high KL divergence between the running statistics and the statistics of the incoming new-domain target batch. By detecting these peaks, it is possible to identify the occurrence of a domain change in the test batch. More plots of $\bar{\alpha}^t$ on other datasets can be found in the supplementary.

data, the need for significant adjustments diminishes, and the running statistics better capture the underlying distribution of the domain. However, when a domain change occurs, the feature statistics of the current test batch deviate significantly from the running statistics. In such cases, the adaptive momentum $\bar{\alpha}^t$ should assume a higher weight, indicating the necessity for a larger correction to align the running feature statistics with the new domain.

To design this momentum term, we propose to use the KL divergence as a metric to quantify the domain shift. Assuming the batch statistics per channel of the BN layers as a univariate Gaussian distribution, we calculate the divergence between the running statistics up to the current time instance (approximated as $\mathcal{N}(\mu_{l_c}^{t-1}, (\sigma_{l_c}^{t-1})^2)$) and the incoming test batch-norm statistics (approximated as $\mathcal{N}(\tilde{\mu}_{l_c}, (\tilde{\sigma}_{l_c})^2)$). This allows us to obtain the *unnormalized* value of the adaptive momentum for each layer as follows,

$$\begin{aligned}\alpha_l^t &= \frac{1}{C_l}\sum_{c=1}^{C_l}\mathcal{D}_{KL}\left[\mathcal{N}\left(\mu_{l_c}^{t-1}, (\sigma_{l_c}^{t-1})^2\right), \mathcal{N}\left(\tilde{\mu}_{l_c}, \tilde{\sigma}_{l_c}^2\right)\right] \\ &= \frac{1}{C_l}\sum_{c=1}^{C_l}\log\left(\frac{\tilde{\sigma}_{l_c}}{\sigma_{l_c}^{t-1}}\right) + \frac{\left(\sigma_{l_c}^{t-1}\right)^2 + \left(\mu_{l_c}^{t-1} - \tilde{\mu}_{l_c}\right)^2}{2\tilde{\sigma}_{l_c}^2} - \frac{1}{2}\,. \end{aligned} \tag{3}$$

Where C_l is the number of channels in l-th layer. Finally, the aggregated value across all layers, α^t, is calculated as follows,

$$\alpha^t = \frac{\sum_l \alpha_l^t}{L} \tag{4}$$

where L is the total number of layers. We track the highest value of the KL divergence to normalize each α^t to $\bar{\alpha}^t$, ensuring that the values remain within the range of 0 to 1. To initialize the highest domain distance value, we use the KL divergence between the pre-trained BN statistics and the statistics of the first target test batch. If a test batch yields an even higher KL divergence, this value is updated accordingly. Thus, $\bar{\alpha}^t$ assumes a value of 1 initially.

Furthermore, it is not necessary to consider all layers of the oracle-model to compute the momentum values. Recent work [16] suggests that the penultimate layers are more sensitive to domain shifts. In our experiments, we only consider the layers of the last block in the model for the calculation.

In summary, we propose an effective approach that gradually aligns the running statistics of each test batch with the global statistics of the underlying domain. This enables effective detection of domain shifts, as the running statistics undergo significant changes when a new domain is encoun-

Algorithm 1 Overall Framework

Require: Source model f_{θ_0}, Oracle-model $f_{\theta_{orc}}$

Initialize $\mu_{l_c}^0$ and $(\sigma_{l_c}^0)^2$ of the oracle-model with source pre-trained BN-statistics and set $\alpha_{max} \rightarrow 0^+$
for each incoming test batch t **do**
  for each layer l **do**
    Calculate adaptive momentum α_l^t for layer l using Eqn. 3
  end for
  Calculate mean adaptive momentum α^t of L layers using Eqn. 4
  if $\alpha^t > \alpha_{max}$ **then**
    $\alpha_{max} = \alpha^t$
  end if
  Normalize to get $\bar{\alpha}^t = \frac{\alpha^t}{\alpha_{max}}$
  for each layer l **do**
    for each channel c in layer l **do**
      Update running averages using Eqn. 1 and 2
    end for
  end for
  Input the $\bar{\alpha}^t$ to the peak detection algorithm (Algorithm 2 in the Supplementary)
  if peak **then**
    Restore source model parameters to f_{θ_0}
  end if
  Predict on test batch t
  Adapt model from $f_{\theta_{t-1}}$ to f_{θ_t} with selected TTA algorithm
end for

tered. This can be trivially incorporated into any existing TTA algorithm to reset the model parameters whenever a new domain is encountered, as illustrated in Algorithm 1.

3.3. Peak Detection

Figure 3 plots $\bar{\alpha}^t$ for incoming batches in our experiments. The value of $\bar{\alpha}^t$ exhibits a significant increase whenever a domain change occurs, reaching a peak in its neighborhood. As the model encounters more test batches from the same domain, the running statistics gradually align with the statistics of the domain, resulting in a decrease in the KL divergence. This leads to a decrease in the momentum value. By monitoring the online trend of $\bar{\alpha}^t$ and identifying the peak, we can detect domain changes in real time.

We utilize the online z-score algorithm [11] for peak detection. This algorithm maintains an exponential running average of the mean and standard deviation of inputs using a sliding window. For a query data point, it calculates the z-score, which represents the number of standard deviations the data point deviates from the running mean. An anomaly or peak is detected when the z-score exceeds a predefined threshold. The details of the algorithm is provided in the supplementary.

This is to be noted that the peak detection algorithm exclusively operates on the oracle model, while the original source model is consistently adapted (using any user preferred TTA algorithm) with the arrival of test batches. Whenever the oracle model identifies a domain shift, it restores the adapted source model to its original state, and the process of dynamic adaptation continues.

4. Experiments

4.1. Datasets

• **CIFAR10C, CIFAR100C, and ImageNet-C**: CIFAR10 and CIFAR100 [12] are popular image classification datasets consisting of 10,000 test images. To evaluate the robustness of trained models, CIFAR10C and CIFAR100C [8] were developed. These datasets introduce 15 distinct types of noise at varying severity levels (1 to 5) to the original CIFAR10 and CIFAR100 test images. Similarly, ImageNet-C [8] is the noisy counterpart of the ImageNet dataset [6]. These datasets serve as widely-used benchmarks in the field of continual TTA [32].

• **Digits:** The Digit benchmark is a standard dataset for digit classification, comprising ten classes. For our experiments, we utilized five domains: MNIST (MT), USPS (UP), SVHN (SV), MNIST-M (MM), and Synthetic Digits (SY).

• **Office-Home:**: The Office-Home dataset [30] comprises four domains, namely Art (Ar), Clipart (Cl), Product (Pr), and Real World (Re), each containing 65 classes.

• **Cityscapes to ACDC:** Cityscapes [4] is a large-scale dataset that has dense pixel-level annotations for 30 classes grouped into 8 categories. The Adverse Conditions Dataset [22] has images corresponding to fog, night-time, rain, and snow weather conditions. The number of classes is the same as the evaluation classes of the Cityscapes dataset. Keeping accordance with the standard setting, we evaluate our model on 19 semantic labels without considering the void label.

4.2. Baseline Methods

We utilize TENT [31] as the primary adaptation method and incorporate it with our proposed approach on detecting domain changes and restoring model parameters. We selected TENT primarily due to its lightweight and efficient nature, as it does not require additional modules and can perform adaptation with a single back-propagation step. We also compare our method with a variant of TENT called 'TENT-Online' [32]. TENT-Online assumes the availability of an oracle that can detect domain changes and resets the model accordingly. This serves as an upper-bound comparison for our method, representing the best-case scenario. However, in practice, such an oracle is not practical as this requires additional ground truth domain knowledge. We show that our method achieves comparable results to TENT-

Table 1. Classification error rate ↓ (in %) for the standard CIFAR100-to-CIFAR100C and ImageNet to ImageNet-C continual test-time adaptation task on corruption severity level 5. Note that the "online" methods represent best-case scenarios that require manual restoration to source weight upon a domain change, and they serve as upper bounds for their corresponding continuous adaptation counterparts. Online model rows are highlighted to emphasize best-case scenario.

	Time	t ——————>															
Dataset	Domains	Gaussian	Shot	Impulse	Defocus	Glass	Motion	Zoom	Snow	Frost	Fog	Brightness	Contrast	Elastic	Pixelate	JPEG	Mean
CIFAR100C	DUA Online	43.7	39.8	42.8	33.1	44.4	30.5	27.8	34.9	33.5	43.6	25.8	32.3	39.9	41.4	41.3	37.0
	DUA Continual	43.7	**39.5**	**42.2**	56.8	47.6	40.2	33.6	39.9	36.8	48.2	28.7	43.2	49.5	77.1	47.1	44.9
	DUA + Ours	**43.7**	39.7	42.7	**33.1**	**44.1**	**30.4**	**29.0**	**34.5**	**33.5**	**43.8**	**25.9**	**32.5**	**39.8**	**41.4**	**41.0**	**37.0**
	Tent Online	37.5	35.3	32.7	25.7	37.5	27.5	25.8	30.6	32.3	33.1	24.4	28.1	33.2	28.7	37.3	31.3
	Tent Continual	**37.5**	37.1	44.3	41.3	56.5	55.6	57.9	69.7	75.0	83.3	86.2	93.7	95.5	96.2	96.9	68.4
	Tent + Ours	37.8	**35.5**	**32.7**	**25.8**	**38.0**	**27.6**	**26.9**	**30.8**	**32.4**	**33.9**	**24.7**	**28.3**	**33.2**	**28.8**	**37.9**	**31.6**
ImageNet-C	DUA Online	85.7	84.5	84.6	96.5	88.3	75.9	61.9	69.2	68.7	60.2	35.6	84.0	60.6	51.4	60.3	71.2
	DUA Continual	**85.7**	88.5	**82.5**	99.8	99.8	98.8	89.9	88.6	79.8	84.4	49.5	96.3	78.5	71.5	60.3	83.6
	DUA + Ours	85.7	**84.6**	83.0	**99.4**	**90.8**	**76.3**	**62.1**	**69.4**	**68.9**	**61.7**	**35.7**	**84.5**	**62.8**	**52.2**	**59.9**	**71.8**
	Tent Online	74.7	71.4	73.8	75.5	75.4	61.9	52.6	54.5	61.3	44.0	33.8	79.3	46.5	43.6	50.1	59.9
	Tent Continual	**74.7**	**71.7**	**69.7**	76.3	75.0	70.4	62.0	68.8	71.0	62.7	51.1	82.5	67.8	66.9	71.4	69.5
	Tent + Ours	74.8	73.6	72.3	**75.4**	**74.9**	**62.5**	**52.7**	**55.1**	**62.3**	**44.0**	**33.6**	**79.3**	**47.7**	**43.5**	**50.0**	**60.1**

Online without the need for such ground truth knowledge.

Apart from TENT, we also integrate our method with DUA [19] to illustrate the versatility of the proposed domain change detection module with respect to the underlying TTA algorithm. By detecting domain changes and restoring DUA parameters, we extend its capability to perform adaptation in dynamic environments. We also compare our results with CoTTA [32] and EATA [20]. These two methods specifically focus on continual test time adaptation. We use the official implementations for all baselines.

4.3. Implementation Details

For CIFAR100C and ImageNet-C experiments, we adopt the pre-trained ResNeXt-29 [34] and ResNet-50 respectively from Robustbench [5] for all methods. We use ResNet-18 for the Digit and Office-home experiments. Our batch size is set to 64 for ImageNet-C and 128 for other classification experiments. For the TENT approach adopted for our method, we use Adam optimizer with a learning rate of 0.001. Similar to [32], we utilize the validation set compiled by RobustBench for ImageNet-C dataset. For the peak detection algorithm, we use a sliding window of 10 and a momentum of 0.1 for incoming values. The threshold is taken as 15 standard deviations. We maintain consistency with the respective papers of the compared methods by using the same learning rate, optimizer, and rest of the hyperparameters. For all the comparing methods we use their official implementations.

4.4. Experiments on CIFAR100C and ImageNet-C

We first evaluate our method on CIFAR100C and ImageNet-C. Particularly, given a pre-trained model on CIFAR100/ImageNet, we adapt the model sequentially to 15 types of unseen domains/noise sets. Each of the noise sets has a total of $10,000$ images. We show the results in Table 1. DUA-Online and TENT-Online models are manually reset when there is a domain change and thus, the models act as an upper-bound for comparison with our method. On the other hand, TENT and DUA Continual are continually adapted (lifelong) to test data without manual resetting. It can be observed from the table that the performance of both TENT and DUA deteriorates over time, as the models continually adapt to unseen test samples. In the case of ImageNet-C the deterioration is much more prominent because ImageNet-C has a total of 1000 classes, which makes it difficult for the model to produce reliable pseudo-labels and thus contributing to error accumulation. The deviation of the results from the corresponding online models also highlights the poor performance of TENT and DUA. The performance in fact deteriorates over time in comparison to their corresponding online models due to catastrophic forgetting and gradual error accumulation. On the other hand, when our method is added to TENT and DUA in order to automatically detect a domain change and reset the model parameters accordingly, the performance of both models improves by a big margin. The results also get very much close to their corresponding online models which verifies that our method can effectively mimic the online models without requiring any domain knowledge.

4.5. Experiments on Digits and Office-Home

We next perform experiments on digit and officehome datasets. In order to simulate a dynamic environment, we train a model on the train set of one dataset, and sequentially adapt on the test sets of rest of the datasets for a total of 10 cycles. The results are shown in Table 3 and Table 4. The columns in the tables show the dataset in which the source model is trained on. For example, in case of the

Table 2. Classification error rate ↓ (in %) for the standard CIFAR100-to-CIFAR100C and ImageNet to ImageNet-C adaptation task on corruption severity level 5 against current state-of-the-art models on continual test time adaptation.

Dataset	Time	t ——————>															
	Domains	Gaussian	Shot	Impulse	Defocus	Glass	Motion	Zoom	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixelate	JPEG	Mean
CIFAR100C	Source	72.3	67.4	39.0	29.4	53.6	30.5	28.8	39.1	45.5	50.3	29.7	55.4	37.2	74.8	41.0	46.3
	BN-Stat [23]	42.1	40.7	42.7	27.6	41.9	29.7	27.9	34.9	35	41.5	26.5	30.3	35.7	32.9	41.2	35.4
	EATA [20]	39.7	37.2	37.0	26.9	40.0	28.4	26.5	32.0	32.9	38.2	25.2	29.9	34.3	30.5	39.0	33.2
	CoTTA [32]	40.1	37.7	39.7	26.9	38.0	27.9	**26.4**	32.8	**31.8**	40.3	24.7	**26.9**	**32.5**	**28.3**	**33.5**	32.5
	Tent + Ours	**37.8**	**35.5**	**32.7**	**25.8**	**38.0**	**27.6**	26.9	**30.8**	32.4	**33.9**	**24.7**	28.3	33.2	28.8	37.9	**31.6**
ImageNetC	Source	97.9	96.9	98.2	81.9	89.7	84.9	78.2	83.3	77.3	76.2	41.2	94.4	82.9	79.2	68.7	82.1
	BN-Stat [23]	85	83.7	85	84.7	84.3	73.7	61.2	66	68.2	52.1	34.9	82.7	55.9	51.3	59.8	68.6
	EATA [20]	82.4	76.9	73.9	77.4	**73.1**	63.9	54.0	60.9	61.2	49.1	36.0	67.3	49.4	45.6	49.9	61.4
	CoTTA [32]	84.5	81.9	79.8	80.8	78.2	67.3	57.6	60.5	**60.4**	48.2	36.5	**64.0**	**47.3**	**41.1**	**45.2**	62.2
	Tent + Ours	**74.8**	**73.6**	**72.3**	**75.4**	74.9	**62.5**	**52.7**	**55.1**	62.3	**44.0**	**33.6**	79.3	47.7	43.5	50.0	**60.1**

Table 3. Classification error rate on continual adaptation task on digit datasets. The column header represents the source dataset where the model is trained on. Mean classification error over all the cycles is reported here.

Method	MM	MT	UP	SV	SY	Avg
DUA Online	36.3	71.6	74.7	29.7	24.1	47.3
DUA Continual	36.6	73.8	76.0	29.8	24.4	48.1
DUA + Ours	**36.4**	**71.8**	**74.7**	**29.7**	**24.1**	**47.3**
Tent Online	37.1	72.5	75.4	27.7	22.5	47.0
Tent Continual	43.1	87.9	86.0	28.1	23.4	53.7
Tent + Ours	**38.0**	**72.8**	**75.6**	**27.8**	**22.5**	**47.3**

Table 4. Classification error rate on continual adaptation task on office-home dataset.

Method	Ar	Cl	Pr	Rw	Avg
DUA Online	46.7	47.0	51.5	41.7	46.7
DUA Continual	49.6	47.5	52.4	43.2	48.2
DUA + Ours	**47.0**	**47.2**	**52.1**	**41.9**	**47.1**
Tent Online	48.1	46.4	51.4	41.8	46.9
Tent Continual	79.2	63.6	75.2	47.7	66.4
Tent + Ours	**48.9**	**47.3**	**51.4**	**42.1**	**47.4**

'MM' column of 3, the source model is trained on MNIST-M dataset and then during test-time, the model is adapted to sequential unseen domains in the following order: MT → UP → SV → SY. This whole cycle goes on for a total of 10 times. Similarly, in case of the 'Art' column of Table 4, the source model is trained on Art dataset and then, the model is adapted sequentially to unseen domains in the following order: Cl → Pr → Rw for a total of 10 times. The mean classification error over the whole sequence is reported at the tables. It can be observed from both the table that our method performs better than TENT and DUA Continual. Our results are very close to the online models, while in some cases, i.e., SYNDIG, SVHN, Pr it is equal.

4.6. Experiments on Cityscapes to ACDC

In this experiment, we evaluate our method on the more complex continual test-time adaptation scenario - semantic segmentation of Cityscapes to ACDC. In order to simulate real-life situations where comparable environments may be encountered, we iterate an identical four condition sequence for a total of ten times, resulting in a total of 40 repetitions: Fog → Rain → Snow → Night → Fog . . . Specifically, we train the model on the clean weather condition of Cityscapes dataset and adapt to this unseen weather condition sequence from ACDC. We use DeepLab v3+ [3] with a ResNet-18 encoder for the experiment. The batch size used is 4. We report the mean performance over all the sequences.

Table 5. Semantic segmentation results (mIoU in %) on the Cityscapes-to-ACDC online continual test-time adaptation task. We evaluate the four test conditions continually for ten times and report the mean here. 'O' refers to the online model.

Method	DUA-O	DUA-C	+Ours	Tent-O	Tent-C	+Ours
mIoU	20.8	20.5	**20.7**	17.1	14.7	**17.0**

The results are shown in Table 5. It can be observed that the continual models perform worse than the online models. The performance gap between the two models is even more highlighted in the case of TENT. On the other hand, in both cases, adding our method helps DUA and TENT to reach performance on par with the online model.

4.7. Comparison with State-of-the-Art

In this section, we evaluate our method against state-of-the-art approaches specifically designed for the continual adaptation task. We conduct experiments on CIFAR100-C and ImageNet-C. We compare our method with CoTTA [32] and EATA [20]. CoTTA mitigates catastrophic forgetting by employing stochastic restoration of source weights, while EATA incorporates Fisher regularizer to limit drastic changes in important model parameters, thereby preserving source knowledge. In comparison, our method offers a more structured and intuitive approach to retaining source knowledge. Additionally, we also compare our method with BN-Stat [23], which replaces the pre-trained batch-norm

statistics with statistics estimated from the test batch.

In the initial experiment, we conduct adaptation to the 15 corruptions from CIFAR100-C and ImageNet-C, each with a severity level of 5 [32]. The results are presented in Table 2. The table reveals that our method surpasses all the baseline models on both datasets in mean performance. Despite CoTTA and EATA utilizing advanced techniques such as mean teacher and advanced regularizers to retain source knowledge and mitigate error propagation, our simpler approach achieves superior performance compared to both these methods.

We next look at a more challenging adaptation scenario. For this experiment, we gradually change the severity level in CIFAR100C for each corruption set as follows: $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 1$. Hence, we have a total of 15×5 or 75 unseen domains. This specific experiment [32] is designed to highlight the catastrophic forgetting and error accumulation issue more prominently.

Table 6. Results for gradually changing noise level. The mean of the whole sequence is reported here.

Method	EATA	CoTTA	Tent+Ours
Error Rate (in %)	34.9	28.1	**26.8**

As can be seen from the Table, our method again outperforms both CoTTA and EATA in this challenging adaptation task. Notably, our method achieves an impressive 8.1% improvement compared to EATA, further highlighting its effectiveness and superiority in mitigating catastrophic forgetting and error accumulation.

4.8. Analysis on Forgetting

In this section, we demonstrate the robustness of our method against catastrophic forgetting by evaluating the classification accuracy on the source test set after completing adaptation to each domain [20]. Specifically, we take the CIFAR100-C dataset and after adapting the model to each of the 15 unseen domains, we check the accuracy on the test set of the clean CIFAR100 dataset. This helps to quantify the reduction of source knowledge after each adaptation cycle. The results are shown in Figure 4.

From the figure, it is apparent that the test accuracy of the TENT method gradually deteriorates as it encounters new domains. This degradation of source knowledge directly correlates to poor generalization on target domains, as shown in Table 1. For reference, the source accuracy is also plotted, representing the ideal scenario with the curve being completely flat. It can be observed that our method almost overlaps with the source accuracy curve. This shows that we achieve almost no forgetting of source knowledge with our method. CoTTA and EATA outperform TENT by a big margin because these methods are specifically designed to combat forgetting. Nevertheless, these two specialized

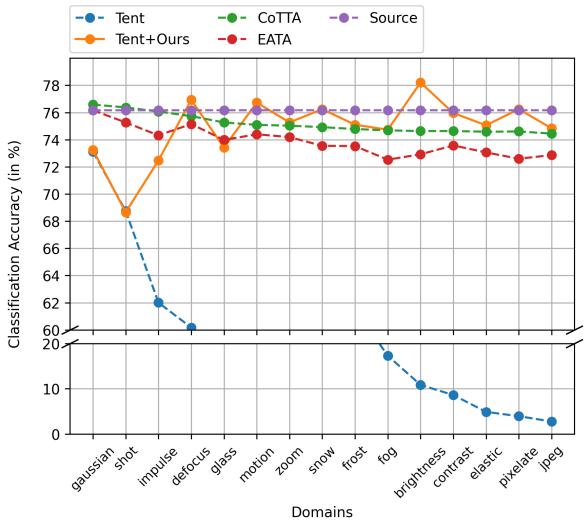

Figure 4. We assess the source knowledge by evaluating the methods on the source test set following the completion of adaptation to each domain of CIFAR-100C. Observing the results, it becomes evident that our method aligns more with the source accuracy, signifying robustness to the forgetting of source knowledge compared to SOTA methods.

methods also are not completely robust to forgetting, as observed by the gap from the flat source accuracy curve in the figure. Our method surpasses even CoTTA and EATA in terms of accuracy. Furthermore, we have carried out additional sensitivity analyses on the threshold value used in the peak detection algorithm to ensure its robustness. Detailed results, along with decision diagram of our method and an analysis of scenarios where the domain gaps between various domains are very small, can be found in the supplementary material.

5. Conclusion

This paper addresses the novel problem of detecting domain changes in dynamic environments. We estimate global batch-norm statistics of a domain using an adaptive momentum, which undergoes a significant change during distribution shifts. By detecting these domain changes and restoring model parameters to their source pre-trained values, we have shown that our method effectively mitigates the issues of catastrophic forgetting and error accumulation observed in traditional TTA methods. This enhances the robustness of TTA methods in dynamic environments, and ensures their performance is maintained over time.

Acknowledgement

The work was partially supported by ONR grant N00014-19-1-2264 and the NSF grants CCF-2008020 and IIS-1724341.

References

[1] Sk Miraj Ahmed, Suhas Lohit, Kuan-Chuan Peng, Michael J Jones, and Amit K Roy-Chowdhury. Cross-modal knowledge transfer without task-relevant source data. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIV*, pages 111–127. Springer, 2022.

[2] Sk Miraj Ahmed, Dripta S Raychaudhuri, Sujoy Paul, Samet Oymak, and Amit K Roy-Chowdhury. Unsupervised multi-source domain adaptation without access to source data. In *CVPR*, 2021.

[3] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

[4] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[5] Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. *arXiv preprint arXiv:2010.09670*, 2020.

[6] 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.

[7] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. In *JMLR*, 2016.

[8] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[9] Han-Kai Hsu, Chun-Han Yao, Yi-Hsuan Tsai, Wei-Chih Hung, Hung-Yu Tseng, Maneesh Singh, and Ming-Hsuan Yang. Progressive domain adaptation for object detection. In *WACV*, 2020.

[10] Minhao Hu, Tao Song, Yujun Gu, Xiangde Luo, Jieneng Chen, Yinan Chen, Ya Zhang, and Shaoting Zhang. Fully test-time adaptation for image segmentation. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part III 24*, pages 251–260. Springer, 2021.

[11] Manish Kejriwal, Pawel Szalachowski, Sugato Basu, Kalpesh Joshi, Balaji Krishnamurthy, and Shubha Venkataraman. Real-time anomaly detection for streaming analytics. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1711–1720. ACM, 2015.

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

[13] Vikash Kumar, Rohit Lal, Himanshu Patil, and Anirban Chakraborty. Conmix for source-free single and multi-target domain adaptation. In *WACV*, 2023.

[14] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016.

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

[16] Hyesu Lim, Byeonggeun Kim, Jaegul Choo, and Sungha Choi. Ttn: A domain-shift aware batch normalization in test-time adaptation. *arXiv preprint arXiv:2302.05155*, 2023.

[17] Yuang Liu, Wei Zhang, and Jun Wang. Source-free domain adaptation for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1215–1224, 2021.

[18] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *ICML*, 2015.

[19] M Jehanzeb Mirza, Jakub Micorek, Horst Possegger, and Horst Bischof. The norm must go on: dynamic unsupervised domain adaptation by normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14765–14775, 2022.

[20] Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Yaofo Chen, Shijian Zheng, Peilin Zhao, and Mingkui Tan. Efficient test-time model adaptation without forgetting. In *International conference on machine learning*, pages 16888–16905. PMLR, 2022.

[21] Dripta S Raychaudhuri, Sujoy Paul, Jeroen Vanbaar, and Amit K Roy-Chowdhury. Cross-domain imitation from observations. In *ICML*, 2021.

[22] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Acdc: The adverse conditions dataset with correspondences for semantic driving scene understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10765–10775, 2021.

[23] Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. *Advances in Neural Information Processing Systems*, 33:11539–11551, 2020.

[24] Inkyu Shin, Yi-Hsuan Tsai, Bingbing Zhuang, Samuel Schulter, Buyu Liu, Sparsh Garg, In So Kweon, and Kuk-Jin Yoon. Mm-tta: multi-modal test-time adaptation for 3d semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16928–16937, 2022.

[25] Junha Song, Jungsoo Lee, In So Kweon, and Sungha Choi. Ecotta: Memory-efficient continual test-time adaptation via self-distilled regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11920–11929, 2023.

[26] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *CVPR*, 2018.

[27] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR*, 2017.

[28] Jeya Maria Jose Valanarasu, Pengfei Guo, Vibashan VS, and Vishal M Patel. On-the-fly test-time adaptation for medical image segmentation. *arXiv preprint arXiv:2203.05574*, 2022.

[29] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[30] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5018–5027, 2017.

[31] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*, 2020.

[32] Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. Continual test-time domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7201–7211, 2022.

[33] Haifeng Xia, Handong Zhao, and Zhengming Ding. Adaptive adversarial network for source-free domain adaptation. In *ICCV*, 2021.

[34] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[35] Hao-Wei Yeh, Baoyao Yang, Pong C Yuen, and Tatsuya Harada. Sofa: Source-data-free feature alignment for unsupervised domain adaptation. In *WACV*, 2021.

[36] Marvin Zhang, Sergey Levine, and Chelsea Finn. Memo: Test time robustness via adaptation and augmentation. *Advances in Neural Information Processing Systems*, 35:38629–38642, 2022.