SloMo-Fast: Slow-Momentum and Fast-Adaptive Teachers for Source-Free Continual Test-Time Adaptation

Md Akil Raihan Iftee[1,*], Mir Sazzat Hossain[1], Rakibul Hasan Rajib[1], A K M Mahbubur Rahman[1],
Tariq Iqbal[2], Md Mofijul Islam[3,†], M Ashraful Amin[1], and Amin Ahsan Ali[1]
[1]Center for Computational & Data Sciences, Independent University, Bangladesh
[2]University of Virginia, USA, [3]Amazon GenAI, USA

Abstract

Continual Test-Time Adaptation (CTTA) is crucial for deploying models in real-world applications with unseen, evolving target domains and restricted access to source data. Existing CTTA methods, however, often rely on source data or prototypes, limiting their applicability in privacy-sensitive and resource-constrained settings. Additionally, these methods suffer from long-term forgetting, which degrades performance on previously encountered domains as target domains shift. To address these challenges, we propose SloMo-Fast, a source-free, dual-teacher CTTA framework designed for enhanced adaptability and generalization. SloMo-Fast includes two complementary teachers: the Slow-Teacher, which adapts gradually to ensure robust generalization by tracking an exponential moving average of the student's parameters, and the Fast-Teacher, which quickly adapts to new domains by constructing prototypes from high-confidence test samples, gathering knowledge across domains. This efficiently approach preserves knowledge of past domains, adapts efficiently to new ones, and reduces computational complexity by relying solely on batch normalization updates. Our extensive experimental results demonstrate that SloMo-Fast consistently outperforms state-of-the-art methods across CTTA benchmarks, achieving mean error rates of 15.79% on CIFAR10-C, 27.01% on CIFAR100-C, and 54.2% on ImageNet-C in the Continual TTA setting, significantly outperformed state-of-the-art methods. Additionally, SloMo-Fast achieves significant performance improvements in Mixed Domain and our proposed new benchmark Mixed domain comes after Continual Domain scenarios along with Cyclic repeatation in continual test time adaptation setting, indicating its ability to learn generalized representations across domains.

1. Introduction

Adapting models to changing environments is crucial for deploying autonomous systems in real-world scenarios. Continual Test-Time Adaptation (CTTA) has emerged as a vital area of research, addressing the need for models to adapt continuously to dynamically changing and unknown domains. This capability is particularly significant in fields like autonomous driving, healthcare, and robotics, where systems must handle evolving conditions without prior knowledge of the changes [3, 21].

Test-Time Adaptation (TTA) methods such as TENT [18], MEMO [24], and EATA [13] focus on adapting models to a single domain. In contrast, CTTA is designed to handle sequences of domains over time, making it suitable for applications like self-driving cars, where weather, lighting, and road conditions change unpredictably [9].

Although there are a number of recent works on CTTA [2, 5, 6, 10–12, 16, 17, 19, 22, 23], there remain significant open research challenges to make it practical and effective for real-world applications. Models must efficiently adapt to evolving data streams in a source-free setting while preserving knowledge from the source domain. Furthermore, robust generalization is essential to avoid forgetting previously encountered domains, as earlier test-time conditions may recur. Many approaches depend on pseudo-labeling within teacher-student frameworks, where ensuring the accuracy and robustness of pseudo-labels is critical. Addressing these challenges is key to unlocking the full potential of CTTA for real-world applications.

To address key CTTA challenges, some methods (e.g., Robust Mean Teacher (RMT) [2] and domain-specific block selection [22]) utilize source prototypes or stored representations from the source domain to guide model adaptation at test time. These prototypes help retain domain-specific features, improving the model's performance with target domain shifts. However, in real-world applications, access to source data or prototypes is often restricted due to privacy concerns [3], storage limitations, or practical constraints re-

[*] Corresponding author: iftee1807002@gmail.com
[†] Work does not relate to position at Amazon.

lated to data transmission and memory capacity [13, 21], which limits the applicability of these methods in privacy-sensitive settings such as healthcare.

In fully source-free settings, some methods aim to prevent catastrophic forgetting of the source domain. For example, CoTTA [19] introduces stochastic restoration of the source model, which is effective for single-domain adaptation. However, CoTTA has limitations: it adapts only one domain at a time, and its stochastic restoration approach can cause the model to forget information about previously encountered test-time domains. Other recent works, such as ROID [11], continuously ensemble parameters from both the source and target models to retain information from past domains. While effective in some continual settings, ROID's approach does not address real-world CTTA scenarios where domains may repeat over time (e.g., in autonomous driving or UAV applications, where weather patterns can recur). Additionally, most CTTA models are not evaluated under conditions where new data from previously seen domains may arrive out of sequence, which is a common scenario in real applications.

Finally, the performance of self-training-based teacher-student CTTA models (e.g., [19, 22, 23]) relies heavily on the quality of pseudo-labels produced by the teacher model. Although state-of-the-art self-training methods have shown promise, they are susceptible to noisy pseudo-labels. High-entropy samples can produce noisy gradients, potentially disrupting model adaptation in continual settings. Moreover, when adapting to long sequences of domains, models can develop biases [11]. To address these issues, [11] employs diversity and certainty-based weighting. However, generating robust pseudo-labels remains an open challenge for teacher-student-based CTTA architectures.

To address the challenges in CTTA, we propose a dual teacher- one student-based framework, SloMo-Fast, that eliminates the need for source data while enhancing adaptability and generalization (Figure 1). SloMo-Fast employs a dual-teacher approach: the Fast-Teacher (T1) adapts quickly to new domains, while the Slow-Teacher (T2) adapts gradually to ensure robust generalization (Figure 2). Unlike existing methods, our framework updates models solely through batch normalization, significantly reducing computational complexity. A key novelty is using class-wise prototypes to capture entropy-based confident feature representations across domains, which are then used to refine the Slow-Teacher through contrastive learning. To maintain generalization during prolonged exposure to a single domain, the Slow-Teacher's weights are periodically restored from the source model. This dual-teacher design enables effective adaptation to current domains while preserving knowledge of previously encountered ones, ensuring reliable pseudo-labels and robust performance in dynamic, continually evolving real-world environments.

Our extensive experimental results demonstrate that SloMo-Fast consistently outperforms state-of-the-art methods across various CTTA benchmarks, including CIFAR10-C, CIFAR100-C, and ImageNet-C. In the Continual TTA setting, SloMo-Fast achieves mean error rates of 15.79% on CIFAR10-C, 27.01% on CIFAR100-C, and 54.2% on ImageNet-C, significantly outperforming methods such as TENT-cont. (20.0% on CIFAR10-C, 62.2% on CIFAR100-C, 82.5% on ImageNet-C), RoTTA (19.3%, 34.8%, 78.1%), and CoTTA (16.5%, 32.8%, 76.0%) on the respective datasets. In the Mixed Domain and Mixed After Continual Domain scenarios, SloMo-Fast achieves error rates of 28.0% on CIFAR10-C, 33.5% on CIFAR100-C, and 54.2% on ImageNet-C, again outperforming all competing methods. These results highlight SloMo-Fast's ability to adapt efficiently and generalize effectively, setting new benchmarks for CTTA performance.

The key contributions of our work are as follows:

- We propose SloMo-Fast, a novel dual-teacher CTTA framework that eliminates the need for source data while enhancing adaptability and generalization. The Fast-Teacher (T1) adapts quickly to new domains, while the Slow-Teacher (T2) ensures robust generalization by adapting gradually.
- SloMo-Fast solely uses batch normalization to update parameters, thus significantly reducing computational complexity.
- We introduce a novel entropy-aware prototype prioritization approach to refine the Slow-Teacher for learning generalized representations across domains.
- Our extensive experiments demonstrate that SloMo-Fast achieves state-of-the-art performance in the Continual TTA setting, with mean error rates of 15.79% on CIFAR10-C, 27.01% on CIFAR100-C, and 54.2% on ImageNet-C, outperforming state-of-the-art methods.
- Additionally, SloMo-Fast outperforms state-of-the-art methods in the Mixed Domain and Mixed After Continual Domain scenarios, achieving error rates of 28.0% on CIFAR10-C, 33.5% on CIFAR100-C, and 54.2% on ImageNet-C.

2. Related Works

In this section, we describe the works on Test-time Adaptation (TTA) models. Then, we provide summary of Continual Test Time Domain Adaptation (CTTA) models followed by their experimental setting such as gradual adaptation.

2.1. Test-time Adaptation (TTA)

TENT [18] introduced one of the earliest approaches to fully test-time adaptation by minimizing entropy with respect to the batch normalization parameters, enabling models to adapt to a single-target domain without requiring

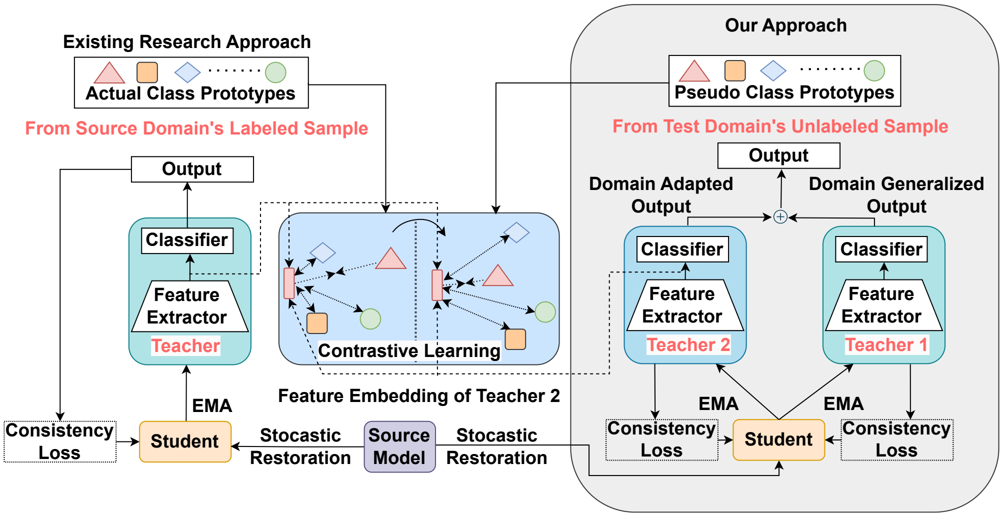

Figure 1. Overview of CTTA approaches with teacher-student models and contrastive learning. SlowMo-FAST (on the right) integrates a second teacher model and dynamically generates prototypes at test time without requiring source data.

source data. MEMO [24] extended this concept by introducing test-time augmentations, encouraging the model to make invariant predictions across different augmented views of the same test data. AdaContrast [1] approached the problem of domain adaptation with contrastive learning, which focuses on maintaining consistency within the target domain by refining pseudo-labels using soft voting from nearest neighbors in the target feature space. EATA [13] developed a robust approach to test-time adaptation through entropy-based sample selection, where reliable and diverse samples are used to guide model updates while unreliable samples are filtered out. To prevent catastrophic forgetting, EATA employed elastic weight consolidation. SAR [14] focused on stabilizing online model updates during test-time adaptation in dynamic environments.

2.2. Continual Test Time Adaptation (CTTA)

CoTTA [19] employed a teacher-student framework to address the challenge of continual test-time adaptation in non-stationary environments. EcoTTA [16] introduced a memory-efficient approach to continual test-time adaptation by leveraging meta-networks and self-distilled regularization. RoTTA [23] built on this approach by introducing a time-aware reweighting strategy that takes into account the timeliness and uncertainty of samples in dynamic environments. DeYo [5] proposed a novel approach to sample selection with a new confidence metric, Pseudo-Label Probability Difference (PLPD). Domain-Specific Block Selection and Paired-View Pseudo-Labeling (DPLOT) [22] advanced the idea of fine-tuning specific parts of the network during test-time adaptation. CMF [6] introduced continual momentum filtering (CMF) to handle catastrophic forgetting during continual test-time adaptation. BECoTTA [4] took a step further by selectively captures domain-specific knowledge through a set of low-rank experts, which are updated dynamically to accommodate new domains. VIDA [10] focused on addressing the trade-off between adaptability and catastrophic forgetting in dynamic environments by introducing high-rank and low-rank domain adapters (ViDAs). Lastly, Parameter-Selective Continual Test-Time Adaptation (PSMT) [17] introduced a method that selectively updates only certain parameters of the network to prevent overfitting during continual adaptation.

2.3. CTTA with Gradual/Mixed settings

RMT [2] focused on continual and gradual domain shifts, introducing a robust mean teacher model that leverages contrastive learning to pull the target domain closer to the source domain. GTTA [12] tackled the issue of gradual domain shift by creating intermediate domains through mixup and lightweight style transfer, addressing both gradual and abrupt domain changes to prevent error accumulation over long test sequences. Universal test-time adaptation (ROID) [11] proposed a new approach for continual and mixed setting. It presents a universal test-time adaptation (TTA) approach aimed at improving model robustness across various environmental conditions by addressing challenges like model bias, catastrophic forgetting, and class prior shifts. Its methodology includes weight ensembling, certainty and diversity weighting, and an adaptive prior correction mechanism to balance model generalization with domain-specific adaptations. The setup focuses on continually adapting the model through normalization layer adjustments, such as using group or layer normalization instead of batch normalization.

2.4. Architecture Evolution

CoTTA [19] employs a teacher-student framework where the student model is updated based on the pseudo-labels generated by the teacher model. The teacher model, in turn, is updated using the Exponential Moving Average (EMA) of the student parameters. While CoTTA demonstrates effective adaptation, it suffers from catastrophic forgetting and lacks the ability to retain long-term domain knowledge.

To address this limitation, RMT [2] introduces source prototypes and utilizes contrastive loss between the source class prototypes and test-time inputs. However, relying on source prototypes is often impractical in real-world scenarios due to their rarity and unavailability in many applications.

In contrast, our SloMo-Fast framework introduces a second teacher model that is more domain-generalized. Instead of using source prototypes, SloMo-Fast constructs class prototypes from confident test samples. This approach eliminates the dependence on source data while enabling long-term retention of domain knowledge, ensuring robust adaptation and generalization across dynamic and evolving domains.

3. Methodology

3.1. Overview

We consider the task of adapting a pre-trained model to perform effectively in a continuously evolving target domain. The initial model, denoted as f_{θ_0} with parameters θ_0 , has been trained on a source dataset (X^s, Y^s) . Our objective is to enhance this model's performance during inference in a dynamic environment, where data distributions (domain) change over time, without access to the source data. Concretely, at time step t, the model receives a new target data x_t as input and is required to generate a prediction $f_{\theta_t}(x_t)$. Simultaneously, the model must adapt its parameters $\theta_t \to \theta_{t+1}$ to improve its performance for subsequent data points. The distribution of the incoming data changes over time and the model is evaluated based on its real-time predictions under this changing distribution.

Fig. 2 gives an overview of our proposed method, which incorporates two teacher models and a student model. All three models share the same architecture, comprising a feature extractor and a classifier. They are initialized with the same pre-trained weigths, θ_0. The models differ in their update strategies. The student model S, with weights θ_S, is updated using a combination of symmetric cross-entropy and differential losses, leveraging pseudo-labels generated from both teacher models. The fast-teacher model, T_1, updates its weights, θ_{T_1}, using an exponential moving average (EMA) of the student model's weights, which provides a smoother version of the student's learning process. In contrast, the slow-teacher model, T_2, initially updates its weights, θ_{T_2}, by optimizing a combination of contrastive loss, mean squared error (MSE) loss, and information maximization loss, enabling T_2 to learn domain-invariant features. Afterward, its parameters are updated by taking the exponential moving average of the student model at each time step. This two-teacher framework offers complementary supervision to the student model, effectively enhancing adaptation and stability across continually shifting distributions. The following subsections detail our proposed approach: selft-training with dual teacher (3.2), slow teacher (T_2) model training (3.3), and prediction ensembling (3.4).

3.2. Self-training with Dual Teacher

For an incoming test sample x_t at time step t, the student model S aims to minimize the discrepancy between its own predictions and those generated by the teacher models T_1 and T_2. Rather than using the standard cross-entropy for discrepancy minimization, we use symmetric cross-entropy[20], which was originally proposed to address noisy labels and has been shown to exhibit better gradient properties compared to standard cross-entropy[2]. For two distributions p and q, the symmetric cross-entropy is defined as:

$$\mathcal{L}_{SCE}(p, q) = -\sum_{c=1}^{C} p(c) \log q(c) - \sum_{c=1}^{C} q(c) \log p(c) \quad (1)$$

where C is the number of classes, $p(c)$ and $q(c)$ represents the probability of class c under distribution p and q, respectively. The training objective for the student model S, leveraging predictions from teacher models T_1 and T_2, results in the following self-training loss:

$$\begin{aligned} \mathcal{L}_{\mathcal{ST}}(x_t) = &\mathcal{L}_{SCE}(f_{\theta_S}(x_t), f_{\theta_{T_1}}(x_t)) + \\ &\mathcal{L}_{SCE}(f_{\theta_S}(x_t), f_{\theta_{T_2}}(x_t)) \end{aligned} \quad (2)$$

After updating the student model S using $\mathcal{L}_{\mathcal{ST}}$, the parameters of the teacher model T_1 are updated through EMA as follows:

$$\theta_{T_1}^{t+1} = \alpha \theta_{T_1}^{t} + (1 - \alpha) \theta_S^{t+1} \quad (3)$$

Here, α is a smoothing factor.

3.3. Domain Generalized (T_2) Model Training

Entropy-based Feature Selection: During the adaptation process, we use feature representations of incoming test samples obtained from T_1 to maintain a fixed-size priority queue for each class. In the absence of ground-truth labels, we rely on pseudo labels to assign class membership for

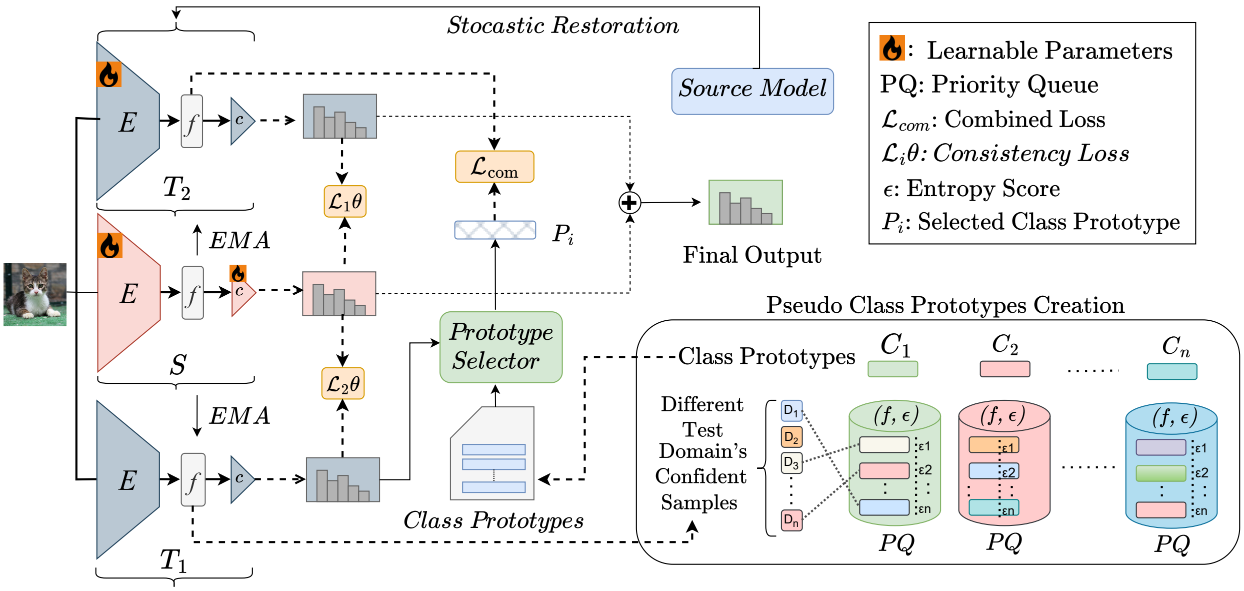

Figure 2. The SlowMo-FAST framework comprises a dual-teacher and student model. The fast teacher T_1 quickly adapts to the current domain by taking the exponential moving average of the student. Confident feature vectors from T_1 are used to construct robust class prototypes via a priority queue, which refine the slow teacher T_2 through contrastive learning. This enables T_2 to learn domain-invariant representations while preserving knowledge from previous domains.

each feature representation. We define pseudo label for a test sample x_t as:

$$\hat{y}_{T_1} = \arg\max_{c}\, y_{T_1}(c) \tag{4}$$

where $y_{T_1}(c)$ is the prediction for c-th class obtained from T_1 for the test sample. This class-wise priority queue is then used to construct class prototypes, with each queue storing features and their associated entropy values up to a maximum size K. To select the most confident features, we prioritize features with lower entropy values in their predictions, ensuring the priority queue retains high-confidence features for constructing robust prototypes. The entropy of the prediction is defined as:

$$\mathcal{H}(y_{T_1}) = -\sum_{c=1}^{C} y_{T_1}(c)\log(y_{T_1}(c)) \tag{5}$$

where C is the number of classes.

Periodically, features with the lowest entropy for each class are removed, allowing new, potentially less confident features from different domains to be included in the priority queue. This approach ensures that the priority queue maintains diverse representations from all domains. The complete process is outlined in the supplementary materials.

Prototype Generation: We utilize confident feature representations stored in the priority queue to generate class prototypes. Instead of simply taking the mean, we compute a weighted average, where the weight is the inverse of the entropy, normalized to ensure consistency. This approach ensures that more confident feature vectors, contribute more to the prototype, making it a representative feature for the corresponding class. Given the priority queue Q_c for class c, the prototype for that class, P_c, is calculated as:

$$P_c = \frac{1}{w} \sum_{(z,\mathcal{H}(z))\in Q_c} w_z z, \tag{6}$$

where z denotes a stored feature, $\mathcal{H}(z)$ is the entropy of the prediction corresponding to that feature, and $w_z = \frac{1}{\mathcal{H}(z)}$. The normalization factor, $w = \sum_{(z,\mathcal{H}(z))\in Q_c} w_z$.

Contrastive Learning with Class Prototype: Contrastive learning with class prototypes enables domain-generalized feature learning by pulling samples from the same class across different domains towards a shared prototype while pushing away samples from different classes. During test time, we first update the priority queue by collecting the features obtained from the teacher model T_1 for the test samples in the current batch X_t. Then, we recompute the prototypes for each class using equation (6). Next, we select samples where T_1 is confident but T_2 is not. This selection process ensures that we are focusing on samples that T_1 understands well, but which T_2 could still benefit from learning. We define a binary variable n to identify whether features of the i-th sample of the current batch will be included for training the T_2 model.

$$n_i = \mathbb{1}[\mathcal{H}(y_{T_1}) \leq \sigma].\mathbb{1}[\mathcal{H}(y_{T_2}) > \sigma] \tag{7}$$

where $y_{T_1}^{(i)}$ and $y_{T_2}^{(i)}$ are the prediction for the i-th sample by T_1 and T_2 model, respectively. where s_i is the feature representation obtained from T_2 for the i-th test sample in the current batch. For each feature in $\mathbb{S}$, we compute the cosine similarity with each class prototype and select the nearest class prototype to form a positive pair. To make the model invariant to small changes in the input space, we also include the corresponding feature of the test sample. This results in a batch size of $3N$, where N is the number of features in $\mathbb{S}$ and each batch consists of the features of original test samples, their augmented views, and the nearest class prototypes. Consider $i \in I := \{1, \ldots, 3N\}$, $A(i) := I \backslash \{i\}$ denote the set of indices excluding i, and $V(i)$ represent the different views of the current sample i. Following [2], we apply a non-linear projection layer to obtain $z = \text{Proj}(s_i)$. The contrastive loss is then defined as:

$$\mathcal{L}_{\mathcal{CL}} = -\sum_{i \in I} \sum_{v \in V(i)} \log \left(\frac{\exp\left(\text{sim}(z_i, z_v)/\tau\right)}{\sum_{a \in A(i)} \exp\left(\text{sim}(z_i, z_a)/\tau\right)} \right) \tag{8}$$

where τ denotes the scalar temperature and $\text{sim}(u, v) = \frac{u^T v}{\|u\| \|v\|}$ is the cosine similarity.

Feature Alignment with MSE Loss: To further encourage the T_2 model to learn domain-generalized features, we apply MSE loss with class prototypes, aligning sample features with class centers across domains. This helps the model focus on class characteristics, reducing domain-specific noise. We compute the MSE loss for each test sample by comparing its feature representation z_i from T_2 with the corresponding class prototype $P_{\hat{y}^i_{T_1}}$:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N} \sum_{i=1}^{N} \|z_i - P_{\hat{y}^i_{T_1}}\|^2 \tag{9}$$

where N is the number of test samples, and $\hat{y}^i_{T_1}$ is the pseudo label for the sample x^i_t as defined in Equation (4).

Information Maximization Loss: Since T_2 serves as a guide to the student model S in our framework, it should exhibit strong individual discriminability and diversity in its predictions. To achieve this, we follow state-of-the-art methods [8][7] for unsupervised domain adaptation and apply an information maximization loss, $\mathcal{L}_{\mathcal{IM}}$ which comprises two components. The entropy loss, $\mathcal{L}_{ent}$, improves the individual certainty of predictions.

$$\mathcal{L}_{ent} = -\mathbb{E}_{x_t \in X_t} \sum_{c=1}^{C} y_{T_2}(c) \log(y_{T_2}(c)) \tag{10}$$

where $y_{T_2} = f_{\theta_{T_2}}(x_t)$. The diversity loss, $\mathcal{L}_{div}$, encourages variations across class distributions.

$$\mathcal{L}_{div} = \sum_{c=1}^{C} \bar{q}(c) \log(\bar{q}(c)) \tag{11}$$

where $\bar{q}(c) = \mathbb{E}_{x_t \in X_t}\left(y_{T_2}(c)\right)$. We then define the information maximization loss as:

$$\mathcal{L}_{\mathcal{IM}} = \mathcal{L}_{ent} - \mathcal{L}_{div} \tag{12}$$

The overall training objective for the teacher model T_2 is defined as follows:

$$\mathcal{L}_{T2} = \lambda_{\text{cl}} \mathcal{L}_{\text{CL}} + \lambda_{\text{mse}} \mathcal{L}_{\text{MSE}} + \lambda_{\text{im}} \mathcal{L}_{\text{IM}} \tag{13}$$

where λ_{cl}, λ_{mse}, and λ_{im} represent the weighting factors for the contrastive loss, the MSE loss, and the information maximization loss, respectively. After updating the T_2 model by taking the gradient of the loss L_{T_2}, we then update T_2 by applying the exponential moving average of the student model S.

Stochastic Restoration for Mitigating Error Accumulation: To address error accumulation from inaccurate predictions due to distribution shifts, we adopt a stochastic restoration method introduced in [19], which preserves knowledge from the pretrained source model. This method mitigates catastrophic forgetting by combining the original and updated weights in the T_2 model after each gradient update. Consider a convolutional layer within the T_2 model with updated weights W_{t+1} after a gradient update at time step t. We apply a Bernoulli mask $M \sim \text{Bernoulli}(p)$, and update the weights as follows:

$$W_{t+1} = M \odot W_0 + (1 - M) \odot W_{t+1} \tag{14}$$

where p is restore probability, and M selectively restores the weights to their original values W_0, preventing the model from diverging too far from the source model.

3.4. Prediction Ensembling

In a typical teacher-student framework, the final output comes from the teacher model. Inspired by [2], we combine the outputs of both the student and T_2 models. The student model adapts quickly to the current domain, while the T_2 model provides generalized predictions across domains. This combination leverages their complementary strengths, improving prediction robustness and accuracy in dynamic environments. For a test sample x_t, the final prediction is:

$$y_t = f_{\theta_S}(x_t) + f_{\theta_{T_2}}(x_t) \tag{15}$$

Prior Correction: It is shown in [11], in continual test-time adaptation, the learned posterior $q(y|x)$ may deviate from the true posterior $p(y|x)$ due to domain shifts, leading to performance degradation. We adapt their strategy for the dual-teacher setting. Following [15], the authors corrected the posterior deviation to recover better performance by rescaling the learned posterior $q(y|x)$ $p(y|x) = q(y|x)\frac{p(y)}{q(y)}$

We perform prior correction by estimating the true class prior $p(y)$ as the sample mean of the current batch's softmax of the final prediction, as defined in equation 15, denoted $\hat{p}_t$, under the assumption that our learned prior is nearly uniform as a result of the information loss objective in equation 12. Finally, we adopt an adaptive smoothing scheme following [11]. to account for limited batch size as follows:

$$\bar{p}_t = \frac{\hat{p}_t + \gamma}{1 + \gamma N_c}, \tag{16}$$

where, γ is the smoothing factor, and N_c is the number of classes.

4. Result and Discussion

In this section, we show the dataset and the settings of domain adaptation.

4.1. Datasets

We assess our approach using diverse domain shifts, including artificial corruptions and natural variations. Building on the setup in [11], we employ the corruption benchmark on CIFAR10-C, CIFAR100-C, and ImageNet-C. These datasets feature 15 corruption types applied at five levels of severity to validation and test images, as described in [11].

- **ImageNet-C:** Contains 1,000 categories or classes, with 50 samples per category for each domain, resulting in 50,000 images per domain.
- **CIFAR100-C:** Comprises 100 categories, with 100 samples per category or class for each domain, yielding a total of 10,000 images per domain.
- **CIFAR10-C:** Consists of 10 classes, with 1,000 samples per class for each domain, amounting to 10,000 images per domain.

4.2. Benchmarks for Continual Test-Time Adaptation

All evaluations are conducted in an *online test-time adaptation (TTA)* setting, where predictions are updated and evaluated immediately. We introduce four distinct evaluation benchmarks to analyze the effectiveness of CTTA:

Continual Domains: Following [11], the model adapts sequentially across K domains $[D_1, D_2, \ldots, D_K]$ without prior knowledge of domain boundaries. For the corruption datasets, the sequence includes all 15 corruption types encountered at severity level 5.

Mixed Domains: In this setting of [11], the test data from different domains is randomly shuffled for adaptation. As a result, consecutive samples during adaptation are likely to come from different domains with different ratio.

Mixed after Continual TTA: Here, the domains are first encountered sequentially, as in the continual setting, but after this sequence, the data came from previously seen domains is shuffled randomly, creating a mixed configuration.

Cyclic Domains: We propose a new benchmark where the domain sequence is repeated after completing a cycle of subgroups categorized by similar corruption types, such as noise, blur, weather, or digital artifacts. In our experiments, we group corruptions into subgroups based on their characteristics: Subgroup 1 includes gaussian noise, shot noise, and impulse noise; Subgroup 2 consists of defocus blur, motion blur, and glass blur; Subgroup 3 contains snow, fog, and frost; Subgroup 4 covers brightness and contrast changes; and Subgroup 5 includes elastic transform, pixelate, and jpeg compression.

4.3. Implementation Details

We use WideResNet-28 (WRN-28) for CIFAR10-C, ResNeXt-29 for CIFAR100-C, and ResNet-50 for ImageNet-C as source model along with presenting the results of our method, **SloMo-Fast**, compared to various baselines in several settings, including continual test-time adaptation (CTTA), mixed domains, mixed after continual domains, and cyclic test-time adaptation.

4.4. Result for Continual Test Time Adaptation

We first evaluate the performance of different methods under the Continual TTA setting on CIFAR10-C, CIFAR100-C, and ImageNet-C benchmarks. Table 1 shows the online classification error rates (%) for each corruption type at the highest severity level (severity 5). For CIFAR10-C, **SloMo-Fast** achieves a mean error rate of **15.79±0.07**, outperforming other methods including state of the art method ROID 16.2. Similarly, for CIFAR100-C, **SloMo-Fast** leads with a mean error rate of **27.01±0.12**, where the state of the art method, ROID, achieves 29.3. For ImageNet-C, **SloMo-Fast** achieves a mean error rate of **54.2±0.10**, which is also the best among all the methods, compared to others showed in the Table 1.

Here, we present an additional assessment demonstrat-

Table 1. Online classification error rate (%) for the corruption benchmarks at the highest severity level 5 for the Continual TTA setting. For CIFAR10-C the results are evaluated on WideResNet-28, for CIFAR100-C on ResNeXt-29, and for ImageNet-C, ResNet-50 are used. We report the performance of each method averaged over 5 runs. *If all of the student parameters are updated otherwise only Batch normalization layers are updated, SloMo-Fast improves the error rates further for CIFAR 10C, CIFAR 100C and ImageNet-C.

Method	*Gaussian*	*shot*	*impulse*	*defocus*	*glass*	*motion*	*zoom*	*snow*	*frost*	*fog*	*bright.*	*contrast*	*elastic*	*pixelate*	*jpeg*	**Mean**
CIFAR10-C																
Source	72.3	65.7	72.9	46.9	54.3	34.8	42.0	25.1	41.3	26.0	9.3	46.7	26.6	58.4	30.3	43.5
TENT-cont.	25.0	20.3	29.0	13.8	31.7	16.2	14.1	18.6	17.6	17.4	10.8	15.6	24.3	19.7	25.1	20.0±1.19
RoTTA	30.3	55.5	70.0	23.8	44.1	20.7	21.0	22.7	16.0	9.4	27.7	27.0	58.6	29.2	33.4	19.3±0.07
CoTTA	24.2	21.9	26.5	12.0	27.9	12.7	10.7	15.2	14.6	12.8	7.9	11.2	18.5	14.0	18.1	16.5±0.16
ROID	23.7	18.7	26.4	11.5	28.1	12.4	10.1	14.7	14.3	12.0	7.5	9.3	19.8	14.5	20.3	16.2±0.05
SloMo-Fast	**22.6**	**19.0**	**24.9**	**13.0**	**25.0**	**14.0**	**12.3**	**15.0**	**14.7**	**13.5**	**10.1**	**12.5**	**17.4**	**13.3**	**16.3**	**16.2**± 0.11
SloMo-Fast*	**22.49**	**18.00**	**24.22**	**12.79**	**25.15**	**13.56**	**12.09**	**14.37**	**14.08**	**12.87**	**9.77**	**12.25**	**17.06**	**12.56**	**15.80**	**15.79** ± 0.07
CIFAR100-C																
Source	73.0	68.0	39.4	29.3	54.1	30.8	28.8	39.4	35.4	30.5	9.3	55.1	37.2	74.7	41.2	46.4
TENT-cont.	37.3	35.6	41.6	37.9	51.3	48.1	48.9	59.8	65.3	73.6	74.2	85.7	89.1	91.1	93.7	62.2±2.17
RoTTA	49.1	44.9	45.5	30.2	42.7	29.5	26.1	32.2	30.7	37.5	24.7	29.1	32.6	30.4	36.7	34.8±0.15
CoTTA	40.5	38.2	39.8	27.2	38.2	28.4	26.4	33.4	32.2	40.6	25.2	27.0	32.4	28.4	33.8	32.8±0.07
ROID	36.5	31.9	33.2	24.9	34.9	26.8	24.3	28.9	28.5	31.1	22.8	24.2	30.7	26.5	34.4	29.3 ±0.04
SloMo-Fast	**37.1**	**33.1**	**34.5**	**24.9**	**35.4**	**27.0**	**24.1**	**29.3**	**28.9**	**33.0**	**22.9**	**25.0**	**30.8**	**27.2**	**34.7**	**29.92**± 0.08
SloMo-Fast*	**37.82**	**32.76**	**33.36**	**26.25**	**31.26**	**26.90**	**24.33**	**26.81**	**26.50**	**28.44**	**23.37**	**24.33**	**26.19**	**24.29**	**27.01**	**27.97** ± 0.12
ImageNet-C																
Source	97.8	97.1	98.2	81.7	89.8	85.2	77.9	83.5	77.1	75.9	41.3	94.5	82.5	79.3	68.6	82.0
TENT-cont.	92.8	91.1	92.5	87.8	90.2	87.2	82.2	82.2	82.0	79.8	48.0	92.5	83.5	75.6	70.4	82.5±0.06
RoTTA	89.4	88.6	89.3	83.4	89.1	86.2	80.0	78.9	76.9	74.2	37.4	89.6	79.5	69.0	59.6	78.1±0.07
CoTTA	89.1	86.6	88.5	80.9	87.2	81.1	75.8	73.3	75.2	70.5	41.6	85.0	78.1	65.6	61.6	76.0±0.17
ROID	76.4	75.3	76.1	77.9	81.7	75.1	69.9	70.9	68.8	64.3	42.5	85.4	69.8	53.0	55.6	54.5±0.13
SloMo-Fast	**68.6**	**65.2**	**64.5**	**68.2**	**66.7**	**57.0**	**49.7**	**51.0**	**56.4**	**43.1**	**33.8**	**57.3**	**43.9**	**41.4**	**45.7**	**54.2±0.10**

Table 2. Online classification error rate (%) for the corruption benchmarks at the highest severity level 5 for the generalization experiments with mixed domains. For CIFAR10-C the results are evaluated on WideResNet-28, for CIFAR100-C on ResNeXt-29, and for ImageNet-C, ResNet-50 are used. We report the performance of each method averaged over 5 runs.

Method	*Gaussian*	*shot*	*impulse*	*defocus*	*glass*	*motion*	*zoom*	*snow*	*frost*	*fog*	*bright.*	*contrast*	*elastic*	*pixelate*	*jpeg*	**Mean**
CIFAR10-C																
Source	72.3	65.7	72.9	46.9	54.3	34.8	42.0	25.1	41.3	26.0	9.3	46.7	26.6	58.4	30.3	43.5
TENT-cont.	73.5	70.1	81.4	31.6	60.3	29.6	28.5	30.8	35.3	25.7	13.6	44.2	32.6	70.2	34.9	44.1 ± 3.82
CoTTA	38.7	36.0	56.1	36.0	36.8	32.3	31.0	19.9	17.6	27.2	11.7	52.6	30.5	35.8	25.7	32.5 ± 1.35
RoTTA	60.0	55.5	70.0	23.8	44.1	20.7	21.3	20.2	22.7	16.0	9.4	22.7	27.0	58.6	29.2	33.4 ± 0.15
RMT	42.8	39.7	55.0	28.5	38.6	26.5	25.9	19.6	18.9	20.6	12.2	27.3	26.9	56.9	25.9	31.0 ± 0.75
ROID	37.1	34.3	50.9	24.8	38.1	22.5	22.0	18.8	18.5	18.8	9.9	25.6	27.2	45.7	26.2	28.0 ± 0.12
SloMo-Fast	**33.4**	**32.1**	**53.9**	**26.4**	**35.0**	**22.7**	**23.4**	**17.9**	**17.8**	**19.8**	**11.4**	**30.1**	**25.9**	**46.4**	**23.4**	**28.0±0.06**
CIFAR100-C																
Source	73.0	68.0	39.4	29.3	54.1	30.8	28.8	39.5	45.8	50.3	29.5	55.1	37.2	74.7	41.2	46.4
TENT-cont.	95.6	95.2	89.2	72.8	82.9	74.4	72.3	78.0	79.7	84.7	71.0	88.5	77.8	96.8	78.7	82.5 ± 1.45
CoTTA	54.4	52.7	49.8	36.0	45.8	36.7	33.9	38.9	35.8	52.0	30.4	60.9	40.2	38.0	41.1	43.1 ± 0.05
RoTTA	65.0	62.3	39.3	33.4	50.0	34.2	32.6	36.6	36.5	45.0	26.4	41.6	40.6	89.5	48.5	45.4 ± 0.14
RMT	52.6	49.9	32.2	31.0	40.5	31.8	30.4	33.4	33.9	40.6	27.8	36.9	35.3	65.0	38.1	38.6 ± 0.15
ROID	40.5	38.0	32.0	28.1	40.5	29.7	27.6	34.1	33.8	41.3	28.7	38.7	34.3	39.7	38.5	35.0 ± 0.04
SloMo-Fast	**41.6**	**39.2**	**29.8**	**28.1**	**36.7**	**29.6**	**27.4**	**31.3**	**31.5**	**37.9**	**27.1**	**34.0**	**32.2**	**42.3**	**34.4**	**33.5** ± **0.02**

ing improved memory efficiency of our method. By updating only the batch normalization layers of the student model

Table 3. Online classification error rate (%) for the corruption benchmarks at the highest severity level 5 for the mixed after continual domains TTA setting. For CIFAR10-C the results are evaluated on WideResNet-28, for CIFAR100-C on ResNeXt-29, and for ImageNet-C, ResNet-50 is used. We report the performance of each method averaged over 5 runs.

Method	*Gaussian*	*shot*	*impulse*	*defocus*	*glass*	*motion*	*zoom*	*snow*	*frost*	*fog*	*bright.*	*contrast*	*elastic*	*pixelate*	*jpeg*	Mixed
CIFAR10-C																
Tent	24.69	19.89	28.48	13.14	31.24	16.87	14.05	18.99	18.35	16.97	11.42	17.20	25.55	19.72	25.23	39.35
CoTTA	24.06	21.82	25.77	11.78	27.53	21.61	10.27	15.06	13.95	12.57	7.57	10.87	18.11	13.57	17.80	26.76
Roid	23.66	18.74	26.59	11.63	28.25	12.59	9.96	14.46	13.95	11.78	7.39	9.34	19.73	14.45	20.57	27.37
SloMo-Fast	22.76	18.54	24.62	12.51	24.78	13.70	12.17	14.43	14.53	12.84	9.65	12.01	16.96	12.67	16.17	**21.34**
CIFAR100-C																
Tent	37.30	35.77	42.16	38.28	51.00	45.96	46.33	55.84	62.18	72.82	72.39	83.91	90.60	92.81	95.35	97.82
CoTTA	40.87	38.01	39.82	27.23	38.06	28.54	26.45	33.44	32.24	40.22	25.17	26.98	32.17	28.45	33.80	40.96
Roid	36.44	31.90	33.68	24.85	34.81	27.01	24.13	29.18	28.56	31.30	22.89	24.22	30.50	26.43	33.98	34.75
SloMo-Fast	37.84	32.92	33.18	26.68	31.61	27.10	24.85	26.58	26.26	28.45	23.64	24.34	26.57	24.51	27.17	**28.00**

Table 4. Our Proposed Cyclic TTA results of SloMo-Fast compared with existing methods on CIFAR10-C and CIFAR100-C for different domain groups. Each subgroup completes a cycle of seeing different test domains twice. (Gaussian, Shot, Impulse): Group 1, (Defocus, Glass, Motion, Zoom): Group 2, (Snow, Frost, Fog): Group 3, (Brightness, Contrast): Group 4, (Elastic, Pixelate, JPEG): Group 5. SloMo-Fast achieves the best performance across both datasets.

Method	Repetition	CIFAR100-C						CIFAR10-C					
		Group 1	Group 2	Group 3	Group 4	Group 5	Avg. Error	Group 1	Group 2	Group 3	Group 4	Group 5	Avg. Error
TENT	Cycle 1	38.28	31.14	32.93	25.04	34.09	32.29	23.66	16.95	15.22	9.07	20.09	17.47
	Cycle 2	47.88	37.12	37.93	25.18	38.95	37.41	23.66	16.95	15.22	9.07	20.09	16.64
	Avg.	43.08	34.13	35.43	25.11	36.52	34.85	23.66	16.95	15.22	9.07	20.09	17.06
COTTA	Cycle 1	36.52	29.43	30.98	23.56	32.75	30.96	23.16	14.98	15.57	10.01	20.63	17.23
	Cycle 2	44.67	34.69	35.93	23.97	36.39	34.69	23.15	15.54	15.15	9.86	20.06	16.28
	Avg.	39.60	32.06	33.46	23.77	34.57	33.70	23.15	15.26	15.36	9.94	20.35	16.75
ROID	Cycle 1	33.94	27.58	30.11	24.09	31.20	29.38	22.16	15.52	13.55	8.48	18.44	16.08
	Cycle 2	32.43	28.31	29.29	23.21	30.55	28.52	22.16	15.52	13.55	8.48	18.44	15.17
	Avg.	**33.18**	27.95	29.70	23.65	30.87	28.95	22.16	15.52	13.55	**8.48**	18.44	15.63
SloMo-Fast	Cycle 1	33.29	27.02	26.96	24.84	25.79	**27.98**	20.62	15.21	13.09	10.23	14.02	**14.89**
	Cycle 2	33.29	27.02	26.96	24.84	25.79	**27.18**	20.62	15.21	13.09	10.23	14.02	**14.38**
	Avg.	33.29	**27.02**	**26.96**	24.84	**25.79**	**27.58**	**20.62**	**15.21**	**13.09**	10.23	**14.02**	**14.63**

during adaptation, we achieve a 98% reduction in trainable parameters compared to fine-tuning the entire model. Notably, SloMo-Fast outperforms state-of-the-art methods on CIFAR10-C and ImageNet-C with 16.2% and 54.2% error rate respectively, even when adapting only the batch normalization layers.

4.5. Results for Mixed Domains

Table 2 presents the results under mixed domain setting. In this setup, the models are trained on mixed data, with each batch incorporating samples from various corruptions. From Table 2, we observe that **SloMo-Fast** continues to outperform other methods in this setting. For CIFAR10-C, **SloMo-Fast** achieves a mean error rate of **28.0±0.06**, which is lower than that of other existing methods. Similarly, for CIFAR100-C, **SloMo-Fast** achieves a mean error rate of **33.5±0.02**, which is also better than the current best ROID of 33.5% error rate. Experimental results demonstrate the robustness of our method to distributional shifts, even within a single batch, showcasing its effectiveness in handling diverse corruptions

4.6. Results for Mixed After Continual Domains

In the Mixed After Continual Domains TTA setting, we evaluate adaptation performance following the continual learning phase. As shown in Table 3, **SloMo-Fast** consistently achieves the lowest error rates across all datasets, significantly outperforming existing methods. For example, on CIFAR10-C, **SloMo-Fast** achieves a mean error rate of **21.34%**, compared to the second-best result of 27.37%. Similarly, on CIFAR100-C, it achieves a mean error rate of **28.01**, surpassing ROID's 34.75%. These results indicate that SloMo-Fast effectively retains knowledge of previously seen domains, whereas other methods often struggle with long-term forgetting of domain-specific knowledge.

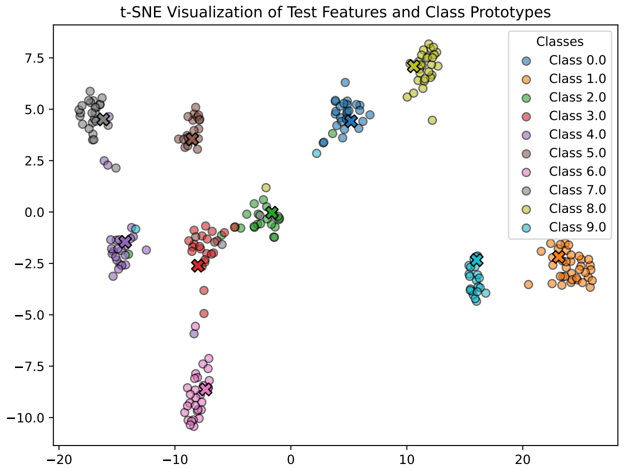

Figure 3. t-SNE visualization of test feature representations (○) and class prototypes (×): The plot demonstrates clear class separation, indicating effective learning of distinct feature representations.

4.7. Results for Cyclic Test-Time Adaptation

We further investigate how our method retains long-term domain-specific knowledge under cyclic test-time adaptation (Cyclic TTA) conditions, as outlined in Table 4. Although detailed results for this specific setting are not included in the current tables, trends observed in previous experiments indicate that **SloMo-Fast** would likely maintain its superior performance in terms of error rates due to its robust continual adaptation capabilities.

In this setup, we consider five groups of domains and repeat the test-time adaptation process in the same sequential order after completing one full cycle. Interestingly, we observe that during the second occurrence of a previously encountered subgroup of domains, SloMo-Fast adapts significantly faster than other methods. This can be attributed to its generalized pseudo-prototypes, which effectively consolidate knowledge from all past domains. Consequently, during repetition, our method leverages this stored knowledge for rapid adaptation and more accurate prediction, resulting in superior performance on repeated domains compared to competing methods.

4.8. Qualitative Results: t-SNE Visualization

Finally, to visualize the effectiveness of our method, we provide t-SNE plots of the feature space at final stage of adaptation in Figure 3. The t-SNE visualization for **SloMo-Fast** shows that the learned representations are well-clustered and exhibit clear separation between the different classes, even under severe corruption conditions.

5. Conclusion

In this paper, we presented SloMo-Fast, a dual-teacher framework for Continual Test-Time Adaptation (CTTA) that eliminates the need for source data while enhancing adaptability, generalization, and computational efficiency. By leveraging two complementary teachers, the Fast-Teacher (T1) and Slow-Teacher (T2), SloMo-Fast adapts to new domains rapidly while maintaining robust generalization to previously encountered domains. Our approach utilizes class-wise prototypes and contrastive learning to refine the Slow-Teacher's representations, and batch normalization updates significantly reduce the computational complexity. Through extensive experiments on CIFAR-10C, CIFAR-100C, and ImageNet-C, we demonstrated that SloMo-Fast outperforms existing CTTA methods across various domain adaptation scenarios. Additionally, we validated the framework's performance in real-world settings, including repetitive domain shifts and mixed-domain scenarios, establishing new benchmarks for generalization and robustness under complex conditions. Our work provides a significant step forward in the development of source-free, continual adaptation methods, and opens up new avenues for applying CTTA in privacy-sensitive and resource-constrained environments.

References

[1] Dian Chen, Dequan Wang, Trevor Darrell, and Sayna Ebrahimi. Contrastive test-time adaptation. In *CVPR*, pages 295–305, 2022. 3

[2] Mario Döbler, Robert A. Marsden, and Bin Yang. Robust mean teacher for continual and gradual test-time adaptation. In *CVPR*, pages 7704–7714, 2023. 1, 3, 4, 6

[3] Neerav Karani, Ertunc Erdil, Krishna Chaitanya, and Ender Konukoglu. Test-time adaptable neural networks for robust medical image segmentation. *Medical Image Analysis*, 68: 101907, 2021. 1

[4] Daeun Lee, Jaehong Yoon, and Sung Ju Hwang. BECoTTA: Input-dependent online blending of experts for continual test-time adaptation. In *ICML*, pages 27072–27093, 2024. 3

[5] Jonghyun Lee, Dahuin Jung, Saehyung Lee, Junsung Park, Juhyeon Shin, Uiwon Hwang, and Sungroh Yoon. Entropy is not enough for test-time adaptation: From the perspective of disentangled factors. In *ICLR*, 2024. 1, 3

[6] Jae-Hong Lee and Joon-Hyuk Chang. Continual momentum filtering on parameter space for online test-time adaptation. In *ICLR*, 2024. 1, 3

[7] Xinyao Li, Zhekai Du, Jingjing Li, Lei Zhu, and Ke Lu. Source-free active domain adaptation via energy-based locality preserving transfer. In *Proceedings of the 30th ACM international conference on multimedia*, pages 5802–5810, 2022. 6

[8] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for un-

supervised domain adaptation. In *International conference on machine learning*, pages 6028–6039. PMLR, 2020. 6

[9] Hong Liu, Mingsheng Long, Jianmin Wang, and Yu Wang. Learning to adapt to evolving domains. In *Advances in Neural Information Processing Systems*, pages 22338–22348. Curran Associates, Inc., 2020. 1

[10] Jiaming Liu, Senqiao Yang, Peidong Jia, Renrui Zhang, Ming Lu, Yandong Guo, Wei Xue, and Shanghang Zhang. ViDA: Homeostatic visual domain adapter for continual test time adaptation. In *ICLR*, 2024. 1, 3

[11] Robert A Marsden, Mario Döbler, and Bin Yang. Universal test-time adaptation through weight ensembling, diversity weighting, and prior correction. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2555–2565, 2024. 2, 3, 7

[12] Robert A. Marsden, Mario Döbler, and Bin Yang. Introducing intermediate domains for effective self-training during test-time. In *IJCNN*, pages 1–10, 2024. 1, 3

[13] Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Yaofo Chen, Shijian Zheng, Peilin Zhao, and Mingkui Tan. Efficient test-time model adaptation without forgetting. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, pages 16888–16905. PMLR, 2022. 1, 2, 3

[14] Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Zhiquan Wen, Yaofo Chen, Peilin Zhao, and Mingkui Tan. Towards stable test-time adaptation in dynamic wild world. In *ICLR*, 2023. 3

[15] Amelie Royer and Christoph H Lampert. Classifier adaptation at prediction time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1401–1409, 2015. 7

[16] Junha Song, Jungsoo Lee, In So Kweon, and Sungha Choi. EcoTTA: Memory-efficient continual test-time adaptation via self-distilled regularization. In *CVPR*, pages 11920–11929, 2023. 1, 3

[17] Jiaxu Tian and Fan Lyu. Parameter-selective continual test-time adaptation. In *arXiv preprint arXiv:2407.02253*, 2024. 1, 3

[18] Dequan Wang, Evan Shelhamer, Shaoteng Liu, B. Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *ICLR*, 2021. 1, 2

[19] Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. Continual test-time domain adaptation. In *CVPR*, pages 7191–7201, 2022. 1, 2, 3, 4, 6

[20] Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 322–330, 2019. 4

[21] Yanshuo Wang, Jie Hong, Ali Cheraghian, Shafin Rahman, David Ahmedt-Aristizabal, Lars Petersson, and Mehrtash Harandi. Continual test-time domain adaptation via dynamic sample selection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1701–1710, 2024. 1, 2

[22] Yeonguk Yu, Sungho Shin, Seunghyeok Back, Minhwan Ko, Sangjun Noh, and Kyoobin Lee. Domain-specific block selection and paired-view pseudo-labeling for online test-time adaptation. In *CVPR*, pages 22723–22732, 2024. 1, 2, 3

[23] Longhui Yuan, Binhui Xie, and Shuang Li. Robust test-time adaptation in dynamic scenarios. In *CVPR*, pages 15922–15932, 2023. 1, 2, 3

[24] Marvin Zhang, Sergey Levine, and Chelsea Finn. MEMO: test time robustness via adaptation and augmentation. In *NeurIPS*, pages 38629–38642, 2022. 1, 3

SloMo-Fast: Slow-Momentum and Fast-Adaptive Teachers for Source-Free Continual Test-Time Adaptation

Supplementary Material

6. Supplementary Experimental Results

6.1. Ablation Study on Losses Applied to T2

The results of the ablation study are summarized in Tables 5 and 6, which evaluate the effect of different loss functions applied to the T_2 model on the CIFAR10-to-CIFAR10C and CIFAR100-to-CIFAR100C online continual test-time adaptation tasks, respectively, evaluations use the WideResNet-28 and ResNeXt-29 model under the highest corruption severity level (level 5). The classification error rates (%) are reported for 15 corruption types, along with the mean error rate as a summary.

In the CIFAR10-to-CIFAR10C task (Table 5), the T_2 model trained with all three losses—mean squared error (MSE), information maximization (IM), and contrastive loss (CL)—achieves the lowest mean error rate of 14.88%. This indicates the strong performance of the full configuration under severe corruption scenarios. Removing the contrastive loss (✓ MSE, ✓ IM) slightly increases the mean error rate to 16.04%, suggesting that CL contributes significantly to robustness. Excluding the information maximization loss (✓ MSE, ✓ CL) results in a mean error rate of 16.17%, highlighting the importance of IM in the adaptation process. When MSE is excluded (✓ IM, ✓ CL), the mean error rate is slightly better at 15.89%, reflecting a strong interaction between IM and CL, even in the absence of MSE.

For the CIFAR100-to-CIFAR100C task (Table 6), similar trends are observed. The T_2 model trained with all three losses achieves the lowest mean error rate of 28.00%. Removing CL (✓ MSE, ✓ IM) increases the mean error rate to 28.57%, demonstrating the importance of CL in enhancing robustness. Excluding IM (✓ MSE, ✓ CL) leads to a mean error rate of 28.35%, showing the critical role of IM in the adaptation process. Finally, removing MSE (✓ IM, ✓ CL) results in a mean error rate of 28.23%, again underscoring the synergy between IM and CL.

The results from both CIFAR10-to-CIFAR10C and CIFAR100-to-CIFAR100C tasks consistently highlight the benefits of integrating all three losses in the T_2 model. This combination achieves the lowest error rates across diverse corruption types, validating the effectiveness of the proposed design for continual test-time adaptation.

6.2. Ablation Study on Prior Correction and Stochastic Restoration

The results of the ablation study are presented in Tables 7 and 8, which evaluate the effect of Prior Correction (PC) applied to the model output and Stochastic Restoration (ST) of the T_2 model on the CIFAR10-to-CIFAR10C and CIFAR100-to-CIFAR100C online continual test-time adaptation tasks, respectively. The evaluations are conducted using the WideResNet-28 and ResNeXt-29 model under the highest corruption severity level (level 5). Classification error rates (%) are reported for 15 corruption types, along with the mean error rate as an overall summary.

In the CIFAR10-to-CIFAR10C task (Table 7), applying PC to the output and using Stochastic Restoration of the T_2 model achieves the lowest mean error rate of 14.88%. This result demonstrates the effectiveness of combining these techniques for robust adaptation. When Stochastic Restoration is removed, and only PC is applied to the output, the mean error rate increases to 16.11%, indicating the critical role of Stochastic Restoration in enhancing the model's robustness under severe corruptions. Conversely, removing PC while retaining Stochastic Restoration results in a mean error rate of 15.78%, suggesting that Prior Correction also significantly contributes to improved performance. These findings highlight the complementary roles of PC and ST in enhancing the adaptation capabilities of the T_2 model.

In the CIFAR100-to-CIFAR100C task (Table 8), a similar trend is observed. Applying PC to the output alongside Stochastic Restoration of the T_2 model achieves the lowest mean error rate of 28.00%. Removing Stochastic Restoration while retaining PC increases the mean error rate to 28.48%, demonstrating the importance of Stochastic Restoration for handling severe corruptions. On the other hand, using only Stochastic Restoration without PC results in a mean error rate of 28.08%, highlighting the significant role of Prior Correction in reducing classification errors.

The results from both CIFAR10-to-CIFAR10C and CIFAR100-to-CIFAR100C tasks consistently demonstrate that the combination of Prior Correction and Stochastic Restoration leads to the most effective adaptation.

6.3. Effect of Consistency Loss

Tables 9 and 10 present the classification error rates (%) for the CIFAR10-to-CIFAR10C and CIFAR100-to-CIFAR100C online continual test-time adaptation tasks, respectively. These results evaluate the effect of applying a consistency loss between the student model and teacher: T_1, T_2, and T_1 with data augmentation input($T_1(aug)$). The evaluations are conducted using WideResNet-28 for CIFAR10C and ResNeXt-29 for CIFAR100C under the largest corruption severity level (level 5). Classification error rates

Table 5. Evaluating the effect of our proposed loss on T_2, evaluated on the CIFAR10-to-CIFAR10C online continual test-time adaptation task. Results are reported as classification error rates (%) using a WideResNet-28 model with corruption severity level 5. Mean squared error (MSE), information maximization (IM), and contrastive loss (CL).

Design Choices			Error Rate (%)															
MSE	IM	CL	Gaussian	shot	impulse	defocus	glass	motion	zoom	snow	frost	fog	brightness	contrast	elastic	pixelate	jpeg	Mean
✓		✓	22.55	18.47	25.11	13.37	24.87	14.05	12.50	14.56	14.34	13.42	10.04	12.40	17.28	13.11	16.50	16.17
✓	✓		23.23	18.98	25.44	12.04	25.53	13.45	11.97	14.47	14.39	12.75	9.48	12.18	17.27	12.78	16.70	16.04
	✓	✓	22.67	18.50	24.65	13.03	24.64	13.65	12.03	14.36	14.11	13.18	9.60	12.14	17.23	12.66	15.95	15.89
✓	✓	✓	22.45	18.51	24.78	11.93	24.69	12.23	10.10	12.73	12.97	11.47	7.51	9.96	16.24	11.70	15.95	14.88

Table 6. Evaluating the effect of our proposed loss on T_2, evaluated on the CIFAR100-to-CIFAR100C online continual test-time adaptation task. Results are reported as classification error rates (%) using a ResNeXt-29 model with corruption severity level 5. Mean squared error (MSE), information maximization (IM), and contrastive loss (CL).

Design Choices			Error Rate (%)															
MSE	IM	CL	Gaussian	shot	impulse	defocus	glass	motion	zoom	snow	frost	fog	brightness	contrast	elastic	pixelate	jpeg	Mean
✓	✓		38.19	33.06	33.95	26.76	32.43	27.63	25.03	27.44	26.68	29.44	23.82	24.86	26.52	24.92	27.81	28.57
✓		✓	38.92	33.29	33.40	26.60	32.09	27.39	24.82	26.73	27.07	28.36	23.65	24.23	26.74	24.76	27.25	28.35
	✓	✓	38.04	32.69	33.16	26.83	31.59	26.91	24.86	27.03	26.93	28.25	23.76	24.67	26.65	24.80	27.24	28.23
✓	✓	✓	37.91	32.51	33.23	26.54	31.42	26.81	24.46	26.58	26.34	28.40	23.51	24.60	26.35	24.20	27.11	28.00

are reported for 15 corruption types, along with the mean error rate as a summary.For CIFAR10-C, the best results are achieved by incorporating the consistency loss between the student predictions and the predictions from both T_1 and T_2. For CIFAR100-C, the best performance is obtained by using the consistency loss between the student predictions and the predictions from T_2 and T_1 with augmented samples.

6.4. CTTA Under Cyclic Domain Settings

In continual test-time adaptation, catastrophic forgetting occurs when the model forgets previously learned knowledge while adapting to new domains. To address this, we propose a second teacher model that learns more generalized knowledge compared to the primary teacher model, which is more adapted to the current domain. This helps retain critical knowledge from past domains while enabling adaptation to new ones, mitigating the risk of forgetting. To validate our approach, we conduct an ablation study in cyclic domain settings, where domains are grouped and presented in a cycle. This setup allows us to compare the effectiveness of various methods designed to tackle catastrophic forgetting. Table 11-17 presents the detailed results on the newly proposed benchmark CTTA under cyclic domain settings.

The experimental results demonstrate that our method improves performance when domains repeat, indicating that it retains past knowledge to some extent while adapting to new domains. Specifically, our approach achieves lower error rates compared to state-of-the-art methods. In CIFAR10-C, our method achieves an error rate of 14.89% in Cycle 1 and 14.38% in Cycle 2, showing improvement in error rate as domains are repeated. In contrast, TENT[18], which does not specifically address continual domain adaptation, results in higher error rates, with Cycle 1 at 17.47% and Cycle 2 at 16.64%. While COTTA[19] shows some improvement initially, it does not exhibit reduction in error rates when domains are repeated. ROID[11], on the other hand, shows limited improvement under cyclic domain settings. Compared to state-of-the-art methods, our method demonstrates better retention of past knowledge, leading to more stable performance across cyclic domains. These results highlight the effectiveness of our approach in mitigating catastrophic forgetting and adapting to domain shifts, outperforming existing methods in terms of reduced error rates.

6.5. Catastrophic Fogetting

The figures illustrate the performance of different CTTA methods, including SloMo-Fast, on the CIFAR10-C benchmark, highlighting challenges like catastrophic forgetting and the ability to retain long-term knowledge.

In the standard CTTA setting, as shown in 4, the SloMo-Fast method achieves consistently low error rates, with a mean error of **15.79%**, outperforming CoTTA (**16.5%**) and ROID (**16.2%**). This demonstrates SloMo-Fast's superior adaptability while avoiding performance degradation seen in other methods.

For mixed domain settings, as shown in 5, SloMo-Fast maintains the best mean error rate of **28.0%**, compared to CoTTA (**32.5%**) and ROID (**28.0%**). This highlights SloMo-Fast's ability to handle mixed corruption scenarios effectively.

When evaluating performance in a mixed-after-continual setting, as in 6, SloMo-Fast achieves the lowest mean error rate of **21.34%**, significantly outperforming ROID (**27.37%**) and CoTTA (**26.76%**), showcasing its resilience to catastrophic forgetting.

Table 7. Classification error rate (%) for the CIFAR10-to-CIFAR10C online continual test-time adaptation task. Results are evaluated using the WideResNet-28 model with corruption severity level 5. Prior Correction (PC) is applied to the model output, and Stochastic Restoration (ST) is applied to the T_2 model.

Design Choices		Error Rate (%)															
PC	ST	Gaussian	shot	impulse	defocus	glass	motion	zoom	snow	frost	fog	brightness	contrast	elastic	pixelate	jpeg	Mean
✓		22.61	17.88	23.86	13.85	24.71	14.98	12.44	14.42	14.33	13.77	10.38	12.54	17.31	12.89	15.71	16.11
	✓	22.58	18.57	24.48	12.82	24.77	13.39	11.76	14.45	14.01	13.02	9.68	11.74	17.08	12.52	15.89	15.78
✓	✓	22.45	18.51	24.78	11.93	24.69	12.23	10.10	12.73	12.97	11.47	7.51	9.96	16.24	11.70	15.95	14.88

Table 8. Classification error rate (%) for the CIFAR100-to-CIFAR100C online continual test-time adaptation task. Results are evaluated using the ResNeXt-29 model with corruption severity level 5. Prior Correction (PC) is applied to the model output, and Stochastic Restoration (ST) is applied to the T_2 model.

Design Choices		Error Rate (%)															
PC	ST	Gaussian	shot	impulse	defocus	glass	motion	zoom	snow	frost	fog	brightness	contrast	elastic	pixelate	jpeg	Mean
✓		37.36	32.60	33.72	27.43	32.28	27.51	25.10	27.20	26.89	29.37	24.01	24.65	27.00	24.61	27.49	28.48
	✓	37.96	32.52	33.07	26.79	31.51	27.20	24.76	26.65	26.42	28.33	23.49	24.50	26.47	24.51	27.00	28.08
✓	✓	37.91	32.51	33.23	26.54	31.42	26.81	24.46	26.58	26.34	28.40	23.51	24.60	26.35	24.20	27.11	28.00

In the cyclic domain adaptation scenario, as shown in 7, SloMo-Fast exhibits stable performance, maintaining an average error rate of **14.63%** across repeated domains, compared to ROID's **15.63%**. This demonstrates SloMo-Fast's ability to retain previously learned knowledge without succumbing to forgetting, a common issue in ROID and CoTTA.

Overall, the results validate SloMo-Fast as a robust solution for CTTA, capable of preserving long-term domain knowledge while achieving state-of-the-art performance.

Table 9. Classification error rate (%) for the standard CIFAR10-to-CIFAR10C online continual test-time adaptation task. Results are evaluated on WideResNet-28 with the largest corruption severity level 5. The consistency loss calculated between student and teachers. T_1 indicates consistency loss calculated between student and teacher 1, T_2 indicates consistency loss calculated between student and teacher 2, $T_1(aug)$ indicates consistency loss calculated between student and teacher 1 where the input of teacher is augementation of input images.

Design Choices			Error Rate (%)															
T1	T_2	T1(aug)	Gaussian	shot	impulse	defocus	glass	motion	zoom	snow	frost	fog	brightness	contrast	elastic	pixelate	jpeg	Mean
✓	✓		22.71	18.12	24.23	12.83	25.57	13.58	11.57	15.07	14.24	13.30	9.74	12.21	17.03	13.34	15.72	15.95
✓		✓	22.70	18.75	25.45	12.98	25.74	14.38	12.37	15.39	15.15	13.40	10.31	13.34	17.83	13.42	17.15	16.56
	✓	✓	22.64	18.22	24.89	13.22	25.14	14.60	12.27	14.56	14.65	13.12	10.22	12.34	17.66	12.92	16.43	16.19

Table 10. Classification error rate (%) for the standard CIFAR100-to-CIFAR100C online continual test-time adaptation task. Results are evaluated on ResNeXt-29 with the largest corruption severity level 5. The consistency loss calculated between student and teachers. T_1 indicates consistency loss calculated between student and teacher 1, T_2 indicates consistency loss calculated between student and teacher 2, $T_1(aug)$ indicates consistency loss calculated between student and teacher 1 where the input of teacher is augementation of input images.

Design Choices			Error Rate (%)															
T1	T_2	T1(aug)	Gaussian	shot	impulse	defocus	glass	motion	zoom	snow	frost	fog	brightness	contrast	elastic	pixelate	jpeg	Mean
✓	✓		38.27	32.97	33.91	26.32	32.02	27.01	24.76	27.31	26.68	28.91	23.74	24.04	26.51	24.34	27.11	28.26
✓		✓	38.18	33.13	33.97	27.13	32.52	27.48	25.41	27.78	27.05	29.06	24.24	25.61	27.74	25.60	28.62	28.90
	✓	✓	37.36	32.72	33.05	26.32	31.67	27.22	24.79	26.94	26.35	28.38	23.64	24.74	26.78	24.62	27.12	28.11

Table 11. Detailed Evaluation Results for TENT on CIFAR10-C under Cyclic Domain Settings

Method	Subgroup	Cycle 1			Cycle 2		
		Domain	Error (%)	Avg	Domain	Error (%)	Avg
Tent	Subgroup 1 (Noise)	gaussian	23.42	23.66	gaussian	24.87	23.66
		shot	21.98		shot	24.37	
		impulse	25.58		impulse	21.74	
	Subgroup 2 (Blur)	defocus	11.81	16.95	defocus	11.81	16.95
		glass	29.76		glass	29.76	
		motion	14.01		motion	14.01	
		zoom	12.23		zoom	12.23	
	Subgroup 3 (Weather)	snow	16.34	15.22	snow	14.98	15.22
		frost	15.94		frost	15.44	
		fog	14.10		fog	14.55	
	Subgroup 4 (Error1)	brightness	7.91	9.07	brightness	7.67	9.07
		contrast	10.81		contrast	9.89	
	Subgroup 5 (Error2)	elastic	22.11	20.09	elastic	20.55	20.09
		pixel	16.22		pixel	15.54	
		jpeg	23.77		jpeg	22.33	
		Cycle 1 Avg: 17.47%			Cycle 2 Avg: 16.64%		

Table 12. Detailed Evaluation Results for TENT on CIFAR100-C under Cyclic Domain Settings

Method	Subgroup	Cycle 1			Cycle 2		
		Domain	Error (%)	Avg	Domain	Error (%)	Avg
TENT	Subgroup 1 (Noise)	gaussian	38.12	38.28	gaussian	47.32	47.88
		shot	38.45		shot	48.23	
		impulse	38.27		impulse	48.09	
	Subgroup 2 (Blur)	defocus	30.87	31.14	defocus	37.00	37.12
		glass	31.19		glass	36.78	
		motion	30.75		motion	37.39	
		zoom	31.27		zoom	37.50	
	Subgroup 3 (Weather)	snow	33.05	32.93	snow	36.88	37.93
		frost	33.21		frost	36.32	
		fog	32.55		fog	38.58	
	Subgroup 4 (Error1)	brightness	25.32	25.04	brightness	24.95	25.18
		contrast	24.76		contrast	25.41	
	Subgroup 5 (Error2)	elastic	33.72	34.09	elastic	39.05	38.95
		pixel	34.56		pixel	39.14	
		jpeg	33.98		jpeg	38.66	
		Cycle 1 Avg: 32.29%			Cycle 2 Avg: 37.41%		

Table 13. Detailed Evaluation Results for COTTA on CIFAR100-C under Cyclic Domain Settings

Method	Subgroup	Cycle 1			Cycle 2		
		Domain	Error (%)	Avg.	Domain	Error (%)	Avg.
COTTA	Subgroup 1 (Noise)	gaussian	36.14	36.52	gaussian	44.23	44.67
		shot	36.84		shot	44.98	
		impulse	36.57		impulse	44.81	
	Subgroup 2 (Blur)	defocus	29.12	29.43	defocus	34.45	34.69
		glass	29.55		glass	34.08	
		motion	28.99		motion	34.72	
		zoom	29.36		zoom	34.51	
	Subgroup 3 (Weather)	snow	31.25	30.98	snow	35.45	35.93
		frost	30.84		frost	35.21	
		fog	30.85		fog	37.13	
	Subgroup 4 (Error1)	brightness	23.28	23.56	brightness	23.95	23.97
		contrast	23.84		contrast	24.09	
	Subgroup 5 (Error2)	elastic	32.48	32.75	elastic	36.54	36.39
		pixel	32.88		pixel	36.19	
		jpeg	32.89		jpeg	36.44	
		Cycle 1 Avg: 30.96%			Cycle 2 Avg: 34.69%		

Table 14. Detailed Evaluation Results for ROID on CIFAR10-C under Cyclic Domain Settings

Method	Subgroup	Cycle 1			Cycle 2		
		Domain	Error (%)	Avg	Domain	Error (%)	Avg
ROID	Subgroup 1 (Noise)	gaussian	23.94	22.16	gaussian	20.62	22.16
		shot	22.41		shot	21.00	
		impulse	20.62		impulse	24.87	
	Subgroup 2 (Blur)	defocus	10.52	15.52	defocus	10.52	15.52
		glass	28.20		glass	28.20	
		motion	12.06		motion	12.06	
		zoom	10.06		zoom	10.06	
	Subgroup 3 (Weather)	snow	15.12	13.55	snow	14.01	13.55
		frost	14.41		frost	13.79	
		fog	12.04		fog	11.92	
	Subgroup 4 (Error1)	brightness	7.76	8.48	brightness	7.37	8.48
		contrast	9.61		contrast	9.17	
	Subgroup 5 (Error2)	elastic	21.08	18.44	elastic	19.16	18.44
		pixel	15.22		pixel	14.51	
		jpeg	20.62		jpeg	20.02	
		Cycle 1 Avg: 16.08%			Cycle 2 Avg: 15.17%		

Table 15. Detailed Evaluation Results for ROID on CIFAR100-C under Cyclic Domain Settings

Method	Subgroup	Cycle 1			Cycle 2		
		Domain	Error (%)	Avg	Domain	Error (%)	Avg
ROID	Subgroup 1 (Noise)	gaussian	32.34	32.53	gaussian	33.67	33.83
		shot	33.12		shot	34.58	
		impulse	32.12		impulse	33.25	
	Subgroup 2 (Blur)	defocus	27.12	26.44	defocus	28.44	28.31
		glass	25.67		glass	27.23	
		motion	26.22		motion	28.23	
		zoom	26.65		zoom	29.34	
	Subgroup 3 (Weather)	snow	28.77	30.11	snow	29.29	29.70
		frost	28.06		frost	28.77	
		fog	33.50		fog	31.03	
	Subgroup 4 (Error1)	brightness	23.64	24.09	brightness	22.46	23.21
		contrast	24.53		contrast	23.95	
	Subgroup 5 (Error2)	elastic	32.08	31.20	elastic	30.86	30.55
		pixel	27.28		pixel	26.90	
		jpeg	34.23		jpeg	33.88	
		Cycle 1 Avg: 29.38%			Cycle 2 Avg: 28.52%		

Table 16. Detailed Evaluation Results for SloMo-Fast on CIFAR10-C under Cyclic Domain Settings

Method	Subgroup	Cycle 1			Cycle 2		
		Domain	Error (%)	Avg	Domain	Error (%)	Avg
SloMo-Fast	Subgroup 1 (Noise)	gaussian	21.65	20.62	gaussian	20.34	20.62
		shot	19.78		shot	21.12	
		impulse	20.43		impulse	20.40	
	Subgroup 2 (Blur)	defocus	15.03	14.79	defocus	14.34	15.21
		glass	14.65		glass	15.92	
		motion	17.07		motion	16.27	
		zoom	12.41		zoom	13.90	
	Subgroup 3 (Weather)	snow	13.72	13.21	snow	13.10	13.09
		frost	13.42		frost	13.23	
		fog	12.48		fog	12.58	
	Subgroup 4 (Error1)	brightness	9.33	10.24	brightness	9.17	10.23
		contrast	11.15		contrast	11.26	
	Subgroup 5 (Error2)	elastic	15.85	13.96	elastic	15.64	14.02
		pixel	11.56		pixel	11.98	
		jpeg	14.46		jpeg	14.61	
		Cycle 1 Avg: 14.89%			Cycle 2 Avg: 14.38%		

Table 17. Detailed Evaluation Results for SloMo-Fast on CIFAR100-C under Cyclic Domain Settings

Method	Subgroup	Cycle 1			Cycle 2		
		Domain	Error (%)	Avg	Domain	Error (%)	Avg
SloMo-Fast	Subgroup 1 (Noise)	gaussian	34.12	33.29	gaussian	34.82	33.29
		shot	32.54		shot	31.89	
		impulse	33.12		impulse	33.84	
	Subgroup 2 (Blur)	defocus	27.01	27.02	defocus	27.03	27.02
		glass	26.97		glass	27.04	
		motion	26.89		motion	27.12	
		zoom	27.22		zoom	26.89	
	Subgroup 3 (Weather)	snow	27.00	26.96	snow	26.98	26.96
		frost	26.90		frost	27.01	
		fog	26.98		fog	26.89	
	Subgroup 4 (Error1)	brightness	24.74	24.84	brightness	24.41	24.84
		contrast	25.05		contrast	25.18	
	Subgroup 5 (Error2)	elastic	25.72	25.79	elastic	25.67	25.79
		pixel	24.79		pixel	24.97	
		jpeg	26.84		jpeg	26.73	
		Cycle 1 Avg: 27.98%			Cycle 2 Avg: 27.18%		

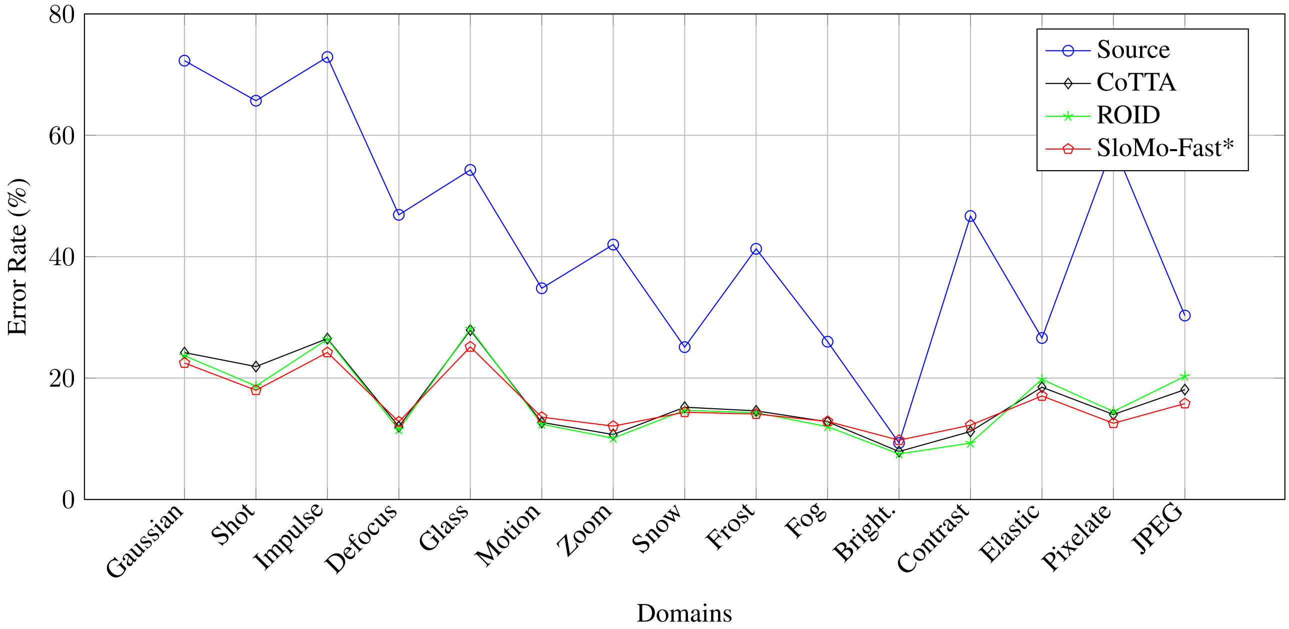

Figure 4. CTTA Error rates (%) for Source (blue), CoTTA (black), ROID (green), and PA (red) across domains in the CIFAR10-C benchmark.

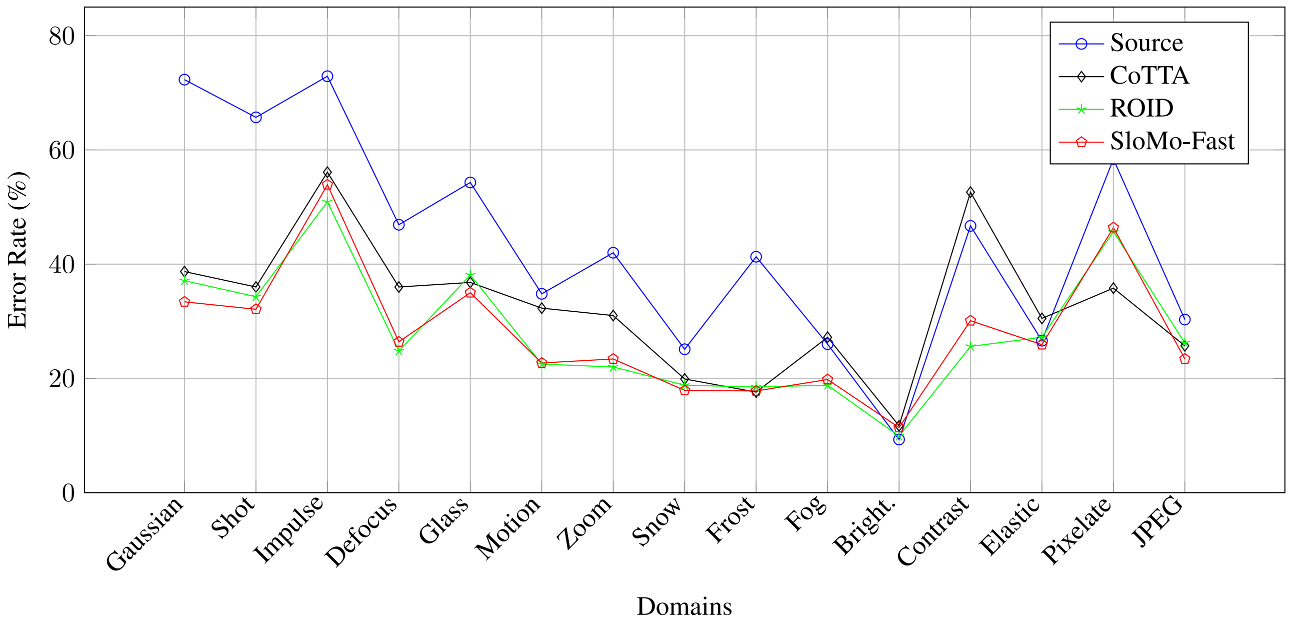

Figure 5. Mixed TTA Error rates (%) for Source (blue), CoTTA (black), ROID (green), and PA (red) methods across domains in the CIFAR10-C benchmark for mixed domains.

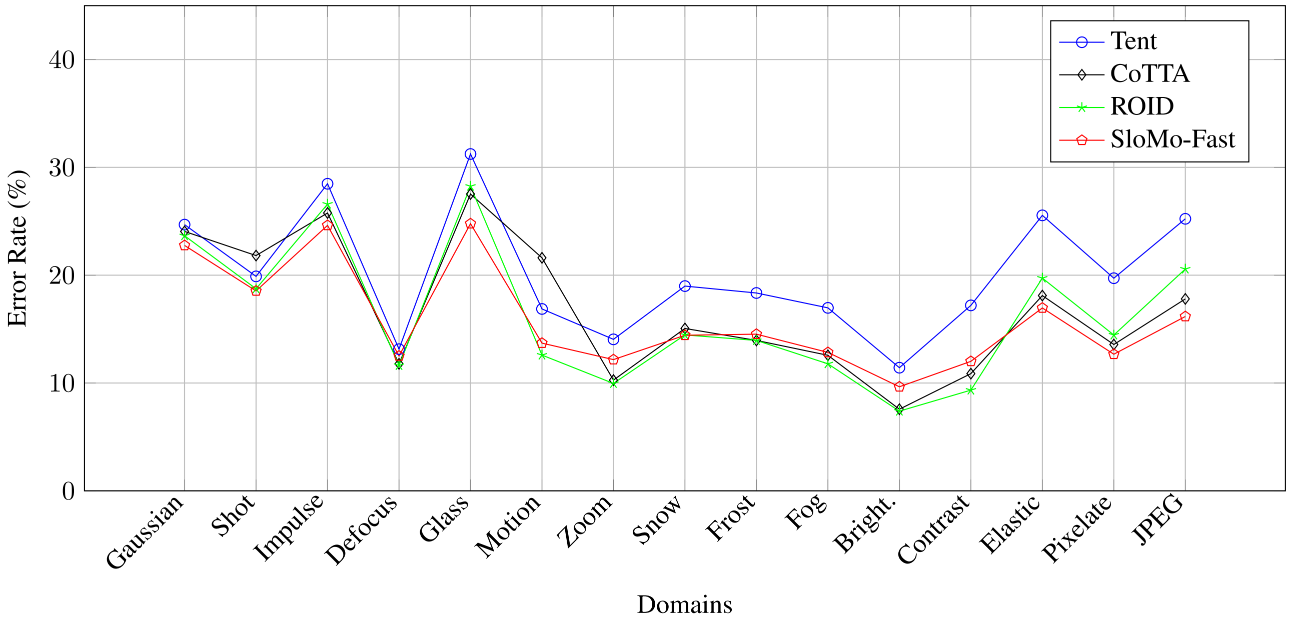

Figure 6. Mixed after Continual TTA Error rates (%) for Tent (blue), CoTTA (black), ROID (green), and PA (red) methods across domains in the CIFAR10-C benchmark for mixed domains after continual learning.

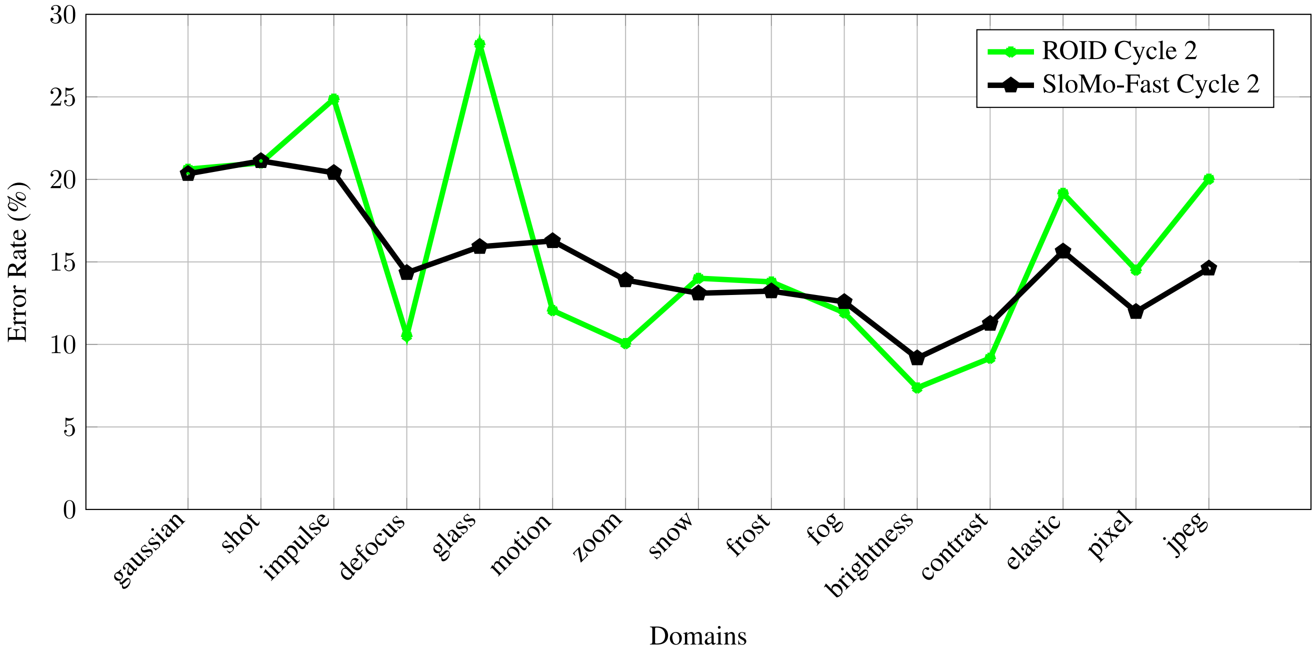

Figure 7. Cyclic TTA Error rates (%) for ROID and PA methods across domains with subgroup boundaries (Cycle 2 only). Here, Existing best ROID is fluctuating and indicates catestrophic forgetting where SloMo-Fast is stable