Turn Down the Noise: Leveraging Diffusion Models for Test-time Adaptation via Pseudo-label Ensembling

Mrigank Raman
Machine Learning Department
Carnegie Mellon University
mrigankr@cmu.edu

Rohan Shah
Machine Learning Department
Carnegie Mellon University
rohans@cmu.edu

Akash Kannan
Machine Learning Department
Carnegie Mellon University
akashkan@cmu.edu

Pranit Chawla
Machine Learning Department
Carnegie Mellon University
pranitc@cmu.edu

Abstract

The goal of test-time adaptation is to adapt a source-pretrained model to a continuously changing target domain without relying on any source data. Typically, this is either done by updating the parameters of the model (model adaptation) using inputs from the target domain or by modifying the inputs themselves (input adaptation). However, methods that modify the model suffer from the issue of compounding noisy updates whereas methods that modify the input need to adapt to every new data point from scratch while also struggling with certain domain shifts. We introduce an approach that leverages a pre-trained diffusion model to project the target domain images closer to the source domain and iteratively updates the model via pseudo-label ensembling. Our method combines the advantages of model and input adaptations while mitigating their shortcomings. Our experiments on CIFAR-10C demonstrate the superiority of our approach, outperforming the strongest baseline by an average of 1.7% across 15 diverse corruptions and surpassing the strongest input adaptation baseline by an average of 18%.

1 Introduction

Supervised deep learning algorithms often assume that the data used for training (source) and testing (target) follow the same ideal pattern, where they are independent and identically distributed (IID). However, in real-world situations, this ideal rarely holds true. Even small differences or changes in data distribution can seriously affect the performance of these models [Quionero-Candela et al., 2009]. This becomes a major challenge when using supervised learning for tasks that involve data with different distributions. To deal with shifts in data distribution, adaptation methods typically use the target data to continuously improve and update their predictions.

Typically, adaptation techniques combat the problem of distribution shift by jointly optimizing on the source and the target distributions during training [Ganin et al., 2016, Hoffman et al., 2018, Saenko et al., 2010]. These methods exhibit strong performance when the shifts in the target domain are well-understood and predefined. However, they tend to falter when faced with the arrival of an entirely new and previously unseen target domain. This underscores the imperative need for development of test-time adaptation, a technique that adapts the model at inference time without relying on the source data and without interrupting the inference process. Broadly speaking, there are two orthogonal ways of performing test-time adaptation. First, model adaptation [Wang et al., 2022, 2020, Chen et al., 2022, Liang et al., 2020] wherein the model is updated iteratively during inference without using test

Neural Information Processing Systems (NeurIPS) 2023 Workshop on Diffusion Models.

labels. Second, Input adaptation [Gao et al., 2022] wherein the input is modified to match the source distribution. Model adaptation helps to continually adapt to changing distributions but may suffer from noisy updates depending on the amount of distribution shift. Moreover, input adaptation must adapt to each example from scratch and sometimes struggle with certain distribution shifts.

Our proposed method, D-TAPE (Diffusion-infused Test-time Adaptation via Pseudo-label Ensembling), seamlessly integrates the strengths of model and input adaptation while minimizing their weaknesses. Utilizing a source domain-trained diffusion model, we align target images more closely with the source domain. However, a direct application of a pre-trained diffusion model risks losing class information. To rectify this, we implement a low-pass filtering technique, akin to [Choi et al., 2021], enabling class-preserving projections. Our model then iteratively updates by ensembling pseudo-labels from both the projected and original test images, offering dynamic adaptability to sudden domain shifts. D-TAPE outperforms existing state-of-the-art approaches like CoTTA [Wang et al., 2022], TENT [Wang et al., 2020], and DDA [Gao et al., 2022]. We also illustrate how D-TAPE effectively brings the input data closer to the source domain compared to the corrupted images, as evidenced by a reduction in the $\mathcal{A}$-distance [Ben-David et al., 2006]. We anticipate that this study will serve as a catalyst for the broader adoption and exploration of diffusion models in future test-time adaptation research.

2 Prior Work

Test Time Adaptation: The goal of test-time adaptation is to adapt a source-pretrained model to a continuously changing target domain without relying on any source data. To that end, the landscape of Test Time Adaptation can broadly be segmented into two paradigms namely Model Adaptation and Input Adaptation. In the realm of Model Adaptation, particularly in source-free and test-time settings, various methodologies have emerged. For instance, models like TENT [Wang et al., 2020], SHOT [Liang et al., 2020], and MEMO [Zhang et al., 2021] employ unsupervised loss minimization techniques specifically tailored to target data distributions. Alternatively, some strategies resort to the generation of pseudo labels via a range of innovative techniques such as a conjugate pseudo-label function [Goyal et al., 2022], weight-averaged teacher models [Wang et al., 2022], and nearest neighbor soft-voting mechanisms [Chen et al., 2022]. These pseudo-labels serve as scaffolding for self-training procedures, effectively amplifying the model's generalization capabilities. In contrast, Input Adaptation strategies predominantly leave the model untouched and only update the input examples from the target domain. Among these, Diffusion Driven Adaptation [Gao et al., 2022] stands out for its application of a diffusion model trained on source data, thereby enabling more effective test-time adaptation.

Diffusion Modelling: Diffusion modeling serves as a pivotal approach in generative techniques, relying on a two-step process that initially introduces noise into data, followed by a denoising phase. Seminal works such as DDPM [Ho et al., 2020] and DDIM [Song et al., 2020] have been instrumental in establishing the framework of this field. An advanced extension is Guided Diffusion [Dhariwal and Nichol, 2021], which leverages class labels to enhance the quality of generated images. Since we do not have access to test labels, we employ DDPM models that have been specifically trained on the CIFAR10 dataset for projecting target images closer to the source domain.

3 Methodology

Diffusion with Interpolated Latent Refinement: Initially, a diffusion model is pre-trained to generate images from the source distribution. Then, at test-time, we run the forward process on the domain-shifted input image by iteratively adding Gaussian noise N times, resulting in a sequence $x_0, \ldots, x_N$. We subsequently apply a modified reverse process to project the image closer to the source domain, using the diffusion model trained to maximize the likelihood of the source data. Initially, we sample $\hat{x}^g_{N-1} \sim p_\theta(x^g_{N-1}|x^g_N = x_N)$ from the reverse denoising process of the diffusion model. Since the diffusion model is not explicitly conditioned to preserve class information, we apply the low-pass filter $h()$ from ILVR [Choi et al., 2021] to the sampled $\hat{x}^g_{N-1}$ to constrain the low-frequency features of $\hat{x}^g_{N-1}$ to be equal to that of x_{N-1} in the same manner as Gao et al. [2022]. This is a sequence of downsampling and upsampling operations that helps preserve the high-level image structure and hence the class information. However, downsampling results in blurring for

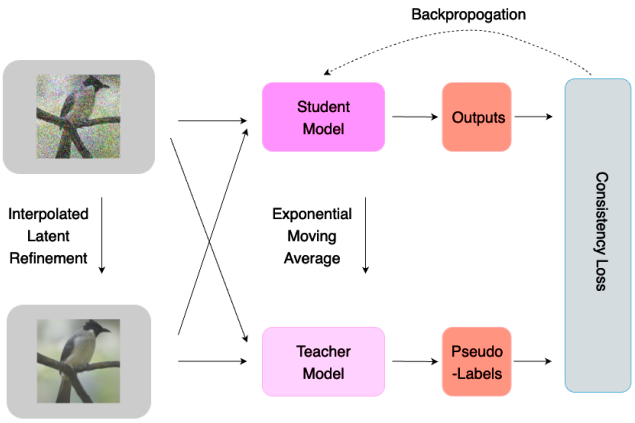

Figure 1: Illustration of our complete adaptation mechanism. Given an image from the target distribution, we first apply our interpolated latent refinement procedure to project the image closer to the source distribution. Then we use a student-teacher framework and update the models by using an ensemble of pseudo-labels generated by the teacher using both the original and projected test image.

low-resolution images. To mitigate this effect, we interpolate the filtered image $h(\hat{x}^g_{N-1})$ with the original sample $\hat{x}^g_{N-1}$ to generate x^g_{N-1}. This is fed back into the diffusion model and the process is repeated to generate x^g_{N-2} and so on. We provide the exact algorithm below.

Model Adaptation via Pseudo-label ensembling: Under the self-training framework, the model weights are adapted using some form of the model predictions themselves. Given a pre-trained model, $f_\theta()$ we follow a strategy similar to that used in CoTTA [Wang et al., 2022] to adapt the model weights. We store two copies of the original model - a student $f_{\theta_s}()$ and a teacher $f_{\theta_t}()$. The teacher generates pseudo-labels that provide supervision to update the student model. The weights of the teacher themselves are adapted by exponential moving average using the student weights. We incorporate the diffusion model output x^g_0 in both student predictions $\hat{y}_S = g(x_0, x^g_0, \theta_S)$ and pseudo-labels $\hat{y}_T = g(x_0, x^g_0, \theta_T)$ where the function g is a combination of a conditional ensembling scheme and logit averaging. We provide ablations for these choices in Table 3. Mathematically,

$$g(x_0, x^g_0, \theta) = \begin{cases} f_\theta(x_0) & \text{if } \max_c f_{\theta_T}(x_0) > \max_c f_{\theta_T}(x^g_0) \\ \frac{f_\theta(x_0) + f_{\theta_T}(x^g_0)}{2} & \text{otherwise} \end{cases}$$

We additionally generate student predictions on augmented versions of the inputs x'_0 and x'^g_0, and denote $\hat{y}'_S = g(x'_0, x'^g_0, \theta_S)$. The loss used to update the student is $\mathcal{L} = -\sum_c 0.5 y^c_T (\log \hat{y}^c_S + \log \hat{y}'^c_S)$.

4 Experimental Setup and Results

Datasets and Evaluation Methodology: We train our models on the CIFAR10 [Krizhevsky et al., 2009] train set and evaluated on CIFAR10-C [Hendrycks and Dietterich, 2019]. CIFAR10-C comprises of 15 different corruptions at 5 different severity levels. We evaluate the continuous adaptability of the models using the approach used in [Wang et al., 2022, Gan et al., 2022] which can briefly described as follows: We arrange the corruption types in the same order as demonstrated in Table 1 and perform inference on the model using batches of test images with a severity level of 5 for each corruption type. This process results in inference on 15 times the size of the clean test data. We exclusively use images with a severity level of 5 to introduce sudden distribution shifts in the batches. We report error rates for each corruption type and calculate the mean error rate across all 15 corruptions. We also provide details and results on a different setting namely gradual TTA in 2

Algorithm 1 Generation using Diffusion

Input: Reference image x_0
Output: Generated image x_0^g
N: diffusion range, $\phi_D(\cdot)$: low-pass filter of scale D
Sample $x_N \sim q(x_N|x_0)$ ▷ perturb input
$x_N^g \leftarrow x_N$
for $t \leftarrow N$ **to** 1 **do**
    $\hat{x}_{t-1}^g \sim p_\theta(x_{t-1}^g|x_t^g)$ ▷ unconditional proposal
    $\hat{x}_0^g \leftarrow \frac{1}{\bar{\alpha}_t} x_t^g - \frac{1}{\bar{\alpha}_{t-1}} \epsilon_\theta(x_t^g, t)$
    $x_{t-1}^g \leftarrow \hat{x}_{t-1}^g - w \nabla_{xt} \|\phi_D(x_0) - \phi_D(\hat{x}_0^g)\|^2$
    $x_{t-1}^g \leftarrow \alpha x_{t-1}^g + (1-\alpha)\hat{x}_{t-1}^g$
end for
return x_0^g

Baselines: We compare D-TAPE to the following test-time adaptation methods.
TENT [Wang et al., 2020]: Minimizes model prediction entropy on test data using the model's own predictions as pseudo labels, updating only normalization parameters.
CoTTA [Wang et al., 2022]: Utilizes a weight-averaged teacher model for pseudo-label generation and updates a student network. It also incorporates augmentation-averaged pseudo labels for low-confidence test inputs and restores a fraction of model weights to their original values.
AdaContrast [Chen et al., 2022]: Maintains a memory queue of target features and predictions with a momentum model. Generates pseudo-labels for using a combination of cross-entropy loss and contrastive loss to update the model.
DDA [Gao et al., 2022]: Employs a diffusion process to transform test images back to the source domain, using the output as predictions. No parameter updates occur during testing, and a low-frequency filter retains class information during diffusion. Following the approach of Wang et al. [2022], we employ a pre-trained WideResNet-28-10 [Zagoruyko and Komodakis, 2016] from RobustBench [Croce et al., 2020] as the backbone for all baselines on CIFAR10-C.

Results Discussion: Our method outperforms the strongest baseline which is CoTTA on 14 out of the 15 corruptions while beating it by an average of 1.7% across all 15 corruptions (Table 1). Additionally, our method also beats DDA which only modifies the input by an average of 18% across all corruptions. This underscores the effectiveness of our method.

Method	gauss	shot	impulse	defocus	glass	motion	zoom	snow	frost	fog	brightness	contrast	elastic	pixelate	jpeg	mean
Source	72.33	65.71	72.92	46.95	54.33	34.76	42.01	25.09	41.30	26.01	9.31	46.71	26.59	58.45	30.30	43.52
DDA	33.09	30.38	38.18	49.51	33.61	37.94	41.28	19.70	24.18	39.10	11.22	61.24	25.38	30.95	19.47	33.02
TENT	24.79	20.48	28.54	14.54	32.27	16.05	13.88	20.21	20.45	17.77	11.17	15.20	24.06	19.96	25.38	20.32
AdaContrast	29.19	22.50	30.09	13.85	32.82	14.00	12.10	16.47	14.73	14.31	8.04	**9.83**	22.11	17.76	19.95	18.52
COTTA	24.30	21.43	25.91	11.72	27.90	12.33	10.66	14.84	13.82	12.36	**7.54**	10.76	18.23	13.61	17.68	16.21
D-TAPE (Ours)	**19.29**	**17.19**	**22.58**	**11.54**	**22.73**	**11.94**	**10.05**	**14.13**	**12.98**	**12.15**	8.17	10.58	**16.87**	**12.70**	**14.54**	**14.50**

Table 1: Classification error rate (%) for the standard CIFAR10-to-CIFAR10-C time adaptation task in the sudden setting. Evaluated on WideResNet-28 with the severity level 5.

Gradual TTA: Unlike the sudden setting, here we make use of all the severity levels. Like the sudden setting, we generate a permutation of all the different types of corruptions and for each type we sample batches with severity in the following order $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 1$ on which we do inference, update weights and then go to the next type of corruption in the same ordering of severity. Note that we do not reset the weights to the pre-trained values anywhere in between. For each type of corruption, the average error on its images is the indicator of performance. D-TAPE outperforms the strongest baseline by $\sim 1\%$ on average across 15 corruptions and improves massively over the best input adaptation technique which is DDA (Table 2).

Model	Gradual CIFAR-10C
Source	43.52
DDA	33.02
TENT	24.56
AdaContrast	15.78
CoTTA	13.99
D-TAPE	**13.09**

Table 2: Mean results over different types of corruption over the gradual CIFAR10-C at level 5 corruption, we report error percentage for each method.

Ablations We perform ablations to demonstrate the importance of using conditional ensembling and logit averaging in the pseudo-label ensembling phase of D-TAPE and report the results in Table 3. We notice that without conditional ensembling, our performance on fog and contrast drastically decreases. This is expected since diffusion fails to project the images closer to the source for these corruptions as seen in Figure 2. We also observe that without logit averaging our performance drops slightly in the case of noises as well where we expect the best performance. Logit averaging helps us to defend against the rare cases where diffusion fails to retain the same object in the image.

Conditional Ensembling	Logit Averaging	gauss	shot	impulse	fog	contrast	mean
✗	✗	26.19	24.9	33.76	33.69	41.19	23.68
✗	✓	19.70	18.10	23.79	16.79	17.37	16.02
✓	✗	20.21	18.57	24.26	14.73	14.14	16.03
✓	✓	**19.29**	**17.19**	**22.58**	**12.15**	**10.58**	**14.50**

Table 3: Ablation on two components of our model, conditional ensembling and logit averaging. Mean is over 15 noises

Does Diffusion help? To answer the question of whether using diffusion helps or not, we use $\mathcal{A}$-distance [Ben-David et al., 2006] to quantify shifts in distribution. We follow Ben-David et al. [2006] and train a linear classifier to separate the two domains for computing the $\mathcal{A}$-distance. We observe that, on average, the generated images closely match the clean image distribution, with particularly close alignment in the case of gaussian, shot, and impulse noises. However, for fog and contrast, the $\mathcal{A}$-distance slightly increases for generated images.(Figure 2). We believe this happens because our interpolated latent refinement scheme retains low frequency noises. We report all the $\mathcal{A}$-distance values in Table 4 which indicate that diffusion is beneficial for almost all corruptions.

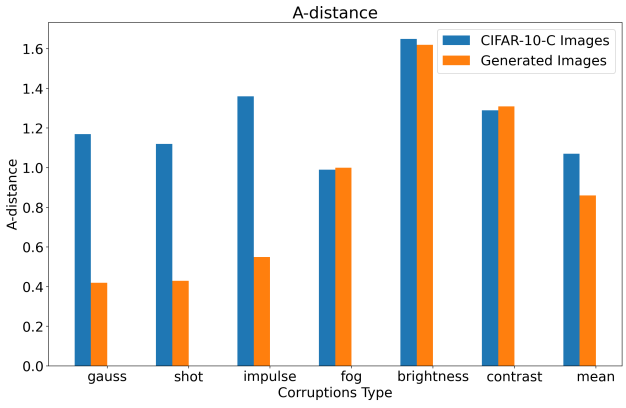

Figure 2: $\mathcal{A}$-distance between the CIFAR-10C dataset and the original CIFAR-10 dataset, and $\mathcal{A}$-distance between generated images and original CIFAR-10 dataset. Mean is over 15 noises

References

Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence. Dataset shift in machine learning. 2009. URL https://api.semanticscholar.org/CorpusID:61294087.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016.

Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pages 1989–1998. Pmlr, 2018.

Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, pages 213–226, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-15561-1.

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.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization, 2020. URL https://arxiv.org/abs/2006.10726.

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.

Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation, 2020. URL https://arxiv.org/abs/2002.08546.

Jin Gao, Jialing Zhang, Xihui Liu, Trevor Darrell, Evan Shelhamer, and Dequan Wang. Back to the source: Diffusion-driven test-time adaptation. *arXiv preprint arXiv:2207.03442*, 2022.

Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. Ilvr: Conditioning method for denoising diffusion probabilistic models, 2021.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006. URL https://proceedings.neurips.cc/paper_files/paper/2006/file/b1b0432ceafb0ce714426e9114852ac7-Paper.pdf.

Marvin Zhang, Sergey Levine, and Chelsea Finn. Memo: Test time robustness via adaptation and augmentation, 2021. URL https://arxiv.org/abs/2110.09506.

Sachin Goyal, Mingjie Sun, Aditi Raghunathan, and J Zico Kolter. Test time adaptation via conjugate pseudo-labels. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=2yvUYc-YNUH.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

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

Yulu Gan, Xianzheng Ma, Yihang Lou, Yan Bai, Renrui Zhang, Nian Shi, and Lin Luo. Decorate the newcomers: Visual domain prompt for continual test time adaptation. *arXiv preprint arXiv:2212.04145*, 2022.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. *arXiv preprint arXiv:2010.09670*, 2020.

.1 $\mathcal{A}$-distance

We report the $\mathcal{A}$-distance values for all the 15 corruptions in Table 4

	CIFAR10-C	Generated
gauss	1.17	0.42
shot	1.12	0.43
impulse	1.36	0.55
defocus	0.95	0.92
glass	0.60	0.50
motion	0.75	0.65
zoom	0.79	0.73
snow	1.51	1.48
frost	1.39	1.36
fog	0.99	1.00
brightness	1.65	1.62
contrast	1.29	1.31
elastic	0.70	0.60
pixelate	1.21	0.85
jpeg	0.58	0.50
mean	1.07	0.86

Table 4: We report the $\mathcal{A}$-distance between CIFAR10-C and CIFAR10 images and also between the images generated using our interpolated latent refinement scheme and CIFAR10 images

.2 Hyperparameter Details

We perform $N = 1000$ steps of the diffusion process, use $\alpha = 0.9$ which is decided using the FID score of the generated images. During the model adaptation stage, we use a learning rate of $1e-3$, batch size of 200, Adam optimizer with a $\beta = 0.9$ and we use 1 optimizer step. We use a momentum of 0.999 to update the teacher via exponential averaging.