CONTRAST: Continual Multi-source Adaptation to Dynamic Distributions

Sk Miraj Ahmed[2,*†], **Fahim Faisal Niloy**[1,*], **Xiangyu Chang**[1], **Dripta S. Raychaudhuri**[3,‡], **Samet Oymak**[4], **Amit K. Roy-Chowdhury**[1]
[1]University of California, Riverside, [2]Brookhaven National Laboratory, [3]AWS AI Labs,
[4]University of Michigan, Ann Arbor
{sahme047@, fnilo001@, cxian008@, drayc001@, amitrc@ece.}ucr.edu, oymak@umich.edu

Abstract

Adapting to dynamic data distributions is a practical yet challenging task. One effective strategy is to use a model ensemble, which leverages the diverse expertise of different models to transfer knowledge to evolving data distributions. However, this approach faces difficulties when the dynamic test distribution is available only in small batches and without access to the original source data. To address the challenge of adapting to dynamic distributions in such practical settings, we propose CONtinual mulTi-souRce Adaptation to dynamic diStribuTions (CONTRAST), a novel method that optimally combines multiple source models to adapt to the dynamic test data. CONTRAST has two distinguishing features. First, it efficiently computes the optimal *combination* weights to combine the source models to adapt to the test data distribution continuously as a function of time. Second, it identifies which of the source model parameters to update so that only the model which is most correlated to the target data is adapted, leaving the less correlated ones untouched; this mitigates the issue of "forgetting" the source model parameters by focusing only on the source model that exhibits the strongest correlation with the test batch distribution. Through theoretical analysis we show that the proposed method is able to optimally combine the source models and prioritize updates to the model least prone to forgetting. Experimental analysis on diverse datasets demonstrates that the combination of multiple source models does at least as well as the best source (with hindsight knowledge), and performance does not degrade as the test data distribution changes over time (robust to forgetting).

1 Introduction

Deep neural networks have shown impressive performance on test inputs that closely resemble the training distribution. However, their performance degrades significantly when they encounter test inputs from a different data distribution. Unsupervised domain adaptation (UDA) techniques [1, 2] aim to mitigate this performance drop. Addressing the distribution shift in case of *dynamic data distributions* is even more challenging and practically relevant - in many real-world applications like autonomous navigation, models often encounter dynamically evolving distributions. Furthermore, test data is often accessed in streaming batches rather than all at once, and source data may not always be available due to privacy and storage concerns.

For domain adaptation to dynamically evolving environments, employing a model ensemble can be beneficial, as it allows leveraging the learned knowledge of different models to more effectively

[*]Equal contribution; Co-first authors listed alphabetically by last name.
[†]Currently at Brookhaven National Laboratory. Work done while the author was at UCR.
[‡]Currently at AWS AI Labs. Work done while the author was at UCR.

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

 https://doi.org/10.52202/079017-4160

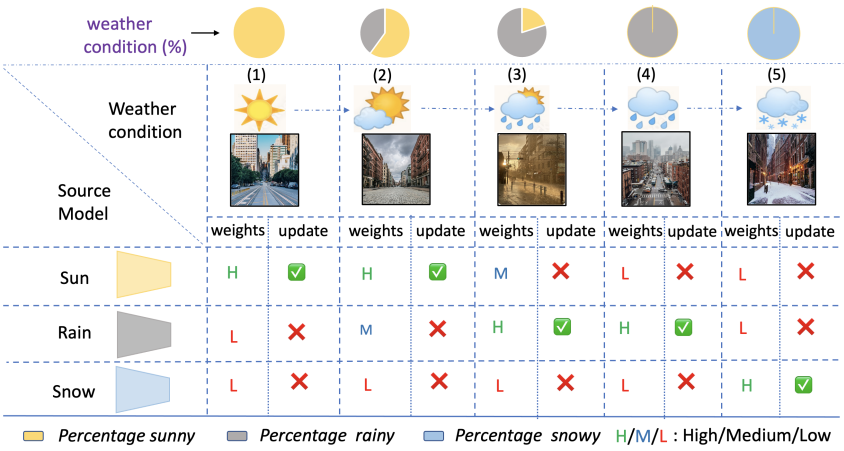

Figure 1: **Problem setup.** Consider several source models trained using data from different weather conditions. During the deployment of these models, they may encounter varying weather conditions that could be a combination of multiple conditions in varying proportions (represented by the pie charts on top). Our goal is to infer on the test data using the ensemble of models by automatically figuring out proper combination weights and adapting the appropriate models on the fly.

mitigate dynamic distribution shifts. Additionally, situations may arise wherein the user has access to a diverse set of pre-trained models across distinct source domains, and no access to source domain data corresponding to each model due to privacy, storage or other constraints. Consequently, training a unified model using the combined source data becomes unfeasible. In those scenarios, it is both reasonable and effective to employ and adapt the entire available array of source models during testing, thereby enhancing performance beyond the scope of single source model adaptation. Moreover, employing a model ensemble provides the flexibility to effortlessly incorporate or exclude models post-deployment, aligning with the user's preferences and the needs of the given task. This flexibility is not achievable with a single domain-generalized model trained on combined source data.

As an example, consider a scenario where a recognition model, initially trained on clear weather conditions, faces data from mixed weather scenarios, like sunshine interspersed with rain (see Figure 1). In such cases, employing multiple models - specifically those trained on clear weather and rain — with appropriate weighting can potentially reduce the test error as opposed to relying on a single source model. In this context, the models for clear weather and rain would be assigned higher weights, while models for other weather conditions would receive relatively lesser weightage.

The main challenge of developing such a model ensembling method is to *learn appropriate combination weights to optimally combine the source model ensemble during the test phase as data is streaming in, such that it results in a test error equal or lower than that of the best source model*. To solve this, we propose CONtinual mulTi-souRce Adaptation to dynamic diStribuTions (CONTRAST) that handles multiple source models and optimally combines them to adapt to the test data.

The efficacy of using multiple source models also extends to preventing *catastrophic forgetting* that may arise when adapting to dynamic distributions for a prolonged time. Consider again the scenario of multiple source models, each trained on a different weather condition. During inference, only the parameters of the models most closely related to the weather encountered during test time will get updates, and the unrelated ones will be left untouched. This ensures that the model parameters do not drift too far from the initial state, since only those related to the test data are being updated. This mechanism mitigates forgetting when the test data distribution varies over a long time scale, as is likely to happen in most realistic conditions. Even if an entirely unrelated distribution appears during testing and there is no one source model to handle it, the presence of multiple sources can significantly reduce the rate at which the forgetting occurs. This is again because only the most closely related models (clear and rainy weather in the example above) are updated, while others (e.g., snow) are left untouched. Our setting is closely related to Test Time Adaptation methods (TTA) [3], and ours is the first to address *adaptation of multiple sources for dynamically shifting distributions* during test time.

Main Contributions. Our proposed approach, CONTRAST, makes the following contributions.

https://doi.org/10.52202/079017-4160

- We propose a framework for multi-source adaptation to dynamic distribution shifts from streaming test data and without access to the source data. Our approach has the ability to merge the source models using appropriate combination weights during test time, enabling it to perform just as well as the best-performing source or even surpass it.
- Our framework achieves performance on par with the best-performing source and also effectively mitigates catastrophic forgetting when faced with long-term, fluctuating test distributions.
- We provide theoretical insights on CONTRAST, illustrating how it addresses domain shift by optimally combining source models and prioritizing updates to the model least prone to forgetting.
- To demonstrate the real-world advantages of our methodology, we perform experiments on a diverse range of benchmark datasets.

2 Related Works

Unsupervised Domain Adaptation. UDA methods have been applied to many machine learning tasks, including image classification [1], semantic segmentation [2], object detection [4] and reinforcement learning [5], in an effort to address the data distribution shift. Most approaches try to align the source and target data distributions, using techniques such as maximum mean discrepancy [6], adversarial learning [7, 1, 5] and image translation [8, 9]. Recently, there has been a growing interest in adaptation using only a pre-trained source model due to privacy and memory storage concerns related to the source data [10–16]. These approaches include techniques such as information maximization [17–19], pseudo labeling [20, 21], and self-supervision [22].

Table 1: **Comparison of our setting to the existing adaptation settings**. Our proposed setting meets all the criteria that are expected in a comprehensive adaptation framework.

Setting	Source Free	Adaptation On the Fly	Dynamic Target	Multi Source
UDA [1]	✗	✗	✗	✓
Source-free UDA [18]	✓	✗	✗	✓
TTA [3]	✓	✓	✓	✗
CONTRAST	✓	✓	✓	✓

Multi-Source Domain Adaptation (MSDA). Both UDA and source-free UDA have been extended to multi-source setting by incorporating knowledge from multiple source models [18, 23]. Notable techniques include discrepancy-based MSDA [24], higher-order moments [25], adversarial methods [26], and Wasserstein distance-based methods [27]. However, these methods are specifically tailored to UDA scenarios, where the whole target data is assumed to be available during adaptation. Whereas, in our setting we consider access to a batch of target data at an instance. Another related field is Domain Generalization (DG) [28, 29] , which refers to training a single model on a combined set of data from different source domains. Hence, DG requires data from all distinct domains to be available altogether during training, which may not be always feasible. Additionally, Model Soups [30] is a popular approach to ensemble models fine-tuned on same data distribution, where the weights of multiple models are averaged for inference. On the other hand, we use a weighting approach for model predictions, where models are pre-trained on different source data distributions. In our problem, inspired by MSDA, *users are only provided with pre-trained source models*.

Adaptation to Dynamic Data. Few works [31–33] have addressed the adaptation to dynamic data distributions. However, these works either require source data or the entire target domain data to be available during adaptation. When additional constraints such as streaming target data batches and no access to source data are considered, the setting closely aligns with Test Time Adaptation (TTA). While UDA methods typically require a substantial volume of target domain data for model adaptation, which is performed offline and prior to deployment, TTA adjusts a model post-deployment, during inference or testing. One of the early works [34] use test-batch statistics for batch normalization adaptation. Tent [3] updates a pre-trained source model by minimizing entropy and updating batch-norm parameters. DUA [35] updates batch-norm stats with incoming test batches. TTA methods have also been applied to segmentation problems [36–39]. When these TTA methods are used to adapt to changing target distribution, they usually suffer from 'forgetting' and 'error accumulation' [40]. In order to solve this, CoTTA [40] restores source knowledge stochastically to avoid drifting of source knowledge. EATA [41] adds a regularization loss to preserve important weights for less forgetting. While motivated by TTA, our method considers multi-source adaptation in a dynamic setting and

https://doi.org/10.52202/079017-4160

has an inherent capability to mitigate forgetting. In Table 1, we illustrate a comparison between our setting and existing settings.

3 CONTRAST Framework

3.1 Problem Setting

In this problem setting, we propose to combine multiple pre-trained models during test time through the application of suitable combination weights, determined based on a limited number of test samples. Specifically, we will focus on the classification task that involves K categories. Consider the scenario where we have a collection of N source models, denoted as $\{\mathrm{f}_S^j\}_{j=1}^N$, that we aim to deploy during test time. In this situation, we assume that a sequence of test data $\{x_i^{(1)}\}_{i=1}^B \rightarrow \{x_i^{(2)}\}_{i=1}^B \rightarrow \ldots \{x_i^{(t)}\}_{i=1}^B \rightarrow \ldots$ are coming batch by batch in an online fashion, where t is the index of time-stamp and B is the number of samples in the test batch. We also denote the test distribution at time-stamp t as $\mathcal{D}_T^{(t)}$, which implies $\{x_i^{(t)}\}_{i=1}^B \sim \mathcal{D}_T^{(t)}$. Motivated by [18], we model the test distribution in each time-stamp t as a linear combination of source distributions where the combination weights are denoted by $\{\mathrm{w}_j^{(t)}\}_{j=1}^N$. Thus, our inference model on test batch t can be written as $\mathrm{f}_T^{(t)} = \sum_{j=1}^N \mathrm{w}_j^{(t)} \mathrm{f}_S^{j(t)}$ where $\mathrm{f}_S^{j(t)}$ is the adapted j-th source in time stamp t. Based on this setup our objective is twofold:

1. We want to determine the optimal combination weights $\{\mathrm{w}_j^{(t)}\}_{j=1}^N$ for the current test batch such that the test error for the optimal inference model is lesser than or equal to the test error of best source model. Mathematically we can write this as follows:
$$\epsilon_{test}^{(t)}(\mathrm{f}_T^{(t)}) \leq \min_{1 \leq j \leq N} \epsilon_{test}^{(t)}(\mathrm{f}_S^j), \tag{1}$$
where $\epsilon_{test}^{(t)}(\cdot)$ evaluates the test error on t-th batch.
2. We also aim for the model to maintain consistent performance on source domains, as it progressively adapts to the changing test conditions. This is necessary to ensure that the model has not catastrophically forgotten the original training distribution of the source domain and maintains its original performance if the source data is re-encountered in the future We would ideally want to have:
$$\epsilon_{src}(\mathrm{f}_S^{j(t)}) \approx \epsilon_{src}(\mathrm{f}_S^j) \quad \forall j, t, \tag{2}$$
where, $\epsilon_{src}(\mathrm{f}_S^j)$ denote the test error of j-th source on its corresponding test data when using the original source model f_S^j, whereas $\epsilon_{src}(\mathrm{f}_S^{j(t)})$ represents the test error on the same test data using the j-th source model adapted up to time step t, denoted as $\mathrm{f}_S^{j(t)}$.

3.2 Overall Framework

Our framework undertakes two operations on each test batch. First, we learn the combination weights for the current batch at time step t by freezing the model parameters. Then, we update the model corresponding to the largest weight with existing state-of-the-art TTA methods, which allows us to fine-tune the model and improve its performance. This implies that the model parameters of source j might get updated up to p times at time-step t, where $0 \leq p \leq (t-1)$.

In other words, the states of the source models evolve over time depending on the characteristics of the test batches up to the previous time step. To formalize this concept, we define the state of the source model j at time-step t as $\mathrm{f}_S^{j(t)}$. In the next section, we will provide a detailed explanation of both aspects of our framework: (i) learning the combination weights, and (ii) updating the model parameters. By doing so, we aim to provide a comprehensive understanding of how our approach works in practice.

3.3 Learning the combination weights

For an unlabeled target sample $x_i^{(t)}$ that arrives at time-stamp t, we denote its pseudo-label, as predicted by source j, as $\hat{y}_{ij}^{(t)} = \mathrm{f}_S^{j(t)}(x_i^{(t)})$, where $\mathrm{f}_S^{j(t)}$ is the state of source j at time-stamp t. Now

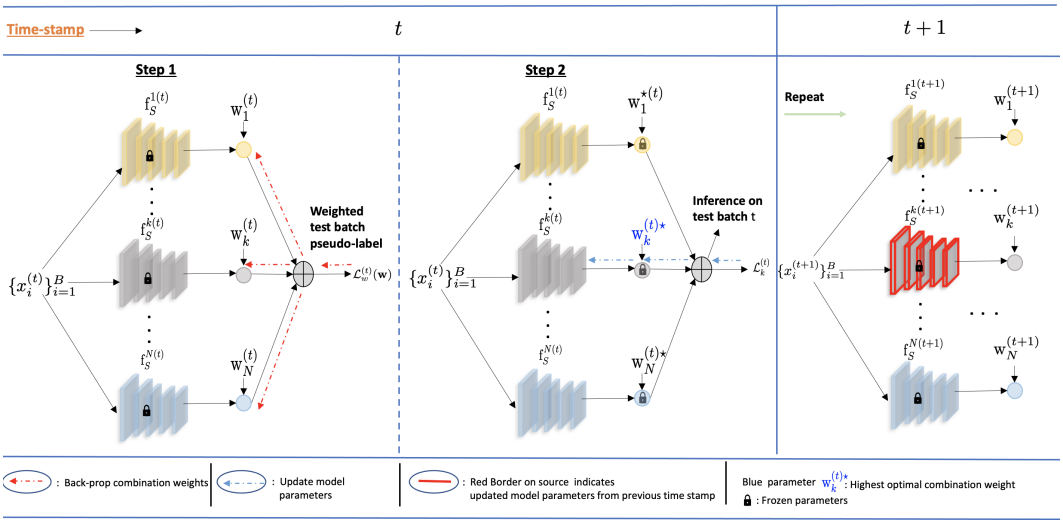

Figure 2: **Overall Framework.** During test time, we aim to adapt multiple source models in a manner such that it optimally blends the sources with suitable weights based on the current test distribution. Additionally, we update the parameters of only one model that exhibits the strongest correlation with the test distribution.

we linearly combine these pseudo-labels by source combination weights $\mathbf{w} = [\mathrm{w}_1\ \mathrm{w}_2 \ldots \mathrm{w}_N]^\top \in \mathbb{R}^N$ to get weighted pseudo-label $\hat{y}_i^{(t)} = \sum_{j=1}^{N} \mathrm{w}_j \hat{y}_{ij}^{(t)}$. Using these weighted pseudo-labels for all the samples in the t-th batch we calculate the expected Shannon entropy as,

$$\mathcal{L}_w^{(t)}(\mathbf{w}) = -\mathbb{E}_{\mathcal{D}_T^{(t)}} \sum_{c=1}^{K} \hat{y}_{ic}^{(t)} \log(\hat{y}_{ic}^{(t)}) \tag{3}$$

Based on this loss we solve the following optimization:

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimize}} \quad & \mathcal{L}_w^{(t)}(\mathbf{w}) \\ \text{subject to} \quad & \mathrm{w}_j \geq 0, \forall j \in \{1, 2, \ldots, N\}, \\ & \sum_{j=1}^{n} \mathrm{w}_j = 1 \end{aligned} \tag{4}$$

Suppose we get $\mathbf{w}^{\star(t)}$ to be the optimal combination weight vector by performing the optimization in (4). In such case, the optimal inference model for test batch t can be expressed as follows:

$$\mathrm{f}_T^{(t)} = \sum_{j=1}^{N} \mathrm{w}_j^{\star(t)} \mathrm{f}_S^{j(t)} \tag{5}$$

Thus, by learning $\mathbf{w}$ in this step, we satisfy Eqn. (1).

Model parameter update. After obtaining $\mathbf{w}^{\star(t)}$, next we select the most relevant source model k given by $k = \arg\max_{1 \leq j \leq N} \mathrm{w}_j^{\star(t)}$. This indicates that the distribution of the current test batch is most correlated with the source model k. We then adapt model k to the test batch t using any state-of-the-art single source method that adapts to dynamic target distributions. Specifically, we employ three distinct adaptation approaches: (i) TENT [3], (ii) CoTTA [40], and (iii) EaTA [41].

Optimization strategy for (4). Solving the optimization problem in Eq. 4 is a prerequisite for inferring the current test batch. As inference speed is critical for test-time adaptation, it is desirable to learn the weights quickly. To achieve this, we design two strategies: (i) selecting an appropriate initialization for $\mathbf{w}$, and (ii) determining an optimal learning rate.
(i) Initialization: Pre-trained models contain information about expected batch mean and variance in their Batch Norm (BN) layers based on the data they were trained on. To leverage this information,

https://doi.org/10.52202/079017-4160

we extract these stored values from each source model prior to adaptation. Specifically, we denote the expected batch mean and standard deviation for the l-th layer of the j-th source model as μ_l^j and σ_l^j, respectively.

During testing on the current batch t, we pass the data through each model and extract its mean and standard deviation from each BN layer. We denote these values as $\mu_l^{T(t)}$ and $\sigma_l^{T(t)}$, respectively. One useful metric for evaluating the degree of alignment between the test data and each source is the distance between their respective batch statistics. A smaller distance implies a stronger correlation between the test data and the corresponding source. Assuming that the batch-mean statistic per node of the BN layers to be a univariate Gaussian, we calculate the distance (KL divergence) between the j-th source (approximated as $\mathcal{N}(\mu_l^j, (\sigma_l^j)^2)$) and the t-th test batch (approximated as $\mathcal{N}(\mu_l^{T(t)}, (\sigma_l^{T(t)})^2)$) as follows (derivation in Appendix Section H):

$$\theta_j^t = \sum_l \mathcal{D}_{KL}\left[\mathcal{N}\left(\mu_l^{T(t)}, (\sigma_l^{T(t)})^2\right), \mathcal{N}\left(\mu_l^j, (\sigma_l^j)^2\right)\right] =$$

$$\sum_{l=1}^{n_j}\sum_{m=1}^{d_j^l} \log\left(\frac{\sigma_{lm}^j}{\sigma_{lm}^{T(t)}}\right) + \frac{\left(\sigma_{lm}^{T(t)}\right)^2 + \left(\mu_{lm}^j - \mu_{lm}^{T(t)}\right)^2}{2\left(\sigma_{lm}^j\right)^2} - \frac{1}{2}$$

where subscript lm denotes the m-th node of l-th layer. After obtaining the distances, we use a softmax function denoted by $\delta(\cdot)$ to normalize their negative values. The softmax function is defined as $\delta_j(a) = \frac{\exp(a_j)}{\sum_{i=1}^N \exp(a_i)}$, where $a \in \mathbb{R}^N$, and $j \in 1, 2, \ldots, N$. If $\theta^t = [\theta_1^t, \theta_2^t \ldots \theta_N^t]^\top \in \mathbb{R}^N$ is the vectorized form of the distances from all the sources, we set

$$\mathbf{w}_{init}^{(t)} = \delta(-\theta^t) \tag{6}$$

where $\mathbf{w}_{init}^{(t)}$ is the initialization for $\mathbf{w}$. As we shall see, this choice leads to a substantial performance boost compared to random initialization.

(ii) Optimal step size: Since we would like to ensure rapid convergence of optimization in Eqn. 4 , we select the optimal step size for gradient descent in the initial stage. Given an initialization $\mathbf{w}_{init}^{(t)}$ and a step size $\alpha^{(t)}$, we compute the second-order Taylor series approximation of the function $\mathcal{L}_w^{(t)}$ at the updated point after one gradient step. Next, we determine the best step size $\alpha_{best}^{(t)}$ by minimizing the approximation with respect to $\alpha^{(t)}$. This is essentially an approximate Newton's method (details in Appendix section I) and has a closed-form solution given by

$$\alpha_{best}^{(t)} = \left[\left(\nabla_{\mathbf{w}}\mathcal{L}_w^{(t)}\right)^\top \left(\nabla_{\mathbf{w}}\mathcal{L}_w^{(t)}\right) / \left(\nabla_{\mathbf{w}}\mathcal{L}_w^{(t)}\right)^\top \mathcal{H}_w \left(\nabla_{\mathbf{w}}\mathcal{L}_w^{(t)}\right)\right]\Bigg|_{\mathbf{w}^{init}}. \tag{7}$$

Here $\nabla_{\mathbf{w}}\mathcal{L}_w^{(t)}$ and $\mathcal{H}_w$ are the gradient and Hessian of $\mathcal{L}_w^{(t)}$ with respect to $\mathbf{w}$. Together with $\mathbf{w}_{init}^{(t)}$ and $\alpha_{best}^{(t)}$, optimization of (4) converges very quickly as demonstrated in the experiments (*in Table 6 of Appendix*). Please note that, *we calculate the Hessian for only n scalar parameters, with n representing the number of source models. Typically, in common application domains, addressing distribution shifts requires only a small number of source models, making the computational overhead of calculating hessian negligible*.

Please refer to Algorithm 1 for a complete overview of CONTRAST.

3.4 Theoretical insights regarding combination weights

Theorem 1 (Convergence of Optimization 4.)**.** *The Optimization 4 converges according to the rule as follows:*

$$\frac{1}{(k+1)}\sum_{j=0}^{k} \|\nabla_{\aleph}\mathcal{L}_w(\mathbf{w}^{(\mathbf{j})})\|_2^2 \leq \frac{2(\mathcal{L}_w(w^{(0)}) - \mathcal{L}_w(w^\star))}{\alpha_{best}^{(t)}(k+1)} \tag{8}$$

where, $\nabla_{\aleph}$ represents the gradient of the objective function over the set of n-simplex $\aleph$ and j represents the iteration number.

Proof. Please refer to the Appendix (Section A) for the proof. □

Implication of Theorem 1. The theorem tells us that to make the optimization converge faster with fewer iterations (small k), it is crucial to start with a good initialization close to the best solution ($(\mathcal{L}(w^{(0)}) - \mathcal{L}(w^\star))$ should be small). By using Eqn. (6), we ensure this condition for quicker convergence. Also, please note that in Theorem 1, j denotes the iteration number in the optimization process, and for simplicity, the batch number t has been intentionally omitted from the notation.

Algorithm 1: Overview of CONTRAST

Input: Pre-trained source models $\{\mathrm{f}_S^j\}_{j=1}^N$, streaming sequential unlabeled test data $\{x_i^{(1)}\}_{i=1}^B \to \{x_i^{(2)}\}_{i=1}^B \to \ldots \{x_i^{(t)}\}_{i=1}^B \to \ldots$

Output: Optimal inference model for t-th test batch $\mathrm{f}_T^{(t)}$ $\forall t$

Initialization: Assign $\mathrm{f}_S^{j(1)} \leftarrow \mathrm{f}_S^j$ $\forall j$

while $t \geq 1$ **do**
- Set initial $\mathbf{w}_{init}^{(t)}$ using Eqn. (6)
- Set $\alpha_{best}^{(t)}$ using Eqn. (7)
- Solve optimization 4 to get $\mathbf{w}^{\star(t)}$
- Infer the test batch t using inference model $\mathrm{f}_T^{(t)}$ using Eqn. (5)
- Find source index k such that $k = \arg\max_{1 \leq j \leq N} \mathrm{w}_j^{\star(t)}$
- Update source model $\mathrm{f}_S^{k(t)}$ according to Model Parameter Update paragraph of Section 3.3 to get $\overline{\mathrm{f}_S^{k(t)}}$
- **for** $1 \leq j \leq N$ **do**
  - **if** $j = k$ **then**
    - Set $\mathrm{f}_S^{j(t+1)} \leftarrow \overline{\mathrm{f}_S^{j(t)}}$
    - **else**
      - Set $\mathrm{f}_S^{j(t+1)} \leftarrow \mathrm{f}_S^{j(t)}$
    - **end**
  - **end**
- **end**

end

3.5 Theoretical insights regarding model update

We now provide theoretical justification on how CONTRAST selects the best source model by optimally trading off model accuracy and domain mismatch. At time t, let $\mathrm{f}_S^{(t)}$ be the set of source models defined as $\left[\mathrm{f}_S^{1(t)}\ \mathrm{f}_S^{2(t)}\ \ldots\ \mathrm{f}_S^{N(t)}\right]$. CONTRAST aims to learn a combination of these models by optimizing weights w on the target domain. For simplicity of exposition, we consider convex combinations $w \in \Delta$ where Δ is the N-dimensional simplex.

To learn $\mathrm{w} \in \Delta$, CONTRAST runs empirical risk minimization on the target task using a loss function $\ell(\cdot)$ with pseudo-labels generated by w-weighted source models. Let $\mathcal{L}(f)$ denote the target population/test risk of a model f (with respect to ground-truth labels) and $\mathcal{L}_T^{\star(t)}$ represent the optimal population risk obtained by choosing the best possible $w \in \Delta$ (i.e. oracle risk). We introduce the functions: **(1)** Ψ which returns the distance between two data distributions and **(2)** φ which returns the distance between two label distributions. We note that, rather than problem-agnostic metrics like Wasserstein, our Ψ, φ definitions are in terms of the loss landscape and source models $\mathrm{f}_S^{(t)}$, hence tighter. We have the following generalization bound at time step t (precise details in Appendix Section A).

Theorem 2. *Consider the model $\mathrm{f}_T^{(t)}$ with combination weights $\mathrm{w}^{\star(t)}$ obtained via CONTRAST by minimizing the empirical risk over B IID target examples per Eqn. 5. Let $\hat{y}_{\mathrm{w}}^{(t)}$ denote the pseudo-label variable of w-weighted source models and $\mathcal{D}_{\mathrm{w}}^{(t)} = \sum_{i=1}^N \mathrm{w}_i^{(t)} \mathcal{D}_{S_i}^{(t)}$ denote weighted source distribution. Under Lipschitz ℓ and bounded $\mathrm{f}_S^{(t)}$, with probability at least $1 - 3e^{-\tau}$ over the target batch, test risk obeys*

$$\underbrace{\mathcal{L}(\mathrm{f}_T^{(t)})}_{\textit{CONTRAST}} - \underbrace{\mathcal{L}_T^{\star(t)}}_{\textit{Optimal}} \leq \min_{\mathrm{w} \in \Delta} \{\underbrace{\Psi(\mathcal{D}_T^{(t)}, \mathcal{D}_{\mathrm{w}}^{(t)})}_{\textbf{\textit{shift}}} + \underbrace{\varphi(\hat{y}_{\mathrm{w}}^{(t)}, y_{\mathrm{w}}^{(t)})}_{\textbf{\textit{quality}}}\} + \sqrt{\tilde{\mathcal{O}}((N+\tau)/B)}.$$

https://doi.org/10.52202/079017-4160

Proof. Please refer to the Appendix (Section A) for the proof. □

Discussion. In a nutshell, this result shows how CONTRAST strikes a balance between: (1) choosing the domain that has the smallest **shift** from target, and (2) choosing a source model that has high-**quality** pseudo-labels on its own distribution (i.e. $\hat{y}_{\mathrm{w}}^{(t)}$ matches $y_{\mathrm{w}}^{(t)}$). From our analysis, it is evident that, rather than adapting the source models to the target distribution, if we simply optimize the combination weights to optimize pseudo-labels for inference, the left side excess risk term ($\mathcal{L}(\mathrm{f}_T^{(t)}) - \mathcal{L}_T^{\star(t)}$) becomes upper bounded by a relatively modest value. This is because the **shift** and **quality** terms on the right-hand side are optimized with respect to w. We note that $\sqrt{N/B}$ is the generalization risk due to finite samples B and search dimension N.

To further refine this, our immediate objective is to tighten the upper bound. This can be achieved by individually adapting each source model to the current test data, all the while maintaining the optimized w constant. Yet, such an approach is not ideal since our second goal is to preserve knowledge from the source during continual adaptation. To attain our desired goal, we must relax the upper bound, reducing our search over $\mathrm{w} \in \hat{\Delta}$. Here, $\hat{\Delta}$ is the discrete counterpart of the simplex Δ. The elements of $\hat{\Delta}$ are one-hot vectors that have all but one entry zero. The elements of $\hat{\Delta}$ essentially represent discrete model selection. Examining the main terms on the right reveals that: (i) source-target distribution shift and (ii) divergence between ground-truth and pseudo-labels are all minimized when we select the source model with the highest correlation to target. This model, denoted by $\mathrm{f}_S^{\star(t)}$, essentially corresponds to the largest entry of $\mathrm{w}^{\star(t)}$ and presents the most stringent upper bound within the $\hat{\Delta}$ search space. Thus, to further minimize the right hand side, the second stage of CONTRAST adapts $\mathrm{f}_S^{\star(t)}$ with the current test data. Crucially, besides minimizing the target risk, this step helps avoids forgetting the source because $\mathrm{f}_S^{\star(t)}$ already does a good job at the target task. Thus, during optimization on target data, $\mathrm{f}_S^{\star(t)}$ will have small gradient and will not move much, resulting in smaller forgetting. Please refer to the Appendix (Section A) for more detailed discussion along with the proof of this theorem.

4 Experiments

Datasets. We demonstrate the efficacy of our approach using both **static target distribution** and **dynamic target data distributions**. For static case, we employ the *Digits* and *Office-Home* datasets [42]. For the dynamic case, we utilize *CIFAR-100C* and *CIFAR-10C* [43]. Detailed descriptions of these datasets along with results on segmentation task can be found in the Appendix.

Baseline Methods. As our problem setting is most closely related to test time adaptation, our baselines are some widely used state-of-the-art (SOTA) single source test time adaptation methods: we specifically compare our algorithm with Tent [3], CoTTA [40] and EaTA [41]. These methods deal with adaptation from small batches of streaming data and without the source data, which is our setting, and hence we compare against these as our baselines. To evaluate the adaptation performance, we follow the protocol similar to [18], where we apply each source model to the test data from a particular test domain individually, which yields X-Best and X-Worst where "X" is the name of the single source adaptation method, representing the highest and lowest performances among the source models adapted using method "X", respectively. For our algorithm, we extend all of the methods "X" in the multi source setting and call the multi-source counterpart of "X" as "X+CONTRAST".

Implementation Details. We use ResNet-18 [44] model for all our experiments. For solving the optimization of Eq. (4), we first initialize the combination weights using Eq. (6) and calculate the optimal learning rate using Eq. (7). After that, we use 5 iterations to update the combination weights using SGD optimizer and the optimal learning rate. For all the experiments we use a batch size of 128, as used by Tent [3]. For more details on implementation and experimental setting see Appendix.

Experiment on CIFAR-100C. We conduct a thorough experiment on this dataset to investigate the performance of our model under dynamic test distribution. We consider 3 corruption noises out of 15 noises from CIFAR-100C, which are adversarial weather conditions namely *Snow*, *Fog* and *Frost*. We add these noises for severity level 5 to the original CIFAR-100C training set and train three source models, one for each noise. Along with these models, we also add the model trained on clean training set of CIFAR-100. During testing, we sequentially adapt the models across the 15 noisy

domains, each with a severity of 5, from the CIFAR-100C dataset [40, 41]. We report the results for the experiment in Table 2. Moreover, we also conduct an experiment on CIFAR10-C with the exact same experimental settings as with CIFAR100-C. *CIFAR-10C results are in Table 5 of Appendix.*

Table 2: **Results on CIFAR-100C.** We take four source models trained on *Clear*, *Snow*, *Fog*, and *Frost*. We employ these models for adaptation on 15 sequential test domains. This table illustrates that even in the dynamic environment X+ CONTRAST performs better than X-Best, which is the direct consequence of optimal aggregation of source models as well as better preservation of source knowledge. (Results in error rate ↓ (in %))

	GN	SN	IN	DB	GB	MB	ZB	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	JPEG	Mean
Source Worst	97.7	96.5	98.2	68.8	78.1	66.1	65.1	53.6	59.3	62.0	55.8	95.4	61.9	71.5	75.2	73.7
Source Best	90.5	89.0	94.5	50.7	48.1	51.9	44.5	30.0	29.5	28.2	39.0	81.9	44.0	38.5	57.1	54.5
Tent Worst	55.9	55.6	71.2	58.0	75.5	78.2	83.3	89.2	92.4	93.7	95.4	96.7	96.5	96.6	96.7	82.3
Tent Best	45.6	43.8	59.1	48.5	59.1	59.1	60.4	65.6	66.1	76.7	75.3	89.8	89.0	91.3	94.2	68.2
Tent + CONTRAST	**42.2**	**40.6**	**55.3**	**28.6**	**40.7**	**31.9**	**29.6**	**31.7**	**32.4**	**30.9**	**28.6**	**41.5**	**38.5**	**34.8**	**49.9**	**37.1**
EaTA Worst	57.7	54.0	66.5	40.6	53.2	41.4	36.8	44.0	43.5	45.4	34.8	45.4	45.7	39.9	55.7	47.0
EaTA Best	48.1	44.7	57.9	37.1	44.1	38.7	34.9	33.7	31.9	31.6	33.2	37.2	40.0	34.7	50.3	39.9
EaTA + CONTRAST	**43.3**	**40.7**	**54.3**	**27.5**	**39.4**	**30.4**	**27.5**	**29.2**	**29.1**	**28.3**	**25.9**	**31.3**	**33.4**	**29.0**	**43.1**	**34.2**
CoTTA Worst	59.2	57.4	68.0	40.1	52.7	42.1	40.5	47.0	46.6	47.2	39.4	43.6	44.5	41.4	47.4	47.8
CoTTA Best	49.8	46.6	58.6	34.0	40.7	36.5	34.2	34.2	32.8	33.0	32.8	34.8	35.3	33.6	41.1	38.5
CoTTA + CONTRAST	**44.6**	**43.8**	**57.2**	**27.8**	**37.6**	**30.6**	**28.0**	**29.3**	**29.3**	**28.2**	**26.6**	**30.0**	**32.5**	**29.7**	**41.4**	**34.4**

From the table, we can draw two key observations:
(i) As anticipated, X+CONTRAST consistently outperforms X-Best across each test distribution, underscoring the validity of our algorithmic proposition.

(ii) Given that the CoTTA and EaTA methods are tailored to mitigate forgetting, the average error post-adaptation across the 15 noises using these methods is significantly lower than that of Tent, which is not designed for this specific challenge. For instance, in Table 2, Tent-Best error is approximately 68.2%, while CoTTA and EaTA-Best are around 39.9% and 38.5%, respectively. However, when these adaptation methods are incorporated into our framework, the final errors are remarkably close: 37.1% for Tent, 34.2% for EaTA, and 36.9% for CoTTA. This suggests that even though Tent is more lightweight and faster compared to the other methods and is not inherently designed to handle forgetting, its performance within our framework is on par with the results obtained when incorporating the other two methods designed to prevent forgetting. This shows the generalizability of our approach to various single-source methods. Note that identical to the experiment on CIFAR100-C, the results on CIFAR10-C in Table 5 follow the same trend where X+CONTRAST outperforms the X-Best.

Table 3: **Results on Office-Home.** We train three source models using three domains in this dataset and use them for testing on the remaining domain under the TTA setting. Our results demonstrate that X+CONTRAST consistently outperforms all of the baselines (X) (% error).

	Ar	Cl	Pr	Rw	Avg.
Source Worst	61.4	64.9	46.2	43.9	54.1
Source Best	42.5	58.5	29.8	35.7	41.6
Tent Worst	57.7	60.4	46.5	42.1	51.7
Tent Best	41.4	54.3	27.9	36.0	39.9
Tent + CONTRAST	**40.7**	**52.5**	**27.4**	**27.4**	**37.0**
EaTA Worst	58.4	64.3	48.0	43.5	53.5
EaTA Best	42.1	57.8	30.3	35.9	41.5
EaTA + CONTRAST	**40.1**	**53.3**	**28.3**	**28.0**	**37.4**
CoTTA Worst	58.3	62.9	47.1	42.8	52.8
CoTTA Best	42.1	55.0	29.0	34.9	40.2
CoTTA + CONTRAST	**40.6**	**53.3**	**28.3**	**29.0**	**37.8**

Experiment on Office-Home. We report the results of the experiment on static distribution using the Office-Home dataset in Table 3. Each column in the table represents a target domain from Office-Home dataset. We train three source models on the remaining Office-Home datasets. For instance, in case of 'Ar' column, 'Ar' is the target domain where three source models trained on 'Cl', 'Pr' and 'Rw' are adapted in test time. We calculate the test error of each incoming test batch and then report the numbers by averaging the error values over all the batches. The table shows that CONTRAST provides a significant reduction of test error compared to the best single source model. This demonstrates that when presented with an incoming test batch, CONTRAST has the capability to effectively blend all available sources using optimal weights, resulting in superior performance compared to the best single source model. It is important to note that each test batch in this experiment is drawn from the same stationary distribution, which represents the distribution of the target domain. We conduct a similar experiment with the same experimental settings on Digits dataset that can be found in Table 4 of Appendix.

https://doi.org/10.52202/079017-4160

Analysis of Forgetting. Here, we demonstrate the robustness of our method against catastrophic forgetting by evaluating the classification accuracy on the source test set after completing adaptation to each domain [41, 45, 46]. For CONTRAST, we use our ensembling method to adapt to the incoming domain. After adaptation, we infer each of the adapted source models on its corresponding source test set. For the baseline single-source methods, every model is adapted individually to the incoming domain, followed by inference on its corresponding source test set. The reported accuracy represents the average accuracy obtained from each of these single-source adapted models.

From Figure 3, we note that our method consistently maintains its source accuracy during the adaptation process across the 15 sequential noises. In contrast, the accuracy for each individual single-source method (X) declines on the source test set as the adaptation process progresses. Specifically, Tent, not being crafted to alleviate forgetting, experiences a sharp decline in accuracy. While CoTTA and EaTA exhibit forgetting, it occurs at a more gradual pace. Contrary to all of these single-source methods, our algorithm exhibits virtually no forgetting throughout the process.

Ablation Study. We conduct an ablation study in Tables 6, 7 in the Appendix to evaluate the impact of various initialization and learning rate strategies on the optimization process described in (4). Our findings demonstrate that the initialization and learning rate configurations generated by our method outperform other alternatives.

Additionally, our experiments in Tables 8, 9 and 10 in the Appendix reveal that selectively updating the most correlated model parameters enhances performance compared to updating all model parameters, the least correlated ones, a selected subset of correlated models or even updating the models according to their combination weights. We report the comparison with MSDA in Table 11 and Model-Soups in Table 12. We also report the values of the combination weights learned by our method. See Section D of the Appendix for detailed observations.

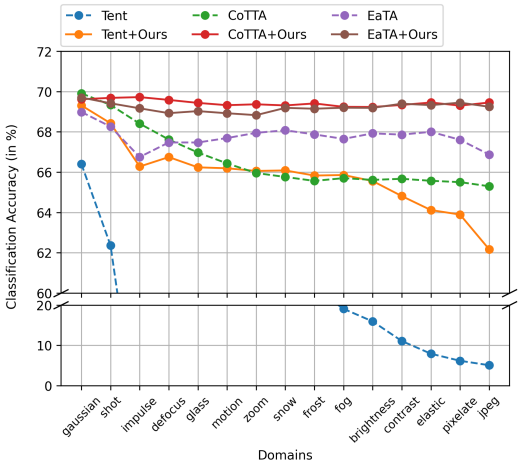

Figure 3: **Comparison with baselines in terms of source knowledge forgetting.** Maintaining the same setting as in Table 2, we demonstrate that by integrating single-source methods with CONTRAST, the source knowledge is better preserved during dynamic adaptation. Unlike all these single-source methods, our algorithm demonstrates virtually no forgetting throughout the entire adaptation process.

5 Conclusions

We propose a novel framework called CONTRAST, that effectively combines multiple source models during test time with small batches of streaming data and without access to the source data. It achieves a test accuracy that is at least as good as the best individual source model. In addition, the design of CONTRAST offers the added benefit of naturally preventing the issue of catastrophic forgetting. To validate the effectiveness of our algorithm, we conduct experiments on a diverse range of benchmark datasets for classification and semantic segmentation tasks. We also demonstrate that CONTRAST can be integrated with a variety of single-source methods. Theoretical analysis of the performance of CONTRAST is also provided.

6 Broader Impact and Limitations

Implementing multiple models for adaptation on dynamic distribution can yield broad impacts. For instance, this approach could find applications in robot navigation, autonomous vehicles or decision making in dynamically evolving scenarios. In all these cases, the algorithm can intelligently select the optimal combination of models during inference, ensuring sustained performance over extended periods. Our method currently assumes that data sampled within a batch comes from the same distribution. In the future, we aim to explore using mixed data samples from different target domains within a batch.

https://doi.org/10.52202/079017-4160

Acknowledgments

The work was partially supported under NSF grant CCF-2008020. Additionally, research was sponsored by the OUSD (R&E)/RT&L and was accomplished under Cooperative Agreement Number W911NF-20-2-0267. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the ARL and OUSD(R&E)/RT&L or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein. This work was also partially supported by the Laboratory Directed Research and Development (LDRD) Program (25-006) of Brookhaven National Laboratory under U.S. Department of Energy Contract No. DE-SC0012704.

References

[1] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR*, 2017.

[2] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *CVPR*, 2018.

[3] 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.

[4] Han-Kai Hsu, Chun-Han Yao, Yi-Hsuan Tsai, Wei-Chih Hung, Hung-Yu Tseng, Maneesh Singh, and Ming-Hsuan Yang. Progressive domain adaptation for object detection. In *WACV*, 2020.

[5] Dripta S Raychaudhuri, Sujoy Paul, Jeroen Vanbaar, and Amit K Roy-Chowdhury. Cross-domain imitation from observations. In *ICML*, 2021.

[6] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *ICML*, 2015.

[7] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. In *JMLR*, 2016.

[8] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *ICML*, 2018.

[9] Luan Tran, Kihyuk Sohn, Xiang Yu, Xiaoming Liu, and Manmohan Chandraker. Gotta adapt 'em all: Joint pixel and feature-level domain adaptation for recognition in the wild. In *CVPR*, 2019.

[10] Shiqi Yang, Yaxing Wang, Joost Van De Weijer, Luis Herranz, and Shangling Jui. Generalized source-free domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8978–8987, 2021.

[11] Jogendra Nath Kundu, Naveen Venkat, R Venkatesh Babu, et al. Universal source-free domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4544–4553, 2020.

[12] Fan Wang, Zhongyi Han, Yongshun Gong, and Yilong Yin. Exploring domain-invariant parameters for source free domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7151–7160, 2022.

[13] Shiqi Yang, Yaxing Wang, Joost Van De Weijer, Luis Herranz, and Shangling Jui. Unsupervised domain adaptation without source data by casting a bait. *arXiv preprint arXiv:2010.12427*, 1 (2):5, 2020.

 https://doi.org/10.52202/079017-4160

[14] Rui Li, Qianfen Jiao, Wenming Cao, Hau-San Wong, and Si Wu. Model adaptation: Unsupervised domain adaptation without source data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9641–9650, 2020.

[15] Ning Ding, Yixing Xu, Yehui Tang, Chao Xu, Yunhe Wang, and Dacheng Tao. Source-free domain adaptation via distribution estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7212–7222, 2022.

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

[17] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *ICML*, 2020.

[18] Sk Miraj Ahmed, Dripta S Raychaudhuri, Sujoy Paul, Samet Oymak, and Amit K Roy-Chowdhury. Unsupervised multi-source domain adaptation without access to source data. In *CVPR*, 2021.

[19] Sk Miraj Ahmed, Suhas Lohit, Kuan-Chuan Peng, Michael J Jones, and Amit K Roy-Chowdhury. Cross-modal knowledge transfer without task-relevant source data. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIV*, pages 111–127. Springer, 2022.

[20] Hao-Wei Yeh, Baoyao Yang, Pong C Yuen, and Tatsuya Harada. Sofa: Source-data-free feature alignment for unsupervised domain adaptation. In *WACV*, 2021.

[21] Vikash Kumar, Rohit Lal, Himanshu Patil, and Anirban Chakraborty. Conmix for source-free single and multi-target domain adaptation. In *WACV*, 2023.

[22] Haifeng Xia, Handong Zhao, and Zhengming Ding. Adaptive adversarial network for source-free domain adaptation. In *ICCV*, 2021.

[23] Sicheng Zhao, Bo Li, Pengfei Xu, and Kurt Keutzer. Multi-source domain adaptation in the deep learning era: A systematic survey. *arXiv preprint arXiv:2002.12169*, 2020.

[24] Jiang Guo, Darsh J Shah, and Regina Barzilay. Multi-source domain adaptation with mixture of experts. *arXiv preprint arXiv:1809.02256*, 2018.

[25] 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/CVF international conference on computer vision*, pages 1406–1415, 2019.

[26] Ruijia Xu, Ziliang Chen, Wangmeng Zuo, Junjie Yan, and Liang Lin. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3964–3973, 2018.

[27] Yitong Li, David E Carlson, et al. Extracting relationships by multi-domain matching. *Advances in Neural Information Processing Systems*, 31, 2018.

[28] Abhimanyu Dubey, Vignesh Ramanathan, Alex Pentland, and Dhruv Mahajan. Adaptive methods for real-world domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14340–14349, 2021.

[29] Zehao Xiao, Xiantong Zhen, Ling Shao, and Cees GM Snoek. Learning to generalize across domains on single test samples. *arXiv preprint arXiv:2202.08045*, 2022.

[30] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pages 23965–23998. PMLR, 2022.

[31] Zuxuan Wu, Xin Wang, Joseph E. Gonzalez, Tom Goldstein, and Larry S. Davis. Ace: Adapting to changing environments for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[32] Mohammad Rostami. Lifelong domain adaptation via consolidated internal distribution. *Advances in neural information processing systems*, 34:11172–11183, 2021.

[33] Theodoros Panagiotakopoulos, Pier Luigi Dovesi, Linus Härenstam-Nielsen, and Matteo Poggi. Online domain adaptation for semantic segmentation in ever-changing conditions. In *European Conference on Computer Vision*, pages 128–146. Springer, 2022.

[34] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016.

[35] M Jehanzeb Mirza, Jakub Micorek, Horst Possegger, and Horst Bischof. The norm must go on: dynamic unsupervised domain adaptation by normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14765–14775, 2022.

[36] Jeya Maria Jose Valanarasu, Pengfei Guo, Vibashan VS, and Vishal M Patel. On-the-fly test-time adaptation for medical image segmentation. *arXiv preprint arXiv:2203.05574*, 2022.

[37] Inkyu Shin, Yi-Hsuan Tsai, Bingbing Zhuang, Samuel Schulter, Buyu Liu, Sparsh Garg, In So Kweon, and Kuk-Jin Yoon. Mm-tta: multi-modal test-time adaptation for 3d semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16928–16937, 2022.

[38] Yuang Liu, Wei Zhang, and Jun Wang. Source-free domain adaptation for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1215–1224, 2021.

[39] Minhao Hu, Tao Song, Yujun Gu, Xiangde Luo, Jieneng Chen, Yinan Chen, Ya Zhang, and Shaoting Zhang. Fully test-time adaptation for image segmentation. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part III 24*, pages 251–260. Springer, 2021.

[40] 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*, pages 7201–7211, 2022.

[41] Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Yaofo Chen, Shijian Zheng, Peilin Zhao, and Mingkui Tan. Efficient test-time model adaptation without forgetting. In *International conference on machine learning*, pages 16888–16905. PMLR, 2022.

[42] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5018–5027, 2017.

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

[44] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[45] Junha Song, Jungsoo Lee, In So Kweon, and Sungha Choi. Ecotta: Memory-efficient continual test-time adaptation via self-distilled regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11920–11929, 2023.

[46] Goirik Chakrabarty, Manogna Sreenivas, and Soma Biswas. Sata: Source anchoring and target alignment network for continual test time adaptation. *arXiv preprint arXiv:2304.10113*, 2023.

[47] Koulik Khamaru and Martin Wainwright. Convergence guarantees for a class of non-convex and non-smooth optimization problems. In *International Conference on Machine Learning*, pages 2601–2610. PMLR, 2018.

 https://doi.org/10.52202/079017-4160

[48] Samet Oymak, Mingchen Li, and Mahdi Soltanolkotabi. Generalization guarantees for neural architecture search with train-validation split. In *International Conference on Machine Learning*, pages 8291–8301. PMLR, 2021.

[49] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[50] Christos Sakaridis, Dengxin Dai, Simon Hecker, and Luc Van Gool. Model adaptation with synthetic and real data for semantic dense foggy scene understanding. In *Proceedings of the european conference on computer vision (ECCV)*, pages 687–704, 2018.

[51] Xiaowei Hu, Chi-Wing Fu, Lei Zhu, and Pheng-Ann Heng. Depth-attentional features for single-image rain removal. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pages 8022–8031, 2019.

[52] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Acdc: The adverse conditions dataset with correspondences for semantic driving scene understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10765–10775, 2021.

[53] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

Appendix Overview:

Contents

A Proof and discussion of Theorems 1 and 2 **16**

B Results on Digits **16**

C Results on CIFAR-10C **17**

D Ablation Study **18**

D.1 Initialization and Learning Rate 18

D.2 Model Update Policy 18

D.2.1 Subset of Model Update 18

D.2.2 Model Update According to Weight 19

D.3 Combination Weight Visualization 20

D.4 Comparison with MSDA 20

D.5 Comparison with Model Soups 20

E Implementation Details **21**

E.1 Stationary Target 21

E.1.1 Digit Classification 21

E.1.2 Object Recognition 21

E.2 Dynamic Target 21

E.2.1 CIFAR-10/100-C 21

F Semantic Segmentation **21**

F.1 Datasets 22

F.2 Experimental setup 22

F.3 Visualization 22

G Additional discussion **23**

H KL divergence between two univariate Gaussians **24**

I Optimal step size in approximate Newton's method **25**

https://doi.org/10.52202/079017-4160

A Proof and discussion of Theorems 1 and 2

Proof of Theroem 1. The optimization (4) has a structure similar to a class of non convex problems as follows:

$$\underset{x \in \chi}{\text{minimize}} \quad g(x) - h(x)) \tag{9}$$

where χ is a closed convex set, $g(x)$ is M_g smooth and $h(x)$ is a continuous convex function. In such cases, the optimization converges as follows [47]:

$$\frac{1}{(k+1)} \sum_{j=0}^{k} \left(\nabla_{\chi} \| f(x^k) \|_2^2 \right) \leq \frac{2(f(x^0) - f^{\star}))}{\alpha(k+1)} \tag{10}$$

where, $f(x) = (g(x) - h(x))$.

In our case $g(x) = c$, where c is a constant (smooth and continous) and $h(x)$ is negative of the Shannon entropy, which is continous and convex. Also, χ is the n-simplex $\aleph$, which is a closed convex set. So, according to the proof derived in [47], we can conclude the bound in Theroem 1. □

Proof of Theorem 2. We adapt the theorem from a corollary (corollary 1) in [48]. In this corollary the following result was derived:

$$\mathcal{L}(\mathrm{f}_{\hat{\alpha}}^{\tau}) \leq \min_{\alpha \in \Delta} (l_{\star}^{\alpha}(\mathcal{D}) + \mathrm{DM}_{\mathcal{D}'}^{\mathcal{D}}(\alpha) + 4\Gamma \mathcal{R}_{n_{\tau}}(\mathcal{F}_{\alpha})) + \sqrt{\tilde{\mathcal{O}}((h_{eff} + t)/n_{\nu})} + \delta$$

Here f^{τ} in the $\mathrm{f}_{\hat{\alpha}}^{\tau}$ is the trained model on the training(τ) distribution $\mathcal{D}'$ and $\hat{\alpha}$ is a hyper-parameter that has been empirically optimized by fine tuning on the validation(ν) distribution $\mathcal{D}$. $\mathcal{L}$ is the expected risk over the distribution $\mathcal{D}$. DM measures the distribution mismatch via difference of sub-optimality gap using the training and validation distribution. $\mathcal{R}_{n_{\tau}}(\mathcal{F}_{\alpha}))$ is the Rademacher complexity of the function class $\mathcal{F}$ with α as the hyper-parameter. The corollary holds for probability of at least $1 - 3e^{-t}$ and h_{eff} is the effective dimension of the hyper-parameter space. Also n_{ν} is the number of samples under the validation. The bound can be first of all easily extended to the source/target scenario instead of train/validation. In our scenario the source models jointly construct the function class $\mathcal{F}_{\alpha}$ where, the hyper-parameter α is the combination weight w. Effective dimension for our case is exactly the number of source model N and instead of t we took τ as the probability variable. For the sake of simplicity we omitted $\delta > 0$ which is a positive constant along with the Rademacher complexity. Also $n_{\nu} = B$ in our setting since we have B number of samples for the target/validation. Now there is a new term in our bound which is φ which was not in the original corollary. This term is used to account for the mismatch between actual and pseudo-labels generated by the source. This is done due to the fact that we do empirical minimization of the entropy of the target pseudo-label since the problem is unsupervised and actual labels are not available. The left side of the inequality is derived using the test/target pseudo-label. Consequently, we can introduce an added distribution mismatch term. This term can be broken down into three components: mismatch from target pseudo to target ground truth (gt), from target gt to source gt, and from source gt to source pseudo label. Of these components, the first two can be readily integrated into the $\Psi(.)$ function , given that it measures the discrepancy between the weighted source and the target. The remaining third component is denoted by the $\varphi(.)$ function. This completes the proof. □

B Results on Digits

We report here the results of digit classification in Table 4. Similar to the experiment on Office-Home dataset, each column of the table represents a target domain dataset. We train four source models on the rest of the digit datasets. For instance, in case of 'MM' column 'MM' is the target domain which is adapted using four source models trained on 'MT', 'UP', 'SV' and 'SY' respectively.

We once again calculate the test error for each incoming test batch and report the results by averaging the errors across all batches. The table demonstrates that CONTRAST achieves a significant reduction

Table 4: **Results on Digits dataset.** We train the source models using four digit datasets to perform inference on the remaining dataset. The column abbreviations correspond to the datasets as follows: 'MM' for MNIST-M, 'MT' for MNIST, 'UP' for USPS, 'SV' for SVHN, and 'SY' for Synthetic Digits.. The table (reporting % error rate(↓)) shows that X+CONTRAST outperforms all of the baselines (X-Best) consistently .

	MM	MT	UP	SV	SY	Avg.
Source Worst	80.5	59.4	50.3	88.5	84.8	72.7
Source Best	47.7	2.2	16.8	18.3	6.7	18.3
Tent Worst	84.2	46.9	41.1	90.1	85.4	69.5
Tent Best	45.2	2.3	16.7	14.4	6.7	17.1
Tent + CONTRAST	**37.5**	**1.9**	**11.2**	**14.2**	**6.7**	**14.3**
EaTA Worst	80.1	48.4	42.6	88.0	83.1	68.4
EaTA Best	47.1	2.7	18.2	18.5	7.2	18.7
EaTA + CONTRAST	**39.5**	**2.0**	**11.5**	**18.0**	**7.0**	**15.6**
CoTTA Worst	80.0	48.3	42.8	87.9	82.9	68.4
CoTTA Best	47.0	2.8	18.6	18.5	7.2	18.8
CoTTA + CONTRAST	**39.6**	**2.0**	**11.7**	**18.1**	**7.1**	**15.7**

in test error compared to the best single source (on average 3% error reduction than the best source). Another baseline exists that simply uses a naive ensemble of the source models, without any weight optimization. In situations where there's a significant performance gap between the best and worst source models adapted using single-source methods, a uniform ensemble of these models produces a predictor that trails considerably behind the best-adapted source, as noted by [18]. Referring to Table 4, when testing on the SVHN dataset, the error disparity between the best and worst adapted source models is approximately 70.7%—a substantial margin. Consequently, using a uniform ensemble in such a scenario results in an error rate of roughly 45.5% (experimentally found, not reported in the table). This is strikingly higher than our method's error rate of around 14.2%. *Given these findings, we deduce that uniform ensembling is not a reliable approach for model fusion. Thus, we exclude it from our experiment section's baseline*.

C Results on CIFAR-10C

Here, we report the results on dynamic target distribution using CIFAR-10C dataset. Note that identical to the experiment on CIFAR100-C in the main paper the results on CIFAR10-C in Table 5 follow the same trend where X+CONTRAST outperforms the X-Best.

Table 5: **Results on CIFAR-10C.** We take four source models trained on *Clear*, *Snow*, *Fog* and *Frost*. We employ these models for adaptation on 15 sequential test domains. This table illustrates that even in the dynamic environment X+CONTRAST performs better than X, which is the direct consequence of better retaining source knowledge. (Results in error rate ↓ (in %))

	GN	SN	IN	DB	GB	MB	ZB	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	JPEG	Mean
Source Worst	84.7	81.1	89.1	42.6	55.6	36.2	32.2	30.6	39.2	28.7	18.5	76.4	26.9	50.0	32.7	48.3
Source Best	72.1	67.8	76.5	22.8	20.4	26.6	18.7	8.1	8.2	6.9	10.6	56.8	18.8	13.9	23.9	30.1
Tent Worst	26.6	22.7	36.1	20.0	34.9	28.8	28.7	32.8	34.4	36.1	30.3	38.2	44.8	41.7	46.8	33.5
Tent Best	19.3	17.6	27.9	14.5	21.1	17.6	13.5	14.3	12.6	14.4	12.4	17.0	19.0	14.3	20.4	17.1
Tent + CONTRAST	**17.2**	**15.6**	**25.7**	**9.1**	**19.1**	**11.7**	**9.0**	**9.9**	**10.1**	**9.7**	**7.7**	**11.7**	**14.5**	**10.3**	**17.4**	**13.2**
EaTA Worst	31.5	30.4	44.8	14.8	33.9	16.1	13.4	20.5	21.6	19.3	11.2	18.9	23.2	19.5	29.6	23.2
EaTA Best	21.9	20.8	33.9	10.5	19.6	14.3	10.6	8.6	9.0	**7.5**	8.5	10.3	16.1	11.4	24.0	15.1
EaTA + CONTRAST	**18.0**	**17.3**	**29.4**	**8.3**	**18.2**	**10.0**	**7.5**	**8.0**	**8.4**	7.9	**6.4**	**9.1**	**13.1**	**10.0**	**18.1**	**12.6**
CoTTA Worst	30.1	26.8	37.8	15.0	28.5	16.6	14.6	19.3	18.6	17.5	12.2	15.9	19.4	15.4	19.3	20.5
CoTTA Best	21.0	18.5	28.0	11.2	**17.3**	13.3	11.1	10.6	10.4	9.5	9.7	11.2	13.1	10.5	15.6	14.1
CoTTA + CONTRAST	**18.4**	**17.0**	**28.0**	**8.4**	17.7	**10.7**	**7.9**	**9.1**	**8.4**	**8.5**	**6.8**	**8.3**	**12.1**	**9.3**	**15.3**	**12.4**

In the single-source scenario, one among the four source models achieves the X-Best (for example CoTTA-Best) accuracy for a specific domain. The determination of which individual model (from the four) will attain the best accuracy for that domain remains uncertain beforehand. Furthermore, the individual source model yielding the X-Best accuracy varies across different domains within CIFAR10-C. However, in our X+CONTRAST approach, the need to deliberate over the selection of one out of the four source models is eliminated. X+CONTRAST reliably outperforms any single source X-model that might achieve the X-Best accuracy.

Individual TTA methods may have distinct advantages. For example, Tent offers several distinct advantages over CoTTA, including its lightweight nature and faster performance. Conversely, CoTTA presents certain benefits over Tent, such as increased resilience against forgetting. Consequently, the

https://doi.org/10.52202/079017-4160

choice between TTA methods is dependent on the user's preferences, aligning with the specific task at hand. In this experiment, we have demonstrated that CONTRAST can be integrated with any TTA method of the user's choosing.

D Ablation Study

D.1 Initialization and Learning Rate

Table 6: **Effect of initialization and step size choice.** Error rate on Office-Home under different choices of initialization and step sizes.

	Step size					
Initialization	$1e-3$	$1e-2$	$1e-1$	$1e0$	$1e1$	Ours
Random	40.7	40.9	40.6	39.6	41.5	39.3
Ours	37.9	37.8	37.5	37.4	39.1	**37.0**

Table 6 presents the error rate results on the Office-Home dataset under the same experimental setting as Table 3 (Appendix) with Tent as the adaptation method, but with different initialization and learning rate choices for solving the optimization in (4). It is evident from the table that our chosen initialization and adaptive learning rate result in the highest accuracy gain.

We additionally show another ablation study in Table 7, where we initialize the combination weights based on the probability of source model predictions. More precisely, we set the initial weights inversely proportional to the entropy of the source model predictions. In simpler terms, a source model with low entropy receives a higher weight, while one with high entropy receives a lower weight.

Table 7: **Initialization based on Entropy.** The table shows the results of entropy based initialization. (Results in error-rate % ↓)

Update Policy	GN	SN	IN	DB	GB	MB	ZB	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	JPEG	Mean
Entropy_init	42.7	41.1	56.9	33.5	46.5	39.4	37.2	41.0	43.2	50.6	46.7	78.6	77.9	79.5	88.7	53.6
Ours	42.2	40.6	55.3	28.6	40.7	31.9	29.6	31.7	32.4	30.9	28.6	41.5	38.5	34.8	49.9	37.1

In the presented table for CIFAR-100C, we note a 16.5% reduction in error resulting from our initialization method. We found that initializing the combination weights using the entropy of the test batch for various sources leads to somewhat uniform initialization. However, when we initialize the combination weights using KL divergence, we achieve a highly effective and peaky prior, favoring the most correlated source model with relatively higher weightage. This clarifies why initializing with entropies fails to converge quickly to the optimum, resulting in significantly poorer outcomes compared to our method.

D.2 Model Update Policy

In Table 8 and 9, we demonstrate that by updating only the model with the highest correlation to the target domain, our method produces the lowest test accuracy. This is in comparison to scenarios where we either update all models or solely the least correlated one. This empirical observation directly supports our theoretical assertion from the theorem: updating the most correlated model is most effective in preventing forgetting, thereby resulting in the smallest test error during gradual adaptation. We also experiment with another model update policy where a subset of model is updated.

D.2.1 Subset of Model Update

In this approach, rather than focusing solely on the most correlated source model, we identify and update a subset of source models that exhibit higher correlation than the rest of the models. Specifically, we select models for updating based on their combination weights, choosing only those whose weights exceed $1/n$, with n representing the total number of models. The intuition behind selecting this threshold $1/n$ for subset selection is grounded in the distance of the combination weight distribution with respect to the uniform distribution. A uniform combination weight implies that all

Table 8: **Choice of model update (CONTRAST+CoTTA).** In our experiments using CoTTA as the model update method on CIFAR100-C, we tested four scenarios: updating all models, updating only the least correlated model, updating subset of model, and updating only the most correlated model. Our results indicate that our model selection approach produces the most favorable outcome. (Results in error rate ↓ (in %))

Update Policy	GN	SN	IN	DB	GB	MB	ZB	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	JPEG	Mean
All Model Update	**44.0**	**42.5**	**54.5**	30.1	38.9	33.4	31.7	32.7	32.1	32.6	30.2	32.8	34.5	32.0	**40.2**	36.2
Least Corr. Update	44.8	44.5	58.9	28.6	38.7	31.0	28.4	29.1	**28.9**	29.5	26.9	30.9	33.8	30.5	44.0	35.2
Subset of Models Update	44.5	43.3	57.1	28.1	**37.5**	30.6	28.4	29.9	29.9	28.8	26.8	30.2	**32.4**	30.2	40.4	34.5
Most Corr. Update	44.6	43.8	57.2	**27.8**	37.6	**30.6**	**28.0**	**29.3**	29.3	**28.2**	**26.6**	**30.0**	32.5	**29.7**	41.4	**34.4**

Table 9: **Choice of model update (CONTRAST+Tent).** In our experiments using Tent as the model update method on CIFAR100-C, we tested four scenarios: updating all models, updating only the least correlated model, updating subset of model, and updating only the most correlated model. Our results indicate that our model selection approach produces the most favorable outcome. (Results in error rate ↓ (in %))

Update Policy	GN	SN	IN	DB	GB	MB	ZB	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	JPEG	Mean
All Model Update	**41.6**	40.9	57.8	47.1	60.2	60.3	62.1	68.6	73.2	80.9	82.1	92.4	91.2	92.5	94.9	69.7
Least Corr. Update	43.8	41.4	56.1	31.2	41.4	34.8	31.4	33.5	33.1	37.5	31.5	41.6	41.5	37.5	53.1	39.3
Subset of Models Update	43.0	41.1	56.4	33.0	47.8	38.7	37.5	41.4	45.3	51.1	46.4	83.6	81.0	60.1	92.4	53.3
Most Corr. Update	42.2	**40.6**	**55.3**	**28.6**	**40.7**	**31.9**	**29.6**	**31.7**	**32.4**	**30.9**	**28.6**	**41.5**	**38.5**	**34.8**	**49.9**	**37.1**

models are equidistant w.r.t the test distribution and should be updated. However, if only one model weight surpasses $1/n$, it signifies that only one model exhibits a high correlation with the overall model.

Results are shown in Table 8 and 9. Several key observations can be extracted from here. Notably, when utilizing the Tent adaptation algorithm, updating a subset of models results in significantly poorer performance compared to updating only the most correlated model. Conversely, with the CoTTA adaptation algorithm, the performance decrement from updating a subset of models is relatively minor compared to updating the most correlated model. This discrepancy can be attributed to the varying degrees of resistance to forgetting exhibited by these adaptation algorithms. Updating multiple models tends to induce forgetting, leading to a decline in overall performance, especially when the adaptation algorithm is not highly resistant to forgetting. Despite the adaptation method's robustness to forgetting, it has been consistently observed that updating the most correlated model not only delivers superior performance but also offers computational advantages over updating a subset of models. This approach simplifies the update process and ensures more efficient use of computational resources.

D.2.2 Model Update According to Weight

Here, we update the model j weighted by w_j. To do so, we need to properly devise an approach that updates models in measures according to their correlation with the test data. Drawing inspiration from recent studies that employ variable learning rates for single-source TTA, we devise a strategy to adjust the learning rate η_j used in updating model j based on their respective combination weights w_j. Specifically, we assigned the highest learning rate $\eta_{max} = 0.001$ (0.001 is the learning rate used for both Tent and CoTTA in our experiments) to the model with the greatest combination weight, while the lowest learning rate $\eta_{min} = 0.0001$, (a tenfold reduction) was allocated to the model with the lowest combination weight. For the remaining models, we interpolated their learning rates proportionally between the highest and lowest rates, based on their respective combination weights following the formula: $\eta_j = [\left(\frac{w_j - w_{min}}{w_{max} - w_{min}}\right) \times (\eta_{\text{max}} - \eta_{\text{min}})] + \eta_{\text{min}}$. In the Table 10, we present the resulting error rates for CIFAR-100C dataset using both Tent and CoTTA.

Table 10: **Model Update according to Weight.** The table shows results of updating model according to their respective weights. (Results in error-rate % ↓)

Update Policy	GN	SN	IN	DB	GB	MB	ZB	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	JPEG	Mean
Tent	41.7	39.7	53.0	33.9	43.9	36.8	34.6	37.8	39.3	41.0	36.8	56.1	49.5	41.4	60.1	43.0
CONTRAST+Tent	42.2	40.6	55.3	28.6	40.7	31.9	29.6	31.7	32.4	30.9	28.6	41.5	38.5	34.8	49.9	37.1
CoTTA	44.5	43.0	56.2	28.1	38.1	30.8	28.6	29.9	29.6	28.7	27.0	29.5	31.8	29.0	38.6	34.2
CONTRAST+CoTTA	44.6	43.8	57.2	27.8	37.6	30.6	28.0	29.3	29.3	28.2	26.6	30.0	32.5	29.7	41.4	34.4

https://doi.org/10.52202/079017-4160

Our investigation reveals that, in scenarios where the update algorithm exhibits limited robustness against forgetting, such as Tent, updating only the model with the highest combination weight proves more advantageous. This is because even marginal updates to uncorrelated models can lead to detrimental forgetting, resulting in poor performance. Conversely, when the update algorithm demonstrates resilience against forgetting (CoTTA), updating the most correlated model impacts performance the most. While updating uncorrelated models does not substantially enhance performance, it significantly increases computational costs. It should also be noted that we have found exactly same finding with our ablation study focused on updating subsets of models. Consequently, we assert that updating the single model with the highest combination weight yields optimal performance across all scenarios.

D.3 Combination Weight Visualization

To provide insight into the combination weight distribution, let's consider an example where the source models are trained on the clean, snow, frost, and fog domains using the training data. We then select one of these domains to collect the average weights over all the test data. When the test data is from the fog domain, the weight distribution appears as follows: [0.05, 0.08, 0.09, 0.78]. On the other hand, when the test domain is frost, we observe the following weight distribution: [0.07, 0.14, 0.69, 0.11]. These results clearly illustrate that the weight distribution accurately reflects the correlation between the source models and target domains.

D.4 Comparison with MSDA

Existing multi-source source-free methods are designed for offline settings where all the target data are available during adaptation. However, in our setting, data is received batch by batch during adaptation. Therefore, theoretically, these methods are expected to perform worse in our setup. Nevertheless, we compared CONTRAST with the seminal paper [18] on source-free multi-source Unsupervised Domain Adaptation (UDA), specifically the DECISION method, to demonstrate its effectiveness in an online adaptation setting. We keep the hyperparameters exactly the same as described in the DECISION and perform adaptation on each incoming batch of test data with the number of epochs specified in DECISION.

Table 11: **Comparison with MSDA.** The table compares the performance of our method with MSDA approach DECISION. (Results in error-rate % ↓)

Update Policy	GN	SN	IN	DB	GB	MB	ZB	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	JPEG	Mean
DECISION	55.0	76.2	90.5	95.2	97.3	97.9	98.2	98.0	98.3	98.4	98.4	98.7	99.0	98.9	98.9	93.3
CONTRAST+Tent	42.2	40.6	55.3	28.6	40.7	31.9	29.6	31.7	32.4	30.9	28.6	41.5	38.5	34.8	49.9	37.1

It is evident from Table 11 that DECISION performs notably poorly in the online setting, with an error rate almost 56% higher than CONTRAST. DECISION utilizes clustering of the entire offline dataset based on the number of classes, a method not feasible to accurately implement in our setting with very small batch sizes. This highlights the necessity of a multi-source method specifically tailored for our setting.

D.5 Comparison with Model Soups

Model Soups [30] is a popular approach for utilizing a set of models by averaging their parameters to create a single model for inference on test data. For completeness, we compare our method against Model Soups.

Table 12: **Comparison with Model Soups.** The table compares the performance our method against model soups. (Results in error-rate % ↓)

Update Policy	GN	SN	IN	DB	GB	MB	ZB	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	JPEG	Mean
Model-Soups	96.82	96.26	97.08	95.17	95.33	95.30	95.22	95.17	95.86	95.28	94.96	97.41	95.04	95.05	95.86	95.72
CONTRAST+Tent	42.2	40.6	55.3	28.6	40.7	31.9	29.6	31.7	32.4	30.9	28.6	41.5	38.5	34.8	49.9	37.1

As shown in Table 12, the performance of Model Soups is significantly worse compared to our method. Model Soups averages the parameters of models fine-tuned on the same data distribution. However, in our setting, we have models trained on different source domains, making the averaging of model parameters suboptimal.

E Implementation Details

In this section, we provide a comprehensive overview of our experimental setup. We conducted two sets of experiments: one on a stationary target distribution, and the other on a dynamic target distribution that changes continuously. The reported results in the main paper are average of three runs with different seeds.

E.1 Stationary Target

E.1.1 Digit Classification

The digit classification task consists of five distinct domains from which we construct five different adaptation scenarios. Each scenario involves four source models, with the remaining domain treated as the target distribution. In total, we construct five adaptation scenarios for our study.

The ResNet-18 architecture was used for all models, with an image size of 64×64 and a batch size of 128 during testing. Mean accuracy over the entire test set is reported in Table 2 of the main paper. For Tent we use a learning rate of 0.01 and for rest of the adaptation method a learning rate of 0.001 is used. We use Adam optimizer for all the adaptation methods. Model parameter update is performed using a single step of gradient descent.

E.1.2 Object Recognition

The object recognition task on the Office-Home dataset comprises of four distinct domains from which we construct four different adaptation scenarios, similar to the digit classification setup. We use the same experimental settings and hyperparameters as the digit classification experiment, with the exception of the image size, which is set to 224×224 in this experiment. The results of this evaluation are reported in Table 3 of the main paper.

E.2 Dynamic Target

E.2.1 CIFAR-10/100-C

In this experiment, we use four ResNet-18 source models trained on different variants of the CIFAR-10/100 dataset: 1) vanilla train set, 2) train set with added fog (severity = 5), 3) train set with added snow (severity = 5), and 4) train set with added frost (severity = 5). To evaluate the models, we use the test set of CIFAR-10/100C (severity = 5) and adapt to each of the domains in a continual manner. The images are resized to 224×224. For all the adaptation methods, a learning rate of 0.001 with Adam optimizer is used.

F Semantic Segmentation

Table 13: **Result on Cityscape to ACDC:** In this experiment, we test our method on the test data from individual weather conditions (static test distribution) of ACDC. The source models are trained on the train set of Cityscape and its noisy variants. Our method clearly outperforms baseline adaptation method. (Results in % mIoU)

Method	Fog	Rain	Snow	Night	Avg.
Tent-Best	25.3	21.0	19.2	12.6	19.5
CONTRAST	**27.7**	**22.8**	**21.1**	**14.0**	**21.4**

Our method is not just limited to image classification tasks and can be easily extended to other tasks like semantic segmentation (sem-seg). We assume access to a set of sem-seg source models $\{\mathrm{f}_S^j\}_{j=1}^N$, where each model classifies every pixel of an input image to some class. Specifically, $\mathrm{f}_S^j : \mathbb{R}^{H \times W} \rightarrow \mathbb{R}^{H \times W \times K}$, where K is the number of classes. In this case, the entropy in Eqn. 3 of the main paper will be modified as follows:

https://doi.org/10.52202/079017-4160

Table 14: **Result on Cityscapes to ACDC for dynamic test distribution:** This table illustrates that over a prolonged cycle of repetitive test distributions, our model can retain performance better than baseline Tent. ((Results in % mIoU))

Time	t												
Round	1				3				5				All
Conditions	Rain	Snow	Fog	Night	Rain	Snow	Fog	Night	Rain	Snow	Fog	Night	Mean
Tent-Best	20.1	21.3	22.3	11.3	18.5	17.2	19.5	8.4	15.8	14.5	17.5	6.8	16.1
CONTRAST	22.1	**21.4**	**24.3**	**13.4**	**21.4**	**18.3**	**23.5**	**11.3**	**18.6**	**15.5**	**21.4**	**10.4**	**18.6**

$$\mathcal{L}_w^{(t)}(\mathbf{w}) = -\mathbb{E}_{\mathcal{D}_T^{(t)}} \sum_{h=1}^{H} \sum_{w=1}^{W} \sum_{c=1}^{K} \hat{y}_{ihwc}^{(t)} \log(\hat{y}_{ihwc}^{(t)}) \tag{11}$$

Where, $\hat{y}_{ihwc}^{(t)}$ is the weighted probability output corresponding to class c for the pixel at location (h, w) at time-stamp t. We modify Eqn. 3 in the main paper, while keeping the rest of the framework the same.

F.1 Datasets

We use the following datasets in our experiments:
• **Cityscapes:** Cityscapes [49] is a large-scale dataset that has dense pixel-level annotations for 30 classes grouped into 8 categories (flat surfaces, humans, vehicles, constructions, objects, nature, sky, and void). There are also fog and rain variants [50, 51] of the Cityscapes dataset, where the clean images of Cityscapes have been simulated to add fog and rainy weather conditions.
• **ACDC:** The Adverse Conditions Dataset [52] has images corresponding to fog, night-time, rain, and snow weather conditions. Also, the corresponding pixel-level annotations are available. The number of classes is the same as the evaluation classes of the Cityscapes dataset.

F.2 Experimental setup

We use Deeplab v3+ [53] with a ResNet-18 encoder as the segmentation model for all the experiments. We resize the input images to a size of 512×512. Following the conventional evaluation protocol [49], we evaluate our model on 19 semantic labels without considering the void label.

We first experiment in a static target distribution setting. Specifically, we train three source models on clean, fog, and rain train splits of Cityscapes. We then evaluate the models on the test set of each of the weather conditions of ACDC dataset using CONTRAST and baseline Tent models. We use a batch size of 16 and report the mean accuracy over all the test batches. Again, we have updated the combination weights of CONTRAST with SGD optimizer using 5 iterations. For updating the source model in CONTRAST that has the most correlation with the incoming test batch, we use the Adam optimizer with a learning rate of 0.001 and updated the batch-norm parameters with one iteration. The baseline Tent models are also updated with the same optimizer and learning rate. The results in Table. 13 clearly demonstrate that CONTRAST outperforms all the baselines on test data from each of the adverse weather conditions.

We also evaluate our method in a dynamic test distribution setting, where we have sequentially incoming test batches from the four weather condition test sets of ACDC dataset. The test sequence includes 5 batches of Rain, followed by 5 batches of Snow, 5 batches of Fog, and finally 5 batches of Night. This sequence is repeated (with the same test images) for a total of 5 rounds. We report the mean accuracy over the 5 batches and include the results for the 1st, 3rd, and 5th rounds in Table 14. We use the same hyperparameters as in the dynamic setting of previous experiments with the exception that the batch-size is 16.

F.3 Visualization

In Fig. 4, we present the input images along with the corresponding predicted masks of the baseline models and CONTRAST from the last round. The figure contains rows of input image samples from

the four different weather conditions of the ACDC dataset, in the order of rain, snow, fog, and night. CONTRAST is compared with baseline adaptation method Tent, and as shown in Fig. 4, it is evident that CONTRAST provides better segmentation results compared to the baselines visually.

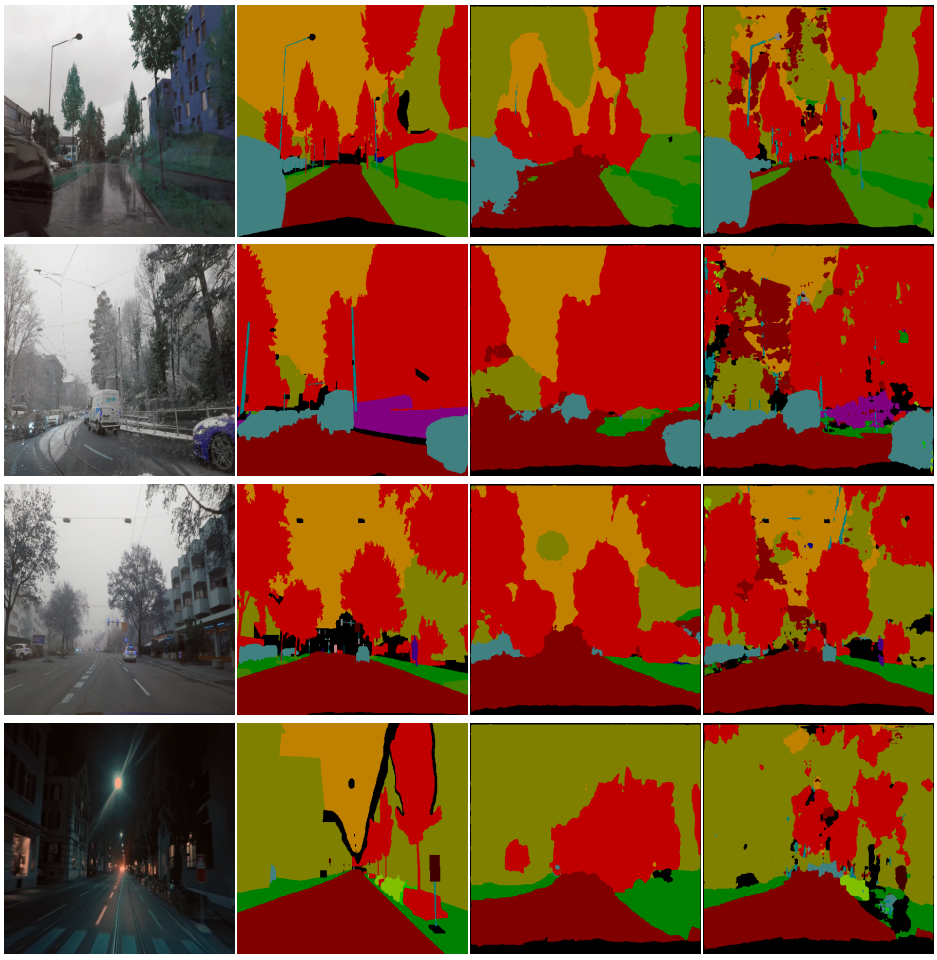

Figure 4: **Visual Comparison of CONTRAST with Baselines for Semantic Segmentation Task.** Each row in the figure corresponds to a different weather condition (rain, snow, fog, and night from top to bottom). It is evident that CONTRAST outperforms the baselines in terms of segmentation results.

G Additional discussion

The $\varphi(.)$ function implies that trained sources should produce high-**quality** pseudo-labels within their own distribution. Essentially, this function evaluates the effectiveness of the model's training. For instance, even if the **shift** between the source and target is minimal, a poorly trained source model might still under-perform on the target. Observe that both the **shift** and the **quality** terms are minimized when we broaden our search space over $\hat{\Delta}$. This allows us to select a model that exhibits the highest correlation with the test domain, thereby providing us with the most strict bound within the discrete simplex.

Examining the issue through the lens of the gradient provides another perspective. By updating the source model that is most correlated with the test data, its gradient will be smaller than those of other models. Over time, this ensures that the model's parameters remain closer to the original source parameters, thereby preventing catastrophic forgetting. let's examine a toy case mathematically of

https://doi.org/10.52202/079017-4160

the most correlated source can give us least gradient.
Let us assume a binary classification task with linear regression where the final activation is sigmoid $\sigma(.)$ function. Now let's take the pseudo-label for a sample x be $\hat{y}$, where $\hat{y} = \sigma(w^\top x)$. Then the entropy h of $\hat{y}$ will be $h = -\hat{y}\log(\hat{y})$. Then we take the derivative of the objective h w.r.t w weight as follows:

$$\begin{aligned} h &= -\hat{y}\log(\hat{y}) \\ \Rightarrow \frac{\partial h}{\partial w} &= (1+\log(\hat{y}))\hat{y}(\hat{y}-1)x \end{aligned}$$

Now we can easily verify that if the source model is closest to the test domain, then the pseudo-label generated by the model has very small entropy which also means $\hat{y}$ is either close to 0 or close to 1. For both of the cases the derivative expression above goes close to zero which validate the claim of having smallest gradient for highest correlated source.

H KL divergence between two univariate Gaussians

During the discussion of initialization of the combination weights in Section 3.5, we come up with θ_j^t which is calculated using the formula for KL divergence between two univariate Gaussians $\mathcal{N}(\mu_1, \sigma_1^2)$ and $\mathcal{N}(\mu_2, \sigma_2^2)$. In this section, we provide the detailed derivation of this below:
From the definition of KL divergence, we know the distance between two distributions p and q is given by,

$$\begin{aligned} \mathcal{D}_{KL}(p,q) &= \int_{-\infty}^{+\infty} p(x)\log\left(\frac{p(x)}{q(x)}\right)\,dx \\ &= \int_{-\infty}^{+\infty} p(x)\log\left(p(x)\right)\,dx - \int_{-\infty}^{+\infty} p(x)\log\left(q(x)\right)\,dx \end{aligned} \tag{12}$$

Here in this problem p and q are univariate Gaussians and can be expressed as follows:

$$p(x) = \frac{1}{(2\pi\sigma_1^2)^{\frac{1}{2}}}\exp\left(-\frac{(x-\mu_1)^2}{2\sigma_1^2}\right), \qquad q(x) = \frac{1}{(2\pi\sigma_2^2)^{\frac{1}{2}}}\exp\left(-\frac{(x-\mu_2)^2}{2\sigma_2^2}\right).$$

Now we compute the second term in Eqn. (12) as follows:

$$\begin{aligned} \int_{-\infty}^{+\infty} p(x)\log\left(q(x)\right)\,dx &= \log\left(\frac{1}{(2\pi\sigma_2^2)^{\frac{1}{2}}}\right) - \int_{-\infty}^{+\infty} p(x)\frac{(x-\mu_2)^2}{2\sigma_2^2}\,dx \\ &= \log\left(\frac{1}{(2\pi\sigma_2^2)^{\frac{1}{2}}}\right) - \frac{\int_{-\infty}^{+\infty} x^2 p(x)\,dx - 2\mu_2\int_{-\infty}^{+\infty} xp(x)\,dx + \mu_2^2}{2\sigma_2^2} \\ &= \log\left(\frac{1}{(2\pi\sigma_2^2)^{\frac{1}{2}}}\right) - \frac{\mathbb{E}\left[X^2\right] - 2\mu_2\mathbb{E}\left[X\right] + \mu_2^2}{2\sigma_2^2} \\ &= \log\left(\frac{1}{(2\pi\sigma_2^2)^{\frac{1}{2}}}\right) - \frac{\text{Var}\left[X\right] + \left(\mathbb{E}\left[X\right]\right)^2 - 2\mu_2\mathbb{E}\left[X\right] + \mu_2^2}{2\sigma_2^2} \\ &= \log\left(\frac{1}{(2\pi\sigma_2^2)^{\frac{1}{2}}}\right) - \frac{\sigma_1^2 + \mu_1^2 - 2\mu_2\mu_1 + \mu_2^2}{2\sigma_2^2} \\ &= \log\left(\frac{1}{(2\pi\sigma_2^2)^{\frac{1}{2}}}\right) - \frac{\sigma_1^2 + (\mu_1-\mu_2)^2}{2\sigma_2^2} \end{aligned} \tag{13}$$

In a similar manner we calculate the first term in Eqn. (12) as follows:

$$
\begin{aligned}
\int_{-\infty}^{+\infty} p(x)\log\left(p(x)\right)\,dx &= \log\left(\frac{1}{(2\pi\sigma_1^2)^{\frac{1}{2}}}\right) - \int_{-\infty}^{+\infty} p(x)\frac{(x-\mu_1)^2}{2\sigma_1^2}\,dx \\
&= \log\left(\frac{1}{(2\pi\sigma_1^2)^{\frac{1}{2}}}\right) - \frac{\int_{-\infty}^{+\infty} x^2 p(x)\,dx - 2\mu_1\int_{-\infty}^{+\infty} xp(x)\,dx + \mu_1^2}{2\sigma_1^2} \\
&= \log\left(\frac{1}{(2\pi\sigma_1^2)^{\frac{1}{2}}}\right) - \frac{\mathbb{E}\left[X^2\right] - 2\mu_1\mathbb{E}\left[X\right] + \mu_1^2}{2\sigma_2^2} \\
&= \log\left(\frac{1}{(2\pi\sigma_2^2)^{\frac{1}{2}}}\right) - \frac{\text{Var}\left[X\right] + \left(\mathbb{E}\left[X\right]\right)^2 - 2\mu_1\mathbb{E}\left[X\right] + \mu_1^2}{2\sigma_1^2} \\
&= \log\left(\frac{1}{(2\pi\sigma_1^2)^{\frac{1}{2}}}\right) - \frac{\sigma_1^2 + \mu_1^2 - 2\mu_1^2 + \mu_1^2}{2\sigma_1^2} \\
&= \log\left(\frac{1}{(2\pi\sigma_1^2)^{\frac{1}{2}}}\right) - \frac{1}{2}
\end{aligned}
\tag{14}
$$

Now combining Eqn. (14) and Eqn. (13), we get the final KL divergence as follows:

$$
\begin{aligned}
\mathcal{D}_{KL}(p,q) &= \log\left(\frac{1}{(2\pi\sigma_1^2)^{\frac{1}{2}}}\right) - \frac{1}{2} - \log\left(\frac{1}{(2\pi\sigma_2^2)^{\frac{1}{2}}}\right) + \frac{\sigma_1^2 + \left(\mu_1-\mu_2\right)^2}{2\sigma_2^2} \\
&= \log\left(\frac{\sigma_2}{\sigma_1}\right) + \frac{\sigma_1^2 + \left(\mu_1-\mu_2\right)^2}{2\sigma_2^2} - \frac{1}{2}
\end{aligned}
\tag{15}
$$

I Optimal step size in approximate Newton's method

In the main paper, we compute the optimal combination weights by solving the optimization below:

$$
\begin{aligned}
\underset{\mathbf{w}}{\text{minimize}} \quad & \mathcal{L}_w^{(t)}(\mathbf{w}) \\
\text{subject to} \quad & \mathrm{w}_j \geq 0, \forall j \in \{1,2,\ldots,N\}, \\
& \sum_{j=1}^{n} \mathrm{w}_j = 1
\end{aligned}
\tag{16}
$$

To solve this problem, we begin by initializing $\mathbf{w}_{init}^{(t)}$ as $\delta(-\theta^t)$. Next, we determine the optimal step size based on the initial combination weights to minimize the loss $\mathcal{L}_w^{(t)}$ as much as possible. Specifically, we use a second-order Taylor expansion to approximate the loss at the updated point after taking a single step with a step size of $\alpha^{(t)}$. Thus, after one step of gradient descent, the updated point becomes:

$$
\mathbf{w}_{init}^{(t)(1)} = \mathbf{w}_{init}^{(t)} - \alpha^{(t)}\left(\nabla_{\mathbf{w}}\mathcal{L}_w^{(t)}\right)\Big|\mathbf{w}^{init} \tag{17}
$$

For notational simplicity let us first denote $\mathbf{w}_{init}^{(t)(1)} = \mathbf{w}^{(1)}$, $\mathbf{w}_{init}^{(t)} = \mathbf{w}^{(0)}$ and $\left(\nabla_{\mathbf{w}}\mathcal{L}_w^{(t)}\right)\Big|\mathbf{w}^{init} = \nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}$. We also denote the hessian of $\mathcal{L}_w^{(t)}$ at $\mathbf{w}^{(0)}$ as $\mathcal{H}_{\mathbf{w}^{(0)}}$. Now, we can write the taylor series

https://doi.org/10.52202/079017-4160

expansion of $\mathcal{L}_w^{(t)}$ at $\mathbf{w}^{(1)}$ as follows:

$$\begin{aligned}
\mathcal{L}_w^{(t)}(\mathbf{w}^{(1)}) &= \mathcal{L}_w^{(t)}(\mathbf{w}^{(0)} - \alpha^{(t)}\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}) \\
&= \mathcal{L}_w^{(t)}(\mathbf{w}^{(0)}) - \alpha^{(t)}\left(\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}\right)^\top\left(\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}\right) + \frac{\left(\alpha^{(t)}\right)^2}{2}\left(\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}\right)^\top\mathcal{H}_{\mathbf{w}^{(0)}}\left(\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}\right) + \mathcal{O}((\alpha^{(t)})^3) \\
&\approx \mathcal{L}_w^{(t)}(\mathbf{w}^{(0)}) - \alpha^{(t)}\left(\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}\right)^\top\left(\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}\right) + \frac{\left(\alpha^{(t)}\right)^2}{2}\left(\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}\right)^\top\mathcal{H}_{\mathbf{w}^{(0)}}\left(\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}\right)
\end{aligned} \tag{18}$$

In order to minimize $\mathcal{L}_w^{(t)}(\mathbf{w}^{(1)})$ we differentiate Eqn. (18) with respect to $\alpha^{(t)}$ and set it zero to get $\alpha_{best}^{(t)}$. Specifically,

$$\begin{aligned}
&\frac{\partial\mathcal{L}_w^{(t)}(\mathbf{w}^{(1)})}{\partial\alpha^{(t)}}\bigg|_{\alpha^{(t)}=\alpha_{best}^{(t)}} = 0 \\
\implies &-\left(\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}\right)^\top\left(\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}\right) + \alpha_{best}^{(t)}\left(\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}\right)^\top\mathcal{H}_{\mathbf{w}^{(0)}}\left(\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}\right) = 0 \\
\implies &\alpha_{best}^{(t)} = \frac{\left(\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}\right)^\top\left(\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}\right)}{\left(\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}\right)^\top\mathcal{H}_{\mathbf{w}^{(0)}}\left(\nabla_{\mathbf{w}^{(0)}}\mathcal{L}_w^{(t)}\right)} = \frac{\left(\nabla_{\mathbf{w}}\mathcal{L}_w^{(t)}\right)^\top\left(\nabla_{\mathbf{w}}\mathcal{L}_w^{(t)}\right)}{\left(\nabla_{\mathbf{w}}\mathcal{L}_w^{(t)}\right)^\top\mathcal{H}_w\left(\nabla_{\mathbf{w}}\mathcal{L}_w^{(t)}\right)}\bigg|_{\mathbf{w}^{init}}
\end{aligned} \tag{19}$$

This is the desired expression of $\alpha_{best}^{(t)}$ in Eqn. 10 in the main paper.

Note that $\mathbf{w}^{(1)}$ does not lie within the simplex. To ensure that the updated $\mathbf{w}$ remains within the simplex, we project it onto the simplex after each gradient step. This can be done by applying the softmax operator ($\delta(.)$ in the main paper), which will ensure that the updated weights are normalized and satisfy the constraints of the simplex. Moreover, in an ideal scenario, one would calculate the optimal step size $\alpha_{best}^{(t)}$ after each gradient step, taking into account the updated point. However, for the purpose of our experiment, we calculate $\alpha_{best}^{(t)}$ only for the first step and use this value as the learning rate for the remaining steps in order to avoid hessian calculation repeatedly. In our experiment, we limit the number of steps to 5 in order to ensure quicker inference. Empirically, we have observed that using the obtained step size as fixed throughout the optimization process works reasonably well.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

1. **Claims**

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

   Answer: [Yes]

   Justification: Main claims made in the abstract and introduction accurately reflect the paper's contributions and scope.

   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: The paper discusses limitations of the work performed.

   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.

https://doi.org/10.52202/079017-4160

- 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: The paper provides full set of assumtions and complete proof.

   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: The paper fully discloses all the information needed to reproduce the main experimental results.

   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: [No]

   Justification: The paper fully discloses all the information needed to reproduce the main experimental results.

   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: Training details is included in the supplementary material.

   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**

https://doi.org/10.52202/079017-4160

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: The paper provides information about the statistical significance of the experiments.

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: The paper provides sufficient information on the computer resources.

   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: Yes, we confirm.

   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: [Yes]

    Justification: Included in the paper.

    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: The paper does not pose such risks.

    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 data and models used are properly cited.

    Guidelines:

https://doi.org/10.52202/079017-4160

- 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: The paper 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: The paper 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: The paper 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.

https://doi.org/10.52202/079017-4160