# DOMAIN SHIFT SIGNAL FOR LOW RESOURCE CONTINUOUS TEST-TIME ADAPTATION

**Goirik Chakrabarty**[1], **Manogna Sreenivas**[2], **Soma Biswas**[2]
[1]Indian Institute of Science Education and Research, Pune, India
goirik.chakrabarty@students.iiserpune.ac.in

[2]Indian Institute of Science, Bangalore, India
{manognas,somabiswas}@iisc.ac.in

## ABSTRACT

Test time domain adaptation has come to the forefront as a challenging scenario in recent times. Although single domain test-time adaptation has been well studied and shown impressive performance, this can be limiting when the model is deployed in a dynamic test environment. We explore this continual domain test time adaptation problem here. Specifically, we question if we can translate the effectiveness of single domain adaptation methods to continuous test-time adaptation scenario. We propose to use the given source domain trained model to continually measure the similarity between the feature representations of the consecutive batches. A domain shift is detected when this measure falls below a certain threshold, which we use as a trigger to reset the model back to source and continue test-time adaptation. We demonstrate the effectiveness of our method by performing experiments across datasets, batch sizes and different single domain test-time adaptation baselines. This can have a significant impact in a variety of applications, from healthcare and agriculture to transportation and finance. As a result, this research has the potential to greatly benefit developing countries by providing new tools and techniques for building more effective and efficient machine learning systems.

## 1 INTRODUCTION

The ability to continually adapt models in real-time is becoming increasingly important in today's fast-paced technological landscape. This is especially true for developing countries, where access to diverse data and changing environments can pose unique challenges for machine learning models. The traditional approach of single domain test-time adaptation, while effective in certain scenarios, can limit the performance of models when deployed in dynamic and ever-changing environments.

This research broadly operates under a stringent assumption that the training and testing data come from the same distribution. This assumption can be problematic when there is a significant difference between the distribution of the training data and the test data, a phenomenon commonly known as *domain shift*. This can result in reduced accuracy and performance of the model, as it has not been trained on data from the testing distribution. To mitigate this vulnerability, various domain adaptation techniques have been developed to make the models more robust to such shifts. These techniques aim to align the distributions of the training and testing data, reducing the negative impact of the domain shift on model performance. The study of robustness of deep networks against distribution shifts has rapidly evolved in recent years, broadly covering the following topics:

**Unsupervised Domain Adaptation (UDA):** This setting assumes access to labeled source domain data along with unlabeled target domain data during training. UDA methods (Ganin et al., 2016; Long et al., 2018; Saito et al., 2018; Xu et al., 2019) primarily aim to align the two domains so that the supervision from source domain can be transferred to that of target.

**Domain Generalization (DG):** DG methods (Li et al., 2021; Kim et al., 2021; Zhou et al., 2021) use multiple source domains to learn robust domain-invariant representations so that the model can better generalize to unseen test domains.

Table 1: Domain adaptation protocols

| Setting | Source-free | Adaptation protocol | | Target domain | |
|---|---|---|---|---|---|
| | | Offline | Online | Single | Continuous |
| UDA | | ✓ | | ✓ | |
| SFDA | ✓ | ✓ | | ✓ | |
| TTA | ✓ | | ✓ | ✓ | |
| CTTA | ✓ | | ✓ | | ✓ |

**Source Free Domain Adaptation (SFDA):** Contrary to UDA and DG, SFDA methods Liang et al. (2020); Yang et al. (2022) attempt to adapt any off-the-shelf model given abundant target domain data. This data is assumed to be available offline and can be shown to the model multiple times for adaptation.

**Test Time Adaptation (TTA):** TTA was first proposed by (Wang et al., 2021) with the objective of leveraging the test data coming in an online manner to adapt a given off-the shelf model. The key challenges here are: (1) No access to labels and therefore inability to recognise and correct wrong predictions; (2) No access to source data; (3) Viewing data in an online manner i.e. you have access to each test minibatch only once.

**Continuous Test Time Adaptation (CTTA):** Taking another step forward from TTA towards reality, a recent work Wang et al. (2022) formalized the CTTA setting where the test domain can dynamically change in time. The state-of-the-art approach Wang et al. (2022) reduces error accumulation through weight-averaged and augmentation-averaged predictions. They further avoid catastrophic forgetting through stochastic restoration of source pre-trained weights. However, this method is computationally very taxing as also acknowledged by the authors. Therefore, our solution can be more effectively deployable in developing countries.

We highlight the difference between various domain adaptation protocols in Table 1. In this work, we specifically question the differences between TTA and CTTA and aim to bridge the gap between the two.

**Why TTA can hurt CTTA?** TTA methods designed for single domain adaptation tend to overfit on the current test domain which can lead to catastrophic forgetting of discriminative information from source in time. This can be extremely harmful when the model could encounter new test domains in the future.

**Can we simulate TTA setting in CTTA?** We recognise that a simplistic approach to CTTA is to adapt to the test domain in a TTA manner i.e. adapt the model using a TTA algorithm and then reset the model back to the source model everytime it encounters a domain shift. This allows the model to learn representations by leveraging the benefits of single domain TTA and at the same time avoid error accumulation in time by not carrying over an overfit model to the next domain.

## 2 PROBLEM SETTING

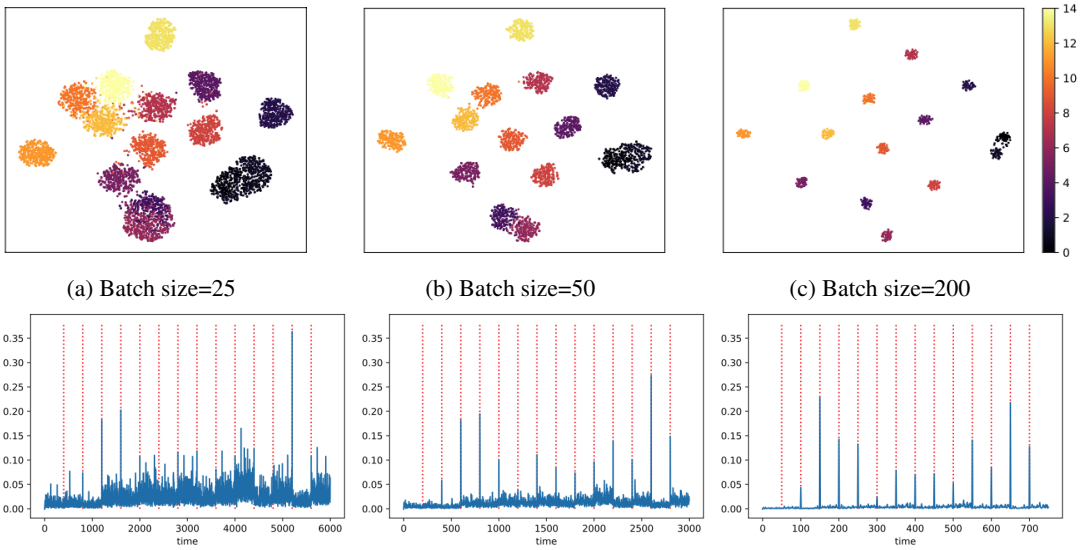
Given an off-the shelf model $h_{\boldsymbol{\theta}}$ comprising of feature extractor $f$ and classifier $g$ trained on a source domain $\mathcal{D}_{train}$, the objective of TTA is to adapt $h_{\boldsymbol{\theta}}$ using test batches $\mathbf{x}_t$ arriving in an online manner from a test domain $\mathcal{D}_{test}$ by minimizing a test time objective as

$$\arg\min_{\theta} \mathcal{L}_{test}(\mathbf{x}_t; \theta) \tag{1}$$

In standard TTA addressed in Wang et al. (2021); Boudiaf et al. (2022); Chen et al. (2022), $\mathbf{x}_t$ comes from a single test domain $\mathcal{D}_{test} \neq \mathcal{D}_{train}$. Here, we address the CTTA setting, where the test domain $\mathcal{D}_{test}$ can continuously change sequentially as $\mathcal{D}_{t1}, \mathcal{D}_{t2}, \mathcal{D}_{t3}, ..., \mathcal{D}_{tN}$, where $\mathcal{D}_{ti} \neq \mathcal{D}_{train} \forall i$.

## 3 METHOD

We first briefly describe some recent source-free adaptation methods, namely Tent Wang et al. (2021) and AaD Yang et al. (2022). Then, we describe our Domain Shift Detection mechanism in detail.




(a) Batch size=25 (b) Batch size=50 (c) Batch size=200

(d) Batch size=25 (e) Batch size=50 (f) Batch size=200

Figure 1: We observe from the t-SNE plots for (a), (b) and (c) that the classes are better clustered and separated as the batch-size increases. The color of these clusters also represent the order in which 15 corruptions are seen. In (d), (e) and (f) we see the corresponding (1 - DSS) signals to the t-SNE. The red dotted lines are where the actual domain shift happens.

**TENT:** They propose to use the test feature statistics in the Batch Normalization (BN) layers instead of those estimated using the source data. Further, they fine-tune the BN's affine parameters to minimize the Shannon entropy $\mathcal{L}_{ent}(x_t) = -\sum_c p_c \log p_c$, where $p_c$ is the softmax score of class $c$ for a test sample $x_t$.

**Attracting and Dispersing (AaD):** AaD is a simple and effective approach recently proposed for SFDA. They treat SFDA as an unsupervised clustering problem where they enforce consistency between predictions of local neighbourhood features while also ensuring diversity in the feature space. The test objective for a sample $x_i$ from a test batch $\mathbf{x}_t$ is $\mathcal{L}(x_i) = -\sum_{p_j \in \mathcal{N}_i} p_i^T p_j + \lambda \sum_{x_m \in \mathbf{x}_t} p_i^T p_m$. Here $\mathcal{N}_i$ is the set of neighbours of $x_i$ and $p_m$ refers to the softmax prediction vector of a sample $x_m \in \mathbf{x}_t$.

The above mentioned methods achieve state-of-the-art performance in single domain adaptation setting. However, these methods suffer from error accumulation due to over-fitting in CTTA. We observe that source model is a more reliable starting point for adaptation than continually adapting. This is because the source model has already been trained on a large amount of data, and it has learned some general representations that can be transferred to the new domain. By adapting the source model on the new domain, the model can adjust its representations to better fit the new data while retaining the knowledge learned from the source domain.

### 3.1 Domain Shift Detection

As mentioned earlier, using TTA methods like TENT can hurt in CTTA setting because of error accumulation. This in turn degrades the model over time. Here, we propose a simple but effective solution to this by resetting the model when a domain shift is encountered.

**Can source model characterize domain shift?** In CTTA, the data distribution changes over time, meaning that each batch of samples can come from a different domain. To handle this challenge, we leverage the feature extractor of the source model $f$, which we empirically observed to capture domain information. The features of each sample $v_f = f(x)$ has two components: (i) Domain-specific component $v_d$ which represents the part of the feature that is unique to a particular domain and distinguishes it from other domains; (ii) Class-specific component $v_c$ that is relevant to the classification task. By separating the features into these two components, the model can learn to

identify and adapt to changes in the distribution of the data between batches, while still maintaining the ability to perform well on the classification task.

We hypothesize that $\mathbf{E}v_f = \mathbf{E}v_d + \mathbf{E}v_c$. Given, the samples come from the same domain, they have the same domain specific component $\mathbf{E}v_d = \mathbf{v}_d$, also the class specific components $v_c$ would be uniformly spread across all classes as $\mathbf{E}v_c = \frac{1}{C}\sum_{k=1}^{C} v_k = \mathbf{v}_c$, where $\mathbf{v}_c$ is a constant vector and C denotes the number of classes. Hence, $\mathbf{E}v_f = \mathbf{v}_d + \mathbf{v}_c$. In this formulation, any change in the domain specific component $\mathbf{E}v_d$ can in-turn be captured by $\mathbf{E}v_f$, which can be empirically estimated.

In CTTA, given a test batch $\mathbf{x}_t = x_1, x_2, ..., x_N$ at time instant $t$, we can estimate $\mathbf{E}v_f(t)$ as the mean feature vector $\mathbf{E}v_f(t) = \frac{1}{N}\sum_{k=1}^{N} v_{f,i}$, where $v_{f,i} = f(x_i)$. This shows that these domain specific components can be used to identify or detect a domain shift. We empirically observe that $\mathbf{v}_c \to 0$ as $N \to \infty$. In Figure 1, we visualize the average batch features using different batch sizes and for 15 corruptions in the CIFAR-100C. As the batch size increases the domain clusters become more compact indicating the aforementioned tendency. Because of this, the domain-specific component becomes more distinctive with larger batch sizes.

This naturally acts as our domain shift signal. We define the cosine similarity of consecutive batches as Domain Shift Signal (DSS), which we compute as

$$\text{DSS} = \text{CosineSimilarity}(\mathbf{E}v_f(t), \mathbf{E}v_f(t-1)) \tag{2}$$

We us this signal to detect a change in domain using a threshold $\tau$. When $\mathbf{E}v_f(t)$ comes from the same domain as $\mathbf{E}v_f(t-1)$, DSS is high, in turn continuing the model adaptation. Otherwise, we trigger a model reset back to the source model. We briefly describe the domain shift detection mechanism below.

**Algorithm 1: Domain Shift Detection module**

**Input:**
Source feature extractor $f$
Threshold for detection $\tau$
**Domain Shift Detection:**
for each batch $\mathbf{x}_t$:
  $v_{f,i} = f(x_{t,i})$
  $\mathbf{E}v_f(t) = \frac{1}{N}\sum_{k=1}^{N} v_{f,i}$
  $DSS(\mathbf{E}v_f(t), \mathbf{E}v_f(t-1)) = \frac{\mathbf{E}v_f(t)^T \mathbf{E}v_f(t-1)}{||\mathbf{E}v_f(t)||||\mathbf{E}v_f(t-1)||}$
  if $DSS(\mathbf{E}v_f(t), \mathbf{E}v_f(t-1)) < \tau$:
    Reset model to source
  Continue TTA

## 4 Experiments and Results

### 4.1 Datasets

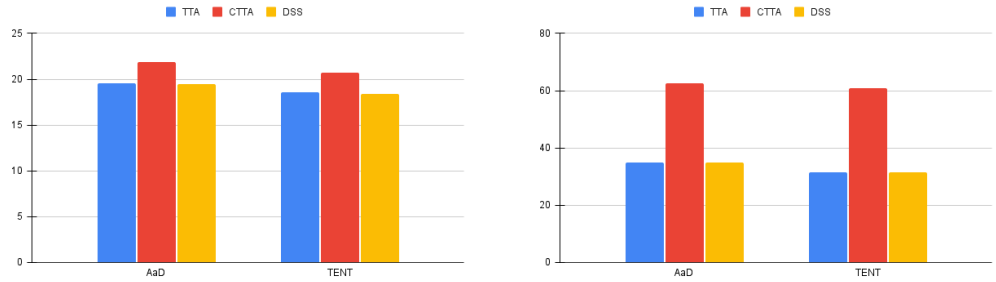
Following the protocol in Wang et al. (2022), we use CIFAR10C and CIFAR100C datasets which are designed to evaluate the robustness of classification networks. These datasets contain images that have been corrupted with 15 different types of corruptions at 5 different levels of severity. In the case of the corruption benchmark, this sequence consists of all 15 corruptions, each encountered at the highest severity level 5.

### 4.2 Baselines

We compare the performance of TENT and AaD in three different scenarios:
**TTA:** Firstly, we consider the TTA setting introduced by TENT Wang et al. (2021) where the model is set to source whenever there is a domain shift. This domain shift information is explicitly provided to the model.

Table 2: Results as error percentages (lower is better) for CIFAR-100C

| Method | *gaussian* | *shot* | *impulse* | *defocus* | *glass* | *motion* | *zoom* | *snow* | *frost* | *fog* | *brightness* | *contrast* | *elastic* | *pixelate* | *jpeg* | Mean |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Source | 73.0 | 68.0 | 39.4 | 29.3 | 54.1 | 30.8 | 28.8 | 39.5 | 45.8 | 50.3 | 29.5 | 55.1 | 37.2 | 74.7 | 41.2 | 46.4 |
| CoTTA | 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-TTA | 37.1 | 34.65 | 33.7 | 25.1 | 37.66 | 27.15 | 25.4 | 30.5 | 31.5 | 33.3 | 23.8 | 27.8 | 32.7 | 28.4 | 36.5 | 31.0 |
| TENT-CTTA | 92.7 | 37.2 | 35.7 | 41.6 | 37.5 | 50.8 | 47.7 | 48.5 | 58.7 | 64.8 | 72.4 | 70.5 | 82.2 | 88.5 | 89.9 | 61.2 |
| **TENT-DSS** | 37.2 | 35.9 | 41.6 | 25.2 | 37.6 | 27.2 | 25.4 | 30.5 | 31.6 | 33.2 | 23.8 | 27.7 | 32.6 | 28.4 | 36.5 | 31.5 |
| AaD-TTA | 41.9 | 39.8 | 42.0 | 27.2 | 41.4 | 29.3 | 27.5 | 34.5 | 34.7 | 40.3 | 26.2 | 30.2 | 35.2 | 32.3 | 40.8 | 34.9 |
| AaD-CTTA | 41.9 | 40.1 | 43.5 | 31.7 | 46.8 | 39.2 | 41.6 | 58.2 | 67.7 | 76.2 | 79.1 | 90.1 | 93.0 | 93.8 | 94.6 | 62.5 |
| **AaD-DSS** | 41.9 | 40.1 | 43.5 | 27.2 | 41.4 | 29.3 | 27.5 | 34.5 | 35.0 | 40.3 | 26.2 | 30.2 | 35.2 | 32.3 | 40.8 | 35.0 |




(a) CIFAR-10C (b) CIFAR-100C

Figure 2: Mean error rates for CIFAR-10C and CIFAR-100C using TENT and AaD

**CTTA:** Next, we consider the CTTA, as introduced by CoTTA Wang et al. (2022). Similar to the TTA setting, the continual benchmark also uses an off-the-shelf model pre-trained on the source domain. However, unlike the standard TTA setting, the continual setting does not require knowledge of when the domain changes, and instead adapts the model online to a sequence of test domains.

**DSS:** Finally, we use our domain shift signal to mimic the TTA setting while we are in the CTTA setting. By using the domain shift signal to dynamically set the model source even without having the underlying domain shift information. Thus the model can adapt to the changing distributions without accumulation of error, effectively mitigating the impact of domain shift.

### 4.3 Implementation details

For the online CTTA task, the source model is a network pre-trained on the clean CIFAR10 or CIFAR100 dataset. The network is evaluated on the largest corruption severity level 5 and is continually adapted to each corruption type in a sequential manner. The CIFAR10 experiments use a WideResNet-28 model while the CIFAR100 experiments use a ResNeXt-29 architecture, both adopted from the RobustBench benchmark. We perform all experiments with a batch size of 200. We update BN layers using the Adam optimizer with a learning rate of 1e-3 for TENT and 1e-4 for AaD. We use threshold $\tau$ of 0.98 for both CIFAR-10C and CIFAR-100C.

## 5 Conclusion

In this work, we propose a modular method for handling the challenge of continual test-time domain adaptation. We address the limitations of traditional single domain adaptation by developing a domain shift detection mechanism that continually measures the similarity between feature representations of consecutive batches. When a shift is detected, our method resets the model back to the source and continues test-time adaptation. Our experiments across standard datasets, batch sizes, and single domain test-time adaptation baselines demonstrate the effectiveness of our approach, making it a promising solution for the continual domain test-time adaptation problem.

## REFERENCES

Malik Boudiaf, Romain Mueller, Ismail Ben Ayed, and Luca Bertinetto. Parameter-free online test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8344–8353, 2022.

Dian Chen, Dequan Wang, Trevor Darrell, and Sayna Ebrahimi. Contrastive test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 295–305, 2022.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016.

Daehee Kim, Youngjun Yoo, Seunghyun Park, Jinkyu Kim, and Jaekoo Lee. Selfreg: Self-supervised contrastive regularization for domain generalization. In *ICCV*, 2021.

Pan Li, Da Li, Wei Li, Shaogang Gong, Yanwei Fu, and Timothy M. Hospedales. A simple feature augmentation for domain generalization. In *ICCV*, 2021.

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.

Mingsheng Long, ZHANGJIE CAO, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *NeurIPS*, volume 31, 2018.

Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*, 2018.

D. Wang, E. Shelhamer, S. Liu, B. Olshausen, and T. Darrell. Tent: Fully test-time adaptation by entropy minimization. In *ICLR*, 2021.

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*, pp. 7201–7211, 2022.

Ruijia Xu, Guanbin Li, Jihan Yang, and Liang Lin. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *ICCV*, October 2019.

Shiqi Yang, Yaxing Wang, Kai Wang, Shangling Jui, et al. Attracting and dispersing: A simple approach for source-free domain adaptation. In *Advances in Neural Information Processing Systems*, 2022.

K. Zhou, Y. Yang, Y. Qiao, and T. Xiang. Domain generalization with mixstyle. In *ICLR*, 2021.