Persistent Test-time Adaptation in Recurring Testing Scenarios

Trung-Hieu Hoang[1] **Duc Minh Vo**[2] **Minh N. Do**[1,3]
[1]Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign
[2]The University of Tokyo
[3]VinUni-Illinois Smart Health Center, VinUniversity
{hthieu, minhdo}@illinois.edu vmduc@nlab.ci.i.u-tokyo.ac.jp

Abstract

Current test-time adaptation (TTA) approaches aim to adapt a machine learning model to environments that change continuously. Yet, it is unclear whether TTA methods can maintain their adaptability over prolonged periods. To answer this question, we introduce a diagnostic setting - **recurring TTA** where environments not only change but also recur over time, creating an extensive data stream. This setting allows us to examine the error accumulation of TTA models, in the most basic scenario, when they are regularly exposed to previous testing environments. Furthermore, we simulate a TTA process on a simple yet representative ϵ**-perturbed Gaussian Mixture Model Classifier**, deriving theoretical insights into the dataset- and algorithm-dependent factors contributing to gradual performance degradation. Our investigation leads us to propose **persistent TTA (PeTTA)**, which senses when the model is diverging towards collapse and adjusts the adaptation strategy, striking a balance between the dual objectives of adaptation and model collapse prevention. The supreme stability of PeTTA over existing approaches, in the face of lifelong TTA scenarios, has been demonstrated over comprehensive experiments on various benchmarks. Our project page is available at https://hthieu166.github.io/petta.

1 Introduction

Machine learning (ML) models have demonstrated significant achievements in various areas [18, 38, 47, 23]. Still, they are inherently susceptible to distribution-shift [46, 13, 48, 21, 6] (also known as the divergence between the training and testing environments), leading to a significant degradation in model performance. The ability to deviate from the conventional testing setting appears as a crucial aspect in boosting ML models' adaptability when confronted with a new testing environment that has been investigated [30, 53, 14]. Among common domain generalization methods [58, 24, 1], *test-time adaptation (TTA)* takes the most challenging yet rewarding path that leverages unlabeled data available at test time for self-supervised adaptation prior to the final inference [57, 39, 8, 41, 59].

Early TTA studies have concentrated on a simply ideal adaptation scenario where the test samples come from a fixed single domain [57, 39, 41]. As a result, such an assumption is far from the ever-changing and complex testing environments. To confront continually changing environments [59, 12], Yuan *et al.* [61] proposed a *practical TTA* scenario where distribution changing and correlative sampling occur [15] simultaneously. Though practical TTA is more realistic than what the previous assumptions have made, it still assumes that any environment only appears once in the data stream, a condition which does not hold true. Taking a surveillance camera as an example, it might accommodate varying lighting conditions recurringly day after day (Fig. 1-left). Based on this reality, we hypothesize that the recurring of those conditions may reveal the error accumulation phenomenon in TTA, resulting in performance degradation over a long period. To verify our hypothesis, we simulate a

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

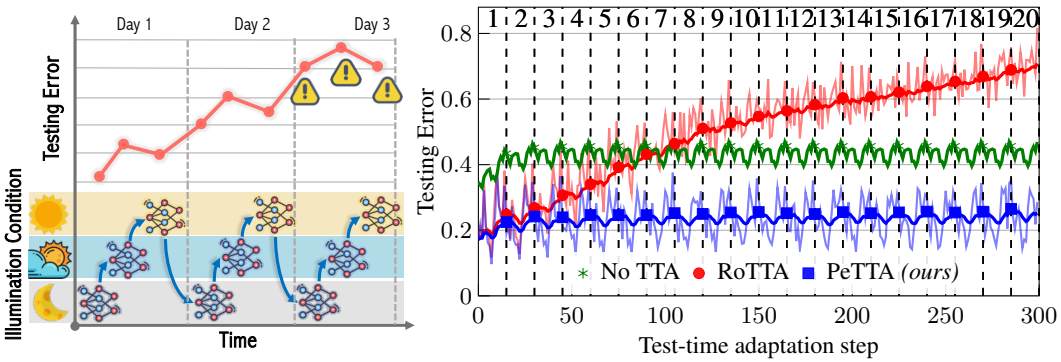


Figure 1: *Recurring Test-time Adaption (TTA)*. (left) Testing environments may change recurringly and preserving adaptability when visiting *the same* testing condition is not guaranteed. (right) The testing error of RoTTA [61] progressively raises (performance degradation) and exceeds the error of the source model (no TTA) while our PeTTA demonstrates its stability when adapting to the test set of CIFAR-10-C [19] 20 times. The **bold** lines denote the running mean and the shaded lines in the background represent the testing error on each domain (excluding the source model, for clarity).

recurring testing environment and observe the increasing error rate by recurringly adapting to the test set of CIFAR-10-C [19] multiple times. We showcase the testing error of RoTTA [61] after 20 cycles of adaptation in Fig. 1-right. As expected, RoTTA can successfully adapt and deliver encouraging outcomes within the first few passes. However, this advantage is short-lived as our study uncovers a significant issue: *TTA approaches in this setting may experience severe and persistent degradation in performance*. Consequently, the testing error of RoTTA gradually escalates over time and quickly surpasses the model without adaptation. This result confirms the risk of TTA deployment in our illustrative scenario, as an algorithm might work well in the first place and gradually degenerate. Therefore, ensuring sustainable quality is crucial for real-world applications, especially given the recurring nature of testing environments.

This study examines whether the adaptability of a TTA algorithm persists over an extended testing stream. Specifically, in the most basic scenario, where the model returns to a previously encountered testing environment after undergoing various adjustments. We thus propose a more general testing scenario than the practical TTA [61], namely *recurring TTA*, where the environments not only change gradually but also recur in a correlated manner over time. We first analyze a simulation using the *$\epsilon-$perturbed Gaussian Mixture Model Classifier ($\epsilon-$GMMC)* on a synthesized dataset and derive a theoretical analysis to confirm our findings, offering insights to tackle similar issues in deep neural networks. The analysis provides hints for reasoning the success of many recent robust continual TTA approaches [61, 12, 59, 15] and leading us to propose a simple yet effective baseline to avoid performance degradation, namely *Persistent TTA (PeTTA)*. PeTTA continuously monitors the chance of collapsing and adjusts the adaptation strategy on the fly, striking a balance between the two objectives: *adaptation* and *collapse prevention*. Our contributions can be summarized as follows:

- First, this work *proposes a testing scenario - recurring TTA*, a simple yet sufficient setup for diagnosing the overlooked *gradual performance degradation* phenomenon of TTA.
- Second, we formally *define the phenomenon of TTA collapsing and undertake a theoretical analysis* on an ϵ-GMMC, shedding light on dataset-dependent and algorithm-dependent factors that contribute to the error accumulation during TTA processes.
- Third, we *introduce persistent TTA (PeTTA)* - a simple yet effective adaptation scheme that surpasses all baseline models and demonstrates a persisting performance.

For more context on related work, readers are directed to visit our discussions in Appdx. A.

2 Background

Test-time Adaptation (TTA). A TTA algorithm operates on an ML classifier $f_t : \mathcal{X} \to \mathcal{Y}$ with parameter $\theta_t \in \Theta$ (parameter space) gradually changing over time ($t \in \mathcal{T}$) that maps an input image $\boldsymbol{x} \in \mathcal{X}$ to a category (label) $y \in \mathcal{Y}$. Let the capital letters $(X_t, Y_t) \in \mathcal{X} \times \mathcal{Y}$ denote a pair of *random variables* with the joint distribution $P_t(\boldsymbol{x}, y) \in \mathcal{P}_d, t \in \mathcal{T}$. Here, $\mathcal{P}_d$ belongs to collection of D sets of testing scenarios (domains) $\{\mathcal{P}_d\}_{d=1}^{D}$. The covariate shift [46] is assumed: $P_t(\boldsymbol{x})$ and $P_{t'}(\boldsymbol{x})$

could be different but $P_t(y|\boldsymbol{x}) = P_{t'}(y|\boldsymbol{x})$ holds $\forall t \neq t'$. At $t = 0$, θ_0 is initialized by a supervised model trained on $P_0 \in \mathcal{P}_0$ (source dataset). The model then explores an online stream of testing data. For each $t > 0$, it receives X_t (typically in form of a batch of N_t testing samples) for adapting itself $f_{t-1} \rightarrow f_t$ before making the final prediction $f_t(X_t)$.

TTA with Mean Teacher Update. To achieve a stable optimization process, the main (*teacher*) model f_t are updated indirectly through a *student* model with parameters θ'_t [57, 61, 12, 15, 55]. At first, the teacher model in the previous step introduces a *pseudo label* [28] $\hat{Y}_t$ for each X_t:

$$\hat{Y}_t = f_{t-1}(X_t). \tag{1}$$

With a classification loss $\mathcal{L}_{\text{CLS}}$ (e.g., cross-entropy [16]), and a model parameters regularizer $\mathcal{R}$, the student model is first updated with a generic optimization operator `Optim`, followed by an exponential moving average (EMA) update of the teacher model parameter θ_{t-1}:

$$\theta'_t = \underset{\theta' \in \Theta}{\texttt{Optim}} \, \mathbb{E}_{P_t} \left[\mathcal{L}_{\text{CLS}} \left(\hat{Y}_t, X_t; \theta' \right) \right] + \lambda \mathcal{R}(\theta'), \tag{2}$$

$$\theta_t = (1 - \alpha)\theta_{t-1} + \alpha \theta'_t, \tag{3}$$

with $\alpha \in (0, 1)$ - the update rate of EMA, and $\lambda \in \mathbb{R}^+$ - the weighting coefficient of the regularization term, are the two hyper-parameters.

Practical TTA. In practical TTA [61], two characteristics of the aforementioned distribution of data stream are noticeable. Firstly, P_t's can be partitioned by t_d's in which $\{P_t\}_{t=t_{d-1}}^{t_d} \subset \mathcal{P}_d$. Here, each partition of consecutive steps follows the same underlying distribution which will *change continually through D domains* [59] ($\mathcal{P}_1 \rightarrow \mathcal{P}_2 \cdots \rightarrow \mathcal{P}_D$). Secondly, the category distribution in each testing batch is *temporally correlated* [15]. This means within a batch, a small subset of categories is dominant over others, making the marginal distribution $P_t(y) = 0, \forall y \notin \mathcal{Y}_t \subset \mathcal{Y}$ even though the category distribution over all batches are balanced. Optimizing under this low intra-batch diversity ($|\mathcal{Y}_t| \ll |\mathcal{Y}|$) situation can slowly degenerate the model [7].

3 Recurring TTA and Theoretical Analysis

This section conducts a theoretical analysis on a concrete failure case of a simple TTA model. The results presented at the end of Sec. 3.2 will elucidate the factors contributing to the collapse (Sec. 3.1), explaining existing good practices (Sec. 3.3) and give insights into potential solutions (Sec. 4).

3.1 Recurring TTA and Model Collapse

Recurring TTA. To study the gradual performance degradation (or model collapse), we propose a *new testing scenario based on practical TTA* [61]. Conducting a single pass through D distributions, as done in earlier studies [61, 59], may not effectively identify the degradation. To promote consistency, our recurring TTA performs *revisiting the previous distributions K times* to compare the incremental error versus the previous visits. For example, a sequence with $K = 2$ could be $\mathcal{P}_1 \rightarrow \mathcal{P}_2 \rightarrow \cdots \rightarrow \mathcal{P}_D \rightarrow \mathcal{P}_1 \rightarrow \mathcal{P}_2 \rightarrow \cdots \rightarrow \mathcal{P}_D$. Appdx. D extends our justifications on constructing recurring TTA.

Definition 1 (**Model Collapse**). *A model is said to be collapsed from step $\tau \in \mathcal{T}, \tau < \infty$ if there exists a non-empty subset of categories $\tilde{\mathcal{Y}} \subset \mathcal{Y}$ such that $\Pr\{Y_t \in \tilde{\mathcal{Y}}\} > 0$ but the marginal $\Pr\{\hat{Y}_t \in \tilde{\mathcal{Y}}\}$ converges to zero in probability:*

$$\lim_{t \to \tau} \Pr\{\hat{Y}_t \in \tilde{\mathcal{Y}}\} = 0.$$

Here, upon collapsing, a model tends to *ignore* almost categories in $\tilde{\mathcal{Y}}$. As it is irrecoverable once collapsed, the only remedy would be resetting all parameters back to θ_0.

3.2 Simulation of Failure and Theoretical Analysis

Collapsing behavior varies across datasets and the adaptation processes. Formally studying this phenomenon on a particular real dataset and a TTA algorithm is challenging. Therefore, we propose a theoretical analysis on ϵ-perturbed binary Gaussian Mixture Model Classifier (ϵ-GMMC) that shares the typical characteristics *by construction* and demonstrates the *same collapsing pattern* in action (Sec. 5.1) as observed on real continual TTA processes (Sec. 5.3).

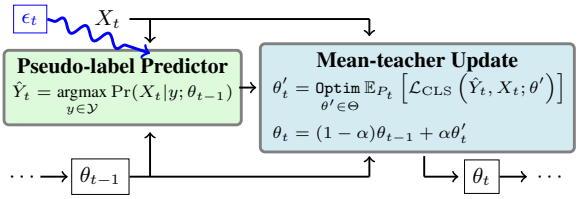


Figure 2: ϵ-perturbed binary Gaussian Mixture Model Classifier, imitating a continual TTA algorithm for theoretical analysis. Two main components include a pseudo-label predictor (Eq. 1), and a mean teacher update (Eqs. 2, 3). The *predictor is perturbed* for retaining a false negative rate of ϵ_t to simulate an undesirable TTA testing stream.

Simulated Testing Stream. Observing a testing stream with $(X_t, Y_t) \in \mathcal{X} \times \mathcal{Y} = \mathbb{R} \times \{0, 1\}$ and the underlying joint distribution $P_t(x, y) = p_{y,t} \cdot \mathcal{N}(x; \mu_y, \sigma_y^2)$. The main task is predicting X_t was sampled from cluster 0 or 1 (negative or positive). Conveniently, let $p_{y,t} \triangleq P_t(y) = \Pr(Y_t = y)$ and $\hat{p}_{y,t} \triangleq \Pr(\hat{Y}_t = y)$ be the marginal distribution of the true label Y_t and pseudo label $\hat{Y}_t$.

GMMC and TTA. GMMC first implies an *equal prior* distribution by construction which is desirable for the actual TTA algorithms (e.g., category-balanced sampling strategies in [61, 15]). Thus, it simplifies f_t into a maximum likelihood estimation $f_t(x) = \text{argmax}_{y \in \mathcal{Y}} \Pr(x|y; \theta_t)$ with $\Pr(x|y; \theta_t) = \mathcal{N}(x; \hat{\mu}_{y,t}, \hat{\sigma}^2_{y,t})$. The goal is estimating a set of parameters $\theta_t = \{\hat{\mu}_{y,t}, \hat{\sigma}^2_{y,t}\}_{y \in \mathcal{Y}}$. A perfect classifier $\theta_0 = \{\mu_y, \sigma_y^2\}_{y \in \mathcal{Y}}$ is initialized at $t = 0$. For the consecutive steps, the simplicity of GMMC allows solving the `Optim` (for finding θ'_t, Eq. 2) perfectly by computing the empirical mean and variance of new samples, approximating $\mathbb{E}_{P_t}$. The mean teacher update (Eq. 3) for GMMC is:

$$\hat{\mu}_{y,t} = \begin{cases} (1-\alpha)\hat{\mu}_{y,t-1} + \alpha \mathbb{E}_{P_t}\left[X_t | \hat{Y}_t\right] & \text{if } \hat{Y}_t = y \\ \hat{\mu}_{y,t-1} & \text{otherwise} \end{cases}. \tag{4}$$

The update of $\hat{\sigma}^2_{y,t}$ is similar. $\hat{Y}_t = f_{t-1}(X_t)$ can be interpreted as a *pseudo label* (Eq. 1).

ϵ-GMMC. Severe distribution shifts or low intra-batch category diversity of recurring TTA/practical TTA *both result in an increase in the error rate of the predictor*. Instead of directly modeling the dynamic changes of $p_{y,t}$ (which can be complicated depending on the dataset), we study an $\epsilon-$pertubed GMMC ($\epsilon-$GMMC), where $p_{y,t}$ is *assumed to be static* (defined below) and the pseudo-label predictor of this model is *perturbed* to *simulate undesirable effects of the testing stream* on the predictor. Two kinds of errors appear in a binary classifier [4]. Let

$$\epsilon_t = \Pr\{Y_t = 1 | \hat{Y}_t = 0\} \tag{5}$$

be the false negative rate (FNR) of the model at step t. Without loss of generality, we study the *increasing type II collapse of ϵ-GMMC*. By intentionally flipping the true positive pseudo labels in simulation, an FNR of ϵ_t is maintained (Fig. 2).

Assumption 1 (Static Data Stream). *The marginal distribution of the true label follows the same Bernoulli distribution* $\text{Ber}(p_0)$*:* $p_{0,t} = p_0$*,* $(p_{1,t} = p_1 = 1 - p_0), \forall t \in \mathcal{T}$.

Lemma 1 (Increasing FNR). *Under Assumption 1, a binary ϵ-GMMC would collapsed (Def. 1) with* $\lim_{t \to \tau} \hat{p}_{1,t} = 0$ *(or* $\lim_{t \to \tau} \hat{p}_{0,t} = 1$*, equivalently) if and only if* $\lim_{t \to \tau} \epsilon_t = p_1$.

Lemma 1 states the negative correlation between $\hat{p}_{1,t}$ and ϵ_t. Unsurprisingly, towards the collapsing point where all predictions are zeros, the FNR also increases at every step and eventually reaches the highest possible FNR of p_1.

Lemma 2 (ϵ-GMMC After Collapsing). *For a binary ϵ-GMMC model, with Assumption 1, if* $\lim_{t \to \tau} \hat{p}_{1,t} = 0$ *(collapsing), the cluster 0 in GMMC converges in distribution to a single-cluster GMMC with parameters:*

$$\mathcal{N}(\hat{\mu}_{0,t}, \hat{\sigma}^2_{0,t}) \xrightarrow{d} \mathcal{N}(p_0\mu_0 + p_1\mu_1, p_0\sigma_0^2 + p_1\sigma_1^2 + p_0p_1(\mu_0 - \mu_1)^2).$$

Lemma 2 states the resulting $\epsilon-$GMMC after collapsing. Cluster 0 now *covers the whole data distribution* (and assigning label 0 for all samples). Furthermore, *collapsing happens when $\hat{\mu}_{0,t}$ moves toward μ_1*. We next investigate the factors and conditions for this undesirable convergence.

Theorem 1 (Convergence of $\epsilon-$GMMC). *For a binary ϵ-GMMC model, with Assumption 1, let the distance from $\hat{\mu}_{0,t}$ toward μ_1 is $d_t^{0\to1} = |\mathbb{E}_{P_t}[\hat{\mu}_{0,t}] - \mu_1|$, then:*

$$d_t^{0\to1} - d_{t-1}^{0\to1} \leq \alpha \cdot p_0 \cdot \left(|\mu_0 - \mu_1| - \frac{d_{t-1}^{0\to1}}{1-\epsilon_t} \right).$$

From Thm. 1, we observe that the distance $d_t^{0\to1}$'s converges (also indicating the convergence to the distribution in Lemma 2) if $d_t^{0\to1} < d_{t-1}^{0\to1}$. The model collapse happens when this condition holds for a sufficiently long period.

Corollary 1 (A Condition for $\epsilon-$GMMC Collapse). *With fixed p_0, α, μ_0, μ_1, $\epsilon-$GMMC is collapsed if there exists a sequence of $\{\epsilon_t\}_{\tau-\Delta_\tau}^{\tau}$ ($\tau \geq \Delta_\tau > 0$) such that:*

$$p_1 \geq \epsilon_t > 1 - \frac{d_{t-1}^{0\to1}}{|\mu_0 - \mu_1|}, \quad t \in [\tau - \Delta_\tau, \tau].$$

Corollary 1 introduces a condition ϵ-GMMC collapse. Here, ϵ_t's are non-decreasing, $\lim_{t\to\tau} \epsilon_t = p_1$.

Remarks. Thm. 1 concludes two sets of factors contributing to collapse: (i) *data-dependent factors*: the prior data distribution (p_0), the nature difference between two categories ($|\mu_0 - \mu_1|$); and (ii) *algorithm-dependent factors*: the update rate (α), the FNR at each step (ϵ_t). ϵ-GMMC analysis sheds light on explaining model collapse on real datasets (Sec. 5.3), reasons the existing approaches (Sec. 3.3) and motivates the development of our baseline (Sec. 4).

3.3 Connection to Existing Solutions

Prior TTA algorithms have already incorporated implicit mechanisms to mitigate model collapse. The theoretical results in the previous section explain the rationale behind these effective strategies.

Regularization Term for θ_t. Knowing that f_0 is always well-behaved, an attempt is restricting the divergence of θ_t from θ_0, e.g. using $\mathcal{R}(\theta_t) \triangleq \|\theta_0 - \theta_t\|_2^2$ regularization [40]. The key idea is introducing a penalty term to avoid an extreme divergence as happening in Thm. 1.

Memory Bank for Harmonizing $P_t(x)$. Upon receiving X_t, samples in this batch are selectively updated to a memory bank $\mathcal{M}$ (which already contains a subset of some instances of $X_{t'}, t' < t$ in the previous steps). By keeping a balanced number of samples from each category, distribution $P_t^{\mathcal{M}}(y)$ of samples in $\mathcal{M}$ is expected to have less zero entries than $P_t(y)$, making the optimization step over $P_t^{\mathcal{M}}$ more desirable. From Thm. 1, $\mathcal{M}$ moderates the extreme value of the category distribution (p_0 term) which typically appears on batches with low intra-batch category diversity.

4 Persistent Test-time Adaptation (PeTTA)

Now we introduce our *Persistent TTA (PeTTA)* approach. Further inspecting Thm. 1, while ϵ_t (Eq. 5) is not computable without knowing the true labels, the measure of divergence from the initial distribution (analogously to $d_{t-1}^{0\to1}$ term) can provide hints to fine-tune the adaptation process.

Key Idea. A proper adjustment toward the TTA algorithm can *break the chain of monotonically increasing* ϵ_t's in Corollary 1 to prevent the model collapse. In the mean teacher update, the larger value of λ (Eq. 2) prioritizes the task of preventing collapse on one hand but also limits its adaptability to the new testing environment. Meanwhile, α (Eq. 3) controls the weight on preserving versus changing the model from the previous step. Drawing inspiration from the exploration-exploitation tradeoff [49, 25] encountered in reinforcement learning [54], we introduce a mechanism for *adjusting λ and α on the fly, balancing between the two primary objectives: adaptation and preventing model collapse*. Our strategy is prioritizing collapse prevention (increasing λ) and preserving the model from previous steps (decreasing α) when there is a significant deviation from θ_0.

In [40, 61, 59], λ and α were fixed through hyper-parameter tuning. This is suboptimal due to varying TTA environments and the lack of validation set [62]. Furthermore, Thm. 1 suggests the convergence rate quickly escalates when ϵ_t increases, making constant λ, α insufficient to prevent collapse.

Sensing the Divergence of θ_t. We first equip PeTTA with a mechanism for *measuring its divergence from θ_0*. Since $f_t(\boldsymbol{x}) = \text{argmax}_{y\in\mathcal{Y}} \Pr(y|\boldsymbol{x};\theta_t)$, we can decompose $\Pr(y|\boldsymbol{x};\theta_t) = [h(\phi_{\theta_t}(\boldsymbol{x}))]_y$, with $\phi_{\theta_t}(\cdot)$ is a θ_t-parameterized deep feature extractor followed by a *fixed* classification head (a linear and softmax layer) $h(\cdot)$. The operator $[\cdot]_y$ extracts the y^{th} component of a vector.

Since $h(\cdot)$ remains unchanged, instead of comparing the divergence in the parameter space (Θ) or between the output probability $\Pr(y|\boldsymbol{x};\theta_t)$ and $\Pr(y|\boldsymbol{x};\theta_0)$, we suggest an *inspection over the feature embedding space* that preserves a *maximum amount of information* in our case (data processing inequality [9]). Inspired by [31] and under Gaussian assumption, the Mahalanobis distance of the first moment of the feature embedding vectors is compared. Let $\boldsymbol{z} = \phi_{\theta_t}(\boldsymbol{x})$, we keep track of a collection of the running mean of feature vector $\boldsymbol{z}$: $\{\hat{\boldsymbol{\mu}}_t^y\}_{y\in\mathcal{Y}}$ in which $\hat{\boldsymbol{\mu}}_t^y$ is EMA updated with vector $\boldsymbol{z}$ if $f_t(\boldsymbol{x}) = y$. The divergence of θ_t at step t, evaluated on class y is defined as:

$$\gamma_t^y = 1 - \exp\left(-(\hat{\boldsymbol{\mu}}_t^y - \boldsymbol{\mu}_0^y)^T \left(\boldsymbol{\Sigma}_0^y\right)^{-1} (\hat{\boldsymbol{\mu}}_t^y - \boldsymbol{\mu}_0^y)\right), \tag{6}$$

where $\boldsymbol{\mu}_0^y$ and $\boldsymbol{\Sigma}_0^y$ are the pre-computed empirical mean and covariant matrix of feature vectors in the source dataset (P_0). The covariant matrix here is diagonal for simplicity. In practice, without directly accessing the training set, we assume a small set of unlabeled samples can be drawn from the source distribution for empirically computing these values (visit Appdx. E.4 for further details).

Here, we implicitly expect the independence of each entry in $\boldsymbol{z}$ and TTA approaches *learn to align feature vectors of new domains back to the source domain* (P_0). Therefore, the accumulated statistics of these feature vectors at each step should be concentrated near the vectors of the initial model. The value of $\gamma_t^y \in [0, 1]$ is close to 0 when $\theta_t = \theta_0$ and increases exponentially as $\hat{\boldsymbol{\mu}}_t^y$ diverging from $\boldsymbol{\mu}_0^y$.

Adaptive Regularization and Model Update. With α_0, λ_0 are initial values, utilizing γ_t^y derived in Eq. 6, a pair of (λ_t, α_t) is *adaptively* chosen at each step:

$$\begin{aligned} &\bar{\gamma}_t = \frac{1}{|\hat{\mathcal{Y}}_t|} \sum_{y\in\hat{\mathcal{Y}}_t} \gamma_t^y, \quad \hat{\mathcal{Y}}_t = \left\{\hat{Y}_t^{(i)} | i = 1, \cdots, N_t\right\}; \\ &\lambda_t = \bar{\gamma}_t \cdot \lambda_0, \qquad \alpha_t = (1 - \bar{\gamma}_t) \cdot \alpha_0, \end{aligned} \tag{7}$$

$\hat{\mathcal{Y}}_t$ is a set of unique pseudo labels in a testing batch ($\hat{Y}_t^{(i)}$ is the i^{th} realization of $\hat{Y}_t$).

Anchor Loss. Penalizing the divergence with regular vector norms in high-dimensional space (Θ) is insufficient (curse of dimensionality [5, 51]), especially with a large model and limited samples. *Anchor loss* $\mathcal{L}_{\text{AL}}$ can nail down the similarity between f_t and f_0 in the probability space [32, 12]:

$$\mathcal{L}_{\text{AL}}(X_t;\theta) = -\sum_{y\in\mathcal{Y}} \Pr(y|X_t;\theta_0) \log \Pr(y|X_t;\theta), \tag{8}$$

which is equivalent to minimizing the KL divergence $D_{KL}\left(\Pr(y|X_t;\theta_0) \| \Pr(y|X_t;\theta)\right)$.

Persistent TTA. Having all the ingredients, we design our approach, PeTTA, following the convention setup of the mean teacher update, with the category-balanced memory bank and the robust batch normalization layer from [61]. Appdx. E.1 introduces the *pseudo code* of PeTTA. For $\mathcal{L}_{\text{CLS}}$, either the self-training scheme [12] or the regular cross-entropy [16] is adopted. With $\mathcal{R}(\theta)$, cosine similarity or L2 distance are both valid metrics for measuring the distance between θ and θ_0 in the parameter space. Fisher regularizer coefficient [40, 27] can also be used, optionally. To sum up, the teacher model update of PeTTA is an *elaborated version* of EMA with λ_t, α_t (Eq. 7) and $\mathcal{L}_{\text{AL}}$ (Eq. 8):

$$\begin{aligned} &\theta_t' = \underset{\theta'\in\Theta}{\texttt{Optim}}\, \mathbb{E}_{P_t}\left[\mathcal{L}_{\text{CLS}}\left(\hat{Y}_t, X_t; \theta'\right) + \mathcal{L}_{\text{AL}}\left(X_t;\theta'\right)\right] + \lambda_t \mathcal{R}(\theta'), \\ &\theta_t = (1-\alpha_t)\theta_{t-1} + \alpha_t \theta_t'. \end{aligned}$$

5 Experimental Results

5.1 $\epsilon-$MMC Simulation Result

Simulation Setup. A total of 6000 samples from two Gaussian distributions: $\mathcal{N}(\mu_0 = 0, \sigma_0^2 = 1)$ and $\mathcal{N}(\mu_1 = 2, \sigma_1^2 = 1)$ with $p_0 = p_1 = \frac{1}{2}$ are synthesized and gradually released in a batch of $B = 10$ samples. For evaluation, an independent set of 2000 samples following the same distribution is used for computing the prediction frequency, and the false negative rate (FNR). $\epsilon-$GMMC update follows Eq. 4 with $\alpha = 5e^{-2}$. To simulate model collapse, the predictor is intercepted and 10% of the true-postive pseudo labels at each testing step are randomly flipped (Corollary 1).

Simulation Result. In action, both the likelihood of predicting class 0 (Fig. 3a-left) and the ϵ_t (Eq. 5) (Fig. 3c-right, solid line) gradually increases over time as expected (Lemma 1). After collapsing,

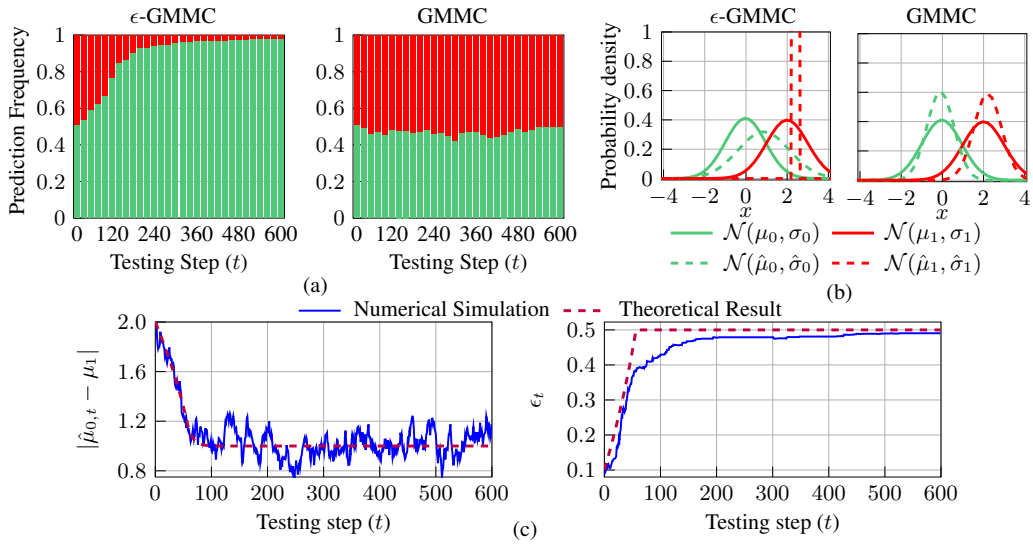


Figure 3: Simulation result on ϵ-perturbed Gaussian Mixture Model Classifier (ϵ-GMMC) and GMMC (perturbed-free). (a) Histogram of model predictions through time. A similar prediction frequency pattern is observed on CIFAR-10-C (Fig. 5a-left). (b) The probability density function of the two clusters after convergence versus the true data distribution. The initial two clusters of ϵ-GMMC collapsed into a single cluster with parameters stated in Lemma 2. In the perturbed-free, GMMC converges to the true data distribution. (c) Distance toward μ_1 ($|\mathbb{E}_{P_t}[\hat{\mu}_{0,t}] - \mu_1|$) and false-negative rate (ϵ_t) in simulation coincides with the result in Thm. 1 (with ϵ_t following Corollary 1).

ϵ-GMMC merges the two initial clusters, resulting in a single one (Fig. 3b-left) with parameters that match Lemma 2. The distance from $\hat{\mu}_{0,t}$ (initialized at μ_0) towards μ_1 converges (Fig. 3c-left, solid line), coincided with the analysis in Thm. 1 when ϵ_t is chosen following Corollary 1 (Fig. 3c, dashed line). GMMC (perturbed-free) stably produces accurate predictions (Fig. 3a-right) and approximates the true data distribution (Fig. 3b-right). The simulation empirically validates our analysis (Sec. 3.2), confirming the vulnerability of TTA models when the pseudo labels are inaccurately estimated.

5.2 Setup - Benchmark Datasets

Datasets. We benchmark the performance on *four* TTA classification tasks. Specifically, CIFAR10 $\rightarrow$ CIFAR10-C, CIFAR100 $\rightarrow$ CIFAR100-C, and ImageNet $\rightarrow$ ImageNet-C [19] are three corrupted images classification tasks (corruption level 5, the most severe). Additionally, we incorporate DomainNet [44] with 126 categories from four domains for the task *real* $\rightarrow$ *clipart, painting, sketch*.

Compared Methods. Besides PeTTA, the following algorithms are investigated: CoTTA [59], EATA [40], RMT [12], MECTA [22], RoTTA [61], ROID [37] and TRIBE [52]. Noteworthy, only RoTTA is specifically designed for the practical TTA setting while others fit the continual TTA setting in general. A parameter-free approach: LAME [7] and a reset-based approach (i.e., reverting the model to the source model after adapting to every $1,000$ images): RDumb [45] are also included.

Recurring TTA. Following the practical TTA setup, multiple testing scenarios from each testing set will gradually change from one to another while the Dirichlet distribution ($\mathrm{Dir}(0.1)$ for CIFAR10-C, DomainNet, and ImageNet-C, and $\mathrm{Dir}(0.01)$ for CIFAR100-C) generates category temporally correlated batches of data. For all experiments, we set the number of revisits $K = 20$ (times) as this number is sufficient to fully observe the gradual degradation on existing TTA baselines.

Implementation Details. We use `PyTorch` [43] for implementation. `RobustBench` [10] and `torchvision` [35] provide pre-trained source models. Hyper-parameter choices are kept as close as possible to the original selections of authors. Visit Sec. G for more implementation details. Unless otherwise noted, for all PeTTA experiments, the EMA update rate for robust batch normalization [61] and feature embedding statistics is set to $5e^{-2}$; $\alpha_0 = 1e^{-3}$ and cosine similarity regularizer is used. On CIFAR10/100-C and ImageNet-C we use the self-training loss in [12] for $\mathcal{L}_{\mathrm{CLS}}$ and $\lambda_0 = 10$ while the regular cross-entropy loss [13] and $\lambda_0 = 1$ (severe domain shift requires prioritizing

Table 1: Average classification error of the task CIFAR-10 → CIFAR-10-C in *recurring TTA*. The lowest error is in **bold**, $^{(*)}$average value across 5 runs (different random seeds) is reported for PeTTA.

Method	*Recurring TTA visit* → 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	**Avg**
Source	43.5																				43.5
LAME [7]	31.1																				31.1
CoTTA [59]	82.2	85.6	87.2	87.8	88.2	88.5	88.7	88.7	88.9	88.9	88.9	89.2	89.2	89.2	89.1	89.2	89.2	89.1	89.3	89.3	88.3
EATA [40]	81.6	87.0	88.7	88.7	88.9	88.7	88.6	89.0	89.3	89.6	89.5	89.6	89.7	89.7	89.3	89.6	89.6	89.8	89.9	89.4	88.8
RMT [12]	77.5	76.9	76.5	75.8	75.5	75.5	75.4	75.4	75.5	75.3	75.5	75.6	75.5	75.5	75.7	75.6	75.7	75.6	75.7	75.8	75.8
MECTA [22]	72.2	82.0	85.2	86.3	87.0	87.3	87.3	87.5	88.1	88.8	88.9	88.9	88.6	89.1	88.7	88.8	88.5	88.6	88.3	88.8	86.9
RoTTA [61]	24.6	25.5	29.6	33.6	38.2	42.8	46.2	50.6	52.2	54.1	56.5	57.5	59.4	60.2	61.7	63.0	64.8	66.1	68.2	70.3	51.3
RDumb [45]	31.1	32.1	32.3	31.6	31.9	31.8	31.8	31.9	31.9	32.1	31.7	32.0	32.5	32.0	31.9	31.6	31.9	31.4	32.3	32.4	31.9
ROID [37]	72.7	72.6	73.1	72.4	72.7	72.8	72.7	72.7	72.9	72.8	72.9	72.9	72.8	72.5	73.0	72.8	72.5	72.5	72.7	72.7	72.7
TRIBE [52]	**15.3**	**16.6**	**16.6**	**16.3**	**16.7**	**17.0**	**17.3**	**17.4**	**17.4**	**18.0**	**17.9**	**18.0**	**17.9**	**18.6**	**18.2**	**18.8**	**18.0**	**18.2**	**18.4**	**18.0**	**17.5**
PeTTA *(ours)*$^{(*)}$	24.3	23.0	22.6	22.4	22.4	22.5	22.3	22.5	22.8	22.8	22.6	22.7	22.7	22.9	22.6	22.7	22.6	22.8	22.9	23.0	22.8

adaptability) are applied in DomainNet experiments. In Appdx. F.5, we provide a sensitivity analysis on the choice of hyper-parameter λ_0 in PeTTA.

5.3 Result - Benchmark Datasets

Recurring TTA Performance. Fig. 1-right presents the testing error on CIFAR-10-C in recurring TTA setting. RoTTA [61] exhibits promising performance in the first several visits but soon raises and eventually exceeds the source model (no TTA). The classification error of compared methods on CIFAR-10→CIFAR-10-C, and ImageNet → ImageNet-C [19] tasks are shown in Tab. 1, and Tab. 2. Appdx. F.1 provides the results on the other two datasets. The observed performance degradation of CoTTA [59], EATA [40], RoTTA [61], and TRIBE [52] *confirms the risk of error accumulation* for an extensive period. While RMT [12], MECTA [22], and ROID [37] remain stable, they failed to adapt to the temporally correlated test stream at the beginning, with a higher error rate than the source model. LAME [7] (parameter-free TTA) and RDumb [45] (reset-based TTA) do not suffer from collapsing. However, their performance is lagging behind, and knowledge accumulation is limited in these approaches that could potentially favor a higher performance as achieved by PeTTA. Furthermore, LAME [7] is highly constrained by the source model, and selecting a precise reset frequency in RDumb [45] is challenging in practice (see Appdx. F.3 for a further discussion).

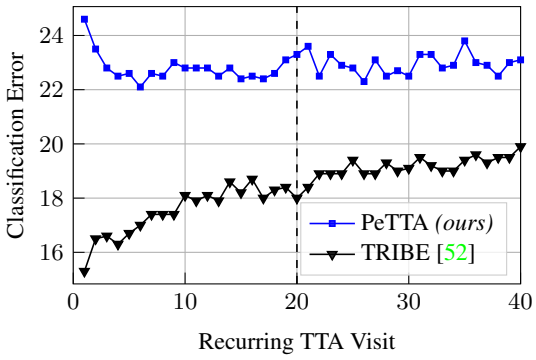


Figure 4: Classification error of TRIBE [52] and PeTTA *(ours)* of the task CIFAR-10→CIFAR10-C task in *recurring* TTA with 40 visits.

In average, PeTTA *outperforms almost every baseline approaches* and *persists across 20 visits* over the three datasets. The only exception is at the case of TRIBE [52] on CIFAR-10-C. While this state-of-the-art model provides stronger adaptability, outweighing the PeTTA, and baseline RoTTA [61] in several recurrences, the risk of the model collapsing still presents in TRIBE [52]. This can be clearly observed when we increase the observation period to 40 recurring visits in Fig. 4. As the degree of freedom for adaptation in PeTTA is more constrained, it takes a bit longer for adaptation but remains stable afterward. Fig. 5b-bottom exhibits the confusion matrix at the last visit with satisfactory accuracy. The same results are also observed when shuffling the order of domain shifts within each recurrence (Appdx. D.3), or extending the number of recurrences to 40 visits (Appdx. F.4).

Continuously Changing Corruption (CCC) [45] Performance. Under CCC [45], Tab. 3 reveals the supreme performance of PeTTA over RoTTA [61] and RDumb [45]. Here, we report the average classification error between two consecutive adaptation step intervals. An adaptation step in this table corresponds to a mini-batch of data with 64 images. The model is adapted to $80,000$ steps in total with more than 5.1M images, significantly longer than 20 recurring TTA visits. Undoubtedly, PeTTA still achieves good performance where the corruptions are algorithmically generated, non-cyclic with two or more corruption types can happen simultaneously. This experiment also empirically justifies the construction of our recurring TTA as a diagnostic tool (Appdx. D.2) where similar observations are concluded on the two settings. Obviously, our recurring TTA is notably simpler than CCC [45].

Table 2: Average classification error of the task ImageNet → ImageNet-C in *recurring TTA* scenario.

Method	*Recurring TTA visit* → 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Avg
Source										82.0											82.0
LAME [7]										80.9											80.9
CoTTA [59]	98.6	99.1	99.4	99.4	99.5	99.5	99.5	99.5	99.6	99.7	99.6	99.6	99.6	99.6	99.6	99.6	99.6	99.6	99.7	99.7	99.5
EATA [40]	60.4	59.3	65.4	72.6	79.1	84.2	88.7	92.7	95.2	96.9	97.7	98.1	98.4	98.6	98.7	98.8	98.8	98.9	98.9	99.0	89.0
RMT [12]	72.3	71.0	69.9	69.1	68.8	68.5	68.4	68.3	70.0	70.2	70.1	70.2	72.8	76.8	75.6	75.1	75.1	75.2	74.8	74.7	71.8
MECTA [22]	77.2	82.8	86.1	87.9	88.9	89.4	89.8	89.9	90.0	90.4	90.6	90.7	90.7	90.8	90.8	90.9	90.8	90.8	90.7	90.8	89.0
RoTTA [61]	68.3	62.1	61.8	64.5	68.4	75.4	82.7	95.1	95.8	96.6	97.1	97.9	98.3	98.7	99.0	99.1	99.3	99.4	99.5	99.6	87.9
RDumb [45]	72.2	73.0	73.2	72.8	72.2	72.8	73.3	72.7	71.9	73.0	73.2	73.1	72.0	72.7	73.3	73.1	72.1	72.6	73.3	73.1	72.8
ROID [37]	62.7	62.3	62.3	62.3	62.5	62.3	62.4	62.4	62.3	62.6	62.5	62.3	62.5	62.4	62.5	62.4	62.4	62.5	62.4	62.5	62.4
TRIBE [52]	**63.6**	64.0	64.9	67.8	69.6	71.7	73.5	75.5	77.4	79.8	85.0	96.5	99.4	99.8	99.9	99.8	99.8	99.9	99.9	99.9	84.4
PeTTA *(ours)*(*)	65.3	**61.7**	**59.8**	**59.1**	**59.4**	**59.6**	**59.8**	**59.3**	**59.4**	**60.0**	**60.3**	**61.0**	**60.7**	**60.4**	**60.6**	**60.7**	**60.8**	**60.7**	**60.4**	**60.2**	**60.5**

Table 3: Average classification error on CCC [45] setting. Each column presents the average error within an adaptation interval (e.g., the second column provides the average error between the 6701 and 13400 adaptation steps). Each adaptation step here is performed on a mini-batch of 64 images.

Method	*CCC [45] Adaptation Step* → 6700	13400	20100	26800	33500	40200	46900	53600	60200	66800	73400	80000	Avg
Source	0.83	0.83	0.83	0.83	0.83	0.84	0.84	0.83	0.84	0.83	0.83	0.83	0.83
RoTTA [61]	0.70	0.85	0.92	0.96	0.98	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.95
RDumb [45]	0.78	0.74	0.75	0.77	0.75	0.72	0.75	0.77	0.75	0.74	0.75	0.75	0.75
PeTTA *(ours)*	**0.67**	**0.63**	**0.62**	**0.65**	**0.65**	**0.64**	**0.64**	**0.68**	**0.63**	**0.63**	**0.65**	**0.65**	**0.64**

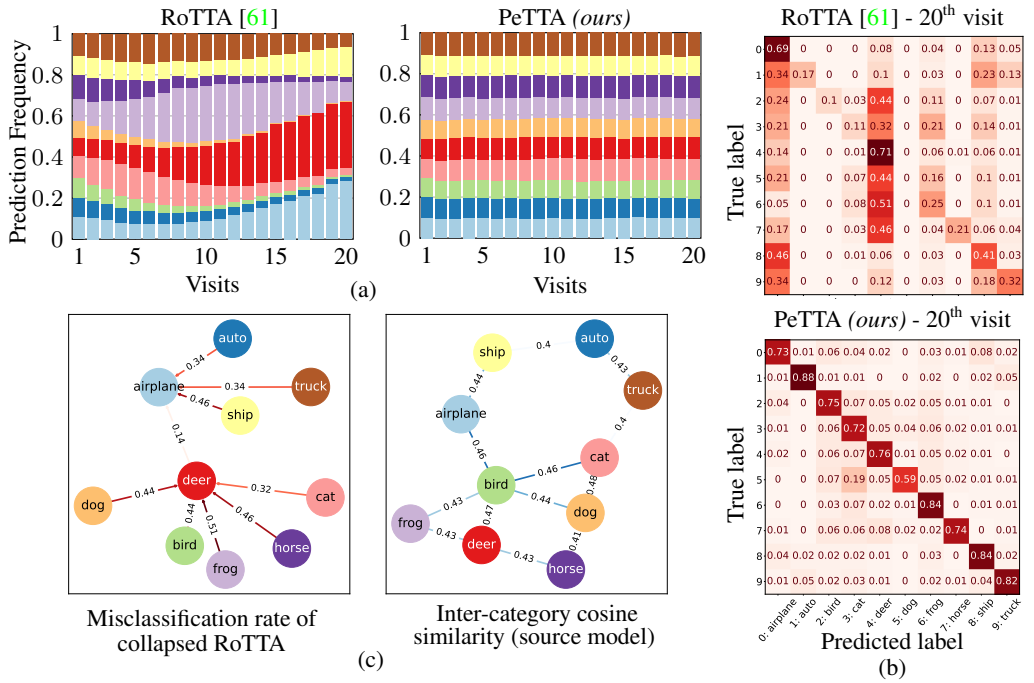


Figure 5: Recurring TTA (20 visits) on CIFAR-10→CIFAR10-C task. (a) Histogram of model predictions (10 labels are color-coded). PeTTA achieves a persisting performance while RoTTA [61] degrades. (b) Confusion matrix at the last visit, RoTTA classifies all samples into a few categories (e.g., *0: airplane*, *4: deer*). (c) Force-directed graphs showing (left) the most prone to misclassification pairs (arrows indicating the portion and pointing from the true to the misclassified category); (right) similar categories tend to be easily collapsed. Edges denote the average cosine similarity of feature vectors (source model), only the highest similar pairs are shown. Best viewed in color.

Collapsing Pattern. The rise in classification error (Fig. 1-right) can be reasoned by the prediction frequency of RoTTA [61] in an recurring TTA setting (Fig. 5a-left). Similar to ϵ-GMMC, the likelihood of receiving predictions on certain categories gradually increases and dominates the others. Further inspecting the confusion matrix of a collapsed model (Fig. 5b-top) reveals two major groups of categories are formed and a single category within each group represents all members, thereby becoming dominant. To see this, Fig. 5c-left simplifies the confusion matrix by only visualizing the

Table 4: Average (across 20 visits) error of multiple variations of PeTTA: without (w/o) $\mathcal{R}(\theta), \mathcal{L}_{\text{AL}}$; $\mathcal{L}_{\text{AL}}$ only; fixed regularization coefficient λ; adaptive coefficient λ_t, update rate α_t; using anchor loss $\mathcal{L}_{\text{AL}}$.

Method	CF-10-C	CF-100-C	DN	IN-C
Baseline w/o $\mathcal{R}(\theta), \mathcal{L}_{\text{AL}}$	42.6	63.0	77.9	93.4
$\mathcal{R}(\theta)$ fixed $\lambda = 0.1\lambda_0$	43.3	65.0	80.0	92.5
$\mathcal{R}(\theta)$ fixed $\lambda = \lambda_0$	42.0	64.6	66.6	92.9
$\mathcal{L}_{\text{AL}}$ only	25.4	56.5	47.5	68.1
PeTTA - λ_t	27.1	55.0	59.7	92.7
PeTTA - λ_t + α_t	23.9	41.4	44.5	75.7
PeTTA - λ_t + $\mathcal{L}_{\text{AL}}$	26.2	36.3	43.2	62.0
PeTTA - λ_t + α_t + $\mathcal{L}_{\text{AL}}$	**22.8**	**35.1**	**42.9**	**60.5**

Table 5: Average (across 20 visits) error of PeTTA. PeTTA favors various choices of regularizers $\mathcal{R}(\theta)$: L2 and cosine similarity in conjunction with Fisher [27, 40] coefficient.

Method		CF-10-C	CF-100-C	DN	IN-C
$\mathcal{R}(\theta)$	Fisher				
L2	✗	23.0	35.6	43.1	70.8
	✓	22.7	36.0	43.9	70.0
Cosine	✗	22.8	**35.1**	**42.9**	**60.5**
	✓	**22.6**	35.9	43.3	63.8

CF: CIFAR, DN: DomainNet, IN: ImageNet

top prone-to-misclassified pair of categories. Here, label *deer* is used for almost every living animal while *airplane* represents transport vehicles. The similarity between categories in the feature space of the source model (Fig. 5c-right) is correlated with the likelihood of being merged upon collapsing. As distance in feature space is analogous to $|\mu_0 - \mu_1|$ (Thm. 1), closer clusters are at a higher risk of collapsing. This explains and showcases that the collapsing behavior is predictable up to some extent.

5.4 Ablation Study

Effect of Each Component. Tab. 4 gives an ablation study on PeTTA, highlighting the use of a regularization term ($\mathcal{R}(\theta)$) with a fixed choice of λ, α not only fails to mitigate model collapse but may also introduce a negative effect (rows 2-3). Trivially applying the anchor loss ($\mathcal{L}_{\text{AL}}$) alone is also incapable of eliminating the lifelong performance degradation in continual TTA (row 4). Within PeTTA, adopting the adaptive λ_t scheme alone (row 5) or in conjunction with either α_t or anchor loss $\mathcal{L}_{\text{AL}}$ (rows 6-7) partially stabilizes the performance. Under the drastic domain shifts with a larger size of categories or model parameters (e.g., on CIFAR-100-C, DomainNet, ImageNet-C), restricting α_t adjustment limits the ability of PeTTA to stop undesirable updates while a common regularization term without $\mathcal{L}_{\text{AL}}$ is insufficient to guide the adaptation. Thus, leveraging all elements secures the persistence of PeTTA (row 8).

Various Choices of Regularizers. The design of PeTTA is not coupled with any specific regularization term. Demonstrated in Tab. 5, PeTTA works well for the two common choices: L2 and cosine similarity. The conjunction use of Fisher coefficent [27, 40] for weighting the model parameter importance is also studied. While the benefit (in terms of improving accuracy) varies across datasets, PeTTA accommodates all choices, as the model collapse is not observed in any of the options.

6 Discussions and Conclusion

On a Potential Risk of TTA in Practice. We provide empirical and theoretical evidence on the risk of deploying continual TTA algorithms. Existing studies fail to detect this issue with *a single pass per test set*. The recurring TTA could be conveniently adopted as a *straightforward evaluation*, where its challenging test stream magnifies the error accumulation that a model might encounter in practice.

Limitations. PeTTA takes one step toward mitigating the gradual performance degradation of TTA. Nevertheless, a complete elimination of error accumulation cannot be guaranteed rigorously through regularization. Future research could delve deeper into expanding our efforts to develop an algorithm that achieves error accumulation-free by construction. Furthermore, as tackling the challenge of the temporally correlated testing stream is not the focus of PeTTA, using a small memory bank as in [61, 15] is necessary. It also assumes the features statistics from the source distribution are available (Appdx. E.3, E.4). These constraints potentially limit its scalability in real-world scenarios.

Conclusion. Towards trustworthy and reliable TTA applications, we rigorously study the *performance degradation problem of TTA*. The proposed *recurring TTA* setting highlights the limitations of modern TTA methods, which struggle to prevent the error accumulation when continuously adapting to demanding test streams. Theoretically inspecting a failure case of $\epsilon-$*GMMC* paves the road for designing PeTTA- a simple yet efficient solution that continuously assesses the model divergence for harmonizing the TTA process, balancing adaptation, and collapse prevention.

Acknowledgements

This work was supported by the Jump ARCHES Endowment through the Health Care Engineering Systems Center, JSPS/MEXT KAKENHI JP24K20830, ROIS NII Open Collaborative Research 2024-24S1201, in part by the National Institute of Health (NIH) under Grant R01 AI139401, and in part by the Vingroup Innovation Foundation under Grant VINIF.2021.DA00128.

References

[1] Kartik Ahuja, Ethan Caballero, Dinghuai Zhang, Jean-Christophe Gagnon-Audet, Yoshua Bengio, Ioannis Mitliagkas, and Irina Rish. Invariance principle meets information bottleneck for out-of-distribution generalization. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=jlchsFOLfeF.

[2] Rahaf Aljundi, Eugene Belilovsky, Tinne Tuytelaars, Laurent Charlin, Massimo Caccia, Min Lin, and Lucas Page-Caccia. Online continual learning with maximal interfered retrieval. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/15825aee15eb335cc13f9b559f166ee8-Paper.pdf.

[3] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/e562cd9c0768d5464b64cf61da7fc6bb-Paper.pdf.

[4] Amitav Banerjee, U. B. Chitnis, S. L. Jadhav, J. S. Bhawalkar, and S. Chaudhury. Hypothesis testing, type I and type II errors. *Industrial Psychiatry Journal*, 18(2):127–131, 2009. ISSN 0972-6748. doi: 10.4103/0972-6748.62274. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2996198/.

[5] Richard Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, USA, 1957.

[6] Arno Blaas, Andrew Miller, Luca Zappella, Joern-Henrik Jacobsen, and Christina Heinze-Deml. Considerations for distribution shift robustness in health. In *ICLR 2023 Workshop on Trustworthy Machine Learning for Healthcare*, 2023. URL https://openreview.net/forum?id=y7XveyWYzIB.

[7] Malik Boudiaf, Romain Mueller, Ismail Ben Ayed, and Luca Bertinetto. Parameter-free online test-time adaptation. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8334–8343, 2022. doi: 10.1109/CVPR52688.2022.00816.

[8] Dian Chen, Dequan Wang, Trevor Darrell, and Sayna Ebrahimi. Contrastive test-time adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, 2022.

[9] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, USA, 2006. ISBN 0471241954.

[10] Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. URL https://openreview.net/forum?id=SSKZPJCt7B.

[11] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7):3366–3385, 2022. doi: 10.1109/TPAMI.2021.3057446.

[12] Mario Döbler, Robert A. Marsden, and Bin Yang. Robust mean teacher for continual and gradual test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7704–7714, June 2022.

[13] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1180–1189, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/ganin15.html.

[14] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. *Domain-Adversarial Training of Neural Networks*, pages 189–209. Springer International Publishing, 2017. doi: 10.1007/978-3-319-58347-1_10. URL https://doi.org/10.1007/978-3-319-58347-1_10.

[15] Taesik Gong, Jongheon Jeong, Taewon Kim, Yewon Kim, Jinwoo Shin, and Sung-Ju Lee. NOTE: Robust continual test-time adaptation against temporal correlation. In *Advances in Neural Information Processing Systems*, 2022.

[16] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In L. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 17, 2004. URL https://proceedings.neurips.cc/paper_files/paper/2004/file/96f2b50b5d3613adf9c27049b2a888c7-Paper.pdf.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1026–1034, 2015.

[19] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019.

[20] Dan Hendrycks, Norman Mu, Ekin D. Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. AugMix: A simple data processing method to improve robustness and uncertainty. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[21] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8320–8329, 2021. doi: 10.1109/ICCV48922.2021.00823.

[22] Junyuan Hong, Lingjuan Lyu, Jiayu Zhou, and Michael Spranger. MECTA: Memory-economic continual test-time model adaptation. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=N92hjSf5NNh.

[23] Fabian Isensee, Paul F. Jaeger, Simon A. A. Kohl, Jens Petersen, and Klaus H. Maier-Hein. nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature Methods*, 18(2):203–211, February 2021. ISSN 1548-7105. doi: 10.1038/s41592-020-01008-z. URL https://www.nature.com/articles/s41592-020-01008-z.

[24] Yusuke Iwasawa and Yutaka Matsuo. Test-time classifier adjustment module for model-agnostic domain generalization. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 2427–2440, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/1415fe9fea0fa1e45dddcff5682239a0-Paper.pdf.

[25] Michael N. Katehakis and Arthur F. Veinott. The multi-armed bandit problem: Decomposition and computation. *Mathematics Operations Research*, 12:262–268, 1987. URL https://api.semanticscholar.org/CorpusID:656323.

[26] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

[27] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114. URL https://www.pnas.org/doi/abs/10.1073/pnas.1611835114.

[28] Dong-Hyun Lee. Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*, 07 2013.

[29] T. Lee, S. Chottananurak, T. Gong, and S. Lee. Aetta: Label-free accuracy estimation for test-time adaptation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 28643–28652, Los Alamitos, CA, USA, jun 2024. IEEE Computer Society. doi: 10.1109/CVPR52733.2024.02706. URL https://doi.ieeecomputersociety.org/10.1109/CVPR52733.2024.02706.

[30] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C. Kot. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[31] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. In *International Conference on Learning Representations Workshop*, 2017. URL https://openreview.net/forum?id=BJuysoFeg.

[32] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2018. doi: 10.1109/TPAMI.2017.2773081.

[33] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? Source hypothesis transfer for unsupervised domain adaptation. In *International Conference on Machine Learning (ICML)*, pages 6028–6039, 2020.

[34] Sen Lin, Peizhong Ju, Yingbin Liang, and Ness Shroff. Theory on forgetting and generalization of continual learning. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23, 2023.

[35] TorchVision maintainers and contributors. Torchvision: Pytorch's computer vision library. https://github.com/pytorch/vision, 2016.

[36] Robert A Marsden, Mario Döbler, and Bin Yang. Gradual test-time adaptation by self-training and style transfer. *arXiv preprint arXiv:2208.07736*, 2022.

[37] 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.

[38] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[39] A. Tuan Nguyen, Thanh Nguyen-Tang, Ser-Nam Lim, and Philip Torr. TIPI: Test time adaptation with transformation invariance. In *Conference on Computer Vision and Pattern Recognition 2023*, 2023. URL https://openreview.net/forum?id=NVh1cy37Ge.

[40] Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Yaofo Chen, Shijian Zheng, Peilin Zhao, and Mingkui Tan. Efficient test-time model adaptation without forgetting. In *The Internetional Conference on Machine Learning*, 2022.

[41] 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 *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=g2YraF75Tj.

[42] K. R. Parthasarathy. *Introduction to Probability and Measure*, volume 33 of *Texts and Readings in Mathematics*. Hindustan Book Agency, Gurgaon, 2005. ISBN 978-81-85931-55-5 978-93-86279-27-9. doi: 10.1007/978-93-86279-27-9. URL http://link.springer.com/10.1007/978-93-86279-27-9.

[43] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019.

[44] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1406–1415, 2019.

[45] Ori Press, Steffen Schneider, Matthias Kuemmerer, and Matthias Bethge. RDumb: A simple approach that questions our progress in continual test-time adaptation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=VfP6VTVsHc.

[46] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence. *Dataset Shift in Machine Learning*. The MIT Press, 2009. ISBN 0262170051.

[47] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

[48] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5389–5400. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/recht19a.html.

[49] Mooweon Rhee and Tohyun Kim. *Exploration and Exploitation*, pages 543–546. Palgrave Macmillan UK, London, 2018. ISBN 978-1-137-00772-8. doi: 10.1057/978-1-137-00772-8_388. URL https://doi.org/10.1057/978-1-137-00772-8_388.

[50] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, , and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=B1gTShAct7.

[51] Tanin Sirimongkolkasem and Reza Drikvandi. On Regularisation Methods for Analysis of High Dimensional Data. *Annals of Data Science*, 6(4):737–763, December 2019. ISSN 2198-5812. doi: 10.1007/s40745-019-00209-4. URL https://doi.org/10.1007/s40745-019-00209-4.

[52] Yongyi Su, Xun Xu, and Kui Jia. Towards real-world test-time adaptation: Tri-net self-training with balanced normalization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(13):15126–15135, 2024.

[53] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9229–9248. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/sun20b.html.

[54] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 2018.

[55] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 1195–1204, 2017. ISBN 9781510860964.

[56] Daniel Vela, Andrew Sharp, Richard Zhang, Trang Nguyen, An Hoang, and Oleg S. Pianykh. Temporal quality degradation in AI models. *Scientific Reports*, 12(1):11654, July 2022. ISSN 2045-2322. doi: 10.1038/s41598-022-15245-z. URL https://www.nature.com/articles/s41598-022-15245-z.

[57] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=uXl3bZLkr3c.

[58] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, and Tao Qin. Generalizing to unseen domains: A survey on domain generalization. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4627–4635. International Joint Conferences on Artificial Intelligence Organization, 8 2021. doi: 10.24963/ijcai.2021/628. URL https://doi.org/10.24963/ijcai.2021/628. Survey Track.

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

[60] Zachary Young and Robert Steele. Empirical evaluation of performance degradation of machine learning-based predictive models – a case study in healthcare information systems. *International Journal of Information Management Data Insights*, 2(1):100070, 2022. ISSN 2667-0968. doi: https://doi.org/10.1016/j.jjimei.2022.100070. URL https://www.sciencedirect.com/science/article/pii/S2667096822000143.

[61] Longhui Yuan, Binhui Xie, and Shuang Li. Robust test-time adaptation in dynamic scenarios. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15922–15932, 2023.

[62] Hao Zhao, Yuejiang Liu, Alexandre Alahi, and Tao Lin. On pitfalls of test-time adaptation. In *ICLR 2023 Workshop on Pitfalls of limited data and computation for Trustworthy ML*, 2023. URL https://openreview.net/forum?id=0Go_RsG_dYn.

Persistent Test-time Adaptation in Recurring Testing Scenarios
Technical Appendices

Table of Contents

A Related Work

Towards Robust and Practical TTA. While forming the basis, early single-target TTA approaches [53, 57, 39, 41, 33] is far from practice. Observing the dynamic of many testing environments, a continual TTA setting is proposed where an ML model continuously adapts to a sequence of multiple shifts [36, 59]. Meanwhile, recent studies [15, 7] point out that the category distribution realistic streams is highly temporally correlated. Towards real-world TTA setting, Yuan *et al.* [61] launch the *practical TTA* which considers the simultaneous occurrence of the two aforementioned challenges.

For a robust and gradual adaptation, an update via the mean teacher [55] mechanism is exploited in many continual TTA algorithms [59, 61, 12, 22]. To moderate the temporally correlated test stream, common approaches utilize a small memory bank for saving a category-balanced subset of testing samples [15, 61], inspired by the replay methods [50, 2] to avoid forgetting in the task of continual learning [34, 3, 11]. Our study emphasizes another perspective: beyond a supreme performance, a desirable TTA should also *sustain it for an extended duration*.

Temporal Performance Degradation. By studying the quality of various ML models across multiple industry applications [56, 60] the issue of AI "aging" with the temporal model degradation progress, even with data coming from a stable process has been confirmed. In TTA, the continuous changes of model parameters through gradient descent aggravate the situation, as also recently noticed in [45]. Apart from observation, we attempt to investigate and provide *theoretical* insights towards the mechanism of this phenomenon.

Accumulated Errors in TTA. In TTA, the issue of accumulated error has been briefly acknowledged. Previous works strive to avoid drastic changes to model parameters as a good practice. Up to some degree, it helps to avoid performance degradation. Nevertheless, it is still *unclear whether their effectiveness truly eliminates the risk*. To preserve in-distribution performance, regularization [27, 40] or replaying of training samples at test-time [12] have been used. Other studies explore reset (recovering the initial model parameters) strategies [59, 45], periodically or upon the running entropy loss approaches a threshold [41]. Unfortunately, knowledge accumulated in the preceding steps will vanish, and a bad heuristic choice of threshold or period leads to highly frequent model resets. Noteworthy, tuning those hyper-parameters is exceedingly difficult due to the unavailability of the validation set [62]. LAME [7] suggests a post-processing step for adaptation (without updating the parameters). This approach, however, still limits the knowledge accumulation. Our PeTTA is *reset-free* by achieving an adaptable continual test-time training.

B Proof of Lemmas and Theorems

In this section, we prove the theoretical results regarding the $\epsilon-$perturbed Gaussian Mixture Model Classifier ($\epsilon-$GMMC) introduced in Sec. 3.2. We first briefly summarize the definition of model collapse and the static data stream assumption:

Definition 1 (**Model Collapse**). *A model is said to be collapsed from step $\tau \in \mathcal{T}, \tau < \infty$ if there exists a non-empty subset of categories $\tilde{\mathcal{Y}} \subset \mathcal{Y}$ such that $\Pr\{Y_t \in \tilde{\mathcal{Y}}\} > 0$ but the marginal $\Pr\{\hat{Y}_t \in \tilde{\mathcal{Y}}\}$ converges to zero in probability:*

$$\lim_{t \to \tau} \Pr\{\hat{Y}_t \in \tilde{\mathcal{Y}}\} = 0.$$

Assumption 1 (**Static Data Stream**). *The marginal distribution of the true label follows the same Bernoulli distribution* $\mathrm{Ber}(p_0)$*: $p_{0,t} = p_0$, ($p_{1,t} = p_1 = 1 - p_0$), $\forall t \in \mathcal{T}$.*

Preliminary. Following the same set of notations introduced in the main text, recall that we denoted $p_{y,t} \triangleq \Pr\{Y_t = y\}$, $\hat{p}_{y,t} \triangleq \Pr\{\hat{Y}_t = y\}$ (marginal distribution of the true label Y_t and pseudo label $\hat{Y}_t$ receiving label y, respectively) and $\epsilon_t = \Pr\{Y_t = 1 | \hat{Y}_t = 0\}$ (the false negative rate (FNR) of

$\epsilon-$GMMC). At testing step t, we obtain the following relations:

$$\mathbb{E}_{P_t}\left[X_t|\hat{Y}_t=0\right]=(1-\epsilon_t)\mu_0+\epsilon_t\mu_1, \tag{9}$$

$$\mathbb{E}_{P_t}\left[X_t|\hat{Y}_t=1\right]=\mu_1, \tag{10}$$

$$\mathrm{Var}_{P_t}\left(X_t|\hat{Y}_t=0\right)=(1-\epsilon_t)\sigma_0^2+\epsilon_t\sigma_1^2+\epsilon_t(1-\epsilon_t)(\mu_0-\mu_1)^2, \tag{11}$$

$$\mathrm{Var}_{P_t}\left(X_t|\hat{Y}_t=1\right)=\sigma_1^2. \tag{12}$$

In addition, under Assumption 1, the marginal distribution $P_t(x)$ (also referred as *data distribution* in our setup) is:

$$P_t(x)=\mathcal{N}(x;p_0\mu_0+p_1\mu_1,p_0\sigma_0^2+p_1\sigma_1^2+p_0p_1(\mu_0-\mu_1)^2)\qquad\forall t\in\mathcal{T}. \tag{13}$$

B.1 Proof of Lemma 1

Lemma 1 (Increasing FNR). *Under Assumption 1, a binary ϵ-GMMC would collapsed (Def. 1) with $\lim_{t\to\tau}\hat{p}_{1,t}=0$ (or $\lim_{t\to\tau}\hat{p}_{0,t}=1$, equivalently) if and only if $\lim_{t\to\tau}\epsilon_t=p_1$.*

Proof. Under Assumption 1, we have $\mathbb{E}_{P_t}[X_t]=p_0\mu_0+(1-p_0)\mu_1$. Also note that:

$$\begin{aligned}\mathbb{E}_{P_t}[X_t]&=\mathbb{E}_{P_t}\left[\mathbb{E}_{P_t}\left[X_t|\hat{Y}_t\right]\right]\\&=\mathbb{E}_{P_t}\left[X_t|\hat{Y}_t=0\right]\hat{p}_{0,t}+\mathbb{E}_{P_t}\left[X_t|\hat{Y}_t=1\right]\hat{p}_{1,t}\\&=\left[(1-\epsilon_t)\mu_0+\epsilon_t\mu_1\right]\hat{p}_{0,t}+\mu_1(1-\hat{p}_{0,t})\\&=\left[(1-\epsilon_t)\hat{p}_{0,t}\right]\mu_0+\left[1-\hat{p}_{0,t}(1-\epsilon_t)\right]\mu_1\\&=p_0\mu_0+(1-p_0)\mu_1,\end{aligned} \tag{14}$$

where the second equality follows Eqs. 9-10. Therefore:

$$\hat{p}_{0,t}=\frac{p_0}{1-\epsilon_t}. \tag{15}$$

Eq. 15 shows positive correlation between $\hat{p}_{0,t}$ and ϵ_t. Given $\lim_{t\to\tau}\epsilon_t=p_1$, taking the limit introduces:

$$\lim_{t\to\tau}\hat{p}_{0,t}=\lim_{t\to\tau}\frac{p_0}{1-\epsilon_t}=\frac{p_0}{1-p_1}=1.$$

Similarly, having $\lim_{t\to\tau}\hat{p}_{0,t}=1$, the false negative rate ϵ_t when $t\to\tau$ is:

$$\lim_{t\to\tau}\epsilon_t=1-p_0=p_1.$$

Since $\hat{p}_{0,t}+\hat{p}_{1,t}=1$, $\lim_{t\to\tau}\hat{p}_{1,t}=0$, equivalently. Towards the collapsing point, the model tends to predict a single label (class 0 in the current setup). In addition, the FNR of the model ϵ_t also raises correspondingly. □

B.2 Proof of Lemma 2.

Lemma 2 (ϵ-GMMC After Collapsing). *For a binary ϵ-GMMC model, with Assumption 1, if $\lim_{t\to\tau}\hat{p}_{1,t}=0$ (collapsing), the cluster 0 in GMMC converges in distribution to a single-cluster GMMC with parameters:*

$$\mathcal{N}(\hat{\mu}_{0,t},\hat{\sigma}_{0,t}^2)\xrightarrow{d}\mathcal{N}(p_0\mu_0+p_1\mu_1,p_0\sigma_0^2+p_1\sigma_1^2+p_0p_1(\mu_0-\mu_1)^2).$$

Proof. From Eqs. 9-10, under the increasing type II collapse of $\epsilon-$GMMC setting, the perturbation does not affect the approximation of μ_1. Meanwhile, when ϵ_t increases, one can expect that $\hat{\mu}_{0,t}$

moves further away from μ_0 toward μ_1. Frist, the mean teacher model of GMMC (Eq. 4, main text) gives:

$$\begin{aligned}
\mathbb{E}_{P_t}\left[\hat{\mu}_{0,t}|\hat{Y}_t=1\right] &= \mathbb{E}_{P_{t-1}}\left[\hat{\mu}_{0,t-1}\right], \\
\mathbb{E}_{P_t}\left[\hat{\mu}_{0,t}|\hat{Y}_t=0\right] &= (1-\alpha)\mathbb{E}_{P_{t-1}}\left[\hat{\mu}_{0,t-1}|\hat{Y}_t=0\right] + \alpha\mathbb{E}_{P_t}\left[X_t|\hat{Y}_t=0\right] \\
&= (1-\alpha)\mathbb{E}_{P_{t-1}}\left[\hat{\mu}_{0,t-1}\right] + \alpha\left(\mathbb{E}_{P_t}\left[X_i|\hat{Y}_t=0\right]\right), \\
\mathbb{E}_{P_t}\left[\hat{\mu}_{1,t}|\hat{Y}_t=1\right] &= (1-\alpha)\mathbb{E}_{P_{t-1}}\left[\hat{\mu}_{1,t-1}|\hat{Y}_t=1\right] + \alpha\mathbb{E}_{P_t}\left[X_t|\hat{Y}_t=1\right] \\
&= (1-\alpha)\mathbb{E}_{P_{t-1}}\left[\hat{\mu}_{1,t-1}\right] + \alpha\left(\mathbb{E}_{P_t}\left[X_i|\hat{Y}_t=1\right]\right), \\
\mathbb{E}_{P_t}\left[\hat{\mu}_{1,t}|\hat{Y}_t=0\right] &= \mathbb{E}_{P_{t-1}}\left[\hat{\mu}_{1,t-1}\right].
\end{aligned}$$

By defining $u_{y,t} = \mathbb{E}_{P_t}\left[\hat{\mu}_{y,t}\right]$, we obtain the following recurrence relation between $u_{0,t}$ and $u_{0,t-1}$:

$$\begin{aligned}
u_{0,t} &= \mathbb{E}_{P_t}\left[\hat{\mu}_{0,t}|\hat{Y}_t=0\right]\hat{p}_{0,t} + \mathbb{E}_{P_t}\left[\hat{\mu}_{0,t}|\hat{Y}_t=1\right]\hat{p}_{1,t} \\
&= \left((1-\alpha)u_{0,t-1} + \alpha\mathbb{E}_{P_t}\left[X_t|\hat{Y}_t=0\right]\right)\hat{p}_{0,t} + u_{0,t-1}\hat{p}_{1,t} \\
&= \left[(1-\alpha)\hat{p}_{0,t} + \hat{p}_{1,t}\right]u_{0,t-1} + \alpha\hat{p}_{0,t}\mathbb{E}_{P_t}\left[X_t|\hat{Y}_t=0\right] \\
&= (1-\alpha\hat{p}_{0,t})u_{0,t-1} + \alpha\hat{p}_{0,t}\mathbb{E}_{P_t}\left[X_t|\hat{Y}_t=0\right] \\
&= (1-\alpha\hat{p}_{0,t})u_{0,t-1} + \alpha\hat{p}_{0,t}\left[(1-\epsilon_t)\mu_0 + \epsilon_t\mu_1\right]. \qquad (16)
\end{aligned}$$

Given $\lim_{t\to\tau}\hat{p}_{0,t} = 1$, it follows that $\lim_{t\to\tau}\epsilon_{0,t} = p_1$ by Lemma 1. From this point:

$$u_{0,t} = (1-\alpha)u_{0,t-1} + \alpha\left(p_0\mu_0 + p_1\mu_1\right) \qquad \forall t > \tau.$$

Taking the limit $t \to \infty$:

$$\begin{aligned}
\lim_{t\to\infty} u_{0,t} &= \lim_{t\to\infty}(1-\alpha)u_{0,t-1} + \alpha\left(p_0\mu_0 + p_1\mu_1\right) \\
&= \lim_{t\to\infty}(1-\alpha)^t\hat{\mu}_{0,0} + \alpha\sum_{i=1}^{t}(1-\alpha)^{i-1}\left(p_0\mu_0 + p_1\mu_1\right) \\
&= \lim_{t\to\infty}(1-\alpha)^t\hat{\mu}_{0,0} + (1-(1-\alpha)^t)(p_0\mu_0 + p_1\mu_1) \\
&= p_0\mu_0 + p_1\mu_1.
\end{aligned}$$

The second equation is obtained by solving the recurrence relation. When $\lim_{t\to\tau}\hat{p}_{0,t} = 1$, $\{\hat{\mu}_{y,t}\}_{y\in\{0,1\}}$ becomes a deterministic values. Hence, giving $u_{y,t} = \mathbb{E}_{P_t}\left[\hat{\mu}_{y,t}\right] = \hat{\mu}_{0,t} (\forall t > \tau)$ and

$$\lim_{t\to\infty}\hat{\mu}_{0,t} = \lim_{t\to\infty} u_{0,t} = p_0\mu_0 + p_1\mu_1. \qquad (17)$$

Repeating the steps above with Eqs. 11-12 in place of Eqs. 9-10, we obtain a similar result for $\sigma^2_{0,t}$:

$$\lim_{t\to\infty}\hat{\sigma}^2_{0,t} = p_0\sigma_0^2 + p_1\sigma_1^2 + p_0p_1(\mu_0-\mu_1)^2. \qquad (18)$$

By *Lévy's continuity theorem* (p. 302, [42]), from Eqs. 17-18, when $t \to \infty$, the estimated distribution of the first cluster $\mathcal{N}(x; \hat{\mu}_{0,t}\hat{\sigma}^2_{0,t})$ converges to the whole data distribution $P_t(x)$ (Eq. 13) when collapsing.

□

B.3 Proof of Theorem 1 and Corollary 1.

Theorem 1 (Convergence of $\epsilon-$GMMC). *For a binary ϵ-GMMC model, with Assumption 1, let the distance from $\hat{\mu}_{0,t}$ toward μ_1 is $d_t^{0\to1} = |\mathbb{E}_{P_t}\left[\hat{\mu}_{0,t}\right] - \mu_1|$, then:*

$$d_t^{0\to1} - d_{t-1}^{0\to1} \leq \alpha\cdot p_0\cdot\left(|\mu_0-\mu_1| - \frac{d_{t-1}^{0\to1}}{1-\epsilon_t}\right).$$

Proof. Substituting Eq. 15 into $\hat{p}_{0,t}$ of Eq. 16 gives:

$$u_{0,t} = \left(1 - \frac{\alpha p_0}{1-\epsilon_t}\right) u_{0,t-1} + \frac{\alpha p_0}{1-\epsilon_t} \left[(1-\epsilon_t)\mu_0 + \epsilon_t \mu_1\right].$$

Hence, we have the distance from $u_{0,t}$ toward μ_1:

$$\begin{aligned}
|u_{0,t} - \mu_1| &= \left| \left(1 - \frac{\alpha p_0}{1-\epsilon_t}\right) u_{0,t-1} + \alpha p_0 \mu_0 + \frac{\alpha p_0 \epsilon_t \mu_1}{1-\epsilon_t} - \mu_1 \right| \\
&= \left| \left(1 - \frac{\alpha p_0}{1-\epsilon_t}\right) (u_{0,t-1} - \mu_1) + \alpha p_0 \mu_0 + \frac{\alpha p_0 \epsilon_t \mu_1}{1-\epsilon_t} - \frac{\alpha p_0 \mu_1}{1-\epsilon_t} \right| \\
&= \left| \left(1 - \frac{\alpha p_0}{1-\epsilon_t}\right) (u_{0,t-1} - \mu_1) + \alpha p_0 \mu_0 - \frac{\alpha p_0 \mu_1 (1-\epsilon_t)}{1-\epsilon_t} \right| \\
&= \left| \left(1 - \frac{\alpha p_0}{1-\epsilon_t}\right) (u_{0,t-1} - \mu_1) + \alpha p_0 (\mu_0 - \mu_1) \right| \\
&\leq \left(1 - \frac{\alpha p_0}{1-\epsilon_t}\right) |u_{0,t-1} - \mu_1| + \alpha p_0 |\mu_0 - \mu_1|.
\end{aligned}$$

The last inequality holds due to the triangle inequality. Equivalently,

$$|u_{0,t} - \mu_1| - |u_{0,t-1} - \mu_1| \leq \alpha \cdot p_0 \cdot \left(|\mu_0 - \mu_1| - \frac{|u_{0,t-1} - \mu_1|}{1-\epsilon_t} \right).$$

Let $d_t^{0\to 1} = |\mathbb{E}_{P_t}[\hat{\mu}_{0,t}] - \mu_1|$, we conclude that:

$$d_t^{0\to 1} - d_{t-1}^{0\to 1} \leq \alpha \cdot p_0 \cdot \left(|\mu_0 - \mu_1| - \frac{d_{t-1}^{0\to 1}}{1-\epsilon_t} \right).$$

□

Corollary 1 (A Condition for $\epsilon-$GMMC Collapse). *With fixed p_0, α, μ_0, μ_1, $\epsilon-$GMMC is collapsed if there exists a sequence of $\{\epsilon_t\}_{\tau-\Delta_\tau}^{\tau}$ ($\tau \geq \Delta_\tau > 0$) such that:*

$$p_1 \geq \epsilon_t > 1 - \frac{d_{t-1}^{0\to 1}}{|\mu_0 - \mu_1|}, \quad t \in [\tau - \Delta_\tau, \tau].$$

Proof. Initialized at μ_0, ϵ-GMMC is collapsing when $\hat{\mu}_{0,t}$ converges to the mid-point $p_0\mu_0 + p_1\mu_1$ (Lemma 2), i.e., moving closer to μ_1. From Thm. 1, the distance towards μ_1 $d_t^{0\to 1} < d_{t-1}^{0\to 1}$ if

$$|\mu_0 - \mu_1| - \frac{|u_{0,t-1} - \mu_1|}{1-\epsilon_t} < 0 \Leftrightarrow |\mu_0 - \mu_1| < \frac{|u_{0,t-1} - \mu_1|}{1-\epsilon_t} \Leftrightarrow \epsilon_t > 1 - \frac{|u_{0,t-1} - \mu_1|}{|\mu_0 - \mu_1|}.$$

When there exists this sequence $\{\epsilon_t\}_{\tau-\Delta_\tau}^{\tau}$ ($\tau \geq \Delta_\tau > 0$) it follows that $d_t^{0\to 1} < d_{t-1}^{0\to 1}$ and $\epsilon_t > \epsilon_{t-1}$ is guaranteed $\forall t \in [\tau - \Delta_\tau, \tau]$. Hence, $\lim_{t\to\tau} \epsilon_t = p_1$ (model collapsed, by Lemma 1). □

C Further Justifications on Gaussian Mixture Model Classifier

One may notice that in ϵ-GMMC (Sec. 4.2), the classifier is defined $f_t(x) = \text{argmax}_{y\in\mathcal{Y}} \Pr(x|y;\theta_t)$ (maximum likelihood estimation) while in general, $f_t(x) = \text{argmax}_{y\in\mathcal{Y}} \Pr(y|x;\theta_t)$ (maximum a posterior estimation), parameterized by a neural network. In this case, since the *equal prior* (i.e., $\Pr(y;\theta_t) = \Pr(y';\theta_t), \forall y, y' \in \mathcal{C}$) is enforced in ϵ-GMMC, the two definitions are *equivalent*.

Proof. Having:

$$\begin{aligned}
\text{argmax}_{y\in\mathcal{Y}} \Pr(y|x;\theta_t) &= \text{argmax}_{y\in\mathcal{Y}} \frac{\Pr(x|y;\theta_t)\Pr(y;\theta_t)}{\sum_{y'\in\mathcal{Y}} \Pr(x|y';\theta_t)\Pr(y';\theta_t)} \\
&= \text{argmax}_{y\in\mathcal{Y}} \Pr(x|y;\theta_t).
\end{aligned}$$

We conclude that the two definitions are equivalent. In fact, it is well-known that maximum likelihood estimation is a special case of maximum a posterior estimation when the prior is uniform. □

D Further Justifications on the Recurring Testing Scenario

D.1 Recurring TTA Follows the Design of a Practical TTA Stream

Note that in recurring TTA, besides the recurrence of environments (or corruptions) as in [59, 40], the distribution of class labels is also temporally correlated (non-i.i.d.) as suggested by [15, 61] to reflect the practical testing stream better. In short, recurring TTA is formed by recurring the environments of *practical TTA* scenario introduced in [61] multiple times (readers are encouraged to visit the original paper for additional motivations on this scenario).

D.2 Recurring TTA as a Diagnostic Tool

Noticeably, CoTTA [59] also performed 10-round repetition across multiple domain shifts to simulate a lifelong TTA testing stream just like our recurring TTA. However, the key difference is CoTTA assumes the distribution of class labels is i.i.d., which does not hold in many real-life testing scenarios as argued in [15, 61]. Our recurring TTA lifts this assumption and allows temporally correlated (non-i.i.d.) label distribution (more challenging, more practical). This extension allows *recurring TTA* to spot the risk of model collapse on CoTTA [59] and other methods. The *over-simplicity* of the repeating scheme in CoTTA for spotting performance degradation is also suggested in [45]. Clearly, it seems not to be a problem at first glance in Tab. 5 of [59] (CoTTA's 10-round repetition), but in fact, the risk in CoTTA remains, as explored in our scenario and also on CCC [45].

The construction of our recurring TTA is notably simple - a technical effort to extend the testing stream. However, this simplicity is on purpose, *serving as a diagnostic tool for lifelong continual TTA*. Counterintuitively, our experiments on four different tasks with the latest methods verify that even if the model is exposed to the same environment *(the most basic case)*, their adaptability and performance are still consistently reduced (demonstrated visually in Fig. 1, quantitatively in Sec. 5.3).

We believe that the extensive testing stream by recurrence in our setup is a *simple yet sufficient scenario* to demonstrate the vulnerability of existing continual TTA methods when facing the issue of model collapse (compared to CCC [45], a notably *more complicated scenario* than our recurring TTA). Indeed, recurring shifts are sufficient to show this failure mode and any lifelong TTA method should necessarily be able to handle recurring conditions.

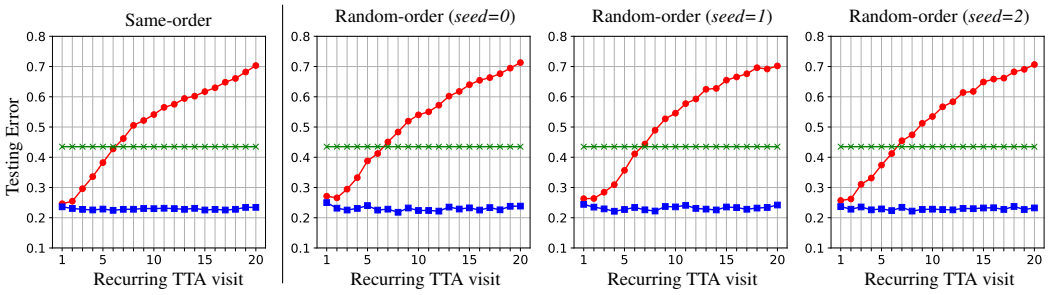
D.3 Recurring TTA with Random Orders

Recall that in Sec. 3.1, *recurring TTA* is constructed by repeating *the same* sequence of D distributions K times. For example, a sequence with $K = 2$ could be $\mathcal{P}_1 \rightarrow \mathcal{P}_2 \rightarrow \cdots \rightarrow \mathcal{P}_D \rightarrow \mathcal{P}_1 \rightarrow \mathcal{P}_2 \rightarrow \cdots \rightarrow \mathcal{P}_D$. For simplicity and consistency that promote reproducibility, the *same order of image corruptions* (following [61]) is used for all recurrences. This section presents supplementary experimental findings indicating that *the order of image corruptions* within each recurrence, indeed, *does not affect* the demonstration of TTA model collapse and the performance of our PeTTA.

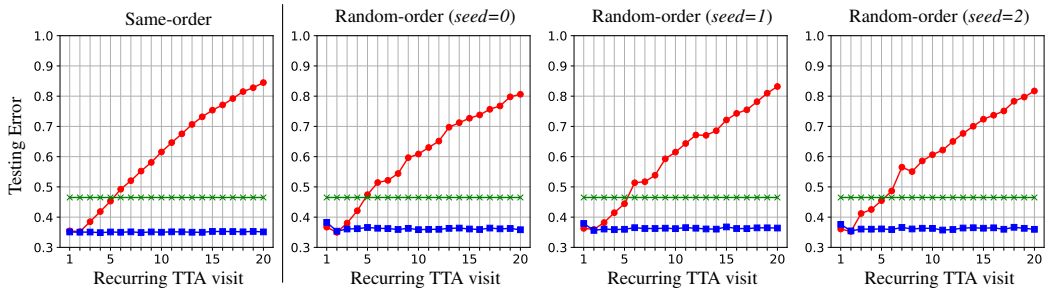
Experiment Setup. We refer to the setting *same-order* as using one order of image corruptions in [61] for all recurrences (specifically, on CIFAR-10/100-C and ImageNet-C: *motion* $\rightarrow$ *snow* $\rightarrow$ *fog* $\rightarrow$ *shot* $\rightarrow$ *defocus* $\rightarrow$ *contrast* $\rightarrow$ *zoom* $\rightarrow$ *brightness* $\rightarrow$ *frost* $\rightarrow$ *elastic* $\rightarrow$ *glass* $\rightarrow$ *gaussian* $\rightarrow$ *pixelated* $\rightarrow$ *jpeg* $\rightarrow$ *impulse*). Conversely, in *random-order*, the order of image corruptions is randomly shuffled at the beginning of each recurrence. Hence, the corruption orders across K recurrences are now entirely different. We redo the experiment of the second setting three times (with different random seeds = $0, 1, 2$). Nevertheless, different TTA methods are ensured to be evaluated on the same testing stream, since it is fixed after generation. Without updating its parameters, the performance of the *source model* is trivially independent of the order of corruptions.

Experimental Result. The experimental results are visualized in Fig. 6. The first column plots the experiments under the *same-order*, while the remaining three columns plot the experiments in the *random-order* setting, with varying random seeds. Note that the message conveyed by each sub-figure entirely matches that of Fig. 1-right.

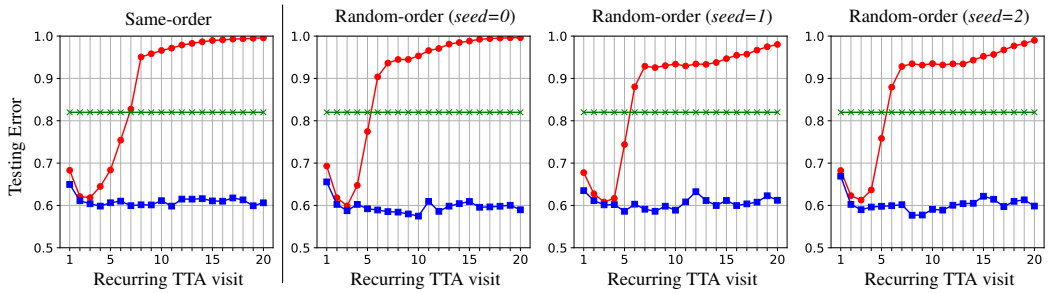
Discussions. Clearly, a similar collapsing pattern is observed in all three TTA tasks, with three combinations of 20 image corruption orders. This pattern also matches the easiest setting using the *same order* of image corruptions we promoted in *recurring TTA*.



(a) CIFAR-10 → CIFAR-10-C task.



(b) CIFAR-100 → CIFAR-100-C task.



(c) ImageNet → ImageNet-C task.

Figure 6: Recurring TTA with different order of corruptions. This figure plots the testing error of two TTA approaches: RoTTA -●- [61], and, PeTTA-■- *(ours)*, and source model-×- as a reference performance under our recurring TTA (with 20 visits) across three TTA tasks. On the *same-order* experiments (column 1), the same order of image corruptions is applied for all 20 visits. Meanwhile, in *random-order*, this order is reshuffled at the beginning of each visit (columns 2-4). Random-order experiments are redone three times with different random seeds. Here, we empirically validate that using the same order of domain shifts (image corruptions) in our recurring TTA is sufficient to showcase the model collapse and evaluate the persistence of our PeTTA. Best viewed in color.

E Further Justifications on Persistent TTA (PeTTA)

E.1 Pseudo Code

We summarize the key steps of our proposed PeTTA in Alg. 1, with the key part (lines 4-13) highlighted in blue. Our approach fits well in the general workflow of a TTA algorithm, *enhancing the regular mean-teacher update step*. Appdx. E.5 elaborates more on our contributions in PeTTA, distinguishing them from other components proposed in previous work. The notations and definitions of all components follow the main text (described in detail in Sec. 4). On line 8 of Alg. 1, as a

Algorithm 1 Persistent TTA (PeTTA)

Input: Classification model f_t and its deep feature extractor ϕ_{θ_t}, both parameterized by $\theta_t \in \Theta$. Testing stream $\{X_t\}_{t=0}^{T}$, initial model parameter (θ_0), initial update rate (α_0), regularization term coefficient (λ_0), empirical mean $(\{\boldsymbol{\mu}_0^y\}_{y\in\mathcal{Y}})$ and covariant matrix $(\{\boldsymbol{\Sigma}_0^y\}_{y\in\mathcal{Y}})$ of feature vectors in the training set, $\hat{\boldsymbol{\mu}}_t^y$ EMA update rate (ν).

$\hat{\boldsymbol{\mu}}_0^y \leftarrow \boldsymbol{\mu}_0^y, \forall y \in \mathcal{Y}$; // Initialization

for $t \in [1, \cdots, T]$ **do**

  $\hat{Y}_t \leftarrow f_{t-1}(X_t)$; // Obtaining pseudo-labels for all samples in X_t

  // Persistent TTA (PeTTA)

  $\hat{\mathcal{Y}}_t \leftarrow \left\{\hat{Y}_t^{(i)} | i = 1, \cdots, N_t\right\}$; // Set of (unique) pseudo-labels in X_t

  $\bar{\gamma}_t \leftarrow 0$;

  for $y \in \hat{\mathcal{Y}}_t$ **do**

    $\gamma_t^y \leftarrow 1 - \exp\left(-(\hat{\boldsymbol{\mu}}_t^y - \boldsymbol{\mu}_0^y)^T (\boldsymbol{\Sigma}_0^y)^{-1} (\hat{\boldsymbol{\mu}}_t^y - \boldsymbol{\mu}_0^y)\right)$; // Divergence sensing term on category y

    $\bar{\gamma}_t \leftarrow \bar{\gamma}_t + \frac{\gamma_t^y}{|\hat{\mathcal{Y}}_t|}$; // Average divergence sensing term for step t

    $\hat{\boldsymbol{\mu}}_t^y \leftarrow (1-\nu)\hat{\boldsymbol{\mu}}_{t-1}^y + \nu\phi_{\theta_{t-1}}(X_t | \hat{Y}_t = y)$; // EMA update of $\hat{\boldsymbol{\mu}}_t^y$ for samples with $\hat{Y}_t = y$

  end

  $\lambda_t \leftarrow \bar{\gamma}_t \cdot \lambda_0$; // Computing adaptive regularization term coefficient

  $\alpha_t \leftarrow (1 - \bar{\gamma}_t) \cdot \alpha_0$; // Computing adaptive update rate

  // Regular Mean-teacher Update

  $\theta_t' \leftarrow \underset{\theta' \in \Theta}{\texttt{Optim}}\, \mathbb{E}_{P_t}\left[\mathcal{L}_{\text{CLS}}\left(\hat{Y}_t, X_t; \theta'\right) + \mathcal{L}_{\text{AL}}\left(X_t; \theta'\right)\right] + \lambda_t \mathcal{R}(\theta')$; // Student model update

  $\theta_t \leftarrow (1-\alpha_t)\theta_{t-1} + \alpha_t \theta_t'$. ; // Teacher model update

  // Final prediction

  yeild $f_t(X_t)$; // Returning the final inference with updated model f_t

end

shorthand notation, $\phi_{\theta_{t-1}}(X_t | \hat{Y}_t = y)$ denotes the empirical mean of all feature vectors of $X_t^{(i)}$ (extracted by $\phi_{\theta_{t-1}}\left(X_t^{(i)}\right)$) if $\hat{Y}_t^{(i)} = y, i = 1, \cdots, N_t$ in the current testing batch.

E.2 Anchor Loss

KL Divergence Minimization-based Interpretation of Anchor Loss. In Sec. 4, we claimed that minimizing the anchor loss $\mathcal{L}_{\text{AL}}$ is equivalent to minimizing the relative entropy (or KL divergence) between the output probability of two models parameterized by θ_0 and θ.

Proof. Having:

$$
\begin{aligned}
D_{KL}\left(\Pr(y|X_t;\theta_0) || \Pr(y|X_t;\theta)\right) &= \sum_{y\in\mathcal{Y}} \Pr(y|X_t;\theta_0) \log \frac{\Pr(y|X_t;\theta_0)}{\Pr(y|X_t;\theta)} \\
&= \underbrace{-\sum_{y\in\mathcal{Y}} \Pr(y|X_t;\theta_0) \log \Pr(y|X_t;\theta)}_{\mathcal{L}_{\text{AL}}(X_t;\theta)} - \underbrace{H(\Pr(y|X_t;\theta_0))}_{\text{constant}}.
\end{aligned}
$$

Hence,

$$
\underset{\theta\in\Theta}{\operatorname{argmin}}\, \mathcal{L}_{\text{AL}}(X_t;\theta) = \underset{\theta\in\Theta}{\operatorname{argmin}}\, D_{KL}\left(\Pr(y|X_t;\theta_0) || \Pr(y|X_t;\theta)\right).
$$

□

Intuitively, a desirable TTA solution should be able to adapt to novel testing distributions on the one hand, but it should *not* significantly diverge from the initial model. $\mathcal{L}_{\text{AL}}$ fits this purpose, constraining the KL divergence between two models at each step.

Connections between Anchor Loss and Regularizer Term. While supporting the same objective (collapse prevention by avoiding the model significantly diverging from the source model), the major difference between Anchor loss ($\mathcal{L}_{\text{AL}}$) and the Regularizer term ($\mathcal{R}(\theta)$) is that the anchor loss operates on the probability space of model prediction while the regularizer term works on the model parameter spaces. Tab. 4 (lines 1 and 5) summarizes the ablation study when each of them is eliminated. We see the role of the regularization term is crucial for avoiding model collapse, while the anchor loss guides the adaptation under the drastic domain shift. Nevertheless, fully utilizing all components is suggested for maintaining TTA persistence.

E.3 The Use of the Memory Bank

The size of Memory Bank. The size of the memory bank in PeTTA is *relatively small, equal to the size of one mini-batch for update* (64 images, specifically).

The Use of the Memory Bank in PeTTA is Fair with Respect To the Compared Methods. Our directly comparable method - RoTTA [61] also takes this advantage (referred to as category-balanced sampling, Sec. 3.2 of [61]). Hence, the comparison between PeTTA and RoTTA *is fair* in terms of additional memory usage. Noteworthy, the use of a memory bank is a *common practice* in TTA literature (e.g., [15, 8, 61]), especially in situations where the class labels are temporally correlated or non-i.i.d. distributed (as we briefly summarized in Appdx. A - Related Work section). CoTTA [59], EATA [40] and MECTA [22] (compared method) assume labels are i.i.d. distributed. Hence, a memory bank is unnecessary, but their performance under temporally correlated label distribution has dropped significantly as a trade-off. The RMT [12] (compared method) does not require a memory bank but it needs to cache a portion of the source training set for replaying (Sec. 3.3 in [12]) which even requires *more* resources than the memory bank.

Eliminating the Need for a Memory Bank. As addressing the challenge of temporally correlated label distribution on the testing stream is not the focus of PeTTA, we have conveniently adopted the use of the memory bank proposed in [61]. Since this small additional memory requirement is not universally applied in every real-world scenario, we believe that this is a reasonable assumption, and commonly adopted in TTA practices. Nevertheless, exploring alternative ways for reducing the memory size (e.g., storing the embedded features instead of the original image) would be an interesting future direction.

E.4 Empirical Mean and Covariant Matrix of Feature Vectors on the Source Dataset

Two Ways of Computing μ_0^y and Σ_0^y in Practice. One may notice that in PeTTA, computing γ_t^y requires the *pre-computed empirical mean (μ_0^y) and covariance (Σ_0^y) of the source dataset*. This requirement may not be met in real-world situations where the source data is unavailable. In practice, the empirical mean and covariance matrix computed on the source distribution can be provided in the following two ways:

1. Most ideally, these values are computed directly by inference on the entire training set once the model is fully trained. They will be provided alongside the source-distribution pre-trained model as a pair for running TTA.
2. With only the source pre-trained model available, assume we can sample a set of unlabeled data from the source distribution. The (pseudo) labels for them are obtained by inferring from the source model. Since the source model is well-performed in this case, using pseudo is approximately as good as the true label.

Accessing the Source Distribution Assumption in TTA. In fact, the second way is typically assumed to be possible in previous TTA methods such as EATA [40], and MECTA [22] (a compared method) to estimate a Fisher matrix (for anti-forgetting regularization purposes). Our work - PeTTA *follows the same second setup* as the previous approaches mentioned above. A variation of RMT [12] (a compared method) approach even requires having the fully labeled source data available at test-time for source replaying (Sec. 3.3 of [12]). This variation is used for comparison in our experiments.

We believe that having the empirical mean and covariant matrix pre-computed on a portion of the source distribution in PeTTA *is a reasonable assumption*. Even in the ideal way, revealing the statistics might not severely violate the risk of data privacy leakage or require notable additional computing resources.

Number of Samples Needed for Computation. To elaborate more on the feasibility of setting (2) mentioned above, we perform a small additional experiment on the performance of PeTTA while varying the number of samples used for computing the empirical mean and covariant matrix on the source distribution. In this setting, we use the test set of CIFAR-10, CIFAR-100, DomainNet validation set of ImageNet (original images, without corruption, or the *real* domain test set of DomainNet), representing samples from the source distribution. The total number of images is $10,000$ in CIFAR-10/A00, $50,000$ in ImageNet, and $69,622$ in DomainNet. We randomly sample $25\%, 50\%, 75\%$, and 100% of the images in this set to run PeTTA for 20 rounds of recurring. The result is provided in Tab. 6 below.

Table 6: Average classification error of PeTTA (across 20 visits) with varying sizes of source samples used for computing feature empirical mean (μ_0^y) and covariant matrix (Σ_0^y).

TTA Task	25%	50%	75%	100%
CIFAR-10 → CIFAR-10-C	22.96	22.99	23.03	22.75
CIFAR-100 → CIFAR-100-C	35.01	35.11	35.09	35.15
DomainNet: *real* → *clip* → *paint* → *sketch*	43.18	43.12	43.15	42.89
ImageNet → ImageNet-C	61.37	59.68	61.05	60.46

The default choice of PeTTA is using 100% samples of the validation set of the source dataset. However, we showcase that it is possible to reduce the number of unlabeled samples from the source distribution to compute the empirical mean and covariant matrix for PeTTA, without significantly impacting its performance.

E.5 Novelty of PeTTA

PeTTA is composed of multiple components. Among them, the anchor loss is an existing idea (examples of previous work utilizing this idea are [32, 12]). Similarly, the mean-teacher update; and regularization are well-established techniques and very useful for the continual or gradual TTA scenario. Hence, we do not aim to improve or alternate these components.

Nevertheless, the novelty of our contribution is the *sensing of the divergence and adaptive model update*, in which the importance of minimizing the loss (adaptation) and regularization (collapse prevention) is changed adaptively. In short, we propose a harmonic way of combining those elements adaptively to achieve a persistent TTA process.

The design of PeTTA draws inspiration from a theoretical analysis (Sec. 3.2), empirically surpassing both the conventional reset-based approach [45] (Appdx. F.3) and other continual TTA approaches [61, 12, 59, 22, 7] on our proposed recurring TTA (Sec. 3.1, Appdx. F.1), as well as the previously established CCC [45] benchmark.

F Additional Experimental Results of PeTTA

F.1 Performance of PeTTA Versus Compared Methods

Performance on CIFAR-100-C and Domainnet Datasets. Due to the length constraint, the classification errors on the tasks CIFAR-100→CIFAR-100-C, and *real* → *clipart, painting, sketch* of DomainNet are provided in Tab. 7 and Tab. 8. To prevent model collapse, the adaptability of PeTTA is more constrained. As a result, it requires more time for adaptation initially (e.g., in the first visit) but remains stable thereafter. Generally, consistent trends and observations are identified across all four TTA tasks.

Standard Deviation of PeTTA Performance Across Multiple Runs. For PeTTA experiments marked with (*) in Tab. 1, Tab. 2, Tab. 7, and Tab. 8, the average performance across five independent runs with different random seeds is reported. Due to the space constraint, the corresponding standard deviation values are now reported in Tab. 9. Generally, the average standard deviation across runs

Table 7: Average classification error of the task CIFAR-100 → CIFAR-100-C in *recurring TTA* scenario. The lowest error is highlighted in **bold**, $^{(*)}$average value across 5 runs (different random seeds) is reported for PeTTA.

Method	*Recurring TTA visit* → 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Avg
Source	46.5																				46.5
LAME [7]	40.5																				40.5
CoTTA [59]	53.4	58.4	63.4	67.6	71.4	74.9	78.2	81.1	84.0	86.7	88.8	90.7	92.3	93.5	94.7	95.6	96.3	97.0	97.3	97.6	83.1
EATA [40]	88.5	95.0	96.8	97.3	97.4	97.2	97.2	97.3	97.4	97.5	97.5	97.5	97.6	97.7	97.7	97.7	97.8	97.8	97.7	97.7	96.9
RMT [12]	50.5	48.6	47.9	47.4	47.3	47.1	46.9	46.9	46.6	46.8	46.7	46.5	46.5	46.6	46.5	46.5	46.5	46.5	46.5	46.5	47.1
MECTA [22]	44.8	44.3	44.6	43.1	44.8	44.2	44.4	43.8	43.8	43.9	44.6	43.8	44.4	44.6	43.9	44.2	43.8	44.4	44.9	44.2	44.2
RoTTA [61]	35.5	35.2	38.5	41.9	45.3	49.2	52.0	55.2	58.1	61.5	64.6	67.5	70.7	73.2	75.4	77.1	79.2	81.5	82.8	84.5	61.4
RDumb [45]	36.7	36.7	36.6	36.6	36.7	36.8	36.7	36.5	36.6	36.5	36.7	36.6	36.5	36.7	36.5	36.6	36.6	36.7	36.6	36.5	36.6
ROID [37]	76.4	76.4	76.2	76.2	76.3	76.1	75.9	76.1	76.3	76.3	76.6	76.3	76.8	76.7	76.6	76.3	76.2	76.0	75.9	76.0	76.3
TRIBE [52]	**33.8**	**33.3**	35.3	**34.9**	35.3	**35.1**	37.1	37.2	37.2	39.1	39.2	41.1	41.0	43.1	45.1	45.1	45.0	44.9	44.9	44.9	39.6
PeTTA *(ours)*$^{(*)}$	35.8	34.4	**34.7**	35.0	**35.1**	**35.1**	**35.2**	**35.3**	**35.3**	**35.3**	**35.2**	**35.3**	**35.2**	**35.2**	**35.1**	**35.2**	**35.2**	**35.2**	**35.2**	**35.2**	**35.1**

Table 8: Average classification error of the task *real* → *clipart* → *painting* → *sketch* on DomainNet dataset in *recurring TTA* scenario.

Method	*Episodic TTA visit* → 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Avg
Source	45.3																				45.3
LAME [7]	45.6																				45.6
CoTTA [59]	96.2	97.1	97.4	97.8	98.1	98.2	98.4	98.4	98.4	98.5	98.6	98.6	98.6	98.6	98.6	98.7	98.7	98.7	98.7	98.7	98.3
RMT [12]	76.2	77.1	77.3	77.3	77.2	77.1	76.8	76.9	76.5	76.4	76.4	76.3	76.4	76.2	76.2	76.1	76.4	76.1	76.0	75.8	76.5
MECTA [22]	94.6	98.4	98.6	98.8	99.1	99.0	99.0	99.0	99.0	99.0	99.0	99.0	99.0	99.0	99.0	99.0	99.0	99.0	99.0	99.0	98.7
RoTTA [61]	44.3	43.8	44.7	46.7	48.7	50.8	52.7	55.0	57.1	59.7	62.7	65.1	68.0	70.3	72.7	75.2	77.2	79.6	82.6	85.3	62.1
RDumb [45]	44.3	44.4	44.3	44.5	44.2	44.2	44.3	44.5	44.4	44.2	44.3	44.3	44.3	44.3	44.5	44.3	44.2	44.3	44.4	44.3	44.3
PeTTA *(ours)*$^{(*)}$	**43.8**	**42.6**	**42.3**	**42.3**	**42.6**	**42.8**	**42.8**	**43.0**	**42.9**	**42.9**	**43.1**	**43.0**	**42.9**	**43.0**	**43.0**	**43.1**	**43.0**	**42.8**	**42.9**	**42.9**	**42.9**

stays within $\pm 0.1\%$ for small datasets (CIFAR-10-C, CIFAR-100-C) and $\pm 0.5\%$ for larger datasets (ImageNet-C, DomainNet).

Table 9: Mean and standard deviation classification error of PeTTA on the four datasets: CIFAR-10-C (CF-10-C), CIFAR-100-C (CF-100-C), DomainNet (DN), and ImageNet-C (IN-C) with *recurring TTA* scenario. Each experiment is run 5 times with different random seeds.

Dataset	*Recurring TTA visit* → 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Avg
CF-10-C	24.3 ±0.4	23.0 ±0.3	22.6 ±0.4	22.4 ±0.3	22.4 ±0.3	22.5 ±0.3	22.3 ±0.4	22.5 ±0.2	22.8 ±0.3	22.8 ±0.4	22.6 ±0.4	22.7 ±0.2	22.7 ±0.1	22.9 ±0.3	22.6 ±0.5	22.7 ±0.2	22.6 ±0.2	22.8 ±0.3	22.9 ±0.4	23.0 ±0.5	22.8 ±0.1
CF-100-C	35.8 ±0.4	34.4 ±0.4	34.7 ±0.2	35.0 ±0.2	35.1 ±0.1	35.1 ±0.1	35.2 ±0.2	35.3 ±0.2	35.3 ±0.1	35.3 ±0.2	35.2 ±0.1	35.3 ±0.2	35.2 ±0.2	35.2 ±0.1	35.1 ±0.1	35.2 ±0.1	35.2 ±0.1	35.2 ±0.1	35.2 ±0.2	35.2 ±0.2	35.1 ±0.1
DN	43.8 ±0.1	42.6 ±0.1	42.3 ±0.2	42.3 ±0.2	42.6 ±0.3	42.8 ±0.3	42.8 ±0.3	43.0 ±0.4	42.9 ±0.4	42.9 ±0.4	43.1 ±0.4	43.0 ±0.4	42.9 ±0.4	43.0 ±0.3	43.0 ±0.3	43.1 ±0.2	43.0 ±0.4	42.8 ±0.3	42.9 ±0.3	42.9 ±0.3	42.9 ±0.3
IN-C	65.3 ±0.6	61.7 ±0.5	59.8 ±0.5	59.1 ±0.5	59.4 ±1.4	59.6 ±1.1	59.8 ±1.0	59.3 ±0.5	59.4 ±0.8	60.0 ±0.9	60.3 ±0.4	61.0 ±0.8	60.7 ±0.9	60.4 ±0.8	60.6 ±0.9	60.7 ±0.8	60.8 ±1.0	60.7 ±0.6	60.4 ±0.6	60.2 ±0.7	60.5 ±0.5

F.2 An Inspection of PeTTA

In Fig. 7, we showcase an inspection of our PeTTA on the task CIFAR-10 → CIFAR-10-C [19] in a typical recurring TTA with 20 visits. Specifically, the visualizations of PeTTA parameters ($\bar{\gamma}_t$, λ_t, and α_t), adaptation losses ($\mathcal{L}_{\text{CLS}}, \mathcal{L}_{\text{AL}}$) and regularization term ($\mathcal{R}(\theta)$) are provided. Here, we observe the values of adaptive parameters λ_t and α_t continuously changing through time, as the testing scenarios evolve during recurring TTA. This proposed mechanism *stabilizes* the value of the loss functions, and regularization term, balancing between the two primary objectives: adaptation and preventing model collapse. Thus, *the error rate persists* as a result. A similar pattern is observed on other datasets (CIFAR-100-C [19] and DomainNet [44]).

F.3 Does Model Reset Help?

Experiment Setup. We use the term *"model reset"* to represent the action of *"reverting the current TTA model to the source model"*. This straightforward approach is named RDumb [45]. We thoroughly conducted experiments to compare the performance of RDumb with PeTTA. The implementation of RDumb in this setting is as follows. We employ RoTTA [61] as the base test-time adaptor due to the characteristics of the practical TTA [61] stream. The model *(including model*

parameters, the optimizer state, and the memory bank) is reset after adapting itself to T images.[1] For each dataset, three values of this hyper-parameter T are selected:

- $T = 1,000$: This is the value selected by the RDumb's authors [45]. Unless specifically stated, we use this value when reporting the performance of RDumb [45] in all other tables.
- $T = 10,000$ (CIFAR-10/100-C), $T = 5,000$ (ImageNet-C) and $T = 24,237$ (DomainNet).[2] This value is equal to the number of samples in the test set of a *single corruption type*, i.e., the model is reset exactly after visiting each $\mathcal{P}_i$'s (see Sec. 3.1 for notations). For DomainNet [44], since the number of images within each domain is unequal, the average number of images is used instead.
- $T = 150,000$ (CIFAR-10/100-C), $T = 75,000$ (ImageNet-C) and $T = 72,712$ (DomainNet). This number is equal to the number of samples *in one recurrence* of our recurring TTA, i.e., the model is reset exactly after visiting $\mathcal{P}_1 \rightarrow \cdots \rightarrow \mathcal{P}_D$. Here, $D = 15$ - types of corruptions [19] for CIFAR-10/100-C and ImageNet-C and $D = 3$ for DomainNet (*clipart, painting, sketch*). For example, the model is reset 20 times within a *recurring TTA* setting with 20 recurrences under this choice of T.

The second and the last reset scheme could be interpreted as assuming the model has access to *an oracle model* with a capability of signaling the transitions between domains, or recurrences. Typically, this is *an unrealistic capability in real-world scenarios*, and a desirable continual TTA algorithm should be able to operate independently without knowing when the domain shift happening.

Experimental Results. An empirical comparison between RDumb [45] and our PeTTA are reported in Tab. 10, Tab. 11, Tab. 12 and Tab. 13 for all four tasks.

Table 10: Average classification error comparison between RDumb [45] (a reset-based approach) with different reset frequencies and our PeTTA on CIFAR-10 $\rightarrow$ CIFAR-10-C task.

	Recurring TTA visit $\longrightarrow$																				
Reset Every	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	**Avg**
$T = 1000$	31.1	32.1	32.3	31.6	31.9	31.8	31.8	31.9	31.9	32.1	31.7	32.0	32.5	32.0	31.9	31.6	31.9	31.4	32.3	32.4	31.9
$T = 10000$	25.8	25.9	26.5	26.1	26.4	25.4	25.8	25.8	26.1	26.2	26.1	26.1	26.1	26.1	26.1	25.9	25.5	25.5	25.7	26.2	26.0
$T = 150000$	24.8	25.3	24.3	24.1	25.3	25.4	25.4	24.5	25.0	24.9	25.0	24.8	25.0	24.5	24.9	24.1	24.0	24.7	24.9	24.4	24.8
PeTTA *(ours)*$^{(*)}$	**24.3**	**23.0**	**22.6**	**22.4**	**22.4**	**22.5**	**22.3**	**22.5**	**22.8**	**22.8**	**22.6**	**22.7**	**22.7**	**22.9**	**22.6**	**22.7**	**22.6**	**22.8**	**22.9**	**23.0**	**22.8**

Table 11: Average classification error comparison between RDumb [45] (a reset-based approach) with different reset frequencies and our PeTTA on CIFAR-100-C dataset.

	Recurring TTA visit $\longrightarrow$																				
Reset Every	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	**Avg**
$T = 1000$	36.7	36.7	36.6	36.6	36.7	36.8	36.7	36.5	36.6	36.5	36.7	36.6	36.5	36.7	36.5	36.6	36.6	36.7	36.6	36.5	36.6
$T = 10000$	43.5	43.6	43.7	43.7	43.4	43.5	43.6	43.4	43.5	43.6	43.8	43.5	43.5	43.6	43.4	43.6	43.5	43.8	43.7	43.6	43.6
$T = 150000$	**35.4**	35.4	35.4	35.3	35.4	35.4	35.5	35.6	35.4	35.4	35.5	**35.3**	**35.2**	35.4	**35.1**	35.8	**35.1**	35.6	35.3	35.8	35.4
PeTTA *(ours)*$^{(*)}$	35.8	**34.4**	**34.7**	**35.0**	**35.1**	**35.1**	**35.2**	**35.3**	**35.3**	**35.3**	**35.2**	**35.3**	**35.2**	**35.2**	**35.1**	**35.2**	35.2	**35.2**	**35.2**	**35.2**	**35.1**

Table 12: Average classification error comparison between RDumb [45] (a reset-based approach) with different reset frequencies and our PeTTA on DomainNet dataset.

	Recurring TTA visit $\longrightarrow$																				
Reset Every	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	**Avg**
$T = 1000$	44.3	44.4	44.3	44.5	44.2	44.2	44.3	44.5	44.4	44.2	44.3	44.3	44.3	44.3	44.5	44.3	44.2	44.3	44.4	44.3	44.3
$T = 24237$	44.1	44.3	43.9	44.2	44.1	44.3	44.2	44.4	44.1	44.1	44.0	44.3	44.1	44.0	44.0	44.2	44.1	44.1	44.1	44.4	44.1
$T = 72712$	44.3	44.3	44.0	44.3	44.1	44.3	44.2	44.4	44.2	44.1	44.0	44.1	44.2	44.1	44.1	44.1	44.1	44.0	44.0	44.3	44.2
PeTTA *(ours)*$^{(*)}$	**43.8**	**42.6**	**42.3**	**42.3**	**42.6**	**42.8**	**42.8**	**43.0**	**42.9**	**42.9**	**43.1**	**43.0**	**42.9**	**43.0**	**43.0**	**43.1**	**43.0**	**42.8**	**42.9**	**42.9**	**42.9**

Discussions. Across datasets and reset frequencies, our PeTTA approach is always *better* than RDumb [45]. The supreme performance holds even when RDumb has access to the oracle information that can reset the model exactly at the transition between each domain shift or recurrence. Importantly, this oracle information is typically unavailable in practice.

[1] A slight abuse of notation. T here is the number of images between two consecutive resets, following the notation on Sec. 3 of [45], *not* the sample indices in our notations.

[2] A subset of $5,000$ samples from ImageNet-C are selected following `RobustBench` [10] for a consistent evaluation with other benchmarks.

Table 13: Average classification error comparison between RDumb [45] (a reset-based approach) with different reset frequencies and our PeTTA on ImageNet-C dataset.

	Recurring TTA visit ⟶																				
Reset Every	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	**Avg**
$T = 1000$	72.2	73.0	73.2	72.8	72.2	72.8	73.3	72.7	71.9	73.0	73.2	73.1	72.0	72.7	73.3	73.1	72.1	72.6	73.3	73.1	72.8
$T = 5000$	70.2	70.8	71.6	72.1	72.4	72.6	72.9	73.1	73.2	73.6	73.7	73.9	74.0	74.0	74.3	74.1	74.1	73.8	73.5	71.9	73.0
$T = 75000$	67.0	67.1	67.2	67.5	67.5	67.6	67.8	67.6	67.6	67.6	67.5	67.7	67.6	67.9	68.1	67.9	67.4	67.5	67.7	67.5	67.6
PeTTA *(ours)*$^{(*)}$	**65.3**	**61.7**	**59.8**	**59.1**	**59.4**	**59.6**	**59.8**	**59.3**	**59.4**	**60.0**	**60.3**	**61.0**	**60.7**	**60.4**	**60.6**	**60.7**	**60.8**	**60.7**	**60.4**	**60.2**	**60.5**

Noteworthy, it is clear that the performance of RDumb varies when changing the choice of the reset frequency. For a given choice of T, the better performance on one dataset does not guarantee the same performance on other datasets. For example, $T = 1,000$ - the best empirical value found by RDumb authors [45] on CCC, does not give the best performance on our recurring TTA scenario; the second choice of T negatively impact the performance on many tasks; the third choice gives the best results, but knowing this exact recurrence frequency of the testing stream is unrealistic. The result highlights the challenge in practice when tuning this parameter (too slow/frequent), especially in the TTA setting where a validation set is unavailable. Our PeTTA, in contrast, is reset-free.

F.4 PeTTA with 40 Recurring Visits

To demonstrate the persistence of PeTTA over an even longer testing stream, in Tab. 14 and Fig. 8, we provide the evaluation results of PeTTA on recurring with 40 recurrences.

F.5 The Sensitivity of Hyper-parameter Choices in PeTTA

Table 15: Sensitivity of PeTTA with different choices of λ_0.

Dataset	$\lambda_0 = 1e^0$	$\lambda_0 = 5e^0$	$\lambda_0 = 1e^1$	$\lambda_0 = 5e^1$	$\lambda_0 = 1e^2$
CIFAR-10-C	22.9	22.7	22.8	23.2	24.1
CIFAR-100-C	35.7	35.3	35.1	35.6	36.1
ImageNet-C	61.2	61.0	60.5	61.3	62.4

There are two hyper-parameters in PeTTA: α_0 and λ_0. The initial learning rate of $\alpha_0 = 1e^{-3}$ is used for all experiments. We do not tune this hyper-parameter, and the choice of α_0 is universal across all datasets, following the previous works/compared methods (e.g., RoTTA [61], CoTTA [59]).

Since λ_0 is more specific to PeTTA, we included a sensitive analysis with different choices of λ_0 on PeTTA, evaluated with images from CIFAR-10/100-C and ImageNet-C in Tab. 15. Overall, the choice of λ_0 is not extremely sensitive, and while the best value is $1e^1$ on most datasets, other choices such as $5e^0$ or $5e^1$ also produce roughly similar performance. Selecting λ_0 is intuitive, the larger value of λ_0 stronger prevents the model from collapsing but also limits its adaptability as a trade-off.

In action, λ_0 is an initial value and will be adaptively scaled with the sensing model divergence mechanism in PeTTA, meaning it does not require careful tuning. More generally, this hyper-parameter can be tuned similarly to the hyper-parameters of other TTA approaches, via an additional validation set, or some accuracy prediction algorithm [29] when labeled data is unavailable.

F.6 More Details on the Ablation Study

We provide the detailed classification error for each visit in the recurring TTA setting of each row entry in Tab. 4 (PeTTA Ablation Study): Tab. 16, Tab. 17, Tab. 18, Tab. 19; and Tab. 5 (PeTTA with various choices of regularizers): Tab. 20, Tab. 21, Tab. 22, Tab. 23.

Fig. 9 presents an additional examination of the ablation study conducted on the task CIFAR-100 $\rightarrow$ CIFAR-100-C [19] for our PeTTA approach. We plot the classification error (top) and the value of $\bar{\gamma}_t$ (bottom) for various PeTTA variations. As the model diverges from the initial state, the value of $\bar{\gamma}_t$ increases. Unable to adjust α_t or constraint the probability space via $\mathcal{L}_{\text{AL}}$ limits the ability of PeTTA to prevent model collapse. In all variations with the model collapse in ablation studies, the rapid saturation of $\bar{\gamma}_t$ is all observed. Therefore, incorporating all components in PeTTA is necessary.

Table 16: Average classification error of multiple variations of PeTTA. Experiments on CIFAR10 → CIFAR10-C [19] task.

Method	*Episodic TTA visit* → 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Avg
Baseline w/o $\mathcal{R}(\theta)$	23.5	24.0	27.4	29.9	33.4	35.6	38.0	40.7	43.1	45.0	46.0	48.6	50.0	49.7	50.8	51.5	52.3	53.3	54.3	55.5	42.6
$\mathcal{R}(\theta)$ fixed $\lambda = 0.1\lambda_0$	23.5	24.0	27.2	29.8	33.4	35.3	37.9	40.5	43.3	45.3	46.8	49.3	50.9	51.0	52.1	53.2	54.0	54.8	56.0	57.6	43.3
$\mathcal{R}(\theta)$ fixed $\lambda = \lambda_0$	23.5	23.6	26.2	28.4	31.6	33.5	36.4	38.7	41.1	43.1	44.8	47.6	49.3	49.5	50.9	52.1	53.1	54.2	55.6	57.0	42.0
PeTTA- λ_t	24.9	25.3	26.0	26.4	27.2	26.5	27.2	27.1	27.4	27.7	27.8	28.0	27.5	28.0	27.7	27.4	27.0	27.6	27.8	27.8	27.1
PeTTA- $\lambda_t + \alpha_t$	25.5	24.5	23.7	23.1	23.2	**22.4**	23.3	23.2	23.7	24.1	23.9	24.5	24.3	24.0	23.8	23.9	23.8	24.1	24.6	24.7	23.9
PeTTA- $\lambda_t + \mathcal{L}_{\text{AL}}$	**23.3**	23.9	24.6	25.3	26.2	25.9	26.4	26.6	26.9	26.6	26.7	26.7	26.7	26.8	26.8	27.2	26.9	26.9	26.8	27.0	26.2
PeTTA $\alpha_t + \mathcal{L}_{\text{AL}}$	24.3	**23.0**	**22.6**	**22.4**	**22.4**	22.5	**22.3**	**22.5**	**22.8**	**22.8**	**22.6**	**22.7**	**22.7**	**22.9**	**22.6**	**22.7**	**22.6**	**22.8**	**22.9**	**23.0**	**22.8**

Table 17: Average classification error of multiple variations of PeTTA. Experiments on CIFAR-100 → CIFAR100-C [19] task.

Method	*Episodic TTA visit* → 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Avg
Baseline w/o $\mathcal{R}(\theta)$	40.2	46.3	51.2	54.4	57.3	59.4	61.3	62.6	63.9	65.1	66.3	67.1	68.1	68.9	69.6	70.3	71.1	71.6	72.4	72.9	63.0
$\mathcal{R}(\theta)$ fixed $\lambda = 0.1\lambda_0$	40.5	46.1	51.5	55.1	58.2	60.5	62.6	64.2	65.7	67.3	68.6	69.5	70.6	71.6	72.5	73.4	74.2	74.9	75.8	76.5	65.0
$\mathcal{R}(\theta)$ fixed $\lambda = \lambda_0$	41.8	47.6	52.6	56.1	58.9	60.7	62.5	63.9	65.0	66.2	67.1	68.3	69.5	70.3	71.4	72.4	73.4	74.1	75.0	75.6	64.6
PeTTA- λ_t	39.4	43.4	46.6	49.1	51.0	52.6	53.8	54.7	55.7	56.5	57.1	57.7	58.3	58.8	59.3	59.9	60.6	61.0	61.6	62.1	55.0
PeTTA- $\lambda_t + \alpha_t$	39.4	40.1	40.8	40.7	41.2	41.5	41.4	41.6	41.5	41.5	41.7	41.6	41.8	41.7	41.8	42.0	41.9	41.9	42.0	41.8	41.4
PeTTA- $\lambda_t + \mathcal{L}_{AL}$	36.2	35.6	35.7	36.1	36.2	36.4	36.4	36.5	36.2	36.2	36.6	36.5	36.5	36.6	36.5	36.6	36.5	36.5	36.3	36.5	36.3
PeTTA $\lambda_t + \alpha_t + \mathcal{L}_{\text{AL}}$	**35.8**	**34.4**	**34.7**	**35.0**	**35.1**	**35.1**	**35.2**	**35.3**	**35.3**	**35.3**	**35.2**	**35.3**	**35.2**	**35.2**	**35.1**	**35.2**	**35.2**	**35.2**	**35.2**	**35.2**	**35.1**

Table 18: Average classification error of multiple variations of PeTTA. Experiments on *real* → *clipart, painting, sketch* task from DomainNet [44] task.

Method	*Recurring TTA visit* → 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Avg
Baseline w/o $\mathcal{R}(\theta)$	52.3	69.0	68.6	68.6	69.4	70.5	71.8	73.4	75.6	77.6	78.8	81.0	82.8	84.3	85.9	87.4	88.5	89.9	90.8	92.1	77.9
$\mathcal{R}(\theta)$ fixed $\lambda = 0.1\lambda_0$	52.5	70.0	69.8	70.0	71.1	72.5	74.6	76.1	77.8	80.4	81.9	83.5	85.2	87.2	89.1	90.2	91.5	93.2	94.1	94.9	80.0
$\mathcal{R}(\theta)$ fixed $\lambda = \lambda_0$	54.6	69.8	63.7	56.0	61.7	76.4	70.4	62.5	58.2	76.0	73.6	66.8	58.6	62.3	80.8	75.5	67.0	59.9	59.3	78.3	66.6
PeTTA- λ_t	49.2	64.5	62.4	60.9	59.6	58.6	57.7	57.8	57.6	57.7	58.0	58.5	59.0	59.5	59.8	61.1	62.0	62.6	63.6	64.9	59.7
PeTTA- $\lambda_t + \alpha_t$	43.9	**42.5**	**42.3**	**42.3**	**42.6**	**42.8**	43.1	43.7	43.9	44.3	44.6	45.1	45.4	45.7	45.7	46.1	46.1	46.2	46.5	46.4	44.5
PeTTA- $\lambda_t + \mathcal{L}_{AL}$	**43.6**	**42.5**	42.6	42.6	42.9	43.0	43.3	43.4	43.1	43.2	**43.1**	43.3	43.3	43.2	43.2	43.9	43.7	43.0	43.2	43.5	43.2
PeTTA $\lambda_t + \alpha_t + \mathcal{L}_{\text{AL}}$	43.8	42.6	**42.3**	**42.3**	**42.6**	**42.8**	**42.8**	**43.0**	**42.9**	**42.9**	**43.1**	**43.0**	**42.9**	**43.0**	**43.0**	**43.1**	**43.0**	**42.8**	**42.9**	**42.9**	**42.9**

Table 19: Average classification error of multiple variations of PeTTA. Experiments on ImageNet → ImageNet-C [19] task.

Method	*Recurring TTA visit* → 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Avg
Baseline w/o $\mathcal{R}(\theta)$	66.9	61.9	72.7	93.6	97.4	97.8	98.0	98.2	98.3	98.3	98.4	98.4	98.5	98.5	98.6	98.6	98.6	98.6	98.7	98.7	93.4
$\mathcal{R}(\theta)$ fixed $\lambda = 0.1\lambda_0$	65.5	70.9	79.1	85.2	90.3	92.6	95.8	95.8	95.4	97.3	96.9	97.7	97.9	98.2	98.0	98.7	98.6	98.4	98.4	98.7	92.5
$\mathcal{R}(\theta)$ fixed $\lambda = \lambda_0$	66.5	62.1	73.0	93.5	97.0	97.2	97.5	97.5	97.6	97.5	97.7	97.7	97.7	97.8	97.9	97.9	98.0	98.0	98.0	97.9	92.9
PeTTA- λ_t	65.9	62.1	76.3	96.7	97.0	96.9	96.9	96.9	97.0	97.1	97.0	97.2	97.0	97.1	97.1	97.0	97.0	97.0	97.0	97.0	92.7
PeTTA- $\lambda_t + \alpha_t$	**64.8**	70.5	74.6	75.8	75.5	75.8	76.1	76.2	76.2	76.5	76.7	77.0	76.9	77.4	77.1	77.3	77.2	77.4	77.6	77.4	75.7
PeTTA- $\lambda_t + \mathcal{L}_{AL}$	**64.8**	**61.1**	60.0	59.8	60.4	60.4	61.2	61.2	61.8	61.9	62.1	62.2	62.1	62.9	62.1	62.8	62.7	62.1	62.8	66.6	62.0
PeTTA *(ours)*$^{(*)}$	**65.3**	**61.7**	**59.8**	**59.1**	**59.4**	**59.6**	**59.8**	**59.3**	**59.4**	**60.0**	**60.3**	**61.0**	**60.7**	**60.4**	**60.6**	**60.7**	**60.8**	**60.7**	**60.4**	**60.2**	**60.5**

Table 20: Average classification error of PeTTA with various choices of regularizers. Experiments on CIFAR-10 → CIFAR-10-C [19] task.

Method	*Episodic TTA visit* → 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Avg
L2	25.6	24.8	23.8	23.1	23.2	22.7	23.0	22.7	22.7	22.7	22.8	22.7	22.8	22.7	22.5	**22.3**	**22.2**	**22.4**	**22.7**	**22.8**	23.0
L2+Fisher	25.2	23.7	22.5	21.8	22.3	21.5	22.3	22.1	22.5	22.8	22.6	22.6	**22.6**	22.8	22.6	22.9	22.6	22.9	23.0	23.3	22.7
Cosine	**24.3**	**23.0**	22.6	22.4	22.4	22.5	22.3	22.5	22.8	22.8	22.6	22.7	22.7	22.9	22.6	22.7	22.6	22.8	22.9	23.0	22.8
Cosine+Fisher	25.1	23.8	**22.2**	**21.6**	**22.0**	**21.4**	**22.0**	**21.8**	**22.1**	**22.3**	**22.5**	**22.4**	**22.6**	**22.6**	**22.4**	22.7	22.6	22.8	22.8	23.3	**22.6**

Table 21: Average classification error of PeTTA with various choices of regularizers. Experiments on CIFAR-100 → CIFAR-100-C [19] task.

Method	*Recurring TTA visit* → 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Avg
L2	36.9	35.5	35.5	35.5	35.7	35.6	35.6	35.5	35.5	35.4	35.6	35.5	35.7	35.7	35.7	35.7	35.8	35.5	35.4	35.5	35.6
L2+Fisher	36.8	35.4	35.4	35.8	35.9	36.0	35.9	35.9	35.9	35.8	36.1	36.1	36.1	36.1	36.1	36.1	36.2	36.0	36.0	35.9	36.0
Cosine	**35.8**	**34.4**	**34.7**	**35.0**	**35.1**	**35.1**	**35.2**	**35.3**	**35.3**	**35.3**	**35.2**	**35.3**	**35.2**	**35.2**	**35.1**	**35.2**	**35.2**	**35.2**	**35.2**	**35.2**	**35.1**
Cosine+Fisher	36.7	35.2	35.5	35.6	35.9	35.9	36.1	36.0	36.0	35.9	36.0	36.0	36.0	36.1	36.0	36.0	35.9	35.9	35.9	36.0	35.9

Table 22: Average classification error of PeTTA with various choices of regularizers. Experiments on *real* → *clipart, painting, sketch* task from DomainNet [44] dataset.

Method	*Recurring TTA visit* → 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Avg
L2	43.8	42.7	42.5	42.4	42.8	42.9	43.0	43.1	43.1	43.2	43.4	43.3	43.2	43.3	43.2	43.2	43.4	43.0	43.1	43.1	43.1
L2+Fisher	43.9	42.8	42.7	43.0	43.2	43.4	43.6	43.8	43.9	44.1	44.0	44.2	44.2	44.2	44.4	44.4	44.5	44.5	44.5	44.5	43.9
Cosine	43.8	42.6	**42.3**	**42.3**	**42.6**	**42.8**	**42.8**	**43.0**	**42.9**	**42.9**	**43.1**	**43.0**	**42.9**	**43.0**	**43.0**	**43.1**	**43.0**	**42.8**	**42.9**	**42.9**	**42.9**
Cosine+Fisher	**43.7**	**42.5**	42.5	42.6	42.9	43.2	43.2	43.5	43.4	43.5	43.4	43.5	43.4	43.6	43.5	43.5	43.4	43.5	43.3	43.4	43.3

Table 23: Average classification error of PeTTA with various choices of regularizers. Experiments on ImageNet → ImageNet-C [19] task.

Method	*Recurring TTA visit* → 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Avg
L2	70.8	72.2	71.5	69.8	72.3	69.3	70.3	70.5	70.0	70.8	70.2	72.1	71.4	70.8	70.9	70.9	69.7	71.0	71.1	70.4	70.8
L2+Fisher	70.5	70.0	69.5	69.4	69.6	69.9	69.2	69.3	72.2	70.4	71.0	70.5	71.7	71.5	71.3	68.4	68.6	68.8	68.7	68.7	70.0
Cosine	65.3	**61.7**	**59.8**	**59.1**	**59.4**	**59.6**	**59.8**	**59.3**	**59.4**	**60.0**	**60.3**	**61.0**	**60.7**	**60.4**	**60.6**	**60.7**	**60.8**	**60.7**	**60.4**	**60.2**	**60.5**
Cosine+Fisher	**65.1**	**61.7**	60.9	61.2	61.9	62.6	62.8	63.2	64.2	63.4	64.3	64.4	63.9	64.3	65.8	65.5	64.9	65.0	65.2	65.2	63.8

F.7 More Confusion Matrices in Recurring TTA Setting

For the task CIFAR-10 → CIFAR-10-C [19] in *recurring TTA* setting (with 20 visits), we additionally showcase the confusion matrix of RoTTA [61] (Fig. 10) and our proposed PeTTA (Fig. 11) at each visit. Our PeTTA persistently achieves competitive performance across 20 visits while RoTTA [61] gradually degrades.

G Experimental Details

G.1 Computing Resources

A computer cluster equipped with an Intel(R) Core(TM) 3.80GHz i7-10700K CPU, 64 GB RAM, and one NVIDIA GeForce RTX 3090 GPU (24 GB VRAM) is used for our experiments.

G.2 Experiments on CCC Testing Stream

In this section, we further evaluate the performance of our PeTTA on the testing data stream of Continuous Changing Corruption (CCC) [45] setting. Here we use the baseline accuracy 20%, transition speed 1000, and random seed 44.[3] The compared methods are source model (ResNet 50), PeTTA, RoTTA [61], and RDumb [45]. Noteworthy, different from recurring TTA, the class labels here are i.i.d. distributed. The adaptation configuration of PeTTA follows the same settings as used on ImageNet-C, while the same setting introduced in Sec. F.3, with $T = 1000$ is used for RDumb [45].

G.3 Test-time Adaptation Methods

Pre-trained Model on Source Distribution. Following previous studies [57, 61, 12, 59], only the batch norm layers are updated. As stated in Sec. 5.2, RobustBench [10] and `torchvision` [35] provide pre-trained models trained on source distributions. Specifically, for ImageNet-C and DomainNet experiments, a ResNet50 model [17] pre-trained on ImageNet V2 (specifically, checkpoint `ResNet50_Weights.IMAGENET1K_V2` of `torchvision`) is used. From `RobustBench`, the model with checkpoint `Standard` and `Hendrycks2020AugMix_ResNeXt` [20] are adopted for CIFAR10-C and CIFAR-100-C experiments, respectively. Lastly, experiments on DomainNet dataset utilize the checkpoint (`best_real_2020`) provided in AdaContrast [8] study.[4]

Optimizer. Without specifically stated, Adam [26] optimizer with learning rate equal $1e^{-3}$, and $\beta = (0.9, 0.999)$ is selected as a universal choice for all experiments.

More Details on PeTTA. Since designing the batch normalization layers, and the memory bank is not the key focus of PeTTA, we conveniently adopt the implementation of the Robust Batch Norm layer and the Category-balanced Sampling strategy using a memory bank introduced in RoTTA [61].

[3] https://github.com/oripress/CCC

[4] https://github.com/DianCh/AdaContrast

G.4 The Use of Existing Assets

Many components of PeTTA is utilized from the official repository of RoTTA [61] [5] and RMT [12]. [6] These two assets are released under MIT license. All the datasets, including CIFAR-10-C, CIFAR-100-C and ImageNet-C [19] are publicly available online, released under Apache-2.0 license.[7] DomainNet dataset [44] (cleaned version) is also released for research purposes.[8]

[5] https://github.com/BIT-DA/RoTTA
[6] https://github.com/mariodoebler/test-time-adaptation
[7] https://github.com/hendrycks/robustness
[8] https://ai.bu.edu/M3SDA/

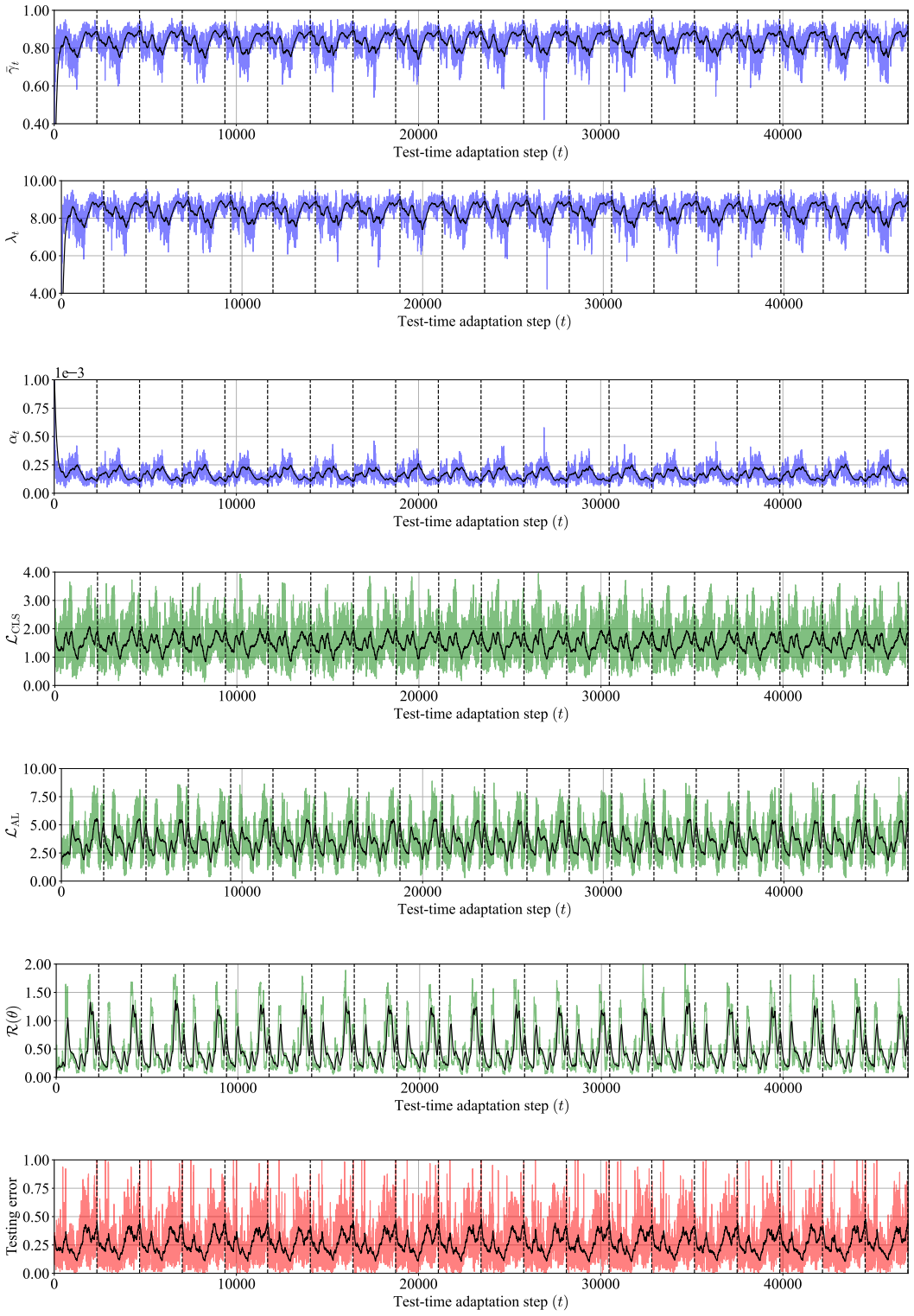


Figure 7: An inspection of PeTTA on the task CIFAR-10 → CIFAR-10-C [19] in a recurring with 20 visits (visits are separated by the *vertical dashed lines*). Here, we visualize (rows 1-3) the dynamic of PeTTA adaptive parameters ($\bar{\gamma}_t, \lambda_t, \alpha_t$), (rows 4-5) the value of the loss functions ($\mathcal{L}_{\text{CLS}}, \mathcal{L}_{\text{AL}}$) and (row 6) the value of the regularization term ($\mathcal{R}(\theta)$) and (row 7) the classification error rate at each step. The **solid line** in the foreground of each plot denotes the running mean. The plots show an adaptive change of λ_t, α_t through time in PeTTA, which stabilizes TTA performance, making PeTTA achieve a persisting adaptation process in all observed values across 20 visits.

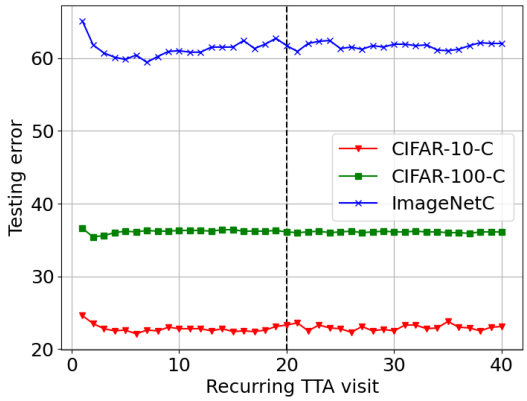


Figure 8: Testing error of PeTTA with 40 recurring TTA visits.

Total Visits	CF-10-C	CF-100-C	IN-C
20 visits	22.8	35.1	60.5
40 visits	22.9	35.1	61.0

Table 14: Average testing error of PeTTA in recurring TTA with 20 and 40 visits. PeTTA demonstrates its persistence over an extended testing time horizon beyond the 20^{th} visit milestone (Fig. 8's horizontal dashed line).

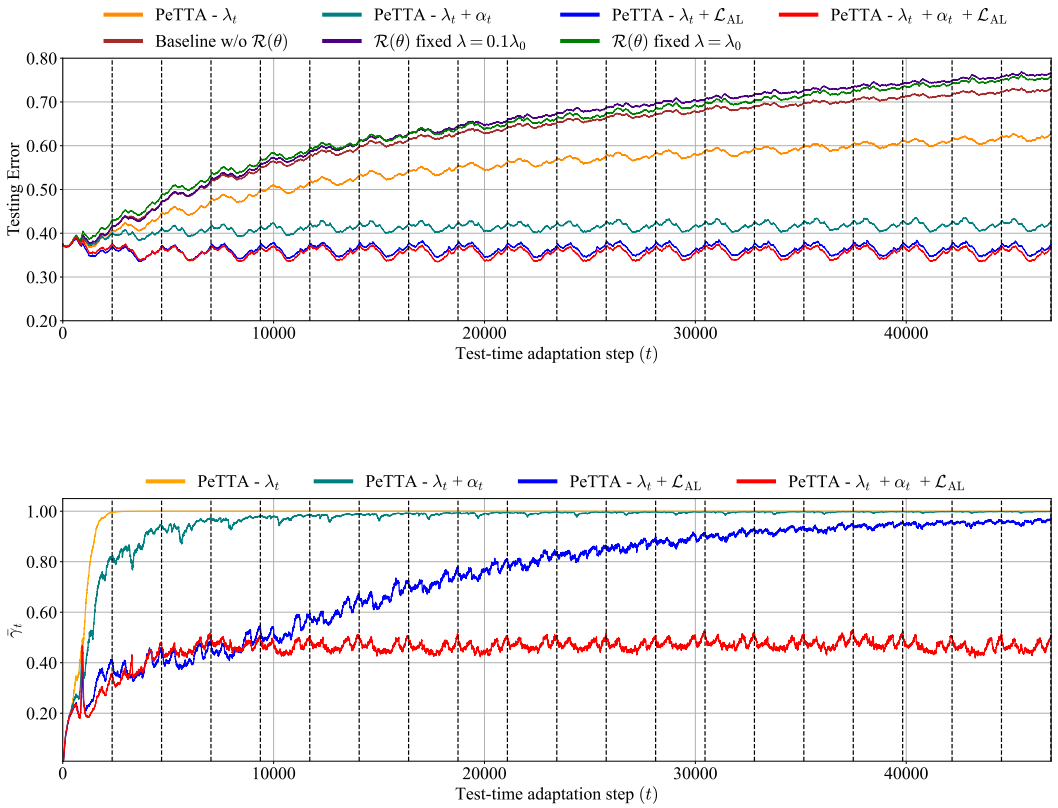


Figure 9: An inspection on the ablation study of multiple variations of PeTTA on the task CIFAR-100 → CIFAR-100-C [19] in an episodic TTA with 20 visits (visits are separated by the vertical dashed lines). **(top)**: testing error of multiple variations of PeTTA. The performance of PeTTA without (w/o) $\mathcal{R}(\theta)$, or fixed regularization coefficient ($\lambda = \lambda_0/0.1\lambda_0$) degrades through time (the top 3 lines). The degradation of PeTTA -λ_t is still happening but at a slower rate (justification below). The performance of the other three variations persists through time with PeTTA -$\lambda_t + \alpha_t + \mathcal{L}_{\text{AL}}$ achieves the best performance. **(bottom)**: changes of $\bar{\gamma}_t$ in multiple variations of PeTTA. When limiting the degree of freedom in adjusting α_t or lacking of supervision from $\mathcal{L}_{\text{AL}}$ (e.g., PeTTA -$\lambda_t + \alpha_t$, PeTTA -$\lambda_t + \mathcal{L}_{\text{AL}}$, and especially PeTTA -$\lambda_t$), the value of γ_t, unfortunately, escalates and eventually saturated. After this point, PeTTA has the same effect as using a fixed regularization coefficient. Therefore, fully utilizing all components is necessary to preserve the persistence of PeTTA. Best viewed in color.

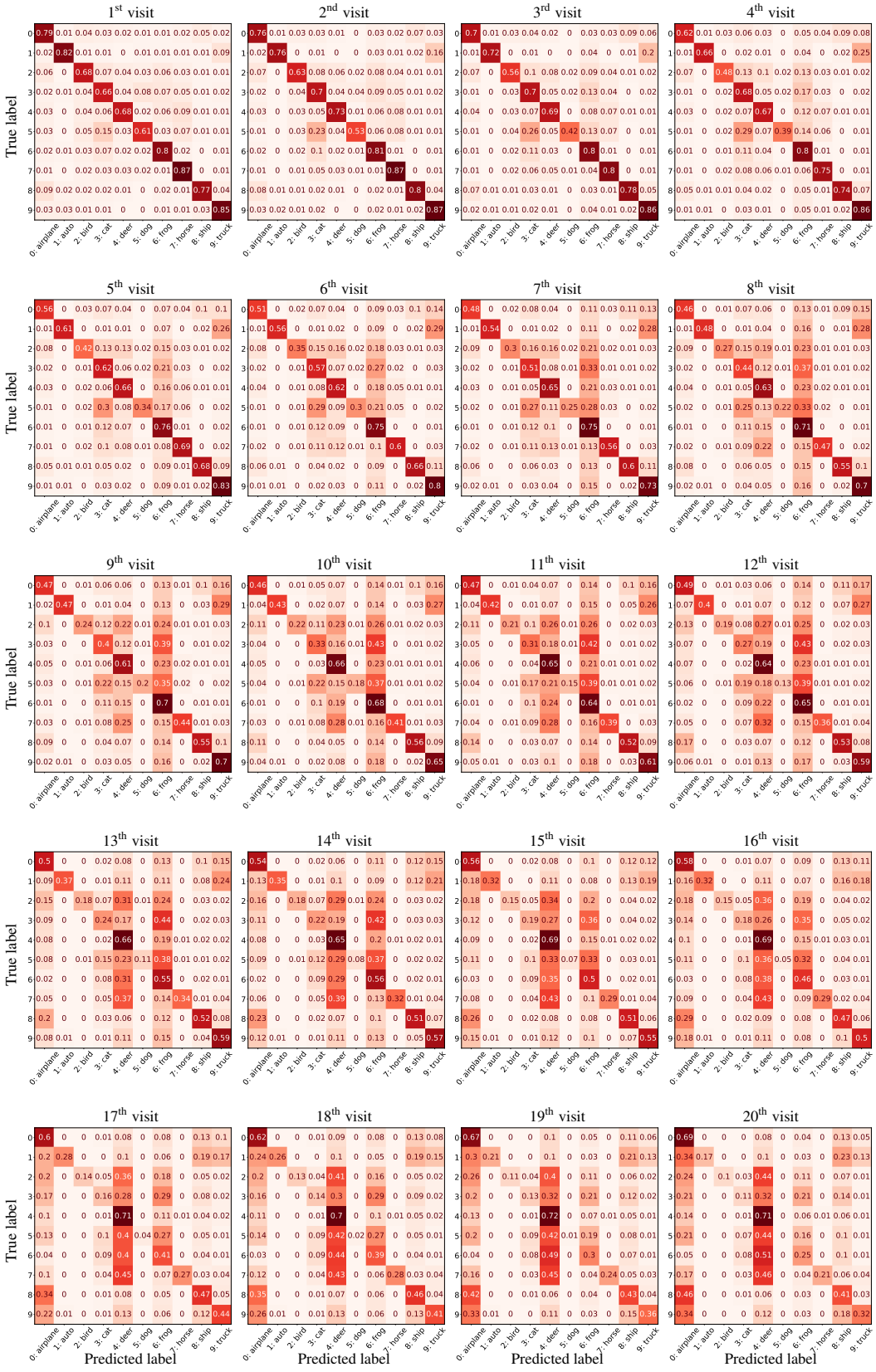


Figure 10: The dynamic of the confusion matrix of RoTTA [61] in episodic TTA with 20 visits.

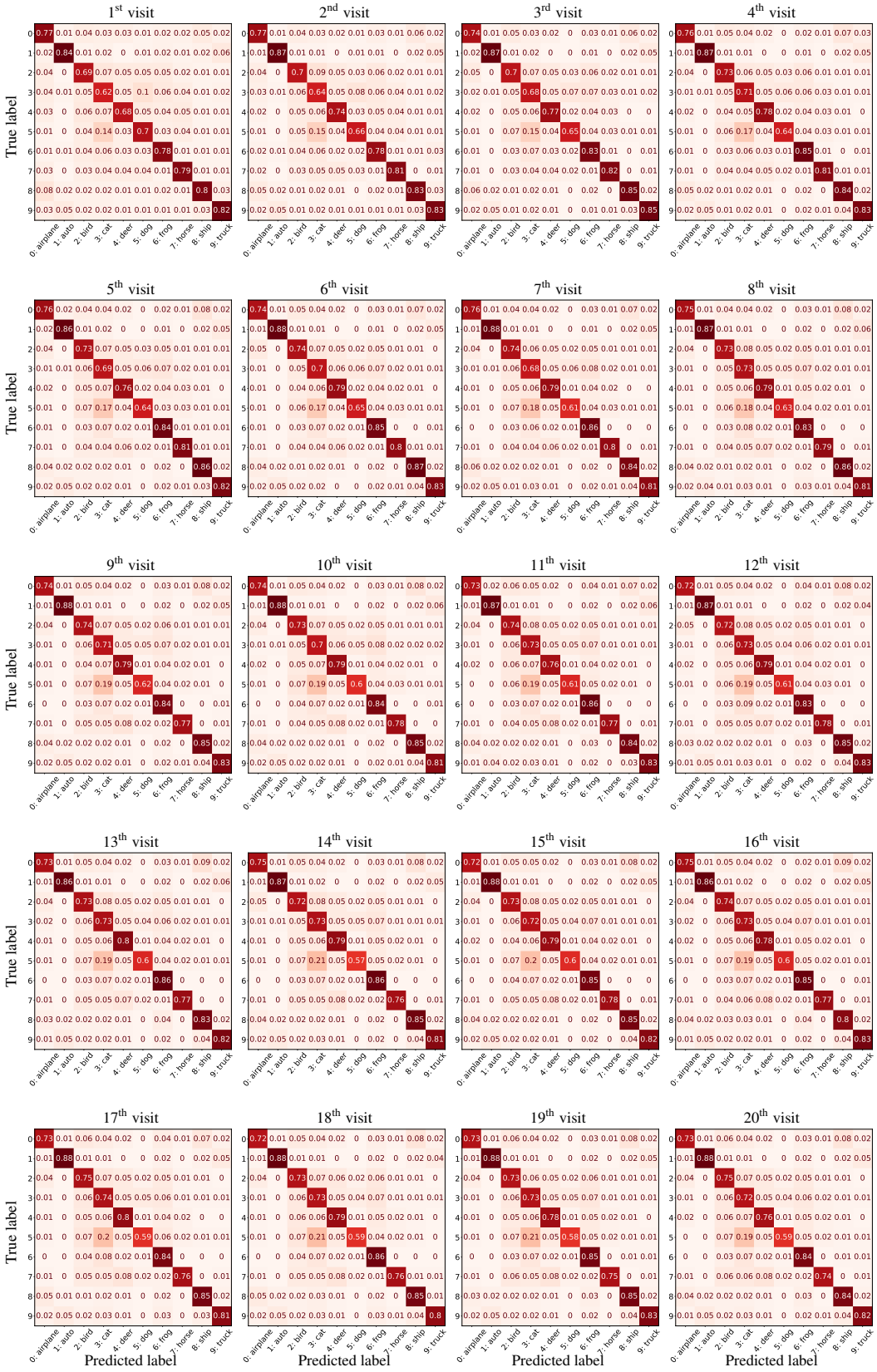


Figure 11: The dynamic of the confusion matrix of PeTTA (*ours*) in episodic TTA with 20 visits.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We have highlighted the three main claims and contributions of our work in both the abstract (highlighted in bold font) and the introduction section (listed as bullet points).

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We have discussed the limitations and potential future work of our study in Sec. 6. Specifically, three main limitations are included: (1) Collapse prevention can not be guaranteed through regularization, PeTTA requires (2) the use of a relatively small memory bank is available and (3) the empirical mean and covariant matrix of feature vectors on the source dataset is computable. We also include discussions in Appdx. E.3 and Appdx. E.4 to further elaborate (2), and (3) respectively.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: We have provided the full proof of all lemmas and theorem in Appdx. B.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: This study propose a new TTA approach - PeTTA. A full description of this approach is given in Sec. 4 with its pseudo-code provided in Appdx. E.1. The implementation of PeTTA in Python is also attached as supplemental material. Additionally, Sec. 5.2 and Appdx. G are dedicated to providing further implementation details for reproducing the main experimental results. Lastly, the construction of recurring TTA is notably simple, and can be easily extended to other TTA streams. Its configuration on each tasks is described in the Recurring TTA paragraph of Sec. 5.2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: This study does not involve any private datasets. All datasets used in our experiments are publicly available online from previous works (more information in Appdx. G.4). The source code of PeTTA is also attached as supplemental material.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The experimental settings of the key results in the paper have been provided in Sec. 5.1 (Simulation Setup) and Sec. 5.2 (Setup - Benchmark Datasets). In the supplementary material, any *additional* experimental results beyond the main paper, such as those in Appdx. D.3, and Appdx. F.3, are consistently preceded by a subsection titled *Experiment Setup* summarizing the experimental details before presenting the results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Due to the limited computing resources, we only extensively evaluate the performance of our proposed method *(PeTTA)* across 5 independent runs, with different random seeds. Specifically, the mean values in 5 runs are reported in Tab. 1, Tab. 2, Tab. 7, and Tab. 8. The corresponding standard deviation values are provided in Appdx. F.1.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We have provided the information on the computing resources used in our experiments in Appdx. G.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The authors have reviewed and to the best of our judgment, this study has conformed to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.

- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: This study advances the research in test-time adaptation area in general, and not tied to particular applications. Hence, there are no significant potential societal consequences of our work which we feel must be specifically highlighted here.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: To the best of our judgment, this study poses no risks for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The original papers that produced the code package or dataset have been properly cited throughout the paper. Further information on the licenses of used assets are provided in Appdx. G.4.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: This study does not release new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: This study does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: This study does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.