pSTarC: Pseudo Source Guided Target Clustering for Fully Test-Time Adaptation

Manogna Sreenivas[†], Goirik Chakrabarty[*], Soma Biswas[†]
[†]IISc Bangalore [*]IISER Pune
{manognas, somabiswas}@iisc.ac.in goirik.chakrabarty@students.iiserpune.ac.in

Abstract

*Test Time Adaptation (TTA) is a pivotal concept in machine learning, enabling models to perform well in real-world scenarios, where test data distribution differs from training. In this work, we propose a novel approach called **p**seudo **S**ource guided **Tar**get **C**lustering (**pSTarC**) addressing the relatively unexplored area of TTA under real-world domain shifts. This method draws inspiration from target clustering techniques and exploits the source classifier for generating pseudo-source samples. The test samples are strategically aligned with these pseudo-source samples, facilitating their clustering and thereby enhancing TTA performance. pSTarC operates solely within the fully test-time adaptation protocol, removing the need for actual source data. Experimental validation on a variety of domain shift datasets, namely VisDA, Office-Home, DomainNet-126, CIFAR-100C verifies pSTarC's effectiveness. This method exhibits significant improvements in prediction accuracy along with efficient computational requirements. Furthermore, we also demonstrate the universality of the pSTarC framework by showing its effectiveness for the continuous TTA framework.*

1. Introduction

Over the past decade, deep networks have shown a continuous upward trend due to the availability of large datasets [4, 6, 19], significant improvements in computing power, and advancements in algorithms [8, 23] and architectures [9, 26]. But while humans can adapt seamlessly to new domains, the performance of deep networks deteriorate significantly when the test and training distributions differ. In practical scenarios, a trained model is often deployed in an unseen test environment, so equipping it with good adaptation capabilities to mitigate the adverse effects of any domain shift is crucial. Additionally, since access to the source data may be difficult because of privacy concerns or storage limitations, there is a significant interest in the following research directions: (i) Source-free Domain Adaptation (SFDA) [17, 33, 34], which assumes access to the source model and a large amount of unlabeled test data and (ii) Test-Time adaptation (TTA) [1, 2, 29], where test data arrives in an online manner, one batch at a time, allowing for one-step model adaptation followed by prediction. SFDA and TTA methods have been developed independently, resulting in fundamentally different approaches.

Here, we address the challenging and more practical task of swiftly adapting models without the need for extensive accumulation of test data, i.e. the TTA setting. Unlike SFDA methods which have been evaluated on real world domain shift datasets like VisDA [22], DomainNet [21] and Office-Home [28], TTA methods have primarily been evaluated within the confines of artificially corrupted data. It is only recently that researchers have started to address the TTA task for such real-world domain shifts [2, 13, 14].

In this work, we propose a simple yet effective TTA strategy termed **p**seudo **S**ource guided **Tar**get **C**lustering (pSTarC). It is inspired by the exceptional performance of SFDA techniques like SHOT [17], NRC [33], and AaD [34] in the context of the real world domain shift benchmarks. Notably, contemporary SFDA methods, including NRC and AaD, concentrate on refining target sample clustering, leveraging the luxury of abundant unlabeled target data. To extend this SFDA principle to TTA, one compelling avenue is the maintenance of a feature bank, which dynamically populates as new target data becomes available, enriching the adaptation process. While approaches like AdaContrast [2] have successfully harnessed this concept for TTA, they need to store auxiliary components like the momentum encoder and key features, a constraint that might not align well with an online TTA framework.

Our proposed pSTarC approach aims to leverage the power of SFDA objectives while adhering to the principle of minimizing memory and storage requirements for TTA task. Generally, the source-trained classifier remains unchanged during TTA to preserve the valuable class-discriminative insights gained from the source. Building on this insight, we introduce a novel strategy: utilizing the classifier to gener-

ate a diverse array of pseudo-source samples, thereby steering the target clustering process. Impressively, our findings reveal that generating as few as 20 pseudo-source samples per class is adequate to achieve state-of-the-art TTA performance, without imposing a significant burden on storage demands. Thus, the main contributions of this work can be summarized as follows:

1. We propose the pSTarC framework, which generates pseudo-source samples to guide the target clustering during test time adaptation.

2. We strive to achieve TTA using SFDA objectives, which not only improves the TTA performance significantly for real domain-shifts, but also helps to unify the seemingly disparate research directions.

3. pSTarC outperforms the state-of-the-art TTA techniques on Office-Home and DomainNet, and at par on VisDA, while requiring much lesser memory.

4. pSTarC also seamlessly works in Continual Test-Time Adaptation (CTTA) [31] scenario, where the test distribution changes with time. Here, its performance is at par with current state-of-the-art approaches on the large-scale DomainNet-126 benchmark.

In a nutshell, pSTarC aligns seamlessly with our objective to pave the way for swift, efficient TTA in the face of real-world domain shifts, building upon the insights garnered from the relationship between SFDA techniques and such demanding benchmarks.

2. Related Works

Here, we discuss the related work in Source-Free Domain Adaptation, Test-Time Adaptation, Continuous Test Time Adaptation and Model Inversion.

Source-free domain adaptation (SFDA) aims to adapt a source domain trained model to a target domain without access to any labeled data from either the source or target domain. SFDA methods typically assume access to abundant unlabeled data from the target domain and leverage the structure of the data to refine the target predictions. [17] proposes to cluster target features by mutual entropy maximization along with pseudo labeling, while keeping the classifier fixed. [2] extends the idea in [17] proposing to refine the pseudo labels using a feature bank, alongside doing self-supervised contrastive learning [3]. Another line of work include [33, 34], where they exploit the inherent semantic structure of the target features extracted from the source model. They reinforce consistency between the predictions of a sample and its local neighbors while also ensuring diversity to avoid degenerate predictions.

Test Time Adaptation (TTA) further relaxes the assumptions on data availability compared to SFDA. TENT [29] proposed the more practical fully test time adaptation setting, where source data cannot be accessed at all, and the model can only utilize the test samples in each batch encountered in an online manner for adaptation. They propose minimizing the entropy of the model predictions on the test data. More recently LAME [1] uses Concave-Convex Procedure (CCCP) to modify the feature vectors to obtain better classification, while AdaContrast [2] addresses SFDA and TTA settings by using contrastive learning with nearest neighbour soft voting for online pseudo label refinement. C-SFDA [14] uses curriculum learning in a Teacher Student framework. Other works like EATA [20] uses a small buffer from source distribution. TTN [18] trains a modified BN layer to leverage source data for improved TTA. In [13], they synthesize source proxy images by condensing the source dataset, which is then used during TTA after stylizing them to match the test distribution. Our work falls in the category of fully test-time adaptation [2, 14, 29, 31].

Continual Test Time Adaptation (CTTA) As a further extension of TTA, the concept of continual test-time adaptation (CTTA) has been recently introduced [31]. This protocol recognizes the dynamic nature of the testing environment, where the test domain evolves over time. CoTTA [31] adopts strategies like weight-averaged and augmentation-averaged predictions in a teacher-student framework to mitigate error accumulation. Additionally, it retains a fraction of neurons with source pre-trained weights during each iteration to prevent catastrophic forgetting, thus enabling model adaptation while preserving source knowledge. RMT [5] is a recent CTTA method that uses symmetric cross-entropy loss and contrastive loss in a teacher student framework.

Model inversion is a recent research direction explored in [15, 24, 30] for image generation where they optimize the input space to generate an image $\hat{x}$ using a pre-trained deep network. To do this, given an arbitrary target y which can be a label or a reference image, a trainable input $\hat{x}$ in the image space is initialized with random noise. This input space is then optimized by minimizing a loss function $\mathcal{L}(\hat{x}, y)$, which is usually cross-entropy loss and a regularizer $\mathcal{R}(\hat{x})$ to induce natural image prior. The training is done in an adversarial manner by alternating between the optimization of the synthesized image and that of the discriminator weights. Inspired by the effectiveness of these methods, here we propose a classifier guided *feature generation* approach, which is used for generating pseudo-source samples for guiding the clustering of the target data.

3. Problem Setting & Motivation

Firstly, the source model is trained using labeled source data. Then, in the Test Time Adaptation (TTA) stage, this model is adapted using the test batches in an online manner.

Source training: The model is first trained using labeled source data $\mathcal{D}_s = \{x_i^s, y_i^s\}_{i=1}^{n_s}$ comprising of C classes. Here, $x_i^s \in \mathcal{X}_s$ and $y_i^s \in \mathcal{Y}_s$ denote the source sample and its class label, and n_s is the number of training samples. We denote the source model as $\mathbf{F_s} = \mathbf{H_s} \circ \mathbf{G_s}$, where $\mathbf{G_s}$ is the feature extractor and $\mathbf{H_s}$ is the classifier. Following [2, 17, 34], the source model $\mathbf{F_s} : \mathcal{X}_s \rightarrow \mathcal{Y}_s$ is trained by minimizing the label-smoothing cross entropy loss as

Test Time Adaptation: Given the source model $\mathbf{F_s}$, during TTA, the target model $\mathbf{F_t}$ is initialized with the source model $\mathbf{F_s}$. We only have access to the unlabeled test samples x_t coming in batches from an unseen test distribution $\mathcal{D}_t$. Here, we address the closed setting where the source and target samples come from the same C classes. The goal is to continuously adapt $\mathbf{F_t} : \mathcal{X}_t \rightarrow \mathcal{Y}_t$ using the unlabeled samples $x_t \in \mathcal{X}_t$ (in batches) in an online manner.

Continual Test Time Adaptation: In addition to the above setup, the test data can come from multiple domains which changes over time such that $D_t^{(1)} \neq D_t^{(2)} \neq \ldots \neq D_s$ leading to the continual test time adaptation scenario.

4. Proposed Framework

The proposed pSTarC framework is based on effectively clustering the target samples which are available during test time. Our formulation is inspired by the clustering framework proposed in the state-of-the-art SFDA technique, AaD [34], which we briefly describe below.

Attracting and Dispersing (AaD): AaD [34] treats SFDA as an unsupervised clustering problem, where consistency is enforced 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

$$\mathcal{L}(x_i) = -\sum_{x_j \in \mathcal{N}_i} p_i^T p_j + \lambda \sum_{x_m \in \mathbf{x}_t} p_i^T p_m \tag{1}$$

where p_i refers to the softmax prediction vector of the sample $x_i \in \mathbf{x}_t$, p_j in the first term corresponds to the prediction vectors in its neighborhood $\mathcal{N}_i$, p_m in the second term corresponds to the prediction vectors of the samples x_m in the current batch $\mathbf{x}_t$.

Now, we describe the proposed pSTarC framework for fully TTA task, which we also illustrate in Fig.(1). In a TTA setting, as mentioned before, the labeled source samples are unavailable, and only the source model is available for adaptation. In addition, since the number of samples in a batch is usually quite low, it is a common practice to freeze the source trained classifier and update only the feature extractor to align target features with those of the source. Hence, we set $\mathbf{H_t} = \mathbf{H_s} = \mathbf{H}$ and only update the feature extractor $\mathbf{G_t}$ using the test data in an online manner. The goal is to adapt the test features such that they align with the source features so that the classifier $\mathbf{H}$ is transferable to test data. The classifier, being trained in a supervised manner using abundant source data, defines the decision boundaries for which the source data is perfectly classified. We leverage this fact to synthesize pseudo-source features, which are used to guide the target clustering. Given the source model, this process is only done once to store few features and corresponding prediction scores, and can be utilized throughout the TTA process. We describe the feature generation and clustering in detail below.

4.1. Pseudo Source Feature Generation

Since the decision boundaries in the feature space remain fixed (due to the classifiers remaining unchanged), it is important to align the target features with the original source features, which will inherently lead to better clustering and hence better classification of the target samples. First, we utilize the fixed source classifier to synthesize pseudo-source features. By aligning the target to these generated features, we hope to improve the adaptation performance of the model and make it more robust to the domain shift between the source and target domains.

Here, we aim to generate, say N pseudo-source features, where $N = C \times n_c$, C being the number of classes and n_c is the number of samples per class. We first randomly initialize a feature bank $\mathbf{f} \in \mathcal{R}^{N \times d}$, where d is the feature dimension. To compute the pseudo-source features, we use the information maximization loss which is a combination of entropy minimization and diversity maximization. These losses have been widely used in unsupervised clustering methods [17] to optimize a feature extractor to make the predictions of unlabeled samples diverse and confident. However, our objectives are very different. While they aim to learn a good feature extractor, our goal is to synthesize pseudo-source features given the source trained classifier $\mathbf{H}$. We want to generate features which are likely to be correctly classified by the source classifier. This is achieved by minimizing the following entropy loss:

$$\mathcal{L}_{ent}\left(\mathbf{f}; \mathbf{H}\right) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{C} \delta_k\left(\mathbf{H}\left(f_i\right)\right) \log \delta_k\left(\mathbf{H}\left(f_i\right)\right) \tag{2}$$

where $\delta_k\left(\mathbf{H}\left(f_i\right)\right)$ is the softmax score of class k for the pseudo-source feature $f_i \in \mathbf{f}$.

Along with this, we use diversity maximization loss to avoid the trivial solution where all feature vectors collapse to the same class. This ensures there are adequate number

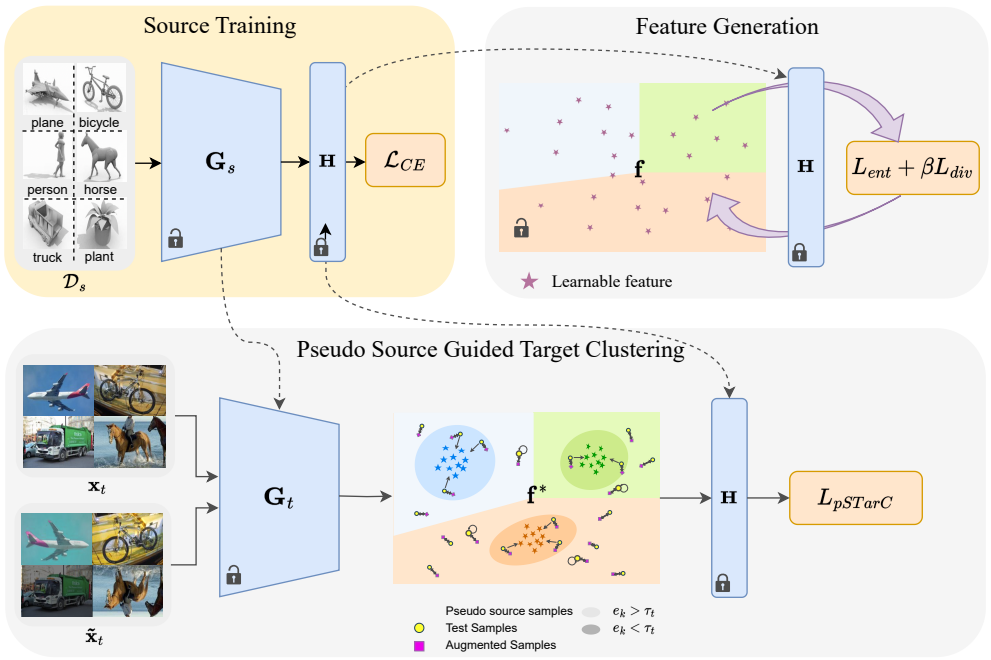

Figure 1. pSTarC Framework: (1) Feature Generation: We randomly initialize a feature bank $\mathbf{f}$ which is iteratively optimized keeping the classifier $\mathbf{H}$ fixed to minimize the entropy of the features while maximizing the diversity across classes using the loss in eqn (4). (2) Given the learnt features, we aim to bring the low entropy samples towards the corresponding pseudo-source features. We anchor the high entropy target samples to its own prediction. We also enforce consistency between the predictions of the test sample and its strong augmentation.

of feature vectors from each class in $\mathbf{f}$.

$$\begin{aligned}\mathcal{L}_{div}\left(\mathbf{f};\mathbf{H}\right) &= \sum_{k=1}^{C} \hat{p}_k \log \hat{p}_k \\ &= D_{KL}\left(\hat{p}, \frac{1}{C}\mathbf{1}_C\right) - \log C\end{aligned} \tag{3}$$

The loss is computed based on the mean softmax score of the test batch $\hat{p} = \mathbb{E}_{f\in\mathbf{f}}\left[\delta\left(h\left(f\right)\right)\right]$. The first term in the equation is the Kullback-Leibler (KL) divergence between the mean prediction vector $\hat{p}$ and the uniform distribution $\frac{1}{C}\mathbf{1}_C$. Here, $\hat{p}$ represents the marginal class distribution of the target data as estimated by the target model $\mathbf{F_t}$, C is the number of classes and $\mathbf{1}_C$ is a vector of ones with length C. The KL divergence measures the dissimilarity between two probability distributions, and in this context, it measures the discrepancy between the class distribution in the feature bank and the ideal case where all classes are equally represented. Overall, the diversity maximization loss encourages the feature bank to have a balanced representation of features across all classes, which is important for improving the clustering performance of the TTA algorithm. To summarize, we optimize the following

$$\mathbf{f}^* = \arg\min_{\mathbf{f}} \mathcal{L}_{ent}(\mathbf{f};\mathbf{H}) + \beta\mathcal{L}_{div}(\mathbf{f};\mathbf{H}) \tag{4}$$

In Fig.(2), we visualize the generated features on setting 20 samples per class for VisDA dataset.

4.2. Pseudo Source Guided Target Clustering

The use of feature bank has proven to be effective in Contrastive learning [7] and SFDA methods like AaD [34] and AdaContrast [2]. The proposed feature bank consists of pseudo-source features which are very different from the target feature bank used in [2, 34]. Unlike target features whose pseudo labels can be noisy, we can obtain clean labels for the generated pseudo-source features. We explain below how the generated features and their label information can be leveraged to better cluster and align the target features. We visually demonstrate the entire pSTarC framework in Fig.(1).

Pseudo-labeling based on confidence thresholding has been used very effectively in several applications [27]. Here, we propose a soft pseudo-labeling approach to cluster the target samples. Specifically, we identify the low entropy test samples based on a threshold τ_t, which we define as the mean entropy of the batch. We aim to align these selected test samples to the nearest pseudo-source samples which belong to the same class as the sample. Formally, given the generated feature bank $\mathbf{f}^*$, we first obtain their softmax score vectors and pseudo labels. We denote $p_i = \delta(\mathbf{H}(f_i))$ as the softmax score vector and $\hat{y}_i = \arg\max_c p_{i,c}$ as the pseudo label for feature f_i, where $p_{i,c}$ is the score of feature i for class c. We partition the features into sets S_c based on

Method	plane	bycyl	bus	car	horse	knife	mcycl	person	plant	sktbrd	train	truck	Average
Source	57.2	11.1	42.4	66.9	55.0	4.4	81.1	27.3	57.9	29.4	86.7	5.8	43.8
CAN* [12]	95.7	88.8	6.9	68.6	94.5	94.8	79.2	70.3	88.7	80.6	83.2	51.7	75.2
MCC* [11]	93.9	78.4	70.4	74.3	92.5	84.2	84.5	58.2	86.6	36.0	86.1	20.6	72.2
Source-Proxy TTA* [13]	92.5	82.4	85.8	74.2	92.7	88.5	83.9	85.8	92.8	62.5	75.2	32.5	79.1
BN-Adapt [25]	87.3	52.1	83.7	52.8	83.7	57.0	83.6	59.2	69.1	54.7	80.0	28.1	66.0
TENT [29]	91.1	45.6	86.4	66.4	88.7	75.1	90.3	76.4	84.4	47.1	83.6	13.7	70.7
AdaContrast [2]	95.0	68.0	82.7	69.6	94.3	80.8	90.3	79.6	90.6	69.7	87.6	36.0	78.7
C-SFDA [14]	95.9	75.6	88.4	68.1	95.4	86.1	94.5	82.0	89.2	80.2	87.3	43.8	**82.1**
pSTarC	95.1	82.1	83.6	61.2	93.8	89.9	87.9	80.7	90.9	81.9	87.6	48.1	81.9

Table 1. Average class accuracy (%) of pSTarC and other TTA methods on VisDA. * refers to methods utilizing source data to enable TTA.

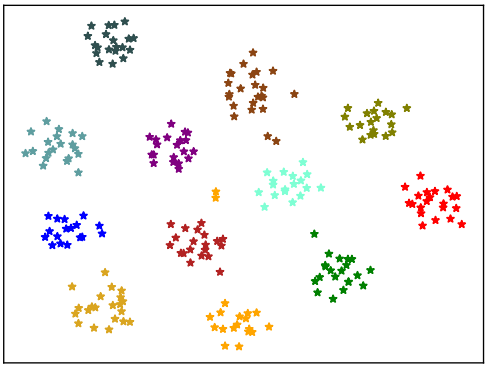

Figure 2. t-SNE plot of 240 generated pseudo-source features for TTA on VisDA dataset comprising of 12 classes.

their pseudo labels as follows:

$$S_c = \{f_i; \quad \hat{y}_i = c, f_i \in \mathbf{f}\}; \quad c \in \{1 \dots C\} \tag{5}$$

These sets are obtained once for the pseudo-source features generated and kept fixed throughout the adaptation process.

Given a test batch $\mathbf{x}_t$, we first obtain their confidence scores and pseudo labels and set the threshold $\tau_t = \mathbb{E}_{x_k \in \mathbf{x}_t}[e_k]$, the mean entropy of the batch. For a test sample $x_k \in \mathbf{x}_t$ (test batch), we denote its pseudo label as $\hat{y}_k$ and compute the sample entropy as e_k. For this sample, we define its positive set $\mathbf{p}^+$ based on its entropy e_k as follows: (1) When $e_k < \tau_t$, we define the positives to be K nearest pseudo-source samples from set $S_{\hat{y}_k}$. (2) For samples which have high entropy, i.e, with $e_k > \tau_t$, as the pseudo labels can be highly noisy, it is not desirable to enforce them to align towards any pseudo-source samples. Instead, we anchor it to its own prediction vector by setting $\mathbf{p}^+ = \{p_k\}$. In addition, we use its strong image augmentation $\tilde{x}_k$ to enforce prediction consistency between p_k and $\tilde{p}_k$, the prediction vector of $\tilde{x}_k$. This helps the model to be invariant to image transformations and improves its generalization ability. We also use the dispersion loss that makes a sample dissimilar to the other samples in the batch, which is representative of the test data in all. This dispersion loss prevents the model from the trivial solution of all test samples collapsing to the same class. Our objective now is to make the predictions of the target embeddings similar to its positives without facing mode collapse, which we achieve by optimizing the following loss:

$$\mathcal{L}_{\text{pSTarC}}(x_k) = \underbrace{-p_k^T \tilde{p}_k}_{L_{aug}} \underbrace{- \sum_{p_j^+ \in \mathbf{p}^+} p_k^T p_j^+}_{L_{attr}} + \lambda \underbrace{\sum_{x_j \in \mathbf{x}_t} p_k^T p_j}_{L_{disp}} \tag{6}$$

We perform one step optimization on test batch $\mathbf{x}_t$ using this loss and then predict their labels. This process is repeated for each batch in the TTA setting.

What makes pSTarC an effective framework?

1. We operate in the *fully test-time scenario*, i.e., we do not assume access to source data in any form unlike some prior methods [11–13], which use the source data to equip the model for future TTA. In pSTarC, we leverage the classifier which is a part of the given source model to synthesize pseudo-source features to enable clustering during test time.

2. Feature banks have been effectively used in AdaContrast [2] to cluster the test data. However, it is expensive to have multiple large memory buffers which have to be continuously updated. We propose a *simple one-step pseudo source generation* framework. These generated features can be used during TTA forever, as the final goal indeed is to align the test distribution to the source distribution.

3. pSTarC is a *memory efficient framework* as we only store the online updating model, in contrast to AdaContrast [2] and C-SFDA [14] where they need to store the student and teacher model. Our framework is also *more efficient in runtime* as we only forward pass the image and its strong augmentation, while the state-of-the-art method C-SFDA [14] uses 12 augmentations.

5. Experimental Evaluation

We evaluate the proposed framework extensively on three real-world domain shift datasets, namely VisDA [22], DomainNet-126 [21] and Office-Home [28] and also on a

Method	A → C	A → P	A → R	C → A	C → P	C → R	P → A	P → C	P → R	R → A	R → C	R → P	Average
Source	44.6	66.5	73.5	51.0	61.9	63.2	51.1	40.5	71.9	64.4	47.1	77.3	59.4
BN-Adapt [25]	38.9	59.9	71.5	55.0	62.0	65.2	54.4	37.3	71.6	65.2	41.3	73.8	58.0
TENT [29]	39.1	60.2	71.6	55.2	62.2	65.5	54.6	37.6	71.8	65.3	41.6	73.9	58.2
AdaContrast [2]	42.2	64.5	73.2	56.2	64.1	66.4	54.7	40.4	73.0	66.7	45.1	75.6	60.2
pSTarC	47.7	68.7	75.4	58.6	68.4	68.9	55.1	45.8	75.6	67.5	51.8	78.7	**63.5**

Table 2. Total accuracy (%) of pSTarC and other TTA methods on Office-Home dataset.

Method	gaussian	shot	impulse	defocus	glass	motion	zoom	snow	frost	fog	brightness	contrast	elastic	pixelate	jpeg	Average
Source	27	32	60.6	70.7	45.9	69.2	71.2	60.5	54.2	49.7	70.5	44.9	62.8	25.3	58.8	53.6
BN Adapt [25]	57.9	59.3	57.3	72.4	58.1	70.3	72.1	65.1	65	58.5	73.5	69.7	64.3	67.1	58.8	64.6
TENT [29]	62.7	65.1	65.5	75.0	62.6	72.5	75.0	69.6	68.1	66.2	76.0	71.8	67.1	71.6	63.1	68.8
AdaContrast [2]	57.3	59.4	61.1	73.4	58.8	71.1	73.4	66.6	67.3	60.7	75.2	71.8	65.4	65.8	60.5	65.9
pSTarC	63.4	65.4	66.5	75	63	73.2	74.9	70.3	69.8	66.5	76.6	73.2	68.0	72.2	63.8	**69.5**

Table 3. Accuracy (%) of different TTA methods on 15 corruptions from CIFAR-100C dataset in TTA setting.

Method	R→C	R→P	P→C	C→S	S→P	R→S	P→R	Average
Source	55.5	62.7	53	46.9	47.3	46.3	75.0	55.2
BN-Adapt [25]	54.1	62.8	54.3	49.4	59.1	47.6	75.0	57.5
TENT [29]	55.6	64.5	55.5	50.8	59.9	49.9	75.9	58.9
AdaContrast [2]	61.1	66.9	60.8	53.4	62.7	54.5	78.9	62.6
C-SFDA [14]	61.6	67.4	61.3	55.1	63.2	54.8	78.5	63.1
pSTarC	60.8	67.7	60.3	55.6	65.3	55.8	80.2	**63.7**

Table 4. Total accuracy (%) of TTA methods on DomainNet-126.

corruption benchmark dataset, namely CIFAR100C [10].

Datasets: **VisDA** is a challenging dataset for object recognition tasks with synthetic to real domain shift. The target domain consists of $55,388$ real object images from 12 classes. **Office-Home** contains four domains - Real, Clipart, Art, Product and 65 classes with a total of $15,500$ images. **DomainNet-126** is a subset of DomainNet consisting of 126 classes from four domains, namely Real, Sketch, Clipart and Painting. **CIFAR-100C** is a corruption benchmark with domain shifts like gaussian noise, blur, weather changes, etc. Following [31], we use severity level 5 corruptions. For VisDA-C, we compare the average of per-class accuracies while for the other datasets, we compare the average of total accuracy across domain shifts.

Model Architecture: For TTA experiments, we use ResNet-50 [9] as the backbone for Office-Home and DomainNet-126 datasets and ResNet-101 [9] for the VisDA dataset. We use the same network architecture as in [2], in which the final part of the network is modified to include fully connected layer and Batch Normalization, and then followed by a classifier, which is a fully connected layer with weight normalization. For CIFAR-100C , we use ResNeXt [32] as used in [5, 31].

Implementation details: We use Pytorch framework and run all experiments on a single NVIDIA A-5000 GPU. For source training, following [2, 14] the model is initialized with ImageNet pre-trained weights and trained for 10, 60 and 50 epochs for VisDA, DomainNet-126 and Office-Home respectively. During test time adaptation, we only update the backbone parameters, keeping the classifier fixed for all experiments. Following [2, 13, 14], we set the batch size to 128 in all experiments for VisDA, DomainNet-126 and Office-Home. We use SGD as the optimizer with learning rate of 5e-4 and momentum 0.9. Following [5, 31], for CIFAR-100C, the batch size is set to 200 and we use Adam [16] optimizer with learning rate of 1e-3. We set β to 5 in eqn.(4) and the number of features per class n_c to 20 in all experiments. We report the results of prior methods from the respective papers. We use the official code provided by AdaContrast [2] to perform experiments on Office-Home and also adapt it to CTTA setting. In the Supplementary material, we describe the image augmentations used, analysis on parameter n_c and provide the pseudo code for pSTarC.

5.1. Evaluation for TTA setting

We compare the performance of our proposed pSTarC framework with the prior TTA approaches [2, 13, 14, 25, 29]. For VisDA dataset, from Table 1, we observe that pSTarC performs at par with the state-of-the-art method C-SFDA [14], while being computationally much more efficient (Table 9). Interestingly, it also outperforms the approaches which assume access to the source data before performing TTA. On Office-Home, we get a significant improvement of 3.5% compared to the prior TTA method AdaContrast [2] as shown in Table 2. On DomainNet-126, from Table 4, we observe that pSTarC achieves an average accuracy of 63.7% across 7 domain shifts, outperforming all the existing approaches including [14]. On CIFAR-100C [10], our method performs 1.1% better than TENT [29] and 3.6% better than AdaContrast [2], suggesting its effectiveness

Method	gaussian	shot	impulse	defocus	glass	motion	zoom	snow	frost	fog	brightness	contrast	elastic	pixelate	jpeg	Average
Source	27	32	60.6	70.7	45.9	69.2	71.2	60.5	54.2	49.7	70.5	44.9	62.8	25.3	58.8	53.6
BN Adapt [25]	57.9	59.3	57.3	72.4	58.1	70.3	72.1	65.1	65	58.5	73.5	69.7	64.3	67.1	58.8	64.6
TENT [29]	62.8	64.2	58.3	62.1	48.8	51.7	51.5	41.6	36.3	28.9	29.6	17.7	12.0	11.5	9.6	39.1
CoTTA [31]	59.9	62.3	60.3	73.1	62.0	72.1	73.6	67.2	68.2	59.7	75.3	73.1	67.5	71.7	66.5	67.5
AdaContrast [2]	57.7	63.2	61.4	72.3	59.9	70.9	72.5	67.1	69.3	61.8	74.1	71.7	66.1	66.7	63.8	66.6
RMT [5]	59.5	63.9	63.7	72.3	66.1	71.5	73.6	71.0	71.0	67.5	74.9	72.6	71.8	73.7	70.7	**69.6**
pSTarC	63.4	67.0	64.0	71.1	62.9	69.3	72.4	67.3	68.7	64.1	72.9	71.9	66.7	70.5	62.9	67.7

Table 5. Accuracy (%) of different methods on 15 corruptions from CIFAR-100C dataset in CTTA setting.

Method	Real→	Clipart→	Painting→	Sketch→	Average
Source only	54.7	50.7	58.3	55.2	54.7
BN Adapt [25]	54.9	54.8	60.5	62.2	58.1
TENT [29]	57.6	55.8	62.8	62.5	59.7
CoTTA [31]	56.6	57.0	63.6	63.7	60.2
AdaContrast [2]	62.2	62.4	67.7	68.1	65.1
RMT [5]	63.0	62.1	68.3	67.9	65.3
pSTarC	62.7	63.6	67.6	68.1	**65.5**

Table 6. Accuracy (%) of different TTA methods on four domain shift sequences from DomainNet-126 in CTTA setting.

even on corruption domain shifts (Table 3).

5.2. Evaluation for CTTA setting

We also study the effectiveness of pSTarC in the CTTA setting where test domains change with time. To do this, we perform experiments on CIFAR-100C and the following four domain sequences from DomainNet-126:
(1) *Real*-World→Clipart→Painting→Sketch;
(2) *Clipart*→Sketch→Real-World→Painting;
(3) *Painting*→Real-World→Sketch→Clipart
(4) *Sketch*→Painting→Clipart→Real-World.

The *first domain* indicates the source domain, which is then adapted to the other three test domains in the above sequence. From Table 6, we observe that pSTarC outperforms all the state-of-the-art approaches in this challenging setting. Specifically, it outperforms CoTTA by a significant margin of 5.3% and also performs favourably compared to the state-of-the-art method RMT [5]. In addition, we also evaluate pSTarC on CIFAR-100C continual setting and report the results in Table 5. It performs favourably compared to AdaContrast [2] and CoTTA [31], while RMT [5] performs the best in this case. But, CoTTA [31] and RMT [5] are computationally more expensive as they need to store teacher and student models, while pSTarC is more lightweight as it only stores one model.

In Figure 3, we summarize the performance of pSTarC with the source model, TENT [29] and AdaContrast [2]. In this plot, the lines farther from the center indicates better performance. We observe that pSTarC outperforms these methods across all domain shifts for both TTA and CTTA.

L_{aug}	L_{attr}	L_{disp}	VisDA	DomainNet-126
✓	✓		68.8	58.8
✓		✓	78.2	59.7
	✓	✓	80.0	63.0
✓	✓	✓	**81.9**	**63.7**

Table 7. Ablation study: Importance of each loss term.

5.3. Additional Analysis

Here, we report the results of additional analysis to better understand the proposed framework.

Ablation Study: The proposed pSTarC framework consists of three loss components. The first component is L_{aug} which enforces consistency between an image and its augmentation. From Table 7, we observe that using strong augmentations can indeed help improve the feature representations, as we get 1.9% and 0.7% improvement on VisDA and DomainNet-126 respectively. The second component L_{attr} aims to align the test features with the pseudo source features. On removing the attraction loss component from L_{pSTarC}, the loss becomes similar to contrastive learning. While this performs reasonably, achieving 78.2% and 59.7% on VisDA and DomainNet respectively, incorporating the pseudo-source features improves the results significantly by 3.7% and 4%, proving that they indeed help model adaptation by correctly aligning the test features so that the source trained classifier can well classify the test data. The third component, L_{disp} is the dispersion term which prevents the model to avoid all the test features collapsing to one cluster, which is a trivial solution when optimizing only the attraction loss L_{attr}. This term plays a role similar to the diversity term and is crucial in any unsupervised adaptation protocols [17, 34] to avoid model collapse, the effect of which we observe in Table 7. The accuracy on VisDA and DomainNet-126 drop to 68.8% and 58.8% respectively, as the test samples would be predicted into lesser number of classes than actually present in the dataset.

Performance on varying batch sizes: In TTA, it is

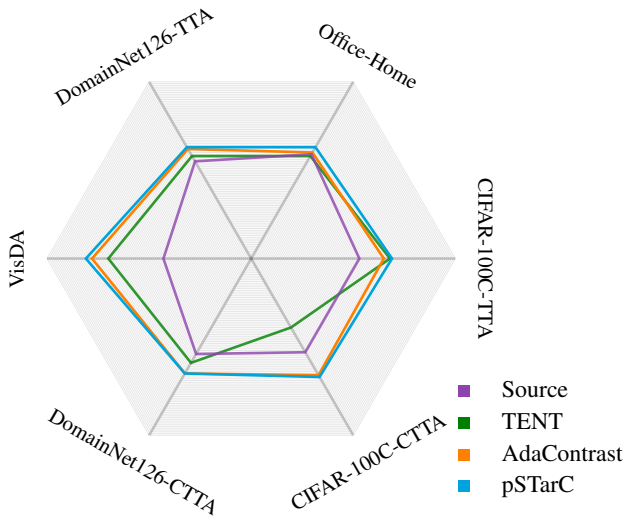

Figure 3. Overall comparison of pSTarC with TTA methods.

crucial for the method to be able to continuously adapt even with very few samples. In this analysis, we vary the batch size from 8 to 128 and perform experiments on the DomainNet-126 dataset. Table8 reports the average accuracy across 7 domain shifts for each batch size. We observe that the proposed pSTarC consistently outperforms both TENT [29] and AdaContrast [2] for all batch sizes. The effect is more pronounced for the smallest batch size 8, where pSTarC outperforms TENT by a huge margin of 15.3% and AdaContrast by 4%. On average, pSTarC does better than TENT by 6.2% and AdaContrast by 1.7%.

Complexity Analysis: Here, we analyse the complexity of pSTarC and three other recent TTA methods: AdaContrast [2], Source-Proxy-TTA [13] and C-SFDA [14] on VisDA dataset. In the TTA setting, it is desirable to have methods that requires storing less additional information due to memory limitations and privacy concerns. The prior methods AdaContrast [2] and C-SFDA [14] are based on the teacher student framework. Hence, it needs to store twice the number of model parameters, while we only store the updating model parameters in pSTarC, as we report in Table 9. AdaContrast stores a memory queue of size 16384 to collect key features (of dimension 256), and its pseudo labels, which is used to retrieve positives for contrastive learning. Alongside, they store another feature bank (of size 1024) and their corresponding scores which is used to retrieve neighbours for soft pseudo-labeling the target samples. Thus, the total memory buffer required for AdaContrast is 16384x(256+1)+1024x(256+12). [13] condenses the source data to save 25 images per class of size 112x112 for VisDA dataset. This accounts to a memory requirement of 37.6M (12x25x112x112). On the other hand, in the pSTarC framework, we only store 20 features per class and the corresponding scores resulting in a memory buffer of $240\text{x}(256+12)$. C-SFDA does not store any features or images. However, they need 13 forward passes (12 augmentations in addition to the actual test sample), while AdaContrast [2] and Source-Proxy-TTA [13] uses 3 augmentations, and pSTarC uses only two augmentations. We summarize this in Table 9, which shows that pSTarC is very efficient, in addition to achieving better or performance comparable to the state-of-the-art across several challenging settings.

Method	Batch size 8	16	32	64	128	Average
TENT	38.8	55.4	58.6	59.1	58.9	54.2
AdaContrast	50.1	57.9	60.8	62.4	62.4	58.7
pSTarC	54.1	59.2	61.3	63.8	63.7	**60.4**

Table 8. Ablation on batch size using DomainNet-126

Method	AdaContrast	Source-Proxy-TTA	C-SFDA	pSTarC
#Parameters	86M	43M	86M	43M
Memory	4.67M	3.76M	-	0.03M
#Forward	3	3	13	2
#Backward	1	1	1	1

Table 9. Complexity Analysis of TTA methods on VisDA

6. Conclusion

In this paper, we have proposed a novel framework termed pSTarC for Test Time Adaptation (TTA) of deep neural networks. pSTarC leverages the fixed source classifier to generate pseudo-source samples, which is then used to align the test samples, which enables the source trained classifier to classify test data from different distributions. Extensive experiments on several real-world domain shift datasets justify the effectiveness of our proposed framework. Additionally, we also show that the method can seamlessly be used in continual test time adaptation scenario, though there is still scope for improvement in the corruption datasets. Overall, our findings highlight the importance of target clustering techniques and leveraging the source classifier for improving test-time adaptation performance in several real-world challenging scenarios.

Acknowledgements This work is partly supported through a research grant from SERB (SPF/2021/000118), Govt. of India. The first author is supported by Prime Minister's Research Fellowship awarded by Govt. of India.

References

[1] Malik Boudiaf, Romain Mueller, Ismail Ben Ayed, and Luca Bertinetto. Parameter-free online test-time adaptation. In *CVPR*, 2022.

[2] Dian Chen, Dequan Wang, Trevor Darrell, and Sayna Ebrahimi. Contrastive test-time adaptation. In *CVPR*, 2022.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[4] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[5] Mario Döbler, Robert A Marsden, and Bin Yang. Robust mean teacher for continual and gradual test-time adaptation. In *CVPR*, 2023.

[6] Mark Everingham, Luc Gool, Christopher K. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010.

[7] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

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

[11] Ying Jin, Ximei Wang, Mingsheng Long, and Jianmin Wang. Minimum class confusion for versatile domain adaptation. In *ECCV*, 2020.

[12] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *CVPR*.

[13] Juwon Kang, Nayeong Kim, Kwon Donghyeon, Jungseul Ok, and Suha Kwak. Leveraging proxy of training data for test-time adaptation. In *ICML*, 2023.

[14] Nazmul Karim, Niluthpol Chowdhury Mithun, Abhinav Rajvanshi, Han-pang Chiu, Supun Samarasekera, and Nazanin Rahnavard. C-sfda: A curriculum learning aided self-training framework for efficient source free domain adaptation. In *CVPR*, 2023.

[15] Yujin Kim, Dogyun Park, Dohee Kim, and Suhyun Kim. Naturalinversion: Data-free image synthesis improving real-world consistency. In *AAAI*, 2022.

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[17] 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.

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

[19] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *ECCV*, 2014.

[20] Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Yaofo Chen, Shijian Zheng, Peilin Zhao, and Mingkui Tan. Efficient test-time model adaptation without forgetting. In *ICML*, 2022.

[21] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *ICCV*, 2019.

[22] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*, 2017.

[23] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *NeurIPS*, 2015.

[24] Mozhdeh Rouhsedaghat, Masoud Monajatipoor, Kai-Wei Chang, C-C Jay Kuo, and Iacopo Masi. Magic: Mask-guided image synthesis by inverting a quasi-robust classifier. *arXiv preprint arXiv:2209.11549*, 2022.

[25] Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. *NeurIPS*, 2020.

[26] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[27] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *NeurIPS*, 2020.

[28] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, 2017.

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

[30] Pei Wang, Yijun Li, Krishna Kumar Singh, Jingwan Lu, and Nuno Vasconcelos. Imagine: Image synthesis by image-guided model inversion. In *CVPR*, 2021.

[31] Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. Continual test-time domain adaptation. In *CVPR*, 2022.

[32] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017.

[33] Shiqi Yang, Joost van de Weijer, Luis Herranz, Shangling Jui, et al. Exploiting the intrinsic neighborhood structure for source-free domain adaptation. *NeurIPS*, 2021.

[34] Shiqi Yang, Yaxing Wang, Kai Wang, Shangling Jui, et al. Attracting and dispersing: A simple approach for source-free domain adaptation. In *NeurIPS*, 2022.