




In Search of Lost Online Test-Time Adaptation: A Survey

Zixin Wang¹ · Yadan Luo¹  · Liang Zheng² · Zhuoxiao Chen¹ · Sen Wang¹ · Zi Huang¹

Received: 14 December 2023 / Accepted: 15 July 2024
© The Author(s) 2024

Abstract

This article presents a comprehensive survey of online test-time adaptation (OTTA), focusing on effectively adapting machine learning models to distributionally different target data upon batch arrival. Despite the recent proliferation of OTTA methods, conclusions from previous studies are inconsistent due to ambiguous settings, outdated backbones, and inconsistent hyperparameter tuning, which obscure core challenges and hinder reproducibility. To enhance clarity and enable rigorous comparison, we classify OTTA techniques into three primary categories and benchmark them using a modern backbone, the Vision Transformer. Our benchmarks cover conventional corrupted datasets such as CIFAR-10/100-C and ImageNet-C, as well as real-world shifts represented by CIFAR-10.1, OfficeHome, and CIFAR-10-Warehouse. The CIFAR-10-Warehouse dataset includes a variety of variations from different search engines and synthesized data generated through diffusion models. To measure efficiency in online scenarios, we introduce novel evaluation metrics, including GFLOPs, wall clock time, and GPU memory usage, providing a clearer picture of the trade-offs between adaptation accuracy and computational overhead. Our findings diverge from existing literature, revealing that (1) transformers demonstrate heightened resilience to diverse domain shifts, (2) the efficacy of many OTTA methods relies on large batch sizes, and (3) stability in optimization and resistance to perturbations are crucial during adaptation, particularly when the batch size is 1. Based on these insights, we highlight promising directions for future research. Our benchmarking toolkit and source code are available at https://github.com/Jo-wang/OTTA_ViT_survey.

Keywords Online test-time adaptation · Domain shift · Transfer learning

1 Introduction

Dataset shift (Quinonero-Candela et al., 2008) poses a notable challenge for machine learning. Models often expe-

rience significant performance drops when confronting test data characterized by superior *distribution differences* from training. Such differences might come from changes in style and lighting conditions and various forms of corruption, making test data deviate from the data upon which these models were initially trained. To mitigate the performance degradation brought by these shifts, test-time adaptation (TTA) has emerged as a promising solution. TTA aims to rectify the dataset shift issue by adapting the model to unseen distributions using unlabeled test data (Liang et al., 2023). Different from unsupervised domain adaptation (Ganin & Lempitsky, 2015; Wang et al., 2022; Wang & Deng, 2018; Chen et al., 2023a, b, 2021; Wang et al., 2020; Luo et al., 2020), TTA does not require access to source data for distribution alignment. Commonly used strategies in TTA include unsupervised proxy objectives, spanning techniques such as pseudo-labeling (Liang et al., 2020), graph-based learning (Luo et al., 2023), and contrastive learning (Chen et al., 2022), applied on the test data through multiple training epochs to enhance model accuracy, such as autonomous vehi-

Communicated by Zhun Zhong.

✉ Yadan Luo
y.luo@uq.edu.au
Zixin Wang
zixin.wang@uq.edu.au
Liang Zheng
liang.zheng@anu.edu.au
Zhuoxiao Chen
zhuoxiao.chen@uq.edu.au
Sen Wang
sen.wang@uq.edu.au
Zi Huang
helen.huang@uq.edu.au

¹ The University of Queensland, Brisbane, Australia

² The Australian National University, Canberra, Australia

cle detection (Hegde et al., 2021; Chen et al., 2024), pose estimation (Lee et al., 2023), video depth prediction (Liu et al., 2023a), frame interpolation (Choi et al., 2021), and medical diagnosis (Ma et al., 2022; Wang et al., 2022c; Saltori et al., 2022). Nevertheless, requiring access to the complete test set at every time step may not always align with practical use. In many applications, such as autonomous driving, adaptation is restricted to using only the **current test batch** processed in a streaming manner. Such operational restrictions make it untenable for TTA to require the full test set tenable.

In this study, our focus is on a specific line of TTA methods, i.e., online test-time adaptation (OTTA), which aims to accommodate *real-time changes* in the test data distribution. We provide a comprehensive overview of existing OTTA studies and evaluate the efficiency and effectiveness of these methods. To facilitate a structured OTTA landscape, we categorize existing approaches into three groups: optimization, data, and model-based OTTA.

- *Optimization-based OTTA* focuses on various optimization methods. Examples are designing new loss functions, updating normalization layers (e.g., BatchNorm) during testing, using pseudo-labeling strategies, teacher-student frameworks, and contrastive learning-based approaches, *etc.*
- *Data-based OTTA* maximizes prediction consistency across diversified test data. Diversification strategies use auxiliary data, improved data augmentation methods, and diffusion techniques and create a saving queue for test data, *etc.*
- *Model-based OTTA* adjusts the model backbone, such as modifying specific layers or their mechanisms, adding supplementary branches, and incorporating prompts.

It is potentially useful to combine methods from different categories for further improvement. An in-depth analysis of this strategy is presented in Sect. 3. Note that this survey does not include the paper if source-stage customization is needed, such as (Thopalli et al., 2022; Döbler et al., 2023; Brahma & Rai, 2023; Jung et al., 2022; Adachi et al., 2023; Lim et al., 2023; Choi et al., 2022; Chakrabarty et al., 2023; Marsden et al., 2022; Gao et al., 2023), as they necessitate extra operation or information at the source pre-training stage, which may not be feasible in some scenarios and cannot form a fair comparison.

Differences from the existing survey. Liang et al. (2023) provided a comprehensive overview of the vast topic of test-time adaptation (TTA), discussing TTAs in diverse configurations and their applicability in vision, natural language processing (NLP), and graph data analysis. One limitation is that the survey does not provide experimental comparisons of existing methods. Mounsaveng et al. (2023) studied fully test-time

adaptation for some specific components, e.g., batch normalization, calibration, class re-balancing, *etc.* However, it does not focus too much on analyzing the existing methods and exploring ViTs. Marsden et al. (2024) conducted comprehensive study on universal test-time adaptation setting. In contrast, our survey concentrates on purely online TTA approaches and provides valuable insight from experimental comparisons, considering various domain shifts, hyperparameter selection, and backbone influence (Zhao et al., 2023b).

Contributions. As Vision Transformer (ViT) architectures gain increasing prominence, a critical question arises: *Do OTTA strategies, originally devised for CNNs, retain their effectiveness when applied to ViT models?* This question stems from the significant architectural differences between ViTs and conventional CNNs, such as ResNets, particularly in their normalization layers and information processing mechanisms. Given the growing adoption of ViTs, investigating their compatibility with existing OTTA strategies is essential. To thoroughly explore this question, we evaluate eight representative OTTA algorithms across diverse distribution shifts, employing a set of metrics to evaluate both effectiveness and efficiency. Below, we summarize the key contribution of this survey:

- *[A focused OTTA survey]* To the best of our knowledge, this is the first focused survey on online test-time adaptation, which provides a thorough understanding of three main working mechanisms. Experimental investigations are conducted in a fair comparison setting.
- *[Comprehensive Benchmarking and Adaptation of OTTA Strategies with ViT]* We reimplemented representative OTTA baselines under the ViT architecture and testified their performance against six benchmark datasets. We drive a set of replacement rules that adapt the existing OTTA methods to accommodate the new backbone.
- *[Both accuracy and efficiency as evaluation Metrics]* Apart from using the traditional recognition accuracy metric, we further provide insights into various facets of computational efficiency by Giga floating-point operations per second (GFLOPs), wall clock time, and GPU memory usage. These metrics are important in real-time streaming applications and can be treated as a supplementary of (Marsden et al., 2024).
- *[Real-world testbeds]* While existing literature extensively explores OTTA methods on both corruption datasets and real-world datasets (Marsden et al., 2024), the diverse difficulty levels of these datasets hinder a fair comparison. Therefore, we further assess OTTA performance on CIFAR-10-Warehouse, a newly introduced, expansive test set of CIFAR-10, to ensure a comprehensive comparison across the same label set. Using the same pre-trained model for adaptation, we provide insights into

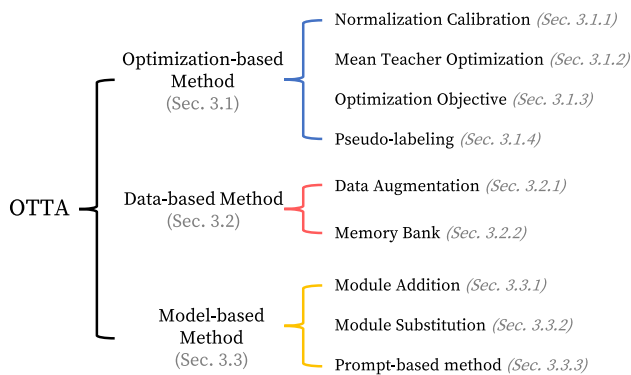


Fig. 1 Taxonomy of existing OTTA methods. The categories, i.e., optimization-, data-, and model-based, inform three mainstream working mechanisms. To provide a clear illustration, methods related to prompts are categorized into model-based methods

various domain shifts that were previously unexplored in the existing survey (Liang et al., 2023).

This work aims to summarize existing OTTA methods with the aforementioned three categorization criteria and analyze some representative approaches by empirical results. Moreover, to assess real-world potential, we conduct comparative experiments to explore the portability, robustness, and environmental sensitivity of the OTTA components. We expect this survey to offer a systematic perspective in navigating OTTA’s intricate and diverse solutions. We also present new challenges as potential future research directions.

Organization of the survey. The rest of this survey will be organized as follows. Section 2 presents the problem definition and introduces widely used datasets, metrics, and applications. Using the taxonomy shown in Fig. 1, Sect. 3 comprehensively reviews existing OTTA methods. With Vision Transformer as new backbones, Sect. 4 empirically analyzes eight state-of-the-art methods by multiple evaluation metrics on corrupted and real-world distribution shifts. We introduce the potential future directions in Sect. 5 and conclude this survey in Sect. 6.

2 Problem Overview

Online Test-time Adaptation (OTTA), with its real-time characteristics, represents a critical approach in test-time adaptation. This section provides a formal definition of OTTA and delves into its fundamental attributes. Furthermore, we explore widely used datasets and evaluation methods, and examine the potential application scenarios of OTTA. To ensure a clear understanding, a comparative analysis is also undertaken to differentiate OTTA from similar settings.

2.1 Problem Definition

In OTTA, we assume access to a trained source model and adapt the model at test time over the test input to make the final prediction. The given source model f_{θ^S} parameterized by θ^S is pre-trained on a labeled source domain $\mathcal{D}_S = \{(\mathbf{x}^S, \mathbf{y}^S)\}$, which is formed by i.i.d. sampling from the source distribution p_S . Unlabeled test data $\mathcal{D}_T = \{\mathbf{x}_1^T, \mathbf{x}_2^T, \mathbf{x}_t^T, \dots, \mathbf{x}_n^T\}$ come in batches, where t indicates the current time step and n is the total number of time steps (i.e., number of batches). Test data often come from one or multiple different distributions $(\mathbf{x}_t^T, \mathbf{y}_t^T) \sim p_T$, where $p_S(\mathbf{x}, \mathbf{y}) \neq p_T(\mathbf{x}, \mathbf{y})$ under the covariate shift assumption (Huang et al., 2006). During TTA, we update the model parameters for batch t , resulting in an adapted model f_{θ^t} .

Before adaptation, the pre-trained model is expected to retain its original architecture, especially the backbone, without modifying its layers or introducing new model branches during training. Additionally, the model is restricted to observing the test data only once and must produce predictions promptly online. By refining the definition of OTTA in this manner, we aim to minimize limitations associated with its application in real-world settings. Note that the model is reset to its original pre-trained state after being adapted to a specific domain, i.e., $f_{\theta^S} \rightarrow f_{\theta^0} \rightarrow f_{\theta^S} \rightarrow f_{\theta^1} \rightarrow f_{\theta^S} \rightarrow \dots \rightarrow f_{\theta^t}$.

Since there is no way to align the source and test set as in unsupervised domain adaptation in OTTA, what optimization objective works in this limited environment? As test data arrive at a fixed pace, how much data is ideal for effective test-time adaptation? Will adaptation work with new backbones (e.g., ViTs)? Does “Test-time Adaptation” lose validity with backbone changes? To address these concerns, we explore OTTA methods by their datasets, evaluations, and applications, decoupling strategies to identify which components work and why with updated backbones.

2.2 Datasets

This survey summarizes datasets in image classification, while recognizing that OTTA has been applied to many downstream tasks (Ma et al., 2022; Ding et al., 2023; Saltori et al., 2022). Testbeds in OTTA usually seek to facilitate adaptation from natural images to corrupted ones. The latter are created by perturbations such as Gaussian noise and Defocus blur. Despite including corruptions at varying severities, these synthetically induced corruptions may not sufficiently mirror the authentic domain shift encountered in real-world scenarios. Our work uses corruption (Croce et al., 2021), generated images, and real-world shift datasets, summarized in Table 1. Details of each testbed are described below.

CIFAR-10-C is a standard benchmark for image classification. It contains 950,000 color images, each of 32×32 pixels,

Table 1 Datasets used in this survey

Datasets	# Domains	# Images	# Classes	Corrupted?	Image size
CIFAR-10-C (Hendrycks & Dietterich, 2018)	19	950,000	10	Yes	32 × 32
CIFAR-100-C (Hendrycks & Dietterich, 2018)	19	950,000	100	Yes	32 × 32
ImageNet-C (Hendrycks & Dietterich, 2018)	19	4,750,000	1000	Yes	224 × 224
CIFAR-10.1 (Recht et al., 2018)	1	2000	10	No	32 × 32
CIFAR-10-Warehouse (Sun et al., 2023)	180	608,691	10	No	224 × 224
OfficeHome (Venkateswara et al., 2017)	4	15,500	65	No	Non-uniform

We list their key statistics

spanning ten distinct classes. CIFAR-10-C retains the class structure of CIFAR-10 but incorporates 19 diverse corruption styles, with severities ranging from levels 1 to 5. This corrupted variant aims to simulate realistic image distortions or corruptions that might arise during processes like image acquisition, storage, or transmission.

CIFAR-100-C has 950,000 colored images with dimensions 32 × 32 pixels, uniformly distributed across 100 unique classes. The CIFAR-100 Corrupted dataset, analogous to CIFAR-10-C, integrates artificial corruptions into the canonical CIFAR-100 images.

ImageNet-C is a corrupted version of ImageNet (Krizhevsky et al., 2012) test set. Produced from ImageNet-1k, ImageNet-C has a similar setup to the CIFAR-10-C and CIFAR-100-C corruption types. For each domain, 5 levels of severity are produced, with 50,000 images per severity from 1,000 classes.

CIFAR-10.1 (Recht et al., 2018) is a real-world test set of CIFAR-10. It contains roughly 2,000 images sampled from the Tiny Image dataset (Yang et al., 2016).

CIFAR-10-Warehouse (Sun et al., 2023) integrates images from both diffusion model (i.e., Stable Diffusion-2-1 (Rombach et al., 2022)) and targeted keyword searches across eight popular search engines. The diffusion model uses the prompt “high quality photo of color class name”, with color chosen from 12 options. The dataset comprises 37 generated and 143 real-world subsets, each containing between 300 and 8,000 images.

OfficeHome is a widely used benchmark in domain adaptation and domain generalization tasks. It has 65 classes within 4 distinct domains: Artistic images (Art), Clip Art, Product images (Product), and Real-World images (RealWorld).

2.3 Evaluation

For a faithful comparison, effectiveness and efficiency are both considered in online test-time adaptation. This survey employs the following evaluation metrics:

mean Error (mErr) is one of the most commonly used metrics to assess model accuracy. It computes the average error rate across all corruption types or domains.

GFLOPs refers to giga floating point operations per second, which quantifies the number of floating-point calculations a model performs in a second. A model with lower GFLOPs is more computationally efficient.

Wall-clock time measures the actual time taken by the model to complete the adaptation process.

GPU memory usage refers to the amount of memory the model uses while running on a GPU. A model with lower GPU memory usage is more applicable to a wider range of devices.

2.4 Relationship with Other Tasks

Offline Test-time Adaptation (TTA) (Liang et al., 2020, 2022; Ding et al., 2022; Yang et al., 2021) aims to adapt a source pre-trained model to the target (i.e., test) set with access to the entire dataset at once. This differs from online test-time adaptation, where test data is given in batches.

Continual TTA Contrary to the classic OTTA setup, where adaptation occurs in discrete steps corresponding to distinct domain shifts, continual TTA (Wang et al., 2022a; Hong et al., 2023; Song et al., 2023; Chakrabarty et al., 2023; Gan et al., 2023) operates under the premise of seamless, continuous adaptation to new data distributions. This process does not require resetting the model with each perceived domain shift. Instead, it emphasizes the importance of a model’s ability to autonomously update and refine its parameters in response to ongoing changes in the data landscape, without explicit indicators of domain boundaries.

Gradual TTA tackles real-world scenarios where domain shifts are gradually introduced through incoming test samples (Marsden et al., 2022; Döbler et al., 2023). An example is the gradual and continuous change in weather conditions. For corruption datasets, existing gradual TTA approaches assume that test data transition from severity level 1 to level 2 and then progress slowly toward the highest level. Both continual and gradual TTA methods ensure online adaptation.

Test-time Training (TTT) introduces an auxiliary task for both training and adaptation (Sun et al., 2020; Gandelsman et al., 2022). During training, the original backbone is modified into a “Y”-shaped structure, with one branch for image clas-

sification and another for an auxiliary task, such as rotation prediction. During adaptation, the auxiliary task continues to be trained in a supervised manner, updating the model parameters. The classification head output serves as the final prediction for each test sample.

Test-time Augmentation (TTAug) applies data augmentations to input data during inference, creating multiple variations of the same test sample, from which predictions are obtained (Shanmugam et al., 2021; Kimura, 2021). The final prediction typically aggregates these predictions through averaging or majority voting. TTAug enhances model performance by providing a range of data views and can be applied to various tasks, including domain adaptation, offline TTA, and OTTA, as it does not require any modification to the model training process.

Domain Generalization (Qiao et al., 2020; Wang et al., 2021b; Xu et al., 2021; Zhou et al., 2023) aims to train models that perform effectively across multiple distinct domains without specific adaptation. It assumes the model learns domain-invariant features applicable across diverse datasets. While OTTA emphasizes dynamic adaptation to specific domains over time, domain generalization seeks to establish domain-agnostic representations.

3 Online Test-time Adaptation

Given the distribution divergence of online data from source training data, OTTA techniques are broadly classified into three categories that hinge on their responses to two primary concerns: managing online data and mitigating performance drops due to distribution shifts. **Optimization-based** methods anchored in designing unsupervised objectives typically lean towards adjusting or enhancing pre-trained models. **Model-based** approaches look to modify or introduce particular layers. On the other hand, **data-based** methods aim to expand data diversity, either to improve model generalization or to harmonize consistency across data views. According to this taxonomy, we sort out existing approaches in Table 9 and review them in detail below.

3.1 Optimization-Based OTTA

Optimization-based OTTA methods consist of three sub-categories: (1) recalibrating statistics in normalization layers, (2) enhancing optimization stability with the mean-teacher model, and (3) designing unsupervised loss functions. A timeline of these methods is illustrated in Fig. 2.

3.1.1 Normalization Calibration

In deep learning, a normalization layer aims to improve the training process and enhance the generalization capac-

ity of neural networks by regulating the statistical properties of activations within a given layer. Batch normalization (BatchNorm) (Ioffe & Szegedy, 2015), the most commonly used normalization layer, stabilizes the training process by utilizing global statistics or a large batch size. By standardizing the mean and variance of activations, BatchNorm reduces the risk of vanishing or exploding gradients during training. Alternatives to BatchNorm include layer normalization (LayerNorm) (Ba et al., 2016), group normalization (GroupNorm) (Wu & He, 2020), and instance normalization (InstanceNorm) (Ulyanov et al., 2016) (Fig. 3). A similar concept to normalization layers is feature whitening, which adjusts features immediately after the activation layer. Both strategies are used in domain adaptation literature (Roy et al., 2019; Carlucci et al., 2017).

Example. Take the most commonly used BatchNorm as an example. Let x_i be the activation for feature channel i in a mini-batch. The BatchNorm layer will first calculate the batch-level mean μ and variance σ^2 by:

$$\mu = \frac{1}{m} \sum_{i=1}^m x_i, \quad \sigma^2 = \frac{1}{m} \sum_{i=1}^m (x_i - \mu)^2, \quad (1)$$

where m is the mini-batch size. Then, the calculated statistics will be applied to standardize the inputs:

$$\hat{x}_i = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}}, \quad y_i = \gamma \hat{x}_i + \beta, \quad (2)$$

where y_i is the final output of the i -th channel from this batch normalization layer, adjusting two learnable affine parameters, γ and β . And ϵ is used to avoid division of 0. For the update, the running mean μ^{run} and variance σ^{run} are computed as a moving average of the mean and variance over all batches seen during training, with a momentum factor α :

$$\mu^{\text{run}} = \alpha \mu + (1 - \alpha) \mu^{\text{run}}, \quad \sigma^{\text{run}} = \alpha \sigma + (1 - \alpha) \sigma^{\text{run}}. \quad (3)$$

Motivation. In domain adaptation, aligning batch normalization statistics helps mitigate performance degradation caused by covariate shifts (Wang et al., 2023b). The hypothesis is that data information is encoded in the weight matrices of each layer, while domain-specific knowledge is conveyed through the statistics of the BatchNorm layer. Therefore, updating the BatchNorm can enhance performance on unseen domains (Li et al., 2017). This idea is broadly applied to online test-time adaptation where we assume for a neural network f trained on a source dataset \mathcal{D}_S with normalization parameters β and γ , updating $\{\gamma, \beta\}$ based on test data x_i^t at each time step t will improve f 's robustness on the test domain.

Building on this assumption, initial investigations in OTTA predominantly focused on fine-tuning by updating only the

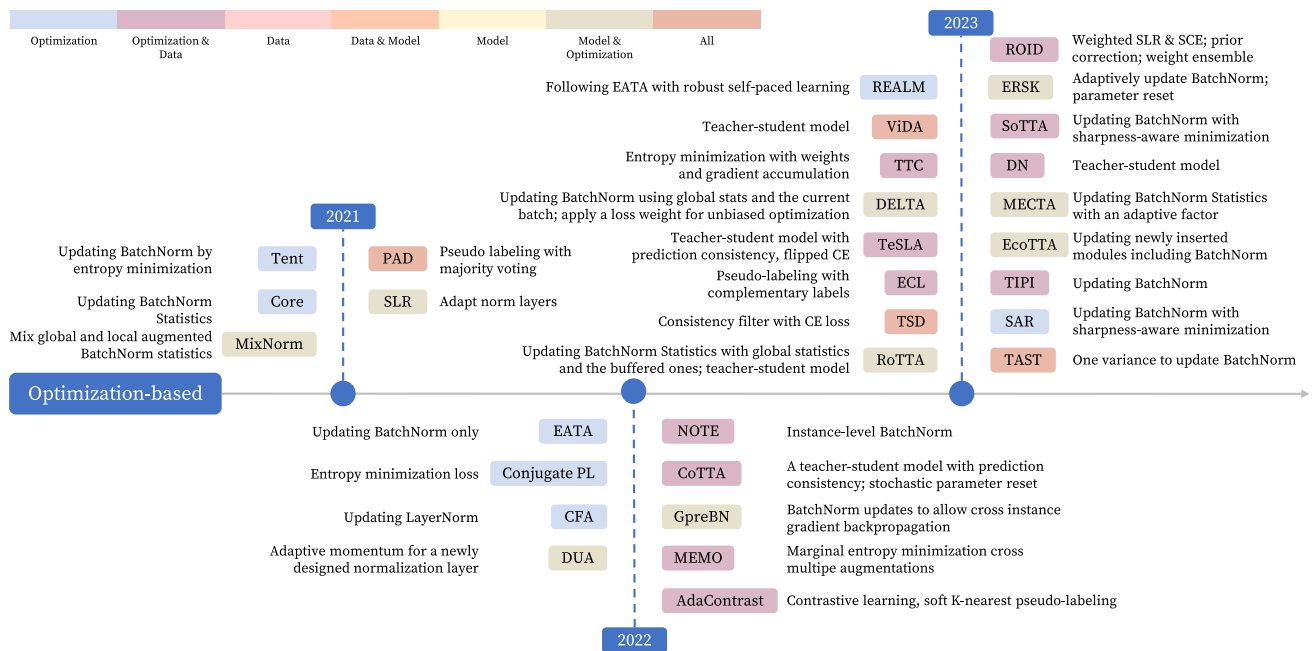


Fig. 2 Timeline of optimization-based OTTA methods

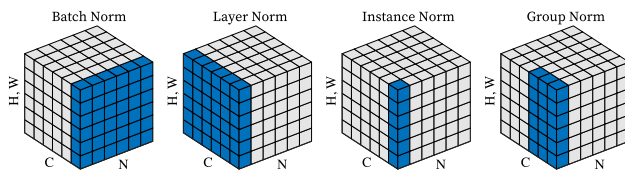


Fig. 3 Visualizing normalization layers (Wu & He, 2020)

normalization layers. This strategy has several popular variations. A common practice adjusts the statistics (μ and σ) and affine parameters (β and γ) in the BatchNorm layer. The choice of normalization techniques, such as LayerNorm (Ba et al., 2016) or GroupNorm (Wu & He, 2020), may depend on the backbone architecture and specific optimization objectives, such as stabilizing entropy minimization (Niu et al., 2023).

Tent (Wang et al., 2021a) and its subsequent works, such as (Jang et al., 2023), are representative approaches within this paradigm. They update the statistics and affine parameters of BatchNorm for each test batch while freezing the remaining parameters. However, the effectiveness of batch-level updates, as seen in Tent, which is updated by minimizing soft entropy, depends on data quality within each batch, introducing potential performance fluctuations. For example, noisy or biased data can significantly affect BatchNorm updates. Methods aimed at stabilization via dataset-level estimates have been proposed to mitigate such performance fluctuations. Gradient preserving batch normalization GpreBN (Yang et al., 2022) allows for cross-instance gradient backpropagation by modifying the BatchNorm

normalization factor.

$$\hat{y}_i = \frac{x_i - \mu_c}{\sigma_c} \bar{\sigma}_c + \bar{\mu}_c - \mu}{\sigma} \gamma + \beta, \quad (4)$$

where $\frac{x_i - \mu_c}{\sigma_c}$ is the standardized input feature \hat{x}_i as in Eq. (2). σ_c and μ_c means stop gradient. GpreBN normalizes \hat{x}_i by arbitrary non-learnable parameters μ and σ . MixNorm (Hu et al., 2021) mixes the statistics of the current batch, produced by augmented sample inputs, with global statistics computed through a moving average. Combining global-level and augmented batch-level statistics bridges the gap between historical context and real-time fluctuations, enhancing performance regardless of batch size. As an alternative, RBN (Yuan et al., 2023a) uses global robust statistics from a memory bank with a fixed momentum for the moving average to ensure high statistic quality. Similarly, Core (You et al., 2021) incorporates a momentum factor in the moving average to fuse source and test set statistics.

Instead of using a fixed momentum factor for the moving average, Mirza et al. (2022) proposed a dynamic approach that determines the momentum based on a decay factor. As model performance deteriorates over time, the decay factor increasingly considers the current batch to avoid biased learning from misled source statistics. ERSK (Niloy et al., 2023) follows a similar idea but determines its momentum by the KL divergence of BatchNorm statistics between the source-pretrained model and the current test batch.

Stabilization via renormalization. Focusing solely on moving averages can undermine the inherent characteristics

of gradient optimization and normalization when updating BatchNorm layers. As noted by Huang et al. (2018), BatchNorm centers and scales activations but does not address their correlation, where decorrelated activations can lead to better feature representation (Schmidhuber, 1992) and generalization (Cogswell et al., 2016). Additionally, batch size significantly influences correlated activations, posing limitations when the batch size is small. The test-time batch renormalization module (TBR) in DELTA (Zhao et al., 2023a) addresses these limitations through a renormalization process. They adjust standardized outputs using two new parameters, r and d . $r = \frac{sg(\hat{\sigma}^{\text{batch}})}{\hat{\sigma}^{\text{ema}}}$ and $d = \frac{sg(\hat{\mu}^{\text{batch}}) - \hat{\mu}^{\text{ema}}}{\hat{\sigma}^{\text{ema}}}$, where $sg(\cdot)$ is stop gradient. Both parameters are computed using batch and global moving statistics, inspired by (Ioffe, 2017), to maintain stable batch statistics. Then \hat{x}_i is further normalized by $\hat{x}_i = \hat{x}_i \cdot r + d$. The above OTTA methods reset the model for each domain, limiting their applicability to scenarios without clear domain boundaries. NOTE (Gong et al., 2022) focuses on continual OTTA under temporal correlation, i.e., distribution changes over time t : $(\mathbf{x}_t, \mathbf{y}_t) \sim P_{\mathcal{T}}(\mathbf{x}, \mathbf{y} | t)$. The authors proposed instance-level BatchNorm to avoid potential instance-wise variations in a domain non-identifiable paradigm.

Stabilization via enlarging batches. To improve the stability of adaptation, another idea is using large batch sizes. In fact, most methods based on batch normalization employ substantial batch sizes such as 200 in (Wang et al., 2021a; Hu et al., 2021). Despite their effectiveness, this practice cannot deal with scenarios where data arrives in smaller quantities due to hardware (e.g., GPU memory) constraints, especially in edge devices.

Alternatives to Batchnorm. To avoid using large-sized batches, viable options include updating GroupNorm (Mumtaz et al., 2021) or LayerNorm, especially in transformer-based tasks (Kojima et al., 2022). In scenarios with limited computational resources, MECTA (Hong et al., 2023) replaces BatchNorm with a customized MECTA norm, reducing memory usage during adaptation. This change mitigates the overhead associated with large batches, extensive channel dimensions, and numerous layers requiring updates. Taking a different tack, EcoTTA (Song et al., 2023) incorporates and exclusively updates meta networks, including BatchNorm layers, effectively reducing computational expenses while maintaining source data discriminability and robust test-time performance. To address performance challenges with smaller batch sizes, TIPI (Nguyen et al., 2023) introduces additional BatchNorm layers alongside existing ones, maintaining two sets of data statistics and leveraging shared affine parameters to enhance consistency across different views of test data.

3.1.2 Mean Teacher Optimization

The mean teacher model, discussed in (Tarvainen & Valpola, 2017), enhances optimization stability in OTTA. This approach initializes both the teacher and student models with a pre-trained source model. For each test sample, weak and strong augmented versions are created and processed by the student and teacher models, respectively. The key lies in using prediction consistency, or consistency regularization, to update the student model. This strategy ensures identical predictions from different data views, reducing model sensitivity to test data changes and improving stability. The teacher model is refined as a moving average of the student across iterations. In OTTA, the mean teacher model and BatchNorm-based methods can be effectively integrated. Incorporating BatchNorm updates into the teacher–student framework can yield more robust results (Sect. 4). Similarly, integrating the mean teacher model with data-driven (Sect. 3.2) or model-driven (Sect. 3.3) methods can further enhance prediction accuracy and stability, marking significant progress in the field.

Model updating strategies. Following the idea of mean-teacher learning, ViDA (Liu et al., 2023b) supervising student output with teacher predictions from augmented input. It introduces high/low-rank adapters to facilitate continual OTTA learning (see Sect. 3.3). Wang et al. (2022a) generally followed the standard consistency learning strategy but introduced a reset method: a fixed number of weights are reset to their source pre-trained states after each iteration to preserve source knowledge and enhance robustness against misinformed updates.

RoTTA (Yuan et al., 2023a) adopts a different approach, focusing on updating only the customized batch normalization layer RBN in the student model, rather than altering all parameters. This strategy leverages consistency regularization and integrates statistics from the test data.

Divergence in augmentations. Drawing inspiration from the prediction consistency strategy in the mean teacher model, Tomar et al. (2023) proposed learning adversarial augmentation to identify the most challenging augmentation policies. These policies drive image feature representations toward uncertain regions near decision boundaries. This method not only achieves clearer decision boundaries but also enhances the separation of class-specific features, significantly improving model robustness to styles of unseen test data.

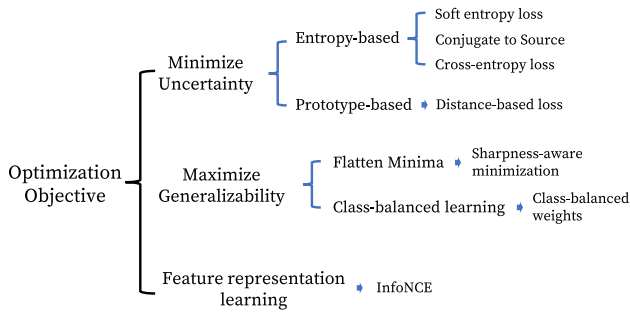


Fig. 4 Common optimization objectives in OTTA

3.1.3 Optimization Objectives

Designing a proper optimization objective is important under the challenges of shifted test data with a limited amount. Commonly seen optimization-based Online Test-time Adaptation (OTTA) are summarized in Fig. 4. Existing literature addresses the optimization problem using three primary strategies:

Optimizing (increasing) confidence. Covariate shifts typically lead to lower model accuracy, which in turn causes the model to express high uncertainty. The latter is often observed. As such, to improve model performance, an intuitive way is to enhance model confidence for the test data.

Entropy-based confidence optimization. This strategy typically aims to minimize the entropy of the softmax output vector:

$$H(\hat{y}) = - \sum_c p(\hat{y}_c) \log p(\hat{y}_c), \quad (5)$$

where \hat{y}_c is the c -th predicted class, $p(\hat{y}_c)$ is its corresponding prediction probability. Intuitively, when the entropy of the prediction decreases, the vector will look sharper, where the confidence, or maximum confidence, increases. In OTTA, minimizing entropy increases the model’s confidence for the current batch without relying on labels, thereby improving accuracy.

There are two main approaches within this strategy: one considers the entire softmax vector, and the other focuses on the maximum entry of the softmax output. `Tent` is a typical method for the former that uses entropy minimization to update the affine parameters in `BatchNorm`. Subsequent studies have expanded upon this strategy. For example, Seto et al. (2023) introduced entropy minimization with self-paced learning, ensuring the learning process progresses adaptively. By integrating general adaptive robust loss (Barron, 2019), the method achieves robustness against large and unstable loss values. `TTPR` (Sivaprasad & Fleuret, 2021) combines entropy minimization with prediction reliability, using a consistency loss across various views of a test image by merging the mean prediction across three augmented versions. Lin et

al. (2023) minimized entropy loss on augmentation-averaged predictions while assigning high weights to low-entropy samples. In `SAR` (Niu et al., 2023), when minimizing entropy, sharpness-aware minimization is used allowing parameters to find “flatter” minimum regions for better model updating stability.

While entropy minimization is widely used, a natural question arises: what makes soft entropy a preferred choice? To reveal the working mechanism of the loss function, `Conj-PL` (Goyal et al., 2022) addresses this by designing a meta-network to parameterize the loss function, observing that the meta-output mirrors the temperature-scaled softmax output. They prove that if cross-entropy loss is used during source pre-training, soft entropy loss is the most appropriate during adaptation.

However, one observation of entropy minimization is the risk of obtaining a degenerate solution where every data point is assigned to the same class. To avoid this, `MuSLA` (Kingetsu et al., 2022) employs mutual information of the sample X and the corresponding prediction \hat{Y} ,

$$I_t(X; \hat{Y}) = H(\hat{p}_0) - \frac{1}{M} \sum_i H(\hat{p}_t^i), \quad (6)$$

where \hat{p}_0 is the prior distribution, M is mini-batch size, i is the index of the sample within the batch. Maximizing $\frac{1}{M} \sum_i H(\hat{p}_t^i)$ could be seen as a regularizer to avoid same-class prediction.

In entropy minimization, gradients can often be dominated by low-confidence predictions. Conversely, cross-entropy loss can be too strict with predicted labels, leading to incorrect updates even with a single wrong prediction. To address this trade-off, Mummadi et al. (2021) proposed the Soft Likelihood Ratio (SLR) loss, which was further employed by Marsden et al. (2024). This approach emphasizes predicted classes while addressing the issue raised by `MuSLA`:

$$\mathcal{L}_{\text{SLR}}(\hat{y}_{ti}) = - \sum_c w_{ti} \hat{y}_{tic} \log \left(\frac{\hat{y}_{tic}}{\sum_{j \neq c} \hat{y}_{tij}} \right), \quad (7)$$

where \hat{y}_{ti} is the softmax probability for the i -th test sample at time step t . w_{ti} is a `ROID`-only weight, combining diversity (the similarity between the recent trend of the model’s prediction and the current model output) and certainty (the negative entropy of the output). If the output confidence for class c is low, the loss calculation is reweighted by the summation over all other predictions $\sum_{j \neq c} \hat{y}_{tij}$ in the denominator, reducing the focus on low-confidence classes.

The cooperation between a teacher and a student is another possible solution to optimize prediction confidence with reliability. Here, the teacher is usually the moving averaged model of interest across iterations. `CoTTA` (Wang et al.,

2022a) uses the Softmax prediction from the teacher to supervise the Softmax predictions from the student under the cross-entropy loss.

To give more precise supervision to the student, ROTTA (Yuan et al., 2023b) adds a twist: the model is updated using samples stored in a memory bank. A reweighting mechanism is introduced to prevent the model from overfitting to “old” samples in the memory bank, prioritizing updates using “new” samples, ensuring a more dynamic and current learning process. See Sect. 3.2.2 for more details about its memory bank strategy.

Supervised by the student output, TeSLA (Tomar et al., 2023) designs its objective as a cross-entropy loss with a regularizer:

$$\mathcal{L}_{pl}(X, \hat{Y}) = -\frac{1}{B} \sum_{i=1}^B \sum_{k=1}^K f_s(\mathbf{x}_i)_k \log((\hat{y}_i)_k) + \sum_{k=1}^K \hat{f}_s(X)_k \log(\hat{f}_s(X))_k, \tag{8}$$

where $\hat{f}_s(X) = \frac{1}{B} \sum_{i=1}^B f_s(\mathbf{x}_i)$ is the marginal class distribution of the student over the batch. \hat{y} is the soft pseudo-label from the teacher model. k is the number of classes. Except for the cross-entropy loss similar to the previous methods, it also maximizes the entropy for the averaged prediction of the student model across the batch to avoid overfitting.

A cross-entropy loss helps minimize model prediction uncertainty but may fail to provide consistent uncertainty scores under different augmentations. This issue is more critical when the teacher–student mechanism is not used. To address this, MEMO (Zhang et al., 2022) computes the average prediction across multiple augmentations for each test sample and then minimizes the entropy of the marginal output distribution over augmentations:

$$\ell(\theta; \mathbf{x}) \triangleq H(\bar{p}_\theta(\cdot | \mathbf{x})) = -\sum_{y \in \mathcal{Y}} \bar{p}_\theta(y | \mathbf{x}) \log \bar{p}_\theta(y | \mathbf{x}), \tag{9}$$

where \bar{p} denotes the averaged Softmax vector. Here, augmentations are randomly generated by AugMix (Hendrycks et al., 2020).

To encourage consistency against smaller perturbations, Marsden et al. (2024) proposed a consistency loss based on symmetric cross-entropy loss (SCE) (Wang et al., 2019):

$$\mathcal{L}_{SCE}(\hat{y}_{ii}; y_{ii}) = -\frac{w'_{ii}}{2} \left(\sum_{c=1}^C \hat{y}_{tic} \log \tilde{y}_{tic} + \sum_{c=1}^C \tilde{y}_{tic} \log \hat{y}_{tic} \right). \tag{10}$$

This loss promotes similar outputs between test images identified as certain and diverse and their augmented views. Here, \hat{y}_{ii} is the softmax probability for the i -th test sample at time step t , \tilde{y}_i is the softmax probability of the augmented view, and w'_{ii} is the weight of the augmented view as in Eq. (7).

Prototype-based optimization. Prototype-based learning (Yang et al., 2018) is a strategy used for unlabeled data by selecting a representative or average for each class and classifying unlabeled data using distance-based metrics. However, its effectiveness can be limited under distribution shifts. To find reliable prototypes, TSD (Wang et al., 2023a) uses a Shannon entropy-based filter to identify class prototypes from target samples with high confidence. A target sample is then used to update the classifier-of-interest if its nearest prototypes are consistent with its class predictions from the same classifier.

Improving generalization ability to unseen target samples. Typically, OTTA uses the same batch of data for model update and evaluation. Here, we would like the model to perform well on upcoming test samples or have good generalization ability. A useful technique is sharpness-aware minimization (SAM) (Foret et al., 2021), where instead of seeking a minima that is “sharp” in its gradients nearby, a “flat” minima region is preferred. Niu et al. (2023) used the following formulation to demonstrate the effectiveness of this strategy.

$$\min_{\Theta} S(\mathbf{x}) E^{SA}(\mathbf{x}; \Theta). \tag{11}$$

Here, $S(\mathbf{x})$ is an entropy-based indicator function that can filter out unreliable predictions based on a predefined threshold. $E^{SA}(\mathbf{x}; \Theta)$ is defined as:

$$E^{SA}(\mathbf{x}; \Theta) \triangleq \max_{\|\epsilon\|_2 \leq \rho} E(\mathbf{x}; \Theta + \epsilon). \tag{12}$$

This term aims to identify a weight perturbation ϵ within a Euclidean ball of radius ρ that maximizes entropy. It quantifies sharpness by measuring the maximal change in entropy between Θ and $\Theta + \rho$. As such, Eq. (12) jointly minimizes entropy and its sharpness. Gong et al. (2023) used same idea for their optimization.

Another difficulty affecting OTTA generalization is class imbalance in a batch: a limited number of data for model updates often cannot reflect the true class frequency. To address this, Dynamic Online Re-weighting (DOT) in DELTA (Zhao et al., 2023a) uses a momentum-updated class-frequency vector. This vector is initialized with equal weights for each class and is updated at every inference step based on the pseudo-label of the current sample and model weight. For a target sample, a significant weight (or frequency) for a particular class prompts DOT to diminish its contribution during subsequent adaptation learning. This prevents biased opti-

mization towards frequent classes, thereby improving model generalization.

Feature representation learning. Since no annotations are assumed for the test data, contrastive learning (van den Oord et al., 2018) can be naturally applied to test-time adaptation tasks. In a self-supervised manner, contrastive learning aims to learn feature representations where positive pairs (a data sample and its augmentations) are close, and negative pairs (different data samples) are pushed away from each other. However, this typically requires multiple epochs of updates, which is incompatible with the online adaptation setting. To suit online learning, AdaContrast (Chen et al., 2022) uses target pseudo-labels to disregard potential same-class negative samples rather than treating all other data samples as negative.

3.1.4 Pseudo-Labeling

Pseudo-labeling is a useful technique in domain adaptation and semi-supervised learning. It assigns labels to samples with high confidence, and these pseudo-labeled samples are then used for training.

In OTTA, where adaptation is confined to the current batch of test data, **batch-level** pseudo-labeling is often employed. For example, MuSLA (Kingetsu et al., 2022) implements pseudo-labeling as a post-optimization step following BatchNorm updates, refining the classifier using pseudo-labels of the current batch to enhance model accuracy.

Furthermore, the teacher–student framework, as seen in models like CoTTA (Wang et al., 2022a), RoTTA (Yuan et al., 2023b), and ViDA (Liu et al., 2023b), also adopts the pseudo-labeling strategy. Here, the teacher outputs are used as soft pseudo-labels, which preventing the model to be overfitted to incorrect predictions.

Reliable pseudo-labels are essential but challenging in OTTA. Continuous data streams limit opportunities for reviewing the prediction. Besides, covariate shifts between the source and test sets degrades pseudo-label reliability.

To address these challenges, TAST (Jang et al., 2023) uses a prototype-based pseudo-labeling strategy. Class centroids are obtained from a support set, initially derived from the source pre-trained classifier’s weights and refined using normalized test data features. To avoid performance degradation brought by unreliable pseudo labels, it calculates centroids only using the nearby support examples and then uses the temperature-scaled output to obtain the pseudo labels. Alternatively, AdaContrast (Chen et al., 2022) uses soft K nearest neighbors voting (Mitchell & Schaefer, 2001) in the feature space to produce reliable pseudo-labels. Wu et al. (2021) suggest using multiple augmentations and majority voting to achieve consistent and trustworthy pseudo-labels.

Complementary pseudo-labeling (PL). One-hot pseudo-labels often result in substantial information loss, especially under domain shifts. To address this, ECL (Han et al., 2023) considers both maximum-probability predictions and predictions that fall below a certain confidence threshold (i.e., complementary labels). The intuition is that if the model is less confident about a prediction, this prediction should be penalized more heavily. This helps prevent the model from making aggressive updates based on incorrect but high-confidence predictions, offering a more stable approach similar to soft pseudo-label updates.

3.1.5 Other Approaches

Deviating from the conventional path of adapting source pre-trained models, Laplacian Adjusted Maximum likelihood Estimation (LAME) (Boudiaf et al., 2022) focuses on refining the model output. This is achieved by discouraging the refined output from deviating from the pre-trained model while encouraging label smoothness according to the manifold smoothness assumption. The final refined prediction is obtained when the energy gap for each refinement step of a batch is small.

To prevent loss of generalization and catastrophic forgetting, weight ensembling offers a solution. RoID (Marsden et al., 2024) continuously ensembles the weights of the initial source model and the weights of the current model at time step t using a moving average, allowing for partial retention of source knowledge. This approach is similar to the parameter reset strategy in CoTTA, commonly used in continual adaptation tasks. Additionally, to address temporal correlation and class imbalance during adaptation, RoID introduces prior correction. The intuition is that if the class distribution within a batch tends to be uniform, strong smoothing is applied to ensure no class is favored. This is indicated by the sample mean over the current softmax prediction \hat{p}_t . Thus, the smoothing scheme is defined as:

$$\bar{p}_t = \frac{\hat{p}_t + \gamma}{1 + \gamma N_c}, \quad (13)$$

where N_c denotes the number of classes and γ is an adaptive smoothing factor.

3.1.6 Summary

Optimization-based methods are the most common in online test-time adaptation, focusing on consistency, stability, and robustness. However, they rely on the availability of sufficient target data to reflect the global test data distribution. The next section will explore data-based methods, which partially address the challenge of limited target data in OTTA.

3.2 Data-Based OTTA

With a limited number of samples in the test batch, encountering unexpected distribution changes is common. We recognize that **data** might be key to bridging the gap between the source and test data.

This section delves into data-centric strategies in OTTA. We explore methods for diversifying data in each batch (Sect. 3.2.1) and preserving high-quality information on a global scale (Sect. 3.2.2). These strategies aim to enhance model generalizability and adapt the model's discriminative capacity to the current data batch (Fig. 5).

3.2.1 Data Augmentation

Data augmentation is important in domain adaptation (Wang & Deng, 2018) and domain generalization (Zhou et al., 2023), as it mimics real-world variations to improve model transferability and generalizability. It is particularly useful for test-time adaptation.

Predefined augmentations. Common data augmentation methods like cropping, blurring, and flipping are effectively incorporated into various OTTA methodologies. An example of this integration is TTC (Lin et al., 2023), which updates the model using averaged predictions from multiple augmentations. Mean teacher models such as RoTTA (Yuan et al., 2023b), CoTTA (Wang et al., 2022a), and ViDA (Liu et al., 2023b), applies predefined augmentations to teacher/student input, and maintains prediction consistency across different augmented views.

To ensure consistent and reliable predictions,ROID (Marsden et al., 2024) uses augmentation for prediction consistency by employing symmetric cross-entropy (SCE). PAD (Wu et al., 2021) employs multiple augmentations of a single test sample for majority voting, believing that if most augmented views yield the same prediction, it is likely correct. TTPR (Sivaprasad & Fleuret, 2021) adopts KL divergence to achieve consistent predictions by aligning the average prediction across three augmented views with the prediction for each view. Another approach, MEMO, uses AugMix (Hendrycks et al., 2020) for test images. For a test data point, a range (usually 32 or 64) of augmentations from the AugMix pool \mathcal{A} is generated to make consistent predictions.

Contextual augmentations. Previously, OTTA methods often predetermined augmentation policies. Given that test distributions can undergo substantial variations in continuously evolving environments, fixed augmentation policies may not be suitable for every test sample. In CoTTA (Wang et al., 2022a), rather than augmenting every test sample by a uniform strategy, augmentations are judiciously applied only when domain differences (i.e., low prediction confidence) are detected, mitigating the risk of misleading the model.

Adversarial augmentation. Traditional augmentation methods always provide limited data views without fully representing the domain differences. TeSLA (Tomar et al., 2023) addresses this by leveraging adversarial data augmentation to identify the most effective augmentation strategy. Instead of using a fixed augmentation set, it creates a policy search space \mathcal{O} as the augmentation pool and assigns a magnitude parameter $m \in [0, 1]$ for each augmentation. A sub-policy ρ consists of augmentations and their corresponding magnitudes. To optimize the policy, the teacher model is adapted using an entropy maximization loss with severity regularization, encouraging prediction variations while avoiding augmentations that are too strong and deviate too far from the original image.

3.2.2 Memory Bank

Going beyond augmentation strategies that could diversify the data batch, a memory bank is a powerful tool to preserve valuable data information for future memory replay. Setting up a memory bank involves two key considerations: (1) Determining **which data to store**, identifying samples valuable for replay during adaptation. (2) **Managing the memory bank**, including adding new instances and removing old ones.

Memory bank strategies are generally time-uniform and class-balanced. Many methods integrate both for maximum effectiveness. To address temporally correlated distributions and class imbalance, NOTE (Gong et al., 2022) introduces Prediction-Balanced Reservoir Sampling (PBRs), saving sample-prediction pairs. PBRs combines time-uniform and prediction-uniform sampling. The time-uniform approach, reservoir sampling (RS), aims for uniform data over a temporal stream. For a sample x predicted as class k , a value p is randomly sampled from a uniform distribution $[0, 1]$. If p is smaller than the proportion of class k in the memory bank, a random sample from the same class is replaced with x . The prediction-uniform strategy (PB) prioritizes predicted labels to maintain majority class balance, replacing a random instance from the majority class with a new sample. PBRs ensures a balanced distribution across time and class, enhancing the model's adaptability.

A similar strategy is employed in SoTTA (Gong et al., 2023) to facilitate class-balanced learning. Each high-confidence sample-prediction pair is stored when the memory bank has available space. If the bank is full, the method opts to replace a sample either from one of the majority classes or from its class if it belongs to the majority. This ensures a more equitable class distribution and strengthens the learning process against class imbalances. Another work, RoTTA (Yuan et al., 2023b), offers a category-balanced sampling with timeliness and uncertainty (CSTU) module, dealing with the batch-level shifted label distribution. CSTU

3.3.1 Module Addition

Input transformation. In an effort to counteract domain shift, Mummadi et al. (2021) introduced to optimize an input transformation module d , along with the BatchNorm layers as discussed in Sect. 3.1.3. This module is built on the top of the source model f , i.e., $g = f \circ d$. Specifically, $d(x)$ is defined as:

$$d(x) = \gamma \cdot [\tau x + (1 - \tau)r_{\psi}(x)] + \beta, \quad (15)$$

where γ and β are channel-wise affine parameters. The component r_{ψ} denotes a network designed to have the same input and output shape, featuring 3×3 convolutions, group normalization, and ReLU activations. The parameter τ facilitates a convex combination of the unchanged and transformed input $r_{\psi}(x)$.

Adaptation modules. To stabilize predictions during model updates, TAST (Jang et al., 2023) integrates 20 adaptation modules to the source pre-trained model. Based on BatchEnsemble (Wen et al., 2020), these modules are appended to the top of the pre-trained feature extractor. The adaptation modules are updated multiple times independently by merging their averaged results with the corresponding pseudo-labels for a batch of data.

For continual adaptation, promptly detecting and adapting to changes in data distribution is essential to address catastrophic forgetting and error accumulation. To achieve this, ViDA (Liu et al., 2023b) employs low/high-rank feature cooperation. Low-rank features retain general knowledge, while high-rank features capture distribution changes. The authors introduced two adapter modules parallel to the linear layers (if the backbone model is ViT) to obtain these features.

Since distribution changes in continual OTTA are unpredictable, strategically combining low/high-rank information is crucial. The authors used MC dropout (Gal & Ghahramani, 2016) to assess model prediction uncertainty about input x . This uncertainty adjusts the weight given to each feature. If the model is uncertain about a sample, the weight of domain-specific knowledge (high-rank feature) is increased; conversely, the weight of domain-shared knowledge (low-rank feature) is increased. This helps the model dynamically recognize distribution changes while preserving its decision-making capabilities.

3.3.2 Module Substitution

Module substitution typically refers to swapping an existing module in a model with a new one. The commonly used techniques are about:

Classifiers. Cosine-distance-based classifier (Chen et al., 2009) offer flexibility and interpretability by leveraging similarity to representative examples for decision-making. TAST

(Jang et al., 2023) formulates predictions by assessing the cosine distance between the sample feature and the support set. TSD (Wang et al., 2023a) employs a similar classifier, comparing features of the current sample against its K-nearest neighbors from a memory bank. PAD (Wu et al., 2021) uses a cosine classifier for predicting augmented test samples in its majority voting process. T3A (Iwasawa and Matsuo, 2021) relies on the dot product between templates in the support set and input data representations for classification.

In the context of updating BatchNorm statistics, any alteration to BatchNorm that extends beyond the standard updating approach falls under this category. This includes techniques such as MECTA norm (Hong et al., 2023), MixNorm (Hu et al., 2021), RBN (Yuan et al., 2023a), and GPreBN (Yang et al., 2022), etc. To maintain focus and avoid redundancy, these specific methods and their intricate details will not be extensively covered again in this section.

3.3.3 Adaptation Techniques Using Prompts

Prompt, widely discussed in vision language models like CLIP (Radford et al., 2021), involves various design and learning strategies, especially when set as a learnable parameter. We consolidate all prompting related adaptation methods here for clarity.¹

“Decorate the Newcomers” (DN) (Gan et al., 2023) uses prompts as supplementary information atop image input. It employs a student-teacher framework with a frozen source pre-trained model to capture both domain-specific and domain-agnostic prompts. For domain-specific knowledge, it optimizes cross-entropy loss between the teacher and student models’ outputs. Additionally, DN introduces a parameter insensitivity loss to reduce the impact of parameters prone to domain shifts. This ensures updated parameters retain domain-agnostic knowledge while learning new, domain-specific information.

DePT (Gao et al., 2022) innovatively segments the transformer into stages, adding learnable prompts at the initial layer of each stage alongside image and CLS tokens. During adaptation, a mean-teacher model updates the learnable prompts and the classifier in the student model. The student model updates based on the cross-entropy loss between pseudo labels and outputs from strongly augmented data. Here, pseudo labels are generated using the averaged predictions of the top-k nearest neighbors from the student’s weakly augmented output within a memory bank. To mitigate errors from incorrect pseudo labels, DePT employs entropy loss between predictions from strongly augmented views of both student and teacher models. Additionally, it minimizes the mean squared error between the combined prompts of the

¹ Refer to the timeline tables for component details.

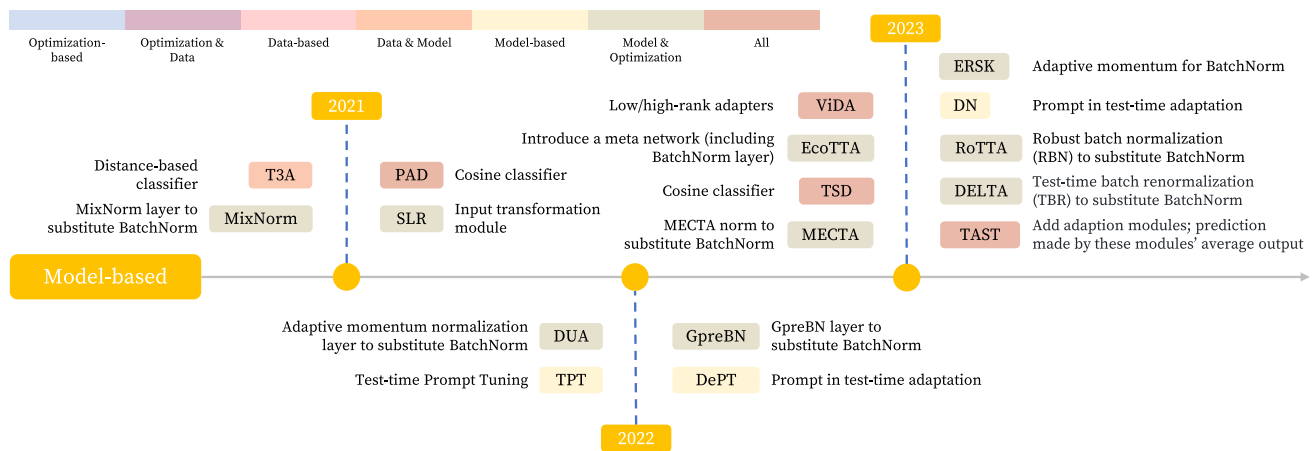


Fig. 6 Timeline of Model-based OTTA methods

student and teacher models at the transformer's output layer. To ensure diversity and prevent trivial solutions, DePT maximizes the cosine distance among the student's combined prompts.

To adapt test data in VLMs, Test Time Prompt Tuning merges an solution. Unlike conventional methods that fine-tune with a fixed number of labeled test samples per class, it learns the prompt by adjusting only the context of the model's input, thus preserving the model's generalization power. TPT (Shu et al., 2022) generates N randomly augmented views of each test image and updates the learnable prompt parameter by minimizing the entropy of the averaged prediction probability distribution. Additionally, a confidence selection strategy filters out outputs with high entropy to avoid noisy updates from unconfident samples. By updating the learnable prompt parameter, the model can adapt more easily to new, unseen domains (Fig. 7).

3.3.4 Summary

Model-based OTTA methods have shown effectiveness but are less prevalent than other groups, mainly due to their reliance on specific backbone architectures. For example, layer substitution primarily based on BatchNorm is inapplicable to ViT-based architectures.

A critical feature of this category is its effective integration with prompting strategies. This combination allows for fewer but more impactful model updates, leading to greater performance improvements. Such efficiency makes model-based OTTA methods especially suitable for complex scenarios.

4 Empirical Studies

Existing OTTA methods predominantly use WideResNet (Zagoruyko & Komodakis, 2016) or ResNet (He et al., 2016)

for experiments, overlooking the evolution of backbones in recent years. In this study, we explore the possibility of decoupling OTTA methods from their conventional CNN backbones. We specifically focus on adapting these methods to the Vision Transformer model (Dosovitskiy et al., 2021). Our work presents strategies for adapting methods, initially developed for CNNs, to work effectively with ViT architectures, thereby examining their flexibility under backbone changes.

Baselines. We evaluate eight OTTA methods, using a standardized testing protocol for fair comparison. We use six diverse datasets: three corrupted datasets (CIFAR-10-C, CIFAR-100-C, and ImageNet-C), two real-world shifted datasets (CIFAR-10.1 and OfficeHome), and one comprehensive dataset (CIFAR-10-Warehouse). CIFAR-10-Warehouse plays a pivotal role in our evaluation, featuring a broad array of subsets, including real-world variations from different search engines with different colors and images created through the diffusion model. Specifically, we used the Google split in CIFAR-10-Warehouse to assess the OTTA model's ability to handle color shifts and mixed object styles. Additionally, we evaluated the Diffusion split to test the effectiveness of OTTA on artificially generated image samples, which have gained popularity in recent years.

4.1 Implementation Details

Optimization details. We use PyTorch for implementation on an NVIDIA RTX A6000. The foundational backbone for all approaches is ViT-base-patch16-224 (Dosovitskiy et al., 2021).² For CIFAR-10-C, CIFAR-10.1, and CIFAR-10-Warehouse as target domains, we trained the source model on CIFAR-10 for 8000 iterations, including a warm-up phase of 1600 iterations. The training used a batch size of 64 and the

² <https://github.com/huggingface/pytorch-image-models>.

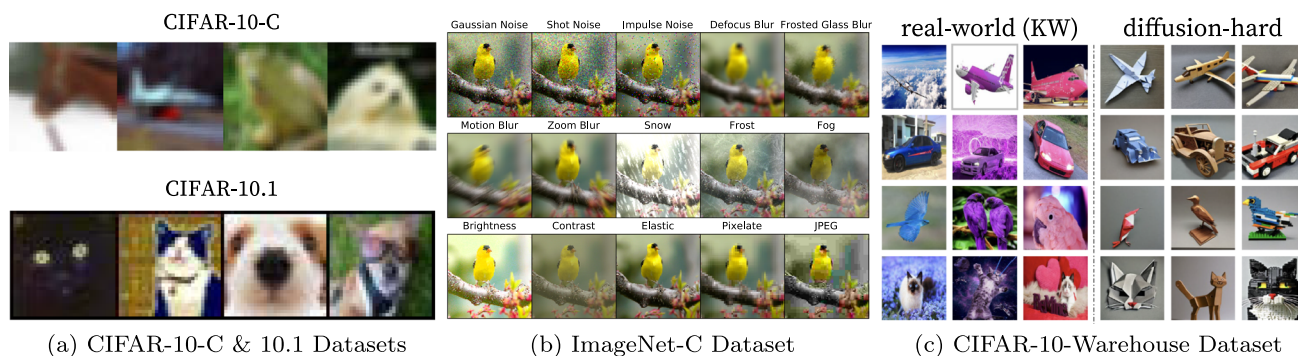


Fig. 7 The exemplars of the adopted datasets. The datasets include color variations, synthetic data, and types of corruption

stochastic gradient descent (SGD) algorithm with a learning rate of $3e-2$. We applied the same configuration for training the source model on CIFAR-100, extending the training to 16,000 iterations with a warm-up period of 4000 iterations. The source model on the ImageNet-1k dataset was obtained from the Timm repository.³ For the OfficeHome dataset, we trained the source model on the clipart domain for 3,500 iterations with 200 iterations of warmup. Basic data augmentation techniques, including random resizing and cropping, were applied across all methods. The common optimization setup employed the Adam optimizer with a momentum term β of 0.9 and a learning rate of $1e-3$. Resizing and cropping techniques were used as default preprocessing steps for all datasets, followed by input normalization (0.5, 0.5, 0.5) to mitigate potential performance fluctuations from external factors beyond the algorithm's core operations.

Component substitution. We develop a series of strategies to adapt the OTTA methods to vision transformers:

- *Switch to LayerNorm:* Due to the absence of the BatchNorm layer in ViT, all BatchNorm-based strategies in the original implementations would be automatically moved onto LayerNorm.
- *Disregard BatchNorm mixup:* Removing statistic mixup strategy originally designed for BatchNorm-based methods because LayerNorm is designed to normalize each data point independently.
- *Sample embedding changes:* For methods that rely on feature representations, an effective solution is to use class embedding as the image feature.
- *Pruning incompatible components:* Any elements incompatible with vision transformer are removed.

Baselines: We select eight methods that represent the eight OTTA categories mentioned in this paper, respectively. They include:

1. **Tent**: [optimization] A fundamental OTTA method rooted in BatchNorm updates for both statistics and affine parameters with entropy minimization. To reproduce it on ViTs, we replace its BatchNorm updates with a LayerNorm updates.
2. **CoTTA** [optimization , data] uses the mean-teacher model, where the teacher model is updated by the moving average of the student. During adaptation, soft entropy is optimized and combined with selective augmentation. For each iteration, it also applies parameter reset (i.e., partially reset the model parameter to the source pre-trained version). While it requires updating the entire student network (i.e., CoTTA-ALL), we further assess CoTTA-LN that only update LayerNorm on its student model. We also examine its parameter reset strategy, resulting in another two variants: updating LayerNorm without parameter reset (CoTTA*-LN), and full network updating without parameter reset (CoTTA*-ALL).
3. **SAR** [optimization] follows Tent while using sharpness-aware minimization (SAM) (Foret et al., 2021) for flat minima.
4. **Conjugate-PL**: [optimization] As the source model is optimized by the cross-entropy loss, this method is Conj-CE. It is similar to Tent but allows the model to interact with the data twice for each iteration: once for updating LayerNorm and another for prediction.
5. **MEMO**: [optimization , data] For each test sample, MEMO applies various data augmentations and adjusts the model parameters to minimize the entropy across the model marginal output distributions from these augmentations. To ensure consistency and avoid unexpected performance fluctuations, we omit all data normalization processes from its set of augmentations. At the same time, we evaluate the performance of MEMO on two versions, LayerNorm update and full update, resulting in MEMO-LN and MEMO-ALL correspondingly.
6. **RoTTA** [optimization , data , model] uses a memory bank to store class-balanced data, considering uncertainty and the “age” of each saved data sample. It

³ vit_base_patch16_224.orig_in21k_ft_in1k.

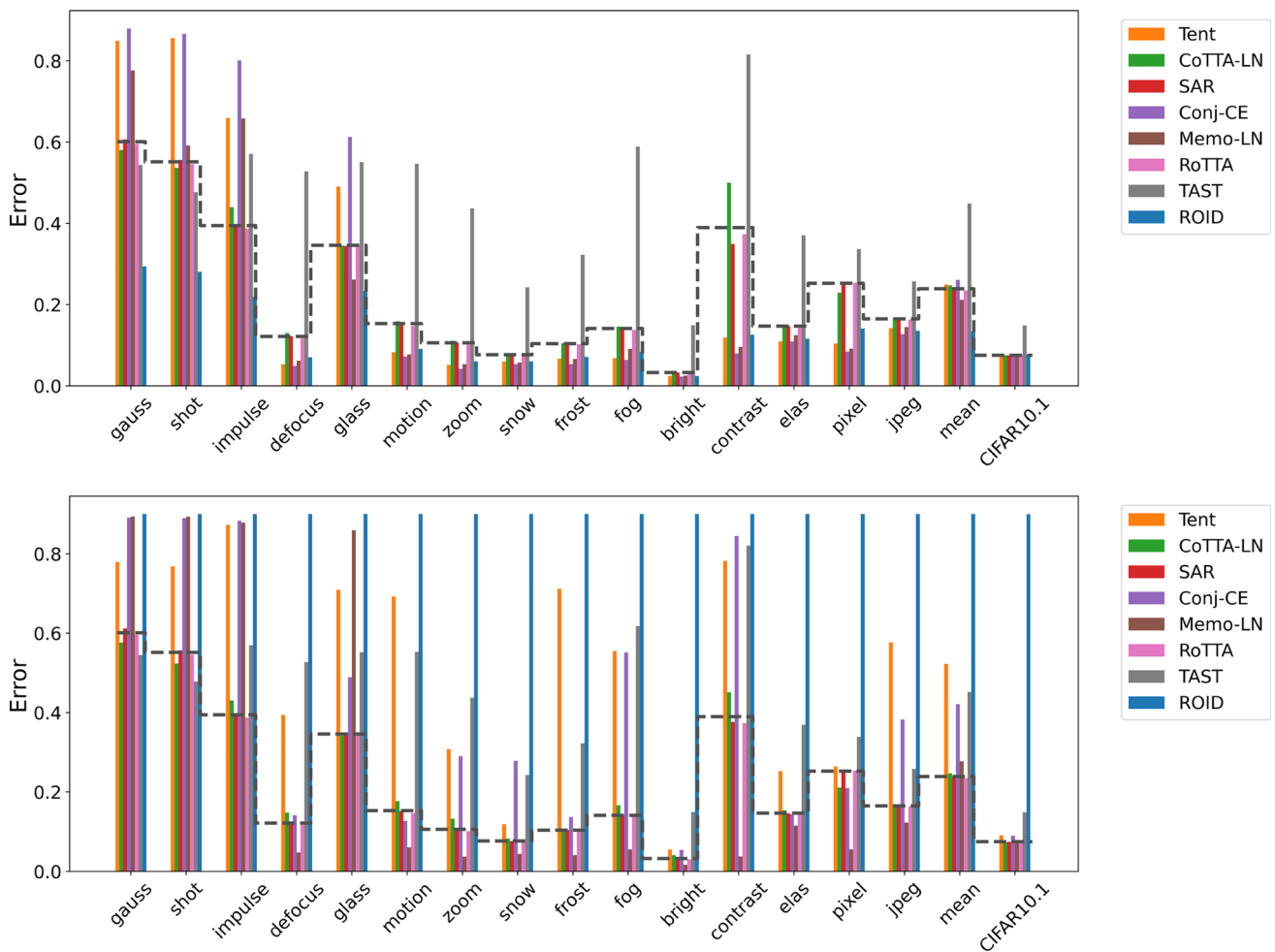


Fig. 8 Comparison of the OTTA performance on the CIFAR-10-C (severity level 5) and CIFAR-10.1 datasets with ViT-base-patch16-224. The top and bottom plots show the experiments conducted with batch

size 16 and 1 based on the LayerNorm updating strategy, respectively. Source-only (i.e., direct inference) performance is shown with the dotted line

also introduces a time-aware reweighting strategy to enhance adaptation stability. For evaluation, given that BatchNorm is not compatible with ViTs, we exclude the RBN module.

7. TAST [optimization , data , model] integrates multiple adaptation modules into the source pre-trained model. Based on BatchEnsemble (Wen et al., 2020), these modules are appended to the top of the pre-trained feature extractor. The adaptation modules are then updated multiple times independently by merging their averaged results with the corresponding pseudo-labels for a batch of data. To accommodate the ViT architecture (especially ViT-base), we use the class embedding as the feature representation.
8. ROID⁴ [optimization , data] is a method for not only typical OTTA but also universal TTA, capable of han-

dling temporal correlation and domain non-stationarity. It incorporates weighted SLR loss with the Symmetric cross-entropy loss, alongside weight ensemble and prior correction mechanisms to ensure efficacy in complex scenarios.

Despite the broad range of available OTTA methods, we believe that a detailed examination of this carefully selected subset will yield valuable insights. Our empirical study is designed to explore the following key questions.

4.2 Does OTTA Still Work with ViT?

To assess the transferability and adaptability of the selected OTTA methods, we compare them against the source-only baseline (i.e., direct inference) on vision transformers. For consistency and controlled variable comparison, each bar

⁴ For fair comparison, we do not use gradient accumulation.

chart included in our analysis is plotted based on the `LayerNorm` updating strategy.

4.2.1 On CIFAR-10-C and CIFAR-10.1 Benchmarks

We evaluate the CIFAR-10-C and CIFAR-10.1 datasets with batch sizes 1 and 16 and show results in Fig. 8. We discuss our observations in three aspects.

Corruption types. Across both batch sizes, most methods exhibit high error in response to noise-induced corruptions, as illustrated in Figs. 8 and 13. In contrast, these methods tend to perform better than noise-induced corruptions when dealing with structured corruptions, such as snow, zoom, or brightness. This pattern suggests that various corruption types reflect different degrees of divergence from the corruption domain to the source dataset. Especially, adapting to noise corruption poses a significant challenge for confidence optimization-based methods (such as `Tent`, `MEMO-LN`, and `Conj-CE`), regardless of the batch size. This difficulty can be linked to the substantial domain gap and the unpredictable nature of noise patterns discussed earlier. Although these strategies aim to increase the model's confidence, they are not equipped to directly correct erroneous predictions. It is worth noting that `ROID` surpasses all other baselines on noise-based corruptions. This may be attributed to the certainty- and diversity-weighted SCE loss, which is designed to address noise issues while avoiding trivial solutions. CIFAR 10.1 stands out as an exception, as it is not associated with any specific corruption. Instead, it represents real-world data that share the same label set as CIFAR 10. Compared to other corruption domains - except the brightness domain - CIFAR 10.1 exhibits a smaller domain gap, as evidenced by the lower error rate achieved through direct inference. In this context, most methods yield results comparable to direct inference. Notably, `Conj-CE` excels with a batch size of 16, while `CoTTA*-LN`, `CoTTA-LN`, and `SAR` stand out when the batch size is reduced to 1.

Batch sizes. Batch size is found to aid pure optimization-based methods like `Tent`, `Conj-CE`, and `MEMO-LN` in outperforming direct inference (i.e., source only). Similarly, batch size significantly influences the performance of `ROID`. It achieves the highest performance, with a significant margin over others, in terms of mean error on CIFAR-10-C when the batch size is 16. Nevertheless, it makes almost entirely erroneous predictions under a batch size of 1. As detailed in Sect. 4.4, our analysis leads to the conclusion that larger batch sizes tend to stabilize loss optimization, which is beneficial for adaptation processes in these methods. This phenomenon is also evident in CIFAR-10.1, where entropy-based methods such as `Tent` and `Conj-CE` face challenges at smaller batch sizes.

Methods that incorporate prediction reliability can effectively navigate the limitations associated with smaller batch sizes.

4.2.2 On CIFAR-100-C Benchmark

The performance on the CIFAR-100-C dataset exhibits a trend similar to that observed in the CIFAR-10-C dataset, as shown in Table 2.

Number of classes. As CIFAR-100-C shares the same corruption setup as CIFAR-10-C, it is important to understand the differences between the CIFAR-10-C and CIFAR-100-C datasets. CIFAR-10-C consists of only 10 classes, allowing all classes to be represented within a single batch if its size is 16. In contrast, CIFAR-100-C, with its 100 classes, introduces a distinct challenge for online streaming adaptation. This scenario is particularly problematic for methods like `Tent`, which rely on updating the `LayerNorm` parameters within the current batch for subsequent use. While no method surpasses the source-only performance, `Tent` experiences a 1.33% accuracy reduction on CIFAR-10-C and a more pronounced 3.94% reduction on CIFAR-100-C.

Batch sizes. A noteworthy finding from the CIFAR-100-C experiments is that `ROID`, with a mean error rate of 41.30% at any experimented batch size, consistently outperforms direct inference (which has a mean error rate of 41.88%). This achievement is likely due to `ROID`'s ability to maintain label diversity within its memory bank, emphasizing the importance of preserving a wide and varied information spectrum to tackle batch-sensitive and complex adaptation tasks effectively. This trend is similarly observed in Figs. 13 and 9, further reinforcing the robustness of `ROID`. Moreover, although `SAR` and `TAST` underperform compared to direct inference, they demonstrate stability across various batch sizes. Considering `ROID`, these methods represent primary approaches to handling different batch sizes: (1) stable optimization, as flat minima are more resilient to gradient fluctuations, and (2) information preservation, providing additional insights for each batch and reducing model sensitivity to batch sizes. Nevertheless, `ROID` and `CoTTA-ALL` exhibit significant performance variations concerning batch sizes.

4.2.3 On ImageNet-C Benchmark

When comparing performance across CIFAR-10-C, CIFAR-100-C, and ImageNet-C, it becomes apparent that ImageNet-C (Fig. 9) exhibits notably poorer performance overall. This could be attributed to the larger number of classes in the ImageNet-C dataset, as a similar trend is observed when comparing CIFAR-10-C (Fig. 8) with CIFAR-100-C Table 2.

Corruption types. Varying domain gaps emerge across datasets when employing the same mechanism to generate

Table 2 Classification error rate (%) for the standard CIFAR-100 → CIFAR-100-C online test-time adaptation task

Method	Glass	Fog	Defoc	Impul	Contr	Gauss	Elastic	Zoom	Pixel	Frost	Snow	JPEG	Motion	Shot	Britt	Mean	
SO	62.39	34.23	32.01	76.94	60.32	76.05	28.86	25.25	39.73	24.20	19.79	35.62	32.56	72.86	7.32	41.88	
<i>Batch size = 16</i>																	
Tent	73.69	35.40	16.33	97.06	85.94	91.95	24.40	15.37	26.78	23.54	17.44	33.36	22.09	93.71	5.53	44.17	
CoTTA-LN	62.68	41.84	38.77	87.01	74.04	80.53	34.62	30.46	36.46	26.47	22.08	39.70	39.89	77.25	7.58	46.63	
CoTTA-ALL	68.75	79.77	45.56	98.58	97.81	94.80	65.83	53.62	38.73	37.63	24.49	67.40	86.00	94.18	13.19	64.42	
CoTTA*-LN	63.12	42.25	39.10	87.11	73.93	80.57	34.94	30.71	36.75	26.30	22.42	39.79	40.22	77.47	7.60	46.82	
CoTTA*-ALL	91.58	91.85	83.09	98.61	97.91	97.39	91.57	90.03	86.26	83.19	79.11	90.57	93.47	97.00	72.50	89.61	
SAR	63.09	31.41	30.72	82.17	64.76	79.59	27.39	24.36	34.76	24.10	19.61	34.01	29.26	77.55	7.32	42.01	
Conj-CE	86.24	37.12	13.06	97.87	94.18	97.49	22.73	11.07	18.64	33.83	14.13	31.64	16.06	97.03	4.86	45.06	
MEMO-LN	83.03	68.62	17.49	96.33	30.39	94.71	25.19	16.58	28.71	19.93	17.13	34.18	20.07	92.96	5.38	43.38	
RoTTA	62.22	33.34	30.92	76.12	58.78	75.42	28.51	24.41	39.61	24.10	19.55	35.49	31.59	72.36	7.14	41.30	
TAST	70.09	65.01	58.18	79.09	84.37	76.04	50.82	52.14	49.78	45.77	41.75	48.93	58.68	72.76	20.65	58.27	
ROID	54.30	24.63	19.05	57.87	33.67	60.84	24.96	17.25	28.49	19.64	17.74	32.55	22.15	57.20	5.88	31.75	
<i>Batch size = 1</i>																	
Tent	94.66	63.74	20.36	98.33	96.17	97.72	42.98	50.11	23.65	65.41	29.05	69.70	83.09	96.28	8.43	62.65	
CoTTA-LN	61.94	42.16	39.05	82.87	69.13	79.16	33.02	31.30	35.29	24.84	21.57	37.19	39.32	75.95	9.32	45.47	
CoTTA-ALL	96.95	93.78	95.34	98.90	97.82	98.86	92.47	93.33	96.36	95.66	98.42	95.79	96.81	98.68	98.04	96.48	
CoTTA*-LN	66.87	46.04	42.94	85.11	71.35	82.93	39.81	35.38	38.61	30.09	27.39	45.03	44.50	79.71	11.59	49.82	
CoTTA*-ALL	98.62	98.36	98.26	98.87	98.60	98.95	98.11	97.64	98.63	98.25	98.62	98.38	98.04	98.90	98.49	98.45	
SAR	62.39	33.95	31.29	80.15	60.48	77.82	28.63	24.83	38.60	24.12	19.87	35.51	31.73	74.89	7.29	42.10	
Conj-CE	97.07	73.10	18.54	98.67	97.21	98.25	43.50	10.90	31.95	46.68	14.56	82.12	16.89	98.18	4.75	55.49	
MEMO-LN	98.08	86.99	10.42	98.69	96.71	98.64	24.14	9.35	12.94	91.24	61.54	64.93	13.12	98.63	4.45	57.99	
RoTTA	62.22	33.34	30.92	76.11	58.78	75.41	28.51	24.40	39.63	24.10	19.55	35.49	31.59	72.36	7.14	41.30	
TAST	70.52	65.33	58.90	79.33	84.50	76.18	51.21	52.46	49.93	46.23	41.92	49.08	59.31	72.95	20.79	58.58	
ROID	98.99	98.99	98.99	99.00	98.99	99.00	98.99	98.99	98.99	99.00	99.00	98.99	98.99	99.00	99.00	98.99	

Bold values indicate the lowest error rate

Results are evaluated on ViT-base-patch16-224 at the severity level 5. Here, SO denotes source only

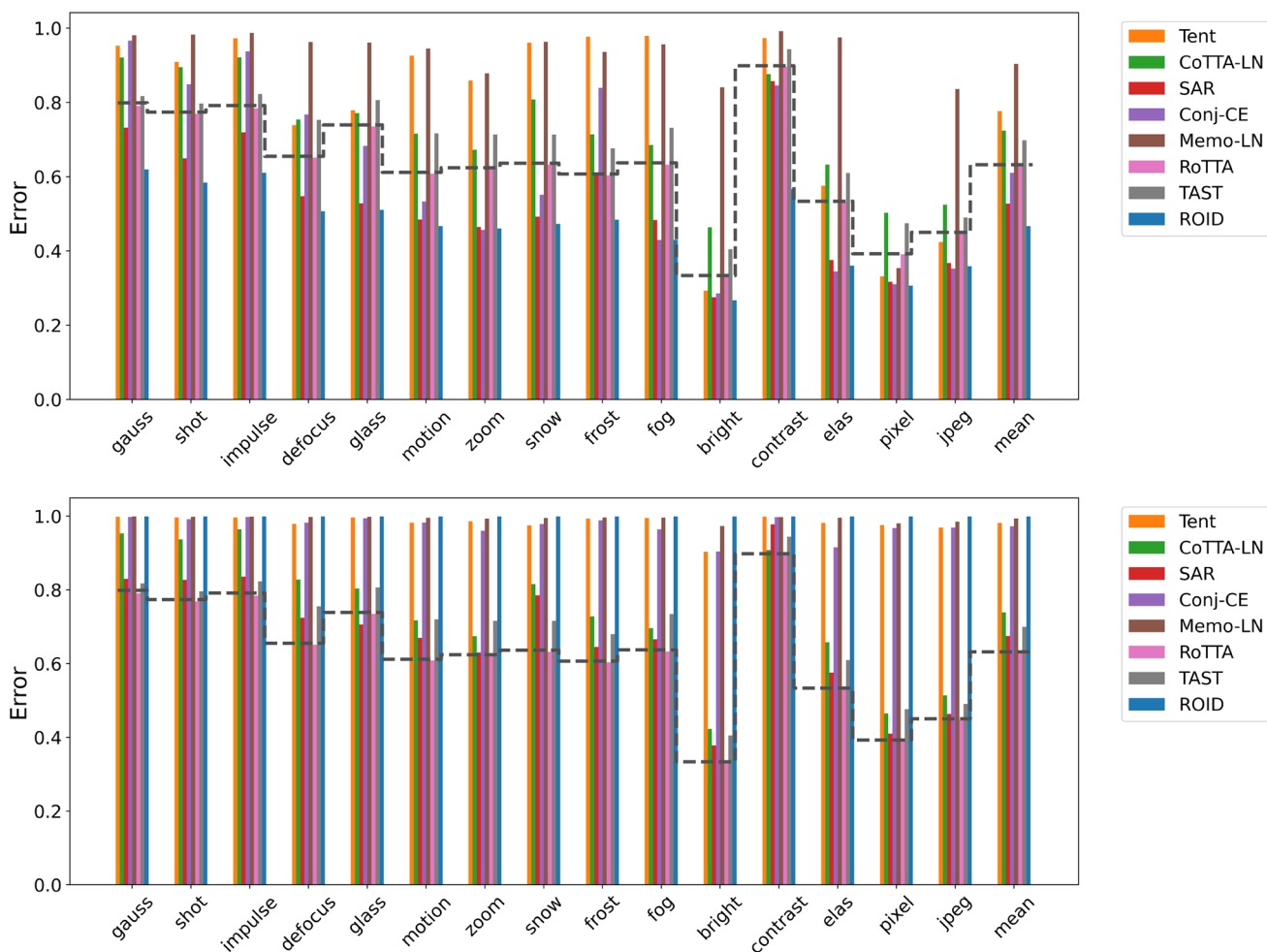


Fig. 9 Comparison of the OTTA performance on the ImageNet-C at the severity level 5. The upper and bottom plots show the experiments conducted with batch size 16 and 1 based on the LayerNorm updating strategy, respectively. Source-only performance is shown with the dotted line

corrupted data. Nevertheless, overall consistency remains, likely influenced by the training process of the source model. Additionally, corruptions representing natural domain gaps (e.g., brightness, zoom, snow) consistently appear simpler than noise-based corruptions (e.g., Gaussian noise, shot noise). Surprisingly, contrast exhibits a notably high error rate for both direct inference and most methods, indicating a substantial domain gap compared to other corruption types. In this case,ROID appears to excel in addressing it. In contrast, brightness appears to have the smallest domain gap with the source data.

Batch sizes. When batch size is 1, optimization-based methods exhibit error rates exceeding 95% across all domains. This also applies toROID, which mainly relies on model optimization. An exception to this trend is SAR, which produces a similar trend as in CIFAR-100-C. However, it produces a larger performance gap in ImageNet-C than CIFAR-100-C compared with direct inference. With different numbers of classes, this further indicates that datasets with increased

complexity and a higher degree of challenge exhibit greater sensitivity to batch-size alterations.

Adaptation strategy. As depicted in Fig. 9, when batch size is 16, SAR, Conj-CE, RoTTA, andROID outperform the source-only model in terms of mean error. In contrast, Tent, MEMO-LN, and CoTTA-LN demonstrate significantly poor results. The error rate of each domain exhibits a similar trend. Notably, Conj-CE, which conducts an additional inference of each batch for final prediction compared with Tent, markedly surpasses Tent across most domains and in mean error. This suggests a significant inter-batch shift in ImageNet-C, such as class differences.

4.2.4 On CIFAR-10-Warehouse Benchmark

In our study, the CIFAR-10-Warehouse dataset (Sun et al., 2023) emerges as an indispensable resource for assessing OTTA methods, perfectly aligning with the CIFAR-10 label set to facilitate comprehensive comparisons across a spec-

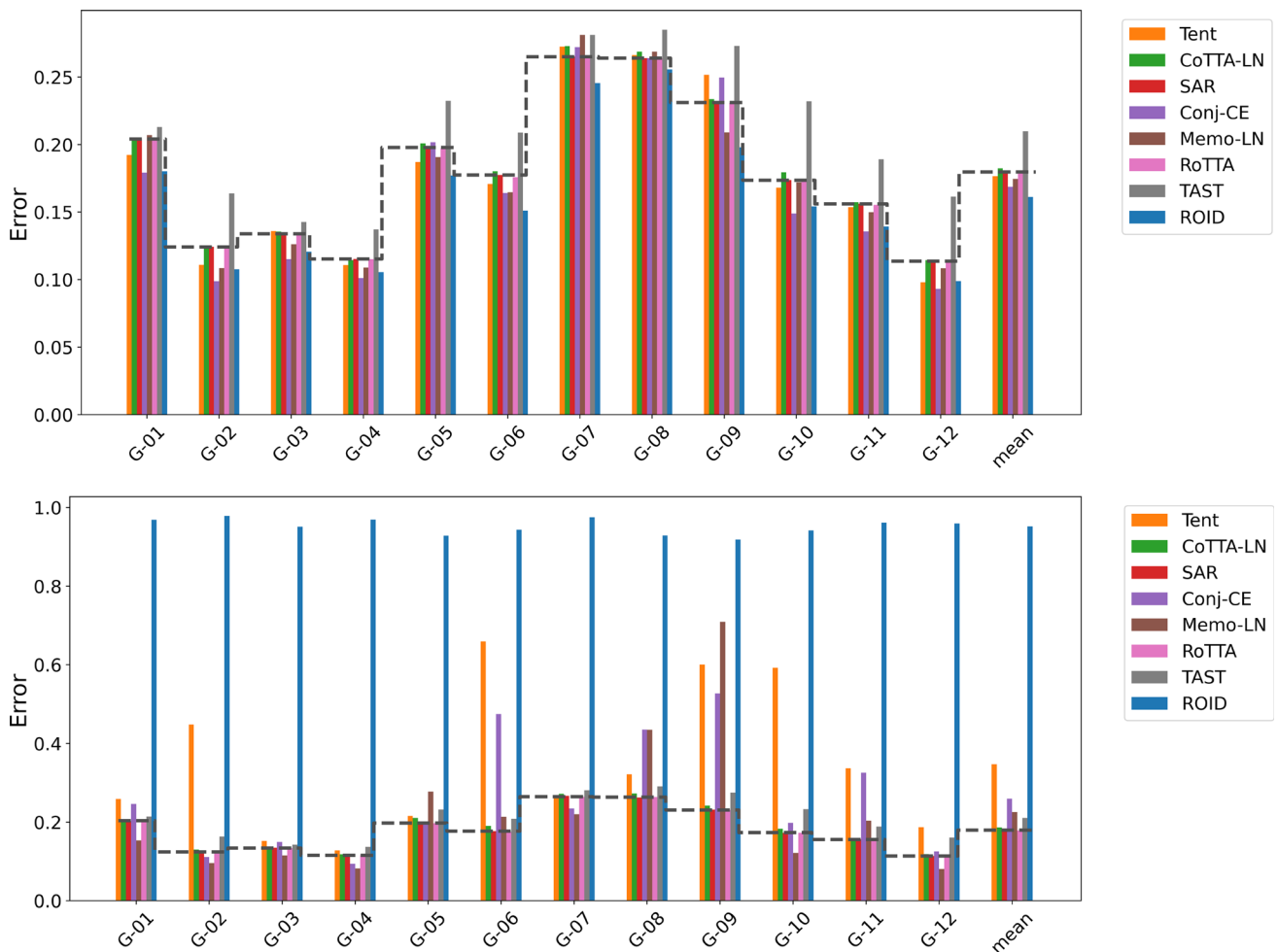


Fig. 10 Comparison of the OTTA performance on the **Google split** of CIFAR-10-Warehouse. The upper and bottom plots show the experiments conducted with batch size 16 and 1 based on the `LayerNorm` updating strategy. Source-only performance is shown with the dotted line

trum of distribution shifts. Specifically, we examine two dataset domains—real-world shifts and diffusion synthesis shifts - to assess the resilience and adaptability of OTTA methods.

Google split. Diverging from previous CIFAR-10 variances, which primarily integrated artificially induced corruptions (i.e., CIFAR-10-C) or relied on sample differences (i.e., CIFAR 10.1), the Google split of the CIFAR-10-Warehouse offers an unparalleled perspective. Sourced from Google search queries, this segment includes 12 subdomains that exhibit a range of color variations within the CIFAR-10 labels. It constructs an essential benchmark to measure contemporary OTTA methods' proficiency against real-world distribution shifts.

Cross-dataset comparison: real-world versus corruptions. Comparing the empirical results for both the Google split (Fig. 10) and CIFAR-10-C (Fig. 8), a notable difference is that the real-world shift presented in CIFAR-10-Warehouse exhibits a smaller domain gap globally, as indicated by the

performances of direct inference. Concerning subdomains, the real-world shifts also demonstrate lower variance compared to corruption domains. This suggests that dealing with attacks or noise in a streaming manner is the most challenging adaptation task, as these two datasets are based on the same source pre-trained model.

Batch sizes. Regarding the differences in batch sizes depicted in Fig. 10, when the batch size is 16, five OTTA methods based on `LayerNorm` updating either match or surpass the performance of direct inference. This observation suggests that the `LayerNorm` updating strategy is generally effective in handling real-world shifts. However, when the batch size is reduced to 1, methods that solely rely on entropy minimization, such as `Tent` and `Conj-CE`, experience performance degradation across most domains. This decline may be attributed to the instability of optimization for single-sample batches. Additionally, the fact that `ROID` performs similarly to corruption-based datasets indicates its sensitivity to changes in batch size.

Adaptation strategy. RoTTA, TAST and SAR demonstrate exceptional stability regardless of batch sizes, which is also shown in CIFAR-100-C. Furthermore, ROID exhibits a smaller margin for surpassing performance compared to corruption-based datasets, indicating its robustness against data noise.

Diffusion split. Furthermore, we incorporate the Diffusion split into our analysis as shown in Table 3. Created using stable diffusion (Rombach et al., 2022), this domain introduces a novel examination of synthetic samples. Considering the increasing prevalence of diffusion-generated imagery, this evaluation offers unique insights into the capability of OTTA methods to adapt to the rising tide of generated images.

Cross-dataset comparison: corruption, real-world, or diffusion? In terms of mean error, the diffusion split has the smallest domain gap among CIFAR-10-based datasets when comparing Fig. 8 with Table 3. This is even true when compared to the CIFAR 10.1 dataset, suggesting that OTTA methods based on ViT can generally handle diffusion-based images.

Adaptation strategy. The outcomes in Table 3 indicate that, on average, each OTTA method performs equally well or better than the baseline direct inference in terms of mean error. However, some methods still exhibit poor performance in certain domains. For instance, while TAST delivers satisfactory results on DM-01, it fails to perform on DM-02, DM-08, DM-09, and DM-12 domains. Similarly, Tent, Conj-CE, and MEMO-LN methods fall short on DM-05.

In contrast, RoTTA and ROID demonstrate their capability to surpass direct inference outcomes for every domain. Similarly, SAR enables the model to reach a region in the optimization landscape that is less sensitive to data variations, resulting in stable predictions.

4.2.5 On OfficeHome Benchmark

OfficeHome (Venkateswara et al., 2017) has four domains. We use the clipart domain as the source dataset and apply OTTA methods to the remaining domains: Art, Product, and RealWorld.

Subdomains. The Art domain is the most difficult to adapt to from clipart based on the given pre-trained source model. However, even with minimal batch size, SAR, RoTTA, TAST, and ROID show competitive results against the baseline in the Art domain. Conversely, in the Product domain, reducing batch sizes significantly impacts the performance of several methods. The RealWorld domain presents the most significant challenge for adaptation, with few methods surpassing direct inference across different batch sizes.

Batch sizes. SAR, RoTTA, and TAST consistently outperform the source-only baseline irrespective of batch size, indicating stable performance. On the other hand, Tent, Conj-CE, and ROID show a decline in performance at a batch size of

Table 3 Classification error rate (%) for the CIFAR-10 → CIFAR-10-Warehouse online test-time adaptation task

	DM-01	DM-02	DM-03	DM-04	DM-05	DM-06	DM-07	DM-08	DM-09	DM-10	DM-11	DM-12	Mean
Diffusion Source-Only	1.84	1.36	3.05	2.32	5.16	3.80	3.18	2.82	1.18	4.41	2.43	2.23	2.82
<i>Batch size = 16</i>													
Tent	1.61	1.05	2.55	1.36	7.02	2.41	1.91	2.57	1.09	4.32	1.68	2.30	2.49
CoTTA-LN	1.84	1.36	3.05	2.34	5.16	3.86	3.21	2.82	1.16	4.39	2.48	2.21	2.82
SAR	1.84	1.36	3.05	2.32	5.16	3.80	3.18	2.82	1.18	4.41	2.43	2.23	2.82
Conj-CE	1.59	0.86	1.77	0.86	5.96	1.73	1.77	2.07	0.64	2.52	1.21	1.98	1.91
MEMO-LN	1.50	1.02	2.14	0.86	5.39	2.05	2.46	2.36	0.82	3.77	1.84	1.86	2.17
RoTTA	1.80	1.36	3.05	2.23	5.07	3.71	3.18	2.80	1.14	4.34	2.43	2.14	2.77
TAST	1.48	1.77	2.68	1.68	4.98	2.41	3.09	2.96	1.73	4.14	3.11	2.84	2.74
ROID	1.64	0.93	2.32	1.27	4.00	2.16	2.43	2.14	0.89	3.55	1.73	1.77	2.07

Bold values indicate the lowest error rate
Results are evaluated on ViT-base-patch16-224 with the Diffusion split

Table 4 Classification error rate (%) for the OfficeHome online test-time adaptation task using a source model trained on Clipart with ViT-base-patch16-224

Method	Art	Product	RealWorld	Mean
SO	39.56	34.42	29.54	34.51
<i>Batch size = 16</i>				
Tent	39.10	33.63	28.69	33.81
CoTTA-ALL	72.23	86.30	79.55	79.36
SAR	39.02	33.30	28.85	33.72
Conj-CE	43.47	32.28	27.01	34.26
MEMO-LN	43.14	34.74	30.25	36.04
RoTTA	39.56	34.42	29.47	34.48
TAST	39.47	32.30	30.02	33.93
ROID	37.74	33.03	28.23	33.00
<i>Batch size = 1</i>				
Tent	39.60	67.97	47.56	51.71
CoTTA-ALL	96.25	97.63	97.41	97.10
SAR	39.27	34.65	29.56	34.49
Conj-CE	68.40	69.59	30.30	56.09
MEMO-LN	48.17	94.05	91.51	93.39
RoTTA	39.56	34.44	29.47	34.49
TAST	39.43	32.28	30.11	33.94
ROID	96.95	98.22	98.03	97.73

Bold values indicate the lowest error rate
SO denotes source-only

1, suggesting a dependency on larger batch sizes for optimal results.

Adaptation strategy. It is observed that CoTTA-ALL and MEMO-LN do not perform well across all domains. This suggests that they heavily rely on augmentations that are not effective enough to bridge the domain gap in OfficeHome. As a result, it is crucial to further examine current augmentation strategies to address domain discrepancies more efficiently.

4.2.6 Conclusion

Based on our extensive experiments, most OTTA methods exhibit similar behavioral patterns across various datasets. This consistency indicates the potential of contemporary OTTA techniques in effectively managing diverse domain shifts. Of particular note are two methods, RoTTA and SAR, which highlight the significance of optimization insensitivity and information preservation, respectively. Additionally, the effectiveness of ROID is demonstrated when the batch size is reasonable, showcasing its capability.

4.3 Is OTTA Efficient?

To assess the efficacy of OTTA algorithms, especially within the constraints of hardware limitations, we employ three

primary evaluation metrics: wall clock time, GPU memory usage, and GFLOPs.

As depicted in Fig. 11, lower values across these metrics are indicative of superior performance. Our analysis reveals that within the context of the ImageNet-C dataset, MEMO-LN exhibits suboptimal performance, accompanied by elevated computational demands, which are deemed disadvantageous.

Meanwhile, RoTTA demonstrates commendable results, characterized by reduced GFLOPs and processing time; however, its memory bank demands greater GPU memory. This requirement may necessitate further hardware provisions or memory optimizations for deployment in practical applications. Note that in our analyses, CoTTA-ALL is excluded to maintain the integrity and informativeness of the comparative showcase. This decision is made since CoTTA-ALL has unexpectedly high GFLOPs and wall clock time cost, along with high error rates, which made the comparative landscape appear skewed.

Conversely, SAR maintains low mean error while ensuring computational efficiency, especially compared with Tent. Similarly, Conj-CE attains significant error reduction with slightly increased resource consumption, performing inference for each batch directly after every iteration. ROID, additionally, achieves a balance between effectiveness and efficiency, as evidenced by its notably low time consumption and GFLOPs, alongside moderate GPU memory usage, while still delivering a superbly low error rate.

4.4 Is OTTA Sensitive to Hyperparameter Selection?

In this section, we investigate whether OTTA methods will be impacted by various hyperparameters. We conduct our experimental study on ImageNet-C with batch size 16 using ViT-base-patch16-224 to assess optimizers, learning rates, and schedulers.

4.4.1 Batch Size Matters, But Only to an Extent

Figure 13 examines the impact of different batch sizes on Tent in the CIFAR-10-C dataset. It reveals that performance significantly varies with batch sizes from 1 to 16 across most corruptions. However, this variability diminishes with larger batch sizes (16 to 128), indicating a reduced influence of LayerNorm updating on batch size, in contrast to traditional BatchNorm settings. This pattern is consistent across other datasets, as depicted in Fig. 12. Nevertheless, **batch size remains crucial for stabilizing the optimization process.** For example, a batch size of 16 outperforms a batch size of 1 in confidence optimization methods in the Google split of the CIFAR-10-Warehouse dataset.

However, **Larger batch sizes are essential for complex datasets** like CIFAR-100-C and ImageNet-C, where direct inference struggles. Besides, Fig. 13 suggests that **increasing**

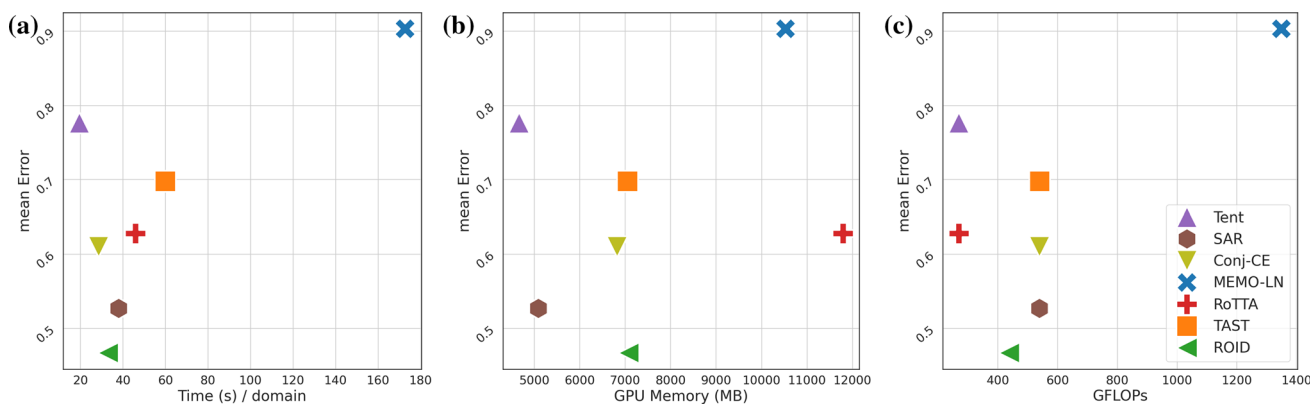


Fig. 11 Mean error versus **a** average wall-clock time per domain, **b** GPU memory usage, and **c** GFLOPs. All experiments are conducted on a severity level of 5 on the ImageNet-C dataset

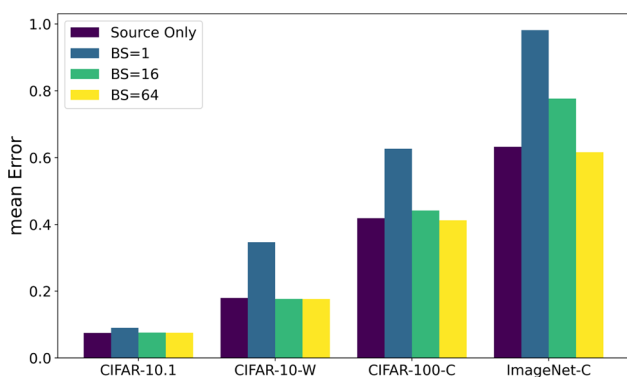


Fig. 12 Impact of varying batch sizes for Tent on CIFAR-10.1, CIFAR-10-Warehouse **Google split**, CIFAR-100-C, and ImageNet-C

batch sizes does not fully address challenging corruptions, such as Gaussian and Shot noise. This indicates the necessity for more advanced adaptation strategies beyond mere batch size adjustments in complex learning scenarios. Additionally, it is worth noting that gradient accumulation

(Marsden et al., 2024) could be treated as a solution to allow single-sample adaptation.

4.4.2 Optimization Layer Matters!

To assess the critical role of LayerNorm, we compare LayerNorm update with the full model update, as summarized in Table 5. This ablation study primarily focused on CoTTA and MEMO, evaluating the impact of optimizing LayerNorm alone. A notable observation is, for all methods, LayerNorm update plays an important role in gaining high performance, underscoring its effectiveness in boosting model performance by avoiding significant forgetting of the source knowledge. This is also proven by Tent, SAR, and Conj-CE, as shown in Fig. 15 in the appendix.

4.4.3 Optimizer Matter?

We evaluate Adam and SGD across eight methods with a batch size of 16 on the ImageNet-C dataset. Our findings,

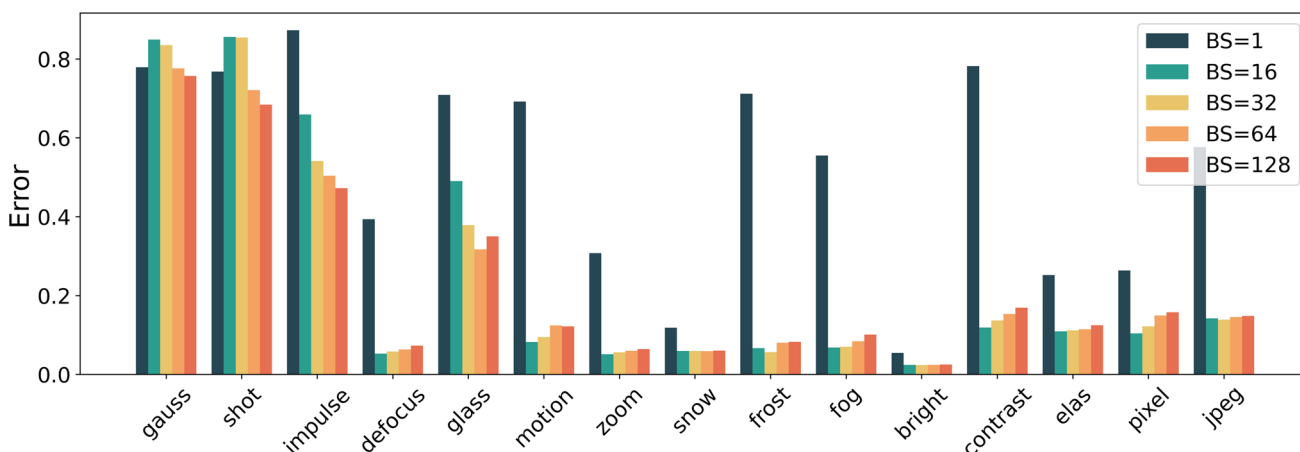


Fig. 13 Impact of batch size on CIFAR-10-C severity level 5. The base model is Tent optimized on LayerNorm

Table 5 Comparisons of model update strategies across five benchmarks: LayerNorm (LN) versus full model (ALL)

Method	CIFAR-10-C		CIFAR-10.1		CIFAR-100-C		ImageNet-C		CIFAR-Warehouse	
	LN	ALL	LN	ALL	LN	ALL	LN	ALL	LN	ALL
CoTTA	0.2471	0.5247	0.0750	0.1155	0.4663	0.6442	0.7237	0.8516	0.1823	0.2804
CoTTA*	0.2468	0.7708	0.0750	0.1640	0.4682	0.8961	0.7229	0.9054	0.1820	0.5463
MEMO	0.2116	0.8999	0.0780	0.9000	0.4338	0.9900	0.9032	0.9995	0.1746	0.8955
		Δ Err (%)		Δ Err (%)		Δ Err (%)		Δ Err (%)		Δ Err (%)
		-52.91		-35.06		-27.62		-15.02%		-34.99
		-67.98		-54.27		-47.75		-20.16%		-66.68
		-76.49		-91.33		-56.18		-96.35%		-80.50

Bold values indicate the lowest error rate

* indicates the variant of CoTTA with no parameter reset mechanism

detailed in Table 6, reveal several key observations. Firstly, methods that adapt to test data under soft supervision, such as soft entropy, appear more susceptible to changes in the optimizer. For instance, switching to SGD resulted in a significant mean error reduction of 9.84% for Tent. Similarly, when updated by soft entropy, SGD enabled CoTTA-ALL to achieve a mean error difference of over 10%. This pattern was also observed in Conj-CE and MEMO-LN. This disparity likely stems from the superior generalization capabilities of SGD, as discussed by Wilson et al. (2017). The notable exception within this group is SAR, which utilizes Sharpness-Aware Minimization (SAM) to find stable minima. This approach may diminish the impact of the choice between optimizers.

Another distinct category includes RoTTA and TAST. These methods, significantly reliant on their memory banks or the averaging of predictions across multiple adaptation modules, demonstrate reduced sensitivity to the choice of optimization strategy. RoID instead exhibits high sensitivity in terms of the optimizer changes. Here, Adam shows its stability due to its adaptive learning rate and moment estimation for noisy and non-stationary objectives.

4.4.4 Impact of Learning Rate

To further explore the impact of hyperparameters, we examine the learning rate within the range [0.0001, 0.0005, 0.001, 0.005, 0.01] on the ImageNet-C dataset, as detailed in Fig. 14. Results indicate that lower learning rates benefit online adaptation.

The empirical results shown in Fig. 14 confirm a general idea: a high learning rate can cause quick overfitting of the current batch in online adaptation. Lower rates enhance model stability, enabling smoother pattern adaptation. Conversely, a large learning rate could disrupt the finely learned knowledge from the source model thus causing performance degradation.

However, this is not universally applicable. Methods such as RoTTA and TAST consistently demonstrate stable performance across varying learning rates. By incorporating more information for prediction, they alleviate the impact of batch-specific label variations, bolstering model stability. Moreover, CoTTA-ALL, SAR, and RoID attain their optimal performance at specific learning rate values, excluding the lowest one, underscoring the continued relevance of studying learning rates to optimize adaptation performance.

4.4.5 Impact of Scheduler

We examine the effectiveness of three different learning rate schedulers to ascertain their influence on performance. Compared with the default learning rate setup of 0.001 (Def) and

Table 6 Comparison between Adam and SGiD (highlighted in italic) on ImageNet-C using ViT-base-patch16-224 at the severity level 5 and batch size 16, SO denotes source-only

Method	Glass	Fog	Defoc	Impul	Contr	Gauss	Elastic	Zoom	Pixel	Frost	Snow	JPEG	Motion	Shot	Brit	Mean
Source-Only	73.92	63.76	65.54	79.18	89.84	79.88	53.38	62.42	39.26	60.70	63.66	45.06	61.20	77.38	33.38	63.24
Tent	77.86	97.88	73.88	97.24	97.28	95.28	57.54	85.92	33.12	97.64	96.06	42.40	92.60	90.88	29.30	77.66
Tent	74.86	95.34	65.28	87.18	95.04	87.58	38.44	50.74	31.82	92.78	88.74	36.92	56.98	87.98	27.64	67.82
CoTTA-ALL	73.72	94.82	92.80	99.14	92.94	99.06	78.06	84.14	79.40	88.92	96.70	54.48	86.58	98.10	58.48	85.16
CoTTA-ALL	76.80	69.22	75.64	91.60	87.22	92.20	62.90	67.06	50.60	71.36	81.18	51.46	70.78	89.58	45.78	72.23
SAR	52.78	48.34	54.74	71.92	85.74	73.18	37.54	46.48	31.70	61.14	49.24	36.74	48.48	64.92	27.52	52.70
SAR	53.34	45.68	52.52	82.40	98.30	72.24	37.72	47.82	31.42	49.90	47.82	37.10	48.86	77.32	27.52	54.00
Conj-CE	68.30	42.96	76.74	93.70	84.52	96.60	34.46	45.64	31.00	83.90	55.12	35.28	53.32	84.88	28.60	61.00
Conj-CE	53.38	44.32	51.96	57.62	61.60	60.62	37.02	47.24	31.42	50.14	48.22	36.34	47.66	57.96	28.38	47.59
MEMO-LN	96.10	95.62	96.26	98.68	99.18	98.08	97.48	87.80	35.40	93.58	96.30	83.56	94.50	98.26	84.06	90.32
MEMO-LN	93.16	88.46	96.10	97.60	98.66	96.72	95.86	85.20	33.10	94.42	94.70	57.30	89.48	98.66	33.36	83.52
RoTTA	73.50	63.24	65.14	78.40	89.58	79.02	53.10	61.94	39.12	60.34	63.24	44.62	60.80	76.88	33.12	62.80
RoTTA	73.68	64.00	65.56	76.72	89.86	77.72	53.46	62.28	39.46	60.98	63.92	44.26	60.92	75.18	33.42	62.76
TAST	80.60	73.16	75.26	82.22	94.32	81.68	60.98	71.34	47.48	67.62	71.34	48.98	71.68	79.66	40.46	69.79
TAST	80.54	73.04	75.32	82.18	94.30	81.74	60.90	71.28	47.46	67.52	71.42	48.88	71.80	79.66	40.22	69.75
ROID	51.06	43.02	50.68	61.02	56.70	61.94	36.04	46.02	30.66	48.44	47.30	35.88	46.70	58.38	26.72	46.70
ROID	98.30	98.08	98.64	99.48	99.80	99.50	98.04	99.18	78.34	98.96	99.54	98.82	99.06	98.54	43.00	93.82

Bold values indicate the lowest error rate

Fig. 14 Impact of varying learning rates on ImageNet-C severity level 5

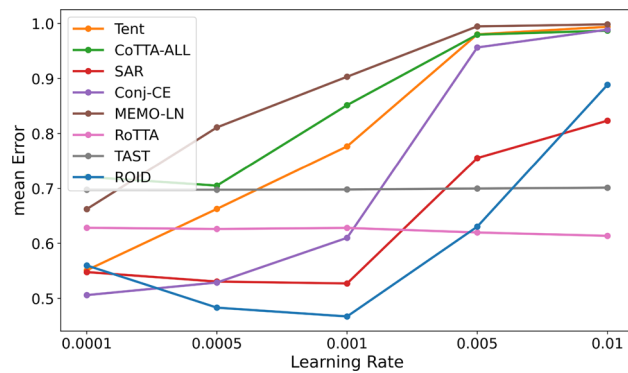


Table 7 Variations of learning rate schedulers on ImageNet-C severity level 5 with batch size 16

Method	Exp	Cos	Cyc	Def	Min
Tent	57.01	72.07	69.30	77.66	55.10
CoTTA-ALL	77.23	82.99	77.48	85.16	72.08
SAR	58.47	52.43	53.99	52.70	54.77
Conj-CE	52.64	54.98	53.03	61.00	50.58
MEMO-LN	61.03	92.59	92.99	90.32	66.22
RoTTA	62.97	62.91	62.91	62.80	62.82
TAST	69.79	69.79	69.79	69.79	69.76
ROID	59.93	48.13	47.87	46.70	55.98

Bold values indicate the lowest error rate

a minimum learning rate of 0.0001 (Min) for the Adam optimizer, they are:

- ExponentialLR (Exp): divides the learning rate every iteration by the decay rate of 0.9, adjusting from the first domain to the last.
- CosineAnnealingWarmRestarts (Cos): periodically resets the learning rate to a higher value and then decreases it following a part of the cosine curve. Here, we gradually decrease the learning rate to 0.0001 for each domain, with the cycle length matching the number of iterations per domain.
- CycleLR (Cyc): cyclically varying the learning rate between two boundary values over a set number of training iterations. Here, we put it to begin with a maximum

learning rate of 0.001 in each domain and decay to 0.0001 by the final iteration of the domain.

As shown in Table 7, cyclical adjustments such as Cos and Cyc of Tent and Conj-CE can surpass the default setup (Def) but do not perform as well as the minimum learning rate (Min), indicating the need for a lower learning rate from the outset of the adaptation process. Meanwhile, RoTTA and TAST demonstrate very stable performance across different learning rate strategies, showcasing their robustness. Notably, RoTTA exceeds the source-only performance (i.e. 63.24%) under any condition. Additionally, ROID demonstrates superior performance when utilizing the default learning rate without a scheduler.

One exception is MEMO-LN, where only ExponentialLR (Exp) can compete with the minimum learning rate. Furthermore, as illustrated in Fig. 14, MEMO-LN produces significant differences in performance among the schedulers and learning rates, indicating its high sensitivity to the learning rate.

4.5 Is OTTA Effective with Other Vision Transformer Variants?

We further evaluate the effectiveness of selected OTTA methods on the SwinTransformer⁵ as a point of comparison to the foundational ViT, as shown in Table 8. Switching the backbone architecture to SwinTransformer can greatly

⁵ swin_base_patch4_window7_224.ms_in22k_ft_in1k.

Table 8 Impact of backbone on ImageNet-C

	SO	Tent	CoTTA-ALL	SAR	Conj-CE	MEMO-LN	RoTTA	TAST	ROID
<i>Batch size = 16</i>									
ViT	63.24	77.66	85.16	52.70	61.00	90.32	62.80	69.79	46.70
Swin	54.61	78.48	88.80	55.44	83.77	85.21	54.38	65.53	31.75
<i>Batch size = 1</i>									
ViT	63.24	98.16	98.72	67.49	97.26	99.32	62.81	69.93	99.91
Swin	54.61	96.98	99.35	56.18	95.61	99.02	54.39	65.84	99.90

Bold values indicate the lowest error rate

improve performance. This is evidenced by the direct inference of SwinTransformer, which outperforms all ViT outputs except for SAR and ROID. This emphasizes the crucial role of backbone architecture in achieving high performance and suggests that improvements in backbone designs can sometimes overshadow the enhancements brought about by advanced OTTA methodologies. Transitioning to a new backbone requires reassessing the effectiveness of individual OTTA methods. Specifically, architecture-agnostic methods like RoTTA (without the RBN module) and ROID outperform the source-only performance with a batch size of 16 using SwinTransformer. Additionally, RoTTA remains stable with a batch size of 1, demonstrating its robustness and adaptability across various architectures.

These findings highlight a dynamic interplay between backbone architectures and OTTA methods. Consequently, evaluating OTTA strategies in the context of evolving transformer models is essential.

5 Future Directions

Our initial evaluations of the Vision Transformer revealed that many Online Test-Time Adaptation methods are not fully optimized for this architecture, resulting in suboptimal outcomes. Based on these findings, we propose several key attributes for an ideal OTTA approach, suitable for future research and tailored to advanced architectures like ViT:

- *Refining OTTA in realistic settings:* Future OTTA methods should undergo testing in realistic environments such as practical TTA (Marsden et al., 2024), weaken the domain boundaries, employing advanced architectures, practical testbeds, and reasonable batch sizes. This approach aims to gain deeper and more relevant insights.
- *Addressing multimodal challenges and exploring prompting techniques:* With the evolution towards foundation models like CLIP (Radford et al., 2021), OTTA is confronted with new challenges. These models may face shifts across various modalities, necessitating innovative OTTA strategies that extend beyond reliance on images alone. Exploring prompt-based methods could offer significant breakthroughs in OTTA.
- *Hot-swappable OTTA:* Keeping pace with the rapid evolution of backbone architectures is crucial. Future OTTA methods should focus on adaptability and generalizability to seamlessly integrate with evolving architectures.
- *Stable and robust optimization for OTTA:* Stability and robustness in optimization remain paramount. Given that larger batch sizes demonstrate limited effectiveness in ViT, future research should investigate more universal optimization improvements. Such advancements aim to

consistently enhance model performance, independent of external factors like batch size.

6 Conclusion

In this survey, we comprehensively examine Online Test-Time Adaptation (OTTA), covering existing methods, relevant datasets, evaluation benchmarks, and their implementations. We conduct extensive experiments to assess the effectiveness and efficiency of current OTTA methods applied to vision transformers. Our findings suggest that noise-synthesized domain shifts often pose greater challenges than other shifts, such as those encountered in real-world or diffusion environments. Additionally, a large number of classes in a dataset can lead to noticeable batch discrepancies, potentially affecting OTTA models' ability to maintain consistent knowledge and increasing the risk of severe forgetting. To address these challenges, we find that updating the normalization layer with a memory bank or optimization flatness, along with appropriate batch size selection, can effectively stabilize the adaptation process and reduce forgetting. We hope this survey serves as a foundational reference, offering valuable insights for researchers and practitioners interested in the evolving field of OTTA.

Appendix A

This appendix offers tables about the categorization of OTTA methods and experiments conducted on the eight selected methods across the chosen datasets.

- Table 9: The summary of existing OTTA methods.
- Table 10: Experiments on the CIFAR-10 \rightarrow CIFAR-10-C task.
- Table 11: Experiments on the ImageNet-1k \rightarrow ImageNet-C task.
- Table 12: Experiments on the CIFAR-10 \rightarrow CIFAR-10-Warehouse Google split task.
- Fig. 15: Ablation study on `LayerNorm` update strategy.

Table 9 The summary of existing OTTA published in the top-tier conferences before Dec-2023

Year	Method	Model-based			Data-based		Optimization-based				
		Add.	Subst.	Prompt	Mem.	Aug.	Loss.	Pl.	T-s.	Norm.	Other
2021	Tent (Wang et al., 2021a)						✓			✓	
2021	MixNorm (Hu et al., 2021)		✓							✓	
2021	PAD (Wu et al., 2021)		✓			✓		✓			
2021	TTPR (Sivaprasad & Fleuret, 2021)					✓	✓				
2021	Core (You et al., 2021)									✓	
2021	SLR (Mummadi et al., 2021)	✓					✓			✓	
2021	T3A (Iwasawa and Matsuo, 2021)		✓		✓						
2022	NOTE (Gong et al., 2022)				✓					✓	
2022	CoTTA (Wang et al., 2022a)					✓			✓		
2022	Conj-PL (Goyal et al., 2022)						✓	✓		✓	
2022	GpreBN (Yang et al., 2022)		✓							✓	
2022	CFA (Kojima et al., 2022)									✓	
2022	DUA (Mirza et al., 2022)		✓			✓				✓	
2022	MEMO (Zhang et al., 2022)					✓	✓			✓	
2022	AdaContrast (Chen et al., 2022)				✓	✓	✓	✓			
2022	TPT (Shu et al., 2022)			✓							
2022	DePT (Gao et al., 2022)			✓							
2022	LAME (Boudiaf et al., 2022)										✓
2023	DN (Gan et al., 2023)			✓							
2023	TAST (Jang et al., 2023)	✓	✓		✓					✓	
2023	MECTA (Hong et al., 2023)		✓							✓	
2023	DELTA (Zhao et al., 2023a)		✓							✓	
2023	TeSLA (Tomar et al., 2023)				✓	✓	✓		✓		
2023	RoTTA (Yuan et al., 2023b)		✓		✓	✓			✓	✓	
2023	EcoTTA (Song et al., 2023)	✓								✓	
2023	TIPi (Nguyen et al., 2023)					✓				✓	
2023	TSD (Wang et al., 2023a)		✓		✓		✓				
2023	SAR (Niu et al., 2023)						✓			✓	
2023	ECL (Han et al., 2023)				✓			✓			
2023	REALM (Seto et al., 2023)						✓			✓	
2023	TTC (Lin et al., 2023)					✓	✓				
2023	SoTTA (Gong et al., 2023)				✓		✓			✓	
2023	ViDA (Liu et al., 2023b)	✓				✓			✓		
2023	ERSK (Niloy et al., 2023)		✓							✓	
2023	ROID (Marsden et al., 2024)					✓	✓			✓	✓

Table 10 Classification error rate (%) for the standard CIFAR-10 → CIFAR-10-C/CIFAR 10.1 online test-time adaptation task

Method	Glass	Fog	Defoc	Impul	Contr	Gauss	Elastic	Zoom	Pixel	Frost	Snow	JPEG	Motion	Shot	Brit	Mean	CIFAR 10.1	
SO	34.60	14.14	12.20	39.47	38.95	60.12	14.70	10.61	25.26	10.39	7.67	16.51	15.30	55.17	3.26	23.89	7.50	
<i>Batch size = 16</i>																		
Tent	49.02	6.81	5.29	65.93	11.91	84.93	10.96	5.12	10.41	6.68	5.96	14.21	8.25	85.60	2.43	24.90	7.60	
CoTTA-LN	34.50	14.57	12.97	43.96	50.01	58.04	15.05	11.09	22.92	10.43	7.75	16.46	15.92	53.65	3.27	24.71	7.50	
CoTTA-ALL	71.68	51.28	38.25	82.60	79.30	85.57	31.22	16.51	39.95	36.75	37.38	37.37	81.00	88.54	9.72	52.47	11.55	
CoTTA*-LN	34.58	14.60	13.02	43.71	49.71	58.00	15.03	11.09	22.96	10.45	7.67	16.53	15.93	53.68	3.24	24.68	7.50	
CoTTA*-ALL	77.17	78.07	70.48	85.98	81.41	86.57	74.54	66.80	68.95	78.34	72.30	79.23	79.62	88.83	67.91	77.08	16.40	
SAR	34.39	14.13	12.18	39.38	34.92	60.67	14.70	10.61	25.17	10.39	7.67	16.51	15.28	55.59	3.26	23.66	7.50	
Conj-CE	61.26	6.36	4.89	80.12	7.94	87.96	10.92	4.20	8.43	5.32	5.33	12.71	7.18	86.63	2.27	26.10	7.35	
MEMO-LN	26.15	9.08	6.17	65.80	9.58	77.68	12.48	5.29	9.14	6.59	5.71	14.45	7.71	59.15	2.49	21.16	7.80	
RoTTA	34.47	13.72	11.70	38.72	37.35	59.55	14.29	10.14	25.38	10.25	7.42	16.31	14.70	54.59	3.07	23.44	7.50	
TAST	55.06	58.89	52.80	57.12	81.54	54.39	37.02	43.68	33.65	32.28	24.21	25.69	54.63	47.69	14.87	44.90	14.85	
ROID	23.31	8.39	7.00	21.91	12.59	29.36	11.58	6.00	14.11	7.07	6.01	13.52	9.10	28.06	2.40	13.36	7.70	
<i>Batch size = 1</i>																		
Tent	70.90	55.50	39.38	87.28	78.20	77.92	25.22	30.77	26.38	71.19	11.85	57.65	69.19	76.82	5.48	52.25	9.05	
CoTTA-LN	34.09	16.63	14.79	43.05	45.14	57.63	15.39	13.23	21.11	10.15	8.25	16.88	17.68	52.36	4.07	24.70	7.30	
CoTTA-ALL	87.81	89.90	86.13	89.17	88.82	89.55	84.91	87.77	86.30	87.42	89.17	88.67	89.06	89.22	81.65	87.70	68.25	
CoTTA*-LN	35.88	18.35	16.96	46.27	47.31	58.21	17.49	15.12	20.95	10.78	9.60	18.15	19.72	52.15	4.83	26.12	7.30	
CoTTA*-ALL	89.79	89.76	86.30	89.63	89.66	89.78	89.17	89.12	89.32	88.04	89.44	89.18	88.76	89.67	87.83	89.03	79.20	
SAR	34.63	14.04	12.03	39.55	37.62	61.17	14.63	10.61	24.85	10.38	7.62	16.63	15.20	55.67	3.27	23.86	7.45	
Conj-CE	48.90	55.11	14.09	88.28	84.49	89.11	14.39	29.01	20.95	13.70	27.84	38.25	12.71	88.95	5.43	42.08	9.00	
MEMO-LN	85.91	5.55	4.77	87.84	3.76	89.33	11.54	3.70	5.52	4.08	4.36	12.28	6.01	89.32	1.65	27.71	7.80	
RoTTA	34.46	13.72	11.71	38.72	37.36	59.55	14.29	10.12	25.39	10.24	7.43	16.32	14.71	54.62	3.07	23.45	7.50	
TAST	55.16	61.79	52.69	56.94	82.01	54.51	36.85	43.73	33.87	32.23	24.27	25.80	55.25	47.80	14.88	45.19	14.85	
ROID	89.99	89.99	89.99	89.99	89.99	89.99	89.99	89.99	89.99	89.99	89.99	89.99	89.99	89.99	89.99	89.99	89.99	

Bold values indicate the lowest error rate

Results are evaluated on ViT-base-patch16-224 with the corruption severity level 5. Here, SO denotes source only

Table 11 Classification error rate (%) for the standard ImageNet-1K \rightarrow ImageNet-C online test-time adaptation task

ImageNet-C	Glass	Fog	Defoc	Impul	Contr	Gauss	Elastic	Zoom	Pixel	Frost	Snow	JPEG	Motion	Shot	Brit	Mean
SO	73.92	63.76	65.54	79.18	89.84	79.88	53.38	62.42	39.26	60.70	63.66	45.06	61.20	77.38	33.38	63.24
<i>Batch size = 16</i>																
Tent	77.86	97.88	73.88	97.24	97.28	95.28	57.54	85.92	33.12	97.64	96.06	42.40	92.60	90.88	29.30	77.66
CoTTA-ALL	73.72	94.82	92.80	99.14	92.94	99.06	78.06	84.14	79.40	88.92	96.70	54.48	86.58	98.10	58.48	85.16
CoTTA-LN	77.06	68.50	75.40	92.10	87.60	92.06	63.28	67.24	50.28	71.36	80.74	52.46	71.58	89.46	46.40	72.37
CoTTA*-ALL	90.30	95.84	95.08	99.28	95.24	99.12	87.90	88.20	90.20	94.84	97.50	68.04	95.82	98.68	62.08	90.54
CoTTA*-LN	77.26	68.60	75.32	91.58	87.08	92.28	63.30	67.18	50.72	70.74	80.64	52.02	71.48	89.72	46.46	72.29
SAR	52.78	48.34	54.74	71.92	85.74	73.18	37.54	46.48	31.70	61.14	49.24	36.74	48.48	64.92	27.52	52.70
Conj-CE	68.30	42.96	76.74	93.70	84.52	96.60	34.46	45.64	31.00	83.90	55.12	35.28	53.32	84.88	28.60	61.00
Conj-Poly	68.26	47.48	54.08	93.68	75.82	96.68	34.08	47.42	29.94	81.26	65.06	36.06	68.36	87.70	29.16	61.00
MEMO-LN	96.10	95.62	96.26	98.68	99.18	98.08	97.48	87.80	35.40	93.58	96.30	83.56	94.50	98.26	84.06	90.32
RoTTA	73.50	63.24	65.14	78.40	89.58	79.02	53.10	61.94	39.12	60.34	63.24	44.62	60.80	76.88	33.12	62.80
TAST	80.60	73.16	75.26	82.22	94.32	81.68	60.98	71.34	47.48	67.62	71.34	48.98	71.68	79.66	40.46	69.79
ROID	51.06	43.02	50.68	61.02	56.70	61.94	36.04	46.02	30.66	48.44	47.30	35.88	46.70	58.38	26.72	46.70
<i>Batch size = 1</i>																
Tent	99.66	99.46	97.90	99.64	99.84	99.80	98.16	98.60	97.56	99.28	97.46	96.90	98.18	99.66	90.34	98.16
CoTTA-ALL	99.24	99.64	99.06	99.54	99.78	99.80	97.94	99.20	97.48	98.08	99.20	98.60	99.46	99.02	94.82	98.72
CoTTA-LN	80.38	69.62	82.82	96.40	90.80	95.30	65.80	67.46	46.52	72.82	81.56	51.42	71.74	93.66	42.28	73.91
CoTTA*-ALL	99.62	99.70	99.54	99.60	99.82	99.80	98.82	99.48	99.30	99.28	99.58	99.20	99.60	99.64	98.70	99.45
CoTTA*-LN	82.84	72.14	84.56	94.98	92.66	94.66	66.16	70.64	48.88	78.10	83.78	51.62	76.08	91.92	45.12	75.61
SAR	70.58	66.64	72.40	83.60	97.76	82.96	57.54	62.98	41.02	64.54	78.52	46.34	66.96	82.72	37.78	67.49
Conj-CE	99.38	96.42	98.18	99.76	99.74	99.76	91.54	96.02	96.74	98.86	97.86	96.90	98.26	99.14	90.38	97.26
MEMO-LN	99.82	99.58	99.78	99.84	99.72	99.90	99.54	99.30	98.04	99.66	99.48	98.46	99.52	99.80	97.30	99.32
RoTTA	73.46	63.26	65.10	78.40	89.58	79.04	53.14	61.94	39.12	60.36	63.18	44.64	60.86	76.88	33.12	62.81
TAST	80.66	73.44	75.50	82.26	94.40	81.74	60.96	71.60	47.64	67.96	71.60	49.06	72.04	79.62	40.48	69.93
ROID	99.90	99.92	99.90	99.92	99.92	99.92	99.90	99.90	99.90	99.90	99.90	99.90	99.90	99.92	99.90	99.91

Bold values indicate the lowest error rate

Results are evaluated on ViT-base-patch16-224 with the corruption severity level 5. SO denotes source only

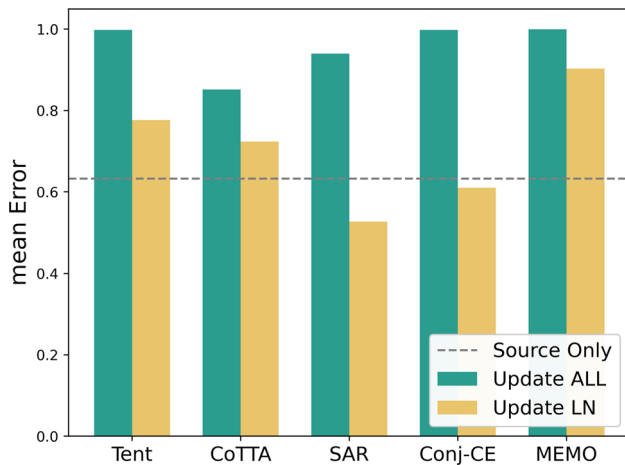
Table 12 Classification error rate (%) for the CIFAR-10 → CIFAR-10-Warehouse online test-time adaptation task

Google SO	G-01	G-02	G-03	G-04	G-05	G-06	G-07	G-08	G-09	G-10	G-11	G-12	Mean
	20.41	12.42	13.39	11.53	19.80	17.74	26.51	26.4	23.12	17.36	15.61	11.37	17.97
<i>Batch size = 16</i>													
Tent	19.23	11.08	13.59	11.07	18.70	17.07	27.25	26.63	25.17	16.80	15.35	9.80	17.65
CoTTA-LN	20.52	12.50	13.56	11.43	20.09	18.02	27.29	26.87	23.36	17.93	15.73	11.43	18.23
CoTTA-ALL	22.28	21.89	29.76	10.55	22.47	20.96	40.86	44.11	41.56	26.07	31.80	24.21	28.04
CoTTA*-LN	20.55	12.48	13.54	11.47	20.04	17.93	27.25	26.71	23.30	17.81	15.88	11.41	18.20
CoTTA*-ALL	40.13	60.93	61.92	33.15	50.54	48.50	56.21	58.46	61.05	61.48	59.86	63.37	54.63
SAR	20.41	12.42	13.39	11.53	19.80	17.74	26.51	26.40	23.15	17.36	15.61	11.37	17.97
Conj-CE	17.91	9.87	11.51	10.11	20.18	16.41	27.22	26.40	24.96	14.89	13.57	9.30	16.86
MEMO-LN	20.70	10.84	12.61	10.88	19.08	16.46	28.11	26.87	20.91	17.21	14.99	10.83	17.46
MEMO-ALL	74.62	90.31	91.95	89.39	92.81	92.49	90.51	92.71	87.48	92.28	89.94	90.07	89.55
RoTTA	20.29	12.37	13.39	11.51	19.73	17.58	26.40	26.40	23.09	17.30	15.52	11.35	17.91
TAST	21.30	16.37	14.26	13.71	23.24	20.89	28.11	28.50	27.30	23.21	18.90	16.13	20.99
ROID	18.02	10.76	12.06	10.55	17.70	15.11	24.55	25.56	19.81	15.41	13.93	9.90	16.11
<i>Batch size = 1</i>													
Tent	25.90	44.81	15.22	12.77	21.57	65.91	26.55	32.14	60.05	59.22	33.68	18.70	34.71
CoTTA-LN	20.44	12.99	13.76	11.78	21.07	19.05	27.22	27.27	24.20	18.30	15.90	11.83	18.65
CoTTA-ALL	84.36	74.83	71.15	83.16	78.50	87.64	81.61	80.46	84.98	85.87	83.04	82.22	81.49
CoTTA*-LN	20.70	13.50	14.11	11.97	21.35	19.31	27.29	28.18	24.57	18.49	15.73	12.33	18.96
CoTTA*-ALL	86.81	85.35	86.21	87.08	88.44	87.41	72.75	90.88	84.03	89.93	85.44	85.36	85.81
SAR	20.29	12.40	13.46	11.45	19.82	17.70	26.66	26.24	23.17	17.30	15.64	11.33	17.95
Conj-CE	24.63	11.08	14.94	9.37	19.58	47.47	23.47	43.52	52.68	19.84	32.57	12.52	25.97
MEMO-LN	15.35	9.53	11.48	8.18	27.78	21.33	21.99	43.48	70.91	12.14	20.35	8.06	22.55
MEMO-ALL	74.62	90.96	91.95	89.39	92.81	92.49	97.44	92.75	87.48	92.28	89.94	90.07	90.18
RoTTA	20.29	12.37	13.36	11.51	19.73	17.56	26.44	26.40	23.09	17.30	15.52	11.37	17.91
TAST	21.41	16.34	14.21	13.67	23.24	20.84	28.11	29.09	27.49	23.32	18.90	16.08	21.06
ROID	96.87	97.85	95.11	96.92	92.81	94.33	97.48	92.87	91.85	94.17	96.16	95.94	95.20

Bold values indicate the lowest error rate

Results are evaluated on ViT-base-patch16-224 with the Google split and diffusion split. Here, SO means source only

Fig. 15 Fully update versus LayerNorm updates on ImageNet-C severity level 5 with batch size 16



Acknowledgements This research is partially supported by the Australian Research Council (DE240100105, DP240101814, DP230101196, DP230101753).

Funding Open Access funding enabled and organized by CAUL and its Member Institutions

Data Availability The data supporting the findings of this study are openly available, as summarized in Table 1. Our open-sourced repository includes the implementation code and experimental configurations of this work.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Adachi, K., Yamaguchi, S., & Kumagai, A. (2023). Covariance-aware feature alignment with pre-computed source statistics for test-time adaptation to multiple image corruptions. In *ICIP* (pp. 800–804). <https://doi.org/10.1109/ICIP49359.2023.10222901>
- Ba, L. J., Kiros, J. R., & Hinton, G. E. (2016). Layer normalization. CoRR [arXiv:1607.06450](https://arxiv.org/abs/1607.06450)
- Barron, J. T. (2019). A general and adaptive robust loss function. In *CVPR* (pp. 4331–4339). <https://doi.org/10.1109/CVPR.2019.00446>
- Boudiaf, M., Müller, R., Ayed, I. B., & Bertinetto, L. (2022). Parameter-free online test-time adaptation. In *CVPR*, pp. 8334–8343. <https://doi.org/10.1109/CVPR52688.2022.00816>
- Brahma, D., & Rai, P. (2023). A probabilistic framework for lifelong test-time adaptation. In *CVPR* (pp. 3582–3591). <https://doi.org/10.1109/CVPR52729.2023.00349>
- Carlucci, F. M., Porzi, L., Caputo, B., Ricci, E., & Bulò, S. R. (2017). Autodial: Automatic domain alignment layers. In *ICCV* (pp. 5077–5085). <https://doi.org/10.1109/ICCV.2017.542>
- Chakrabarty, G., Sreenivas, M., & Biswas, S. (2023). SATA: Source anchoring and target alignment network for continual test time adaptation. CoRR [arXiv:2304.10113](https://arxiv.org/abs/2304.10113)
- Chen, Z., Luo, Y., & Baktashmotlagh, M. (2021). Conditional extreme value theory for open set video domain adaptation. In *MMAasia*. (pp. 20:1–20:8). ACM, <https://doi.org/10.1145/3469877.3490600>
- Chen, Z., Luo, Y., Wang, Z., Baktashmotlagh, M., & Huang, Z. (2023a). Revisiting domain-adaptive 3d object detection by reliable, diverse and class-balanced pseudo-labeling. In *ICCV* (pp. 3691–3703). <https://doi.org/10.1109/ICCV51070.2023.00344>
- Chen, Z., Luo, Y., Wang, Z., Wang, Z., Yu, X., & Huang, Z. (2023b). Towards open world active learning for 3d object detection. CoRR [arXiv:2310.10391](https://arxiv.org/abs/2310.10391)
- Chen, D., Wang, D., Darrell, T., & Ebrahimi, S. (2022). Contrastive test-time adaptation. In *CVPR* (pp. 295–305). <https://doi.org/10.1109/CVPR52688.2022.00039>
- Chen, Z., Wang, Z., Wang, S., Huang, Z., & Luo, Y. (2024). *DPO: Dual-perturbation optimization for test-time adaptation in 3d object detection*. ACM MM.
- Chen, Y., Garcia, E. K., Gupta, M. R., Rahimi, A., & Cazzanti, L. (2009). Similarity-based classification: Concepts and algorithms. *Journal of Machine Learning Research*, 10, 747–776.
- Choi, M., Choi, J., Baik, S., Kim, T. H., & Lee, K. M. (2021). Test-time adaptation for video frame interpolation via meta-learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12), 9615–9628.
- Choi, S., Yang, S., Choi, S., & Yun, S. (2022). Improving test-time adaptation via shift-agnostic weight regularization and nearest source prototypes. *ECCV, Lecture Notes in Computer Science*, 13693, 440–458. https://doi.org/10.1007/978-3-031-19827-4_26
- Cogswell, M., Ahmed, F., Girshick, R. B., Zitnick, L., & Batra, D. (2016). Reducing overfitting in deep networks by decorrelating representations. In *ICLR*.
- Croce, F., Andriushchenko, M., Sehwal, V., Debenedetti, E., Flammarion, N., Chiang, M., Mittal, P., & Hein, M. (2021). Robustbench: A standardized adversarial robustness benchmark. In *NeurIPS benchmarks track*.
- Ding, Y., Liang, J., Jiang, B., Zheng, A., & He, R. (2023). Maps: A noise-robust progressive learning approach for source-free domain adaptive keypoint detection. [arXiv:2302.04589](https://arxiv.org/abs/2302.04589)
- Ding, N., Xu, Y., Tang, Y., Xu, C., Wang, Y., & Tao, D. (2022). Source-free domain adaptation via distribution estimation. In *CVPR* (pp. 7202–7212). <https://doi.org/10.1109/CVPR52688.2022.00707>
- Döbler, M., Marsden, R. A., & Yang, B. (2023). Robust mean teacher for continual and gradual test-time adaptation. In *CVPR* (pp. 7704–7714). <https://doi.org/10.1109/CVPR52729.2023.00744>
- Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houshy, N. (2021). An image is worth 16×16 words: Transformers for image recognition at scale. In *ICLR*.
- Foret, P., Kleiner, A., Mobahi, H., & Neyshabur, B. (2021). Sharpness-aware minimization for efficiently improving generalization. In *ICLR*.
- Gal, Y., & Ghahramani, Z. (2016). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. *ICML, JMLR Workshop and Conference Proceedings*, 48, 1050–1059.
- Gan, Y., Bai, Y., Lou, Y., Ma, X., Zhang, R., Shi, N., & Luo, L. (2023). Decorate the newcomers: Visual domain prompt for continual test time adaptation. In *AAAI* (pp. 7595–7603). <https://doi.org/10.1609/AAAI.V37I6.25922>
- Gandelsman, Y., Sun, Y., Chen, X., & Efros, A. A. (2022). Test-time training with masked autoencoders. In *NeurIPS*.
- Ganin, Y., & Lempitsky, V. S. (2015). Unsupervised domain adaptation by backpropagation. *ICML, JMLR Workshop and Conference Proceedings*, 37, 1180–1189.
- Gao, Y., Shi, X., Zhu, Y., Wang, H., Tang, Z., Zhou, X., Li, M., & Metaxas, D. N. (2022). Visual prompt tuning for test-time domain adaptation. CoRR [arXiv:2210.04831](https://arxiv.org/abs/2210.04831)
- Gao, J., Zhang, J., Liu, X., Darrell, T., Shelhamer, E., & Wang, D. (2023). Back to the source: Diffusion-driven adaptation to test-time corruption. In *CVPR* (pp. 11786–11796). <https://doi.org/10.1109/CVPR52729.2023.01134>
- Gong, T., Jeong, J., Kim, T., Kim, Y., Shin, J., & Lee, S. (2022). NOTE: Robust continual test-time adaptation against temporal correlation. In *NeurIPS*.
- Gong, T., Kim, Y., Lee, T., Chottananurak, S., & Lee, S. (2023). Sotta: Robust test-time adaptation on noisy data streams. CoRR [arXiv:2310.10074](https://arxiv.org/abs/2310.10074)
- Goyal, S., Sun, M., Raghunathan, A., & Kolter, J. Z. (2022). Test time adaptation via conjugate pseudo-labels. In *NeurIPS*.

- Han, J., Zeng, L., Du, L., Ding, W., & Feng, J. (2023). Rethinking precision of pseudo label: Test-time adaptation via complementary learning. CoRR [arXiv:2301.06013](https://arxiv.org/abs/2301.06013)
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *CVPR* (pp. 770–778). <https://doi.org/10.1109/CVPR.2016.90>
- Hegde, D., Sindagi, V., Kilic, V., Cooper, A. B., Foster, M., & Patel, V. (2021). Uncertainty-aware mean teacher for source-free unsupervised domain adaptive 3d object detection. [arXiv:2109.14651](https://arxiv.org/abs/2109.14651)
- Hendrycks, D., & Dietterich, T. G. (2018). Benchmarking neural network robustness to common corruptions and surface variations. [arXiv:1807.01697](https://arxiv.org/abs/1807.01697)
- Hendrycks, D., Mu, N., Cubuk, E. D., Zoph, B., Gilmer, J., & Lakshminarayanan, B. (2020). Augmix: A simple data processing method to improve robustness and uncertainty. In *ICLR*.
- Hong, J., Lyu, L., Zhou, J., & Spranger, M. (2023). MECTA: Memory-economic continual test-time model adaptation. In *ICLR*.
- Hu, X., Uzunbas, M. G., Chen, S., Wang, R., Shah, A., Nevatia, R., & Lim, S. (2021). Mixnorm: Test-time adaptation through online normalization estimation. CoRR [arXiv:2110.11478](https://arxiv.org/abs/2110.11478)
- Huang, J., Smola, A. J., Gretton, A., Borgwardt, K. M., & Schölkopf, B. (2006). Correcting sample selection bias by unlabeled data. In *NIPS* (pp. 601–608).
- Huang, L., Yang, D., Lang, B., & Deng, J. (2018). Decorrelated batch normalization. In *CVPR* (pp. 791–800). <https://doi.org/10.1109/CVPR.2018.00089>
- Ioffe, S. (2017). Batch renormalization: Towards reducing minibatch dependence in batch-normalized models. In *NIPS* (pp. 1945–1953).
- Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ICML, JMLR Workshop and Conference Proceedings*, 37, 448–456.
- Iwasawa, Y., & Matsuo, Y. (2021). Test-time classifier adjustment module for model-agnostic domain generalization. In *NeurIPS* (pp. 2427–2440).
- Jang, M., Chung, S., & Chung, H. W. (2023). Test-time adaptation via self-training with nearest neighbor information. In *ICLR*.
- Jung, S., Lee, J., Kim, N., & Choo, J. (2022). CAFA: Class-aware feature alignment for test-time adaptation. CoRR [arXiv:2206.00205](https://arxiv.org/abs/2206.00205)
- Kimura, M. (2021). Understanding test-time augmentation. *ICONIP, Lecture Notes in Computer Science*, 13108, 558–569. https://doi.org/10.1007/978-3-030-92185-9_46
- Kingetsu H, Kobayashi K, Okawa Y, Yokota Y, Nakazawa K (2022) Multi-step test-time adaptation with entropy minimization and pseudo-labeling. In *ICIP* (pp. 4153–4157). <https://doi.org/10.1109/ICIP46576.2022.9897419>
- Kojima, T., Matsuo, Y., & Iwasawa, Y. (2022). Robustifying vision transformer without retraining from scratch by test-time class-conditional feature alignment. In *IJCAI* (pp. 1009–1016). <https://doi.org/10.24963/IJCAI.2022/141>
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *NIPS* (pp. 1106–1114).
- Lee, T., Tremblay, J., Blukis, V., Wen, B., Lee, B., Shin, I., Birchfield, S., Kweon, I. S., & Yoon, K. (2023). TTA-COPE: Test-time adaptation for category-level object pose estimation. In *CVPR* (pp. 21285–21295). <https://doi.org/10.1109/CVPR52729.2023.02039>
- Li, Y., Wang, N., Shi, J., Liu, J., & Hou, X. (2017). Revisiting batch normalization for practical domain adaptation. In *ICLR*.
- Liang, J., He, R., & Tan, T. (2023). A comprehensive survey on test-time adaptation under distribution shifts. CoRR [arXiv:2303.15361](https://arxiv.org/abs/2303.15361)
- Liang, J., Hu, D., & Feng, J. (2020). Do we really need to access the source data? Source hypothesis transfer for unsupervised domain adaptation. *ICML*, 119, 6028–6039.
- Liang, J., Hu, D., Wang, Y., He, R., & Feng, J. (2022). Source data-absent unsupervised domain adaptation through hypothesis transfer and labeling transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11), 8602–8617.
- Lim, H., Kim, B., Choo, J., & Choi, S. (2023). TTN: A domain-shift aware batch normalization in test-time adaptation. In *ICLR*.
- Lin, G., Lai, H., Pan, Y., & Yin, J. (2023). Improving entropy-based test-time adaptation from a clustering view. CoRR [arXiv:2310.20327](https://arxiv.org/abs/2310.20327)
- Liu, H., Chi, Z., Yu, Y., Wang, Y., Chen, J., & Tang, J. (2023a). Meta-auxiliary learning for future depth prediction in videos. In *WACV* (pp. 5756–5765).
- Liu, J., Yang, S., Jia, P., Lu, M., Guo, Y., Xue, W., & Zhang, S. (2023b). Vida: Homeostatic visual domain adapter for continual test time adaptation. CoRR [arXiv:2306.04344](https://arxiv.org/abs/2306.04344)
- Luo, Y., Huang, Z., Wang, Z., Zhang, Z., Baktashmotlagh, M. (2020). Adversarial bipartite graph learning for video domain adaptation. In *MM* (pp. 19–27). ACM. <https://doi.org/10.1145/3394171.3413897>
- Luo, Y., Wang, Z., Chen, Z., Huang, Z., & Baktashmotlagh, M. (2023). Source-free progressive graph learning for open-set domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(9), 11240–11255.
- Ma, W., Chen, C., Zheng, S., Qin, J., Zhang, H., & Dou, Q. (2022). Test-time adaptation with calibration of medical image classification nets for label distribution shift. *MICCAI, Lecture Notes in Computer Science*, 13433, 313–323. https://doi.org/10.1007/978-3-031-16437-8_30
- Marsden, R. A., Döbler, M., & Yang, B. (2022). Gradual test-time adaptation by self-training and style transfer. CoRR [arXiv:2208.07736](https://arxiv.org/abs/2208.07736)
- Marsden, R. A., Döbler, M., & Yang, B. (2024). Universal test-time adaptation through weight ensembling, diversity weighting, and prior correction. In *WACV* (pp. 2543–2553). <https://doi.org/10.1109/WACV57701.2024.00254>
- Mirza, M. J., Micorek, J., Possegger, H., & Bischof, H. (2022). The norm must go on: Dynamic unsupervised domain adaptation by normalization. In *CVPR* (pp. 14745–14755). <https://doi.org/10.1109/CVPR52688.2022.01435>
- Mitchell, H. B., & Schaefer, P. A. (2001). A “soft” k-nearest neighbor voting scheme. *International Journal of Intelligent Systems*, 16(4), 459–468.
- Mounsaveng, S., Chiaroni, F., Boudiaf, M., Pedersoli, M., & Ayed, I. B. (2023). Bag of tricks for fully test-time adaptation. CoRR [arXiv:2310.02416](https://arxiv.org/abs/2310.02416)
- Mummadi, C. K., Huttmacher, R., Rambach, K., Levinkov, E., Brox, T., Metzger, J. H. (2021). Test-time adaptation to distribution shift by confidence maximization and input transformation. [arXiv:2106.14999](https://arxiv.org/abs/2106.14999)
- Nguyen, A. T., Nguyen-Tang, T., Lim, S., & Torr, P. H. S. (2023). TIPI: Test time adaptation with transformation invariance. In *CVPR* (pp. 24162–24171). <https://doi.org/10.1109/CVPR52729.2023.02314>
- Niloy, F. F., Ahmed, S. M., Raychaudhuri, D. S., Oymak, S., Roy-Chowdhury, A. K. (2023). Effective restoration of source knowledge in continual test time adaptation. CoRR [arXiv:2311.04991](https://arxiv.org/abs/2311.04991)
- Niu, S., Wu, J., Zhang, Y., Wen, Z., Chen, Y., Zhao, P., & Tan, M. (2023). Towards stable test-time adaptation in dynamic wild world. In *ICLR*.
- Qiao, F., Zhao, L., & Peng, X. (2020). Learning to learn single domain generalization. In *CVPR* (pp. 12553–12562). <https://doi.org/10.1109/CVPR42600.2020.01257>
- Quinero-Candela, J., Sugiyama, M., Schwaighofer, A., & Lawrence, N. D. (2008). *Dataset shift in machine learning*. MIT Press.
- Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., & Sutskever, I. (2021). Learning transferable visual models from natural language supervision. *ICML, Proceedings of Machine Learning Research*, 139, 8748–8763.
- Recht, B., Roelofs, R., Schmidt, L., & Shankar, V. (2018). Do CIFAR-10 classifiers generalize to CIFAR-10?. CoRR [arXiv:1806.00451](https://arxiv.org/abs/1806.00451)

- Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In *CVPR* (pp. 10674–10685). <https://doi.org/10.1109/CVPR52688.2022.01042>
- Roy, S., Siarohin, A., Sangineto, E., Bulò, S. R., Sebe, N., & Ricci, E. (2019). Unsupervised domain adaptation using feature-whitening and consensus loss. In *CVPR* (pp. 9471–9480). <https://doi.org/10.1109/CVPR.2019.00970>
- Saltori, C., Krivosheev, E., Lathuilière, S., Sebe, N., Galasso, F., Fiameni, G., Ricci, E., & Poiesi, F. (2022). GIPSO: Geometrically informed propagation for online adaptation in 3d lidar segmentation. *ECCV, Lecture Notes in Computer Science*, 13693, 567–585. https://doi.org/10.1007/978-3-031-19827-4_33
- Schmidhuber, J. (1992). Learning factorial codes by predictability minimization. *Neural Computation*, 4(6), 863–879.
- Seto, S., Theobald, B., Danieli, F., Jaitly, N., & Busbridge, D. (2023). REALM: Robust entropy adaptive loss minimization for improved single-sample test-time adaptation. CoRR [arXiv:2309.03964](https://arxiv.org/abs/2309.03964)
- Shanmugam, D., Blalock, D. W., Balakrishnan, G., Gutttag, J. V. (2021). Better aggregation in test-time augmentation. In *ICCV* (pp. 1194–1203). <https://doi.org/10.1109/ICCV48922.2021.00125>
- Shu, M., Nie, W., Huang, D., Yu, Z., Goldstein, T., Anandkumar, A., & Xiao, C. (2022). Test-time prompt tuning for zero-shot generalization in vision-language models. In *NeurIPS*.
- Sivaprasad, P. T., & Fleuret, F. (2021). Test time adaptation through perturbation robustness. CoRR [arXiv:2110.10232](https://arxiv.org/abs/2110.10232)
- Song, J., Lee, J., Kweon, I. S., & Choi, S. (2023). Ecotta: Memory-efficient continual test-time adaptation via self-distilled regularization. CoRR [arXiv:2303.01904](https://arxiv.org/abs/2303.01904)
- Sun, X., Leng, X., Wang, Z., Yang, Y., Huang, Z., & Zheng, L. (2023). CIFAR-10-warehouse: Broad and more realistic testbeds in model generalization analysis. CoRR [arXiv:2310.04414](https://arxiv.org/abs/2310.04414)
- Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A. A., & Hardt, M. (2020). Test-time training with self-supervision for generalization under distribution shifts. *ICML, Proceedings of Machine Learning Research*, 119, 9229–9248.
- Tarvainen, A., & Valpola, H. (2017). Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *ICLR*.
- Thopalli, K., Turaga, P. K., & Thiagarajan, J. J. (2022). Domain alignment meets fully test-time adaptation. *ACML, Proceedings of Machine Learning Research*, 189, 1006–1021.
- Tomar, D., Vray, G., Bozorgtabar, B., & Thiran, J. (2023). Tesla: Test-time self-learning with automatic adversarial augmentation. CoRR [arXiv:2303.09870](https://arxiv.org/abs/2303.09870)
- Ulyanov, D., Vedaldi, A., & Lempitsky, V. S. (2016). Instance normalization: The missing ingredient for fast stylization. CoRR [arXiv:1607.08022](https://arxiv.org/abs/1607.08022)
- van den Oord, A., Li, Y., & Vinyals, O. (2018). Representation learning with contrastive predictive coding. CoRR [arXiv:1807.03748](https://arxiv.org/abs/1807.03748)
- Venkateswara, H., Eusebio, J., Chakraborty, S., & Panchanathan, S. (2017). Deep hashing network for unsupervised domain adaptation. In *CVPR* (pp. 5385–5394). <https://doi.org/10.1109/CVPR.2017.572>
- Wang, Q., Fink, O., Gool, L. V., Dai, D. (2022a) Continual test-time domain adaptation. In *CVPR* (pp. 7191–7201). <https://doi.org/10.1109/CVPR52688.2022.00706>
- Wang, Z., Luo, Y., Chen, Z., Wang, S., & Huang, Z. (2023b). CalSFDA: Source-free domain-adaptive semantic segmentation with differentiable expected calibration error. In *MM* (pp. 1167–1178). ACM, <https://doi.org/10.1145/3581783.3611808>
- Wang, Z., Luo, Y., Huang, Z., & Baktashmotlagh, M. (2020). Prototype-matching graph network for heterogeneous domain adaptation. In *MM* (pp. 2104–2112). ACM, <https://doi.org/10.1145/3394171.3413662>
- Wang, Z., Luo, Y., Qiu, R., Huang, Z., Baktashmotlagh, M. (2021b). Learning to diversify for single domain generalization. In *ICCV* (pp. 814–823). <https://doi.org/10.1109/ICCV48922.2021.00087>
- Wang, Z., Luo, Y., Zhang, P., Wang, S., & Huang, Z. (2022b). Discovering domain disentanglement for generalized multi-source domain adaptation. In *ICME* (pp. 1–6). <https://doi.org/10.1109/ICME52920.2022.9859733>
- Wang, Y., Ma, X., Chen, Z., Luo, Y., Yi, J., & Bailey, J. (2019). Symmetric cross entropy for robust learning with noisy labels. In *ICCV* (pp. 322–330). <https://doi.org/10.1109/ICCV2019.00041>
- Wang, D., Shelhamer, E., Liu, S., Olshausen, B. A., Darrell, T. (2021a). Tent: Fully test-time adaptation by entropy minimization. In *ICLR*.
- Wang, Z., Ye, M., Zhu, X., Peng, L., Tian, L., Zhu, Y. (2022c). Metateacher: Coordinating multi-model domain adaptation for medical image classification. In *NeurIPS*.
- Wang, S., Zhang, D., Yan, Z., Zhang, J., & Li, R. (2023a). Feature alignment and uniformity for test time adaptation. CoRR [arXiv:2303.10902](https://arxiv.org/abs/2303.10902)
- Wang, M., & Deng, W. (2018). Deep visual domain adaptation: A survey. *Neurocomputing*, 312, 135–153.
- Wen, Y., Tran, D., Ba, J. (2020). Batchensemble: An alternative approach to efficient ensemble and lifelong learning. In *ICLR*.
- Wilson, A. C., Roelofs, R., Stern, M., Srebro, N., & Recht, B. (2017). The marginal value of adaptive gradient methods in machine learning. In *NIPS* (pp. 4148–4158).
- Wu, Q., Yue, X., Sangiovanni-Vincentelli, A. (2021). Domain-agnostic test-time adaptation by prototypical training with auxiliary data. In *NeurIPS workshop on distribution shifts*.
- Wu, Y., & He, K. (2020). Group normalization. *International Journal of Computer Vision*, 128(3), 742–755.
- Xu, Q., Zhang, R., Zhang, Y., Wang, Y., & Tian, Q. (2021). A Fourier-based framework for domain generalization. In *CVPR* (pp. 14383–14392). <https://doi.org/10.1109/CVPR46437.2021.01415>
- Yang, S., Wang, Y., van de Weijer, J., Herranz, L., & Jui, S. (2021). Exploiting the intrinsic neighborhood structure for source-free domain adaptation. In *NeurIPS* (pp. 29393–29405).
- Yang, H., Zhang, X., Yin, F., & Liu, C. (2018). Robust classification with convolutional prototype learning. In *CVPR* (pp. 3474–3482). <https://doi.org/10.1109/CVPR.2018.00366>
- Yang, T., Zhou, S., Wang, Y., Lu, Y., & Zheng, N. (2022). Test-time batch normalization. CoRR [arXiv:2205.10210](https://arxiv.org/abs/2205.10210)
- Yang, J., Meng, X., & Mahoney, M. W. (2016). Implementing randomized matrix algorithms in parallel and distributed environments. *Proceedings of the IEEE*, 104(1), 58–92.
- You, F., Li, J., & Zhao, Z. (2021). Test-time batch statistics calibration for covariate shift. CoRR [arXiv:2110.04065](https://arxiv.org/abs/2110.04065)
- Yuan, L., Xie, B., & Li, S. (2023a). Robust test-time adaptation in dynamic scenarios. CoRR [arXiv:2303.13899](https://arxiv.org/abs/2303.13899)
- Yuan, L., Xie, B., & Li, S. (2023b). Robust test-time adaptation in dynamic scenarios. In *CVPR2023* (pp. 15922–15932). <https://doi.org/10.1109/CVPR52729.2023.01528>
- Zagoruyko, S., & Komodakis, N. (2016). Wide residual networks. In *BMVC*. BMVA Press.
- Zhang, M., Levine, S., & Finn, C. (2022). MEMO: Test time robustness via adaptation and augmentation. In *NeurIPS*.
- Zhao, B., Chen, C., & Xia, S. (2023a). Delta: Degradation-free fully test-time adaptation. In *ICLR*.
- Zhao, H., Liu, Y., Alahi, A., & Lin, T. (2023). On pitfalls of test-time adaptation. *ICML, Proceedings of Machine Learning Research*, 202, 42058–42080.
- Zhou, K., Liu, Z., Qiao, Y., Xiang, T., & Loy, C. C. (2023). Domain generalization: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4), 4396–4415.