Domain-agnostic Test-time Adaptation by Prototypical Training with Auxiliary Data

Qilong Wu[*]
Kuang Yaming Honors School
Nanjing University
Nanjing, Jiangsu, China, 210023
qlw@smail.nju.edu.cn

Xiangyu Yue[*]
Electrical Engineering and Computer Science
UC Berkeley
Berkeley, CA, US, 94720
xyyue@berkeley.edu

Alberto Sangiovanni-Vincentelli
Electrical Engineering and Computer Science
UC Berkeley
Berkeley, CA, US, 94720
alberto@berkeley.edu

Abstract

Test-time adaptation (TTA) aims to achieve high accuracy on out-of-distribution (OOD) target data with only model parameters trained on the source domain and the target data. Standard TTA assumes that the test data is under a single distribution, or the distribution gradually changes with test data streaming in. However, in many scenarios, this assumption does not always hold. For instance, when inference is performed on the cloud, the test data can come from totally different users. In this paper, we try to tackle Domain-agnostic Test-time Adaptation (DaTTA), a new problem setting where the test data distribution is unknown and varies abruptly. To address DaTTA, we propose a framework to perform prototypical training with auxiliary data (PAD). Specifically, we fine-tune the model with augmented test images by consistency loss and further regularize the training process by auxiliary data. We curate a dataset for DaTTA, and the proposed PAD outperforms previous best methods by large margins on both DaTTA and standard TTA.

1 Introduction

Deep networks can achieve high visual recognition accuracy when training and testing data are from the same distribution [1, 2, 3]. However, deep neural nets trained on a specific dataset fail to generalize to new data distributions due to the presence of *domain shift* [4, 5, 6]. Unsupervised Domain Adaptation (UDA) [6, 7, 8, 9] aims to tackle domain shift by training the model together with labeled source $\{(\mathbf{x}_s, y_s)\}$ and unlabeled target $\{\mathbf{x}_t\}$. However, in many applications, source data may not be available during the testing phase. For instance, a hospital could distribute a model trained on the medical data of its patients, but it cannot also send the training data due to privacy concerns.

Test-time Adaptation (TTA) [10, 11] tries to target a more practical problem setting, where during test time, only the unlabeled target data and model trained on labeled source data are available. Test-time Training (TTT) [10] performs test-time training relying on a suitable proxy task, *e.g.*, rotation prediction [12], patch location prediction [13], *etc*. Authors of [10] caution that the proxy must be "both well-defined and non-trivial in the new domain." For example, their method fails on the

[*]Equal contribution

images with black margins as such margins hint at rotation prediction and make it trivial. Moreover, TTT only utilizes the proxy task during adaptation, leaving the original classifier unchanged. Test ENTropy minimization (TENT) [11] performs TTA by minimizing entropy over specific layers without modifying the training process. Large batch size is needed in [11] in order to get stable data statistics as target domain information.

All the previous works have an implicit assumption that the distribution of test data is stable or gradually changes as the data streams in. However, for many applications that perform inference in a centralized manner, the test data distribution is unknown and can change abruptly. For example, for a model running on a server, the test data it needs to process comes from different users, and thus the test data distribution is fairly unpredictable.

In this paper, we propose a new problem setting, termed Domain-agnostic Test-time Adaptation (DaTTA), in which the source data is unavailable during test time, and the test data distribution is unknown and varies abruptly. We show in the experiment section that previous methods suffer in this new practical setting. In order to tackle DaTTA, we propose a novel framework to perform Prototypical training with Auxiliary Data (PAD). In PAD, the model is fine-tuned with augmented test images by consistency loss and in order for the model to not drift away, we regularize the training by auxiliary data. To summarize, our contributions are three-fold:

- We study Domain-agnostic Test-time Adaptation, a new problem setting where only a model trained on source data is available during testing and test data have an unpredictable distribution and can change abruptly.
- We propose a novel framework, termed PAD, to perform prototypical training while using auxiliary data to regularize the training process.
- Our method shows consistently superior performance over previous methods on two datasets Mixed CIFAR-10-C (curated from CIFAR-10-C) and CIFAR-10.1. More surprisingly, PAD achieves better test accuracy on OOD distribution (CIFAR-10.1) than the in-domain distribution (CIFAR-10).

2 Related Work

We relate our method to existing test-time domain adaptation methods. Our method accomplishes test-time adaptation by test-time learning, making it flexible, and only needs to change the last layer to the cosine classifier during training. Test-Time Training (TTT) [10] also learns during testing but differs in its loss and relies on a suitable proxy task, such as recognizing rotations of an image, and so its loss depends on the choice of the proxy, which must be "both well-defined and non-trivial in the new domain."Consequently, their method fails on the image with black margins, which makes predicting rotation trivial. Moreover, TTT leaves the original classification loss unused. Intuitively, TTT only adapts the model to make the test sample not surprising, but our method allows the model to learn from the original task directly.

Test ENTropy minimization (TENT) [11] also focuses on test-time adaptation and does not need to alter training and only needs to minimize entropy over specific layers. However, TENT cannot be used in our setting directly because it utilizes the batch information during testing, which is not accessible in our setting. As the test data come in a stream in our setting, batch information cannot be used for different test samples unless the past samples can be stored. Even if past samples are stored, we must also note that the current sample's prediction cannot depend on the samples after it, and sometimes we cannot do this because of privacy concerns.

3 Method

In domain-agnostic test-time adaptation, target dataset $\mathcal{D}_t = \{(\mathbf{x}_i^t)\}_{i=1}^{N_t}$ is not available during training; and source dataset $\mathcal{D}_s = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^{N_s}$ is not available during testing. Moreover, $\mathcal{D}_t$ streams in during test time, with the distribution changing in an unpredictable manner.

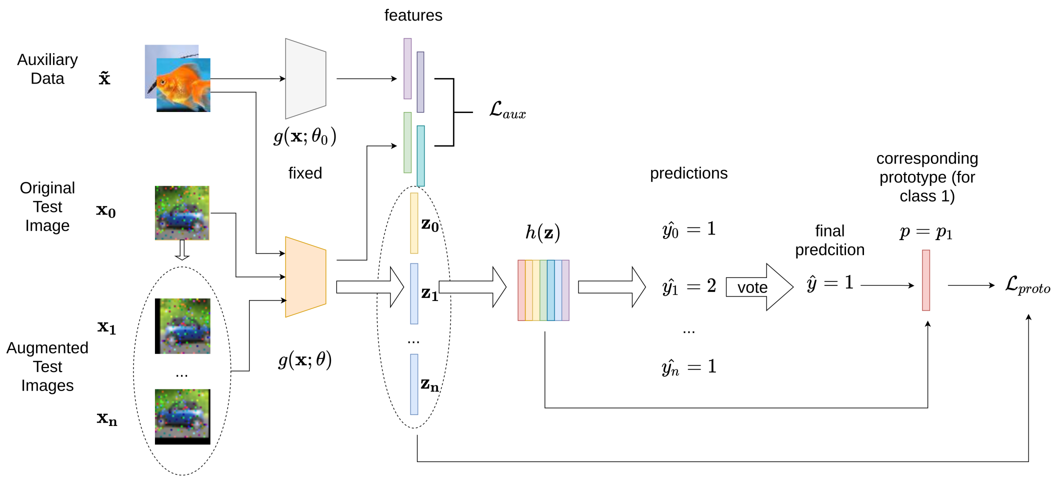

Figure 1: Overview. A single test sample is augmented to a batch of samples (including itself), and then we predict by voting on this batch. After prediction, the model is updated to align features to the corresponding prototype while keeping the relative similarity difference minimal. The test image displayed is from CIFAR-10-C for the class automobile with impulse noise, and the auxiliary data is from tiny-imagenet-200 for class goldfish, Carassius auratus and albatross, mollymawk in this figure.

3.1 Prototypical Learning with Augmented Samples

The base model consists of a feature encoder g, which outputs a normalized feature vector $\mathbf{z} \in \mathbb{R}^d$ and a cosine classifier h, whose weights can be regarded as the prototypes for each category.

During training on the source, we learn prototypes for each category, and they serve as representatives for the feature of each category. Precisely, the model composes of a feature extractor g and a classifier h. h classifies the features into different categories by the cosine similarity between the extracted feature and the prototypes and classifies the feature into the category with the most similarity. Note that unlike [14] which manually updates the prototypes with the average feature for each category, we update the prototypes automatically with backpropagation and normalization.

Since the test data distribution does not evolve in a continuous manner, for a test sample $\mathbf{x}_0$, there is no reliable prior domain information from previous test samples. In order to get more stable prediction, we augment $\mathbf{x}_0$ into $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$, and the final prediction $\hat{y}$ is a majority voting over the predictions $\hat{y}_i$ for each $\mathbf{x}_i, i \in [0, n]$. During testing, we minimize the L1 loss between each feature $\mathbf{z}_i = g(\mathbf{x}_i;\theta)$ and the prototype p corresponding to $\hat{y}$. $\hat{y}$ can be viewed as a pseudo label, and the voting mechanism makes $\hat{y}$ more reliable. Note we only compute the loss on $\mathbf{x}_0$ if the prediction $\hat{y}_0$ on the original image is consistent with $\hat{y}$ and the number of supporting votes is more than half. The loss for test sample $\mathbf{x}_0$ can be written as $\mathcal{L}(\mathbf{x}_0) = \frac{1}{n+1}\sum_{i=0}^{n} l_1(\mathbf{z}_i, p)$, and the loss over $S_c = \{\mathbf{x}_0 | \hat{y}_0 = \hat{y},\ |\{i|\hat{y}_i = \hat{y}, i \in [0,n]\}| > \frac{n}{2},\ \mathbf{x}_0 \in \mathcal{D}_t\}$ is ($|S|$ denotes the number of elements in S):

$$\mathcal{L}_{\text{proto}} = \mathbb{E}_{\mathbf{x}^t \sim S_c}\mathcal{L}(\mathbf{x}^t). \tag{1}$$

3.2 Auxiliary Data Regularization

Model forgetting is a problem in test-time adaptation as source dataset $\mathcal{D}_s$ is not available during adaptation. Excess training on the target without any task supervision can result in severe performance degradation. TENT[11] mitigates this problem by only fine-tuning the batch-norm parameters with all other model parameters fixed. This method works well under the assumption that the test data distribution changes smoothly. In DaTTA, however, the test data distribution could change abruptly, and the accumulated BN parameters cannot represent the upcoming test samples well. As we show in the experiment section, the performance of TENT degrades significantly under DaTTA.

In order to mitigate the above problem, we introduce a novel regularization mechanism that does not require $\mathcal{D}_s$. The motivation is simple: a model that does not suffer from forgetting should keep the relative relationship similar for the same data, even if such data is irrelevant for the desired task. The data is termed "auxiliary data" and is randomly sampled from some other datasets.

We pick auxiliary data different from data used for the base training stage as this can provide more information while not violating this setting where source data is unavailable during testing. This selection is advantageous when data for the desired task is sparse, but we can access other unrelated data, even from significantly different domains.

For a motivating example, we can use our method for the diagnosis of COVID-19 from chest radiographs. Because COVID-19 is a novel epidemic, it is very likely that there is a domain shift in this application, as the training data have limited coverage and test data may come from different hospitals and different instruments, making adaptation necessary. Our method allows us to use other unrelated data from other domains like usual images, which are very easy to obtain, as auxiliary data to help regularize adaptation.

To be more specific, we have two copies feature extractor at the beginning of testing, one is the fixed source-only version $g(\cdot;\theta_0)$, and the other keeps updating $g(\cdot;\theta)$ during test time adaptation (from $g(\cdot;\theta_0)$). The loss is then calculated between $\mathbf{z}_0 = g(x;\theta_0)$ and $\mathbf{z} = g(x;\theta)$. Instead of using L1/L2 loss to directly minimize the difference between $\mathbf{z}_0$ and $\mathbf{z}$, motivated by [15], we employ a similarity-based loss to keep the relative similarity for different auxiliary data samples unchanged. Specifically, let $Z^{l,0} \in \mathbb{R}^{b\times d}$ and $Z^{l} \in \mathbb{R}^{b\times d}$ denote the activations of l-th layer from $g(\cdot;\theta_0)$ and $g(\cdot;\theta)$ respectively, where b is the batch size and d is the dimension of the activations after reshaping. The corresponding similarity matrices are computed as $S^l = \frac{Z^l \cdot Z^{l\mathrm{T}}}{\left\| Z^l \cdot Z^{l\mathrm{T}} \right\|_2}$ and $S^{l,0} = \frac{Z^{l,0} \cdot Z^{l,0\mathrm{T}}}{\left\| Z^{l,0} \cdot Z^{l,0\mathrm{T}} \right\|_2}$, respectively, where $\left\|\cdot\right\|_2$ is a row-wise l_2-norm. We then define the auxiliary data regularization loss of l-th layer as

$$\mathcal{L}^l = \frac{1}{b^2} \left\| S^l - S^{l,0} \right\|_F^2 , \tag{2}$$

where $\left\|\cdot\right\|_F$ is the Frobenius norm. Finally, we compute the sum of auxiliary data regularization losses across a set of intermediate layers L:

$$\mathcal{L}_{\text{aux}} = \sum_{l \in L} \mathcal{L}^l \tag{3}$$

3.3 PAD Learning

The proposed PAD learning framework consists of prototypical learning with augmented samples as well as auxiliary data regularization. Consequently, the overall objective loss function of PAD is:

$$\mathcal{L}_{\text{PAD}} = \mathcal{L}_{\text{proto}} + \lambda \cdot \mathcal{L}_{\text{aux}}, \tag{4}$$

where λ is a hyperparameter balancing the two loss components.

4 Experiments

We evaluated our method on both Mixed CIFAR-10-C (artificial domain shift) for DaTTA and CIFAR-10.1 (unknown natural domain shift) for standard TTA. A subset of Tiny-Imagenet-200 is used for auxiliary data, with no category overlap with CIFAR-10. More details are in Appendix B.

4.1 Results on Mixed CIFAR-10-C and CIFAR-10.1

Baselines We compare our method with other test-time adaptation methods, including TTT, Norm ([16], [17]), and TENT. TENT and Norm use a larger batch size of 128, which violates the setting that test data come in a stream, while TTT and ours keep batch size = 1. TTT (standard) is an offline variant in which they only adapt with a test sample. Results are shown in Table 1.

Mixed CIFAR-10-C On Mixed CIFAR-10-C, our method (9.40%) achieved significant improvement compared with all other baselines (no more than 6.64%) even if our baseline is comparable with that for TTT or already much higher than that for TENT. Additionally, TENT has a much lower baseline as they use batch normalization instead of group normalization, even if batch normalization has a lower error rate on the original dataset.

Table 1: Experiment results on Mixed CIFAR-10-C and CIFAR-10.1.

Methods		Mixed CIFAR-10-C		CIFAR-10.1	
		Test error (%)	Improvement	Test error (%)	Improvement
Source only	Baseline (for TENT w/ Norm)	47.83		14.30	
	TTT baseline (GN w/ joint training)	34.63		16.60	
	TTT baseline (GN w/ joint training)*	33.87		16.70	
	Our baseline (GN w/ cosine classifier)	34.72		18.85	
Test-time adaptation	Norm† [16], [17]	44.60	3.23	73.75	-59.45
	TENT† [11]	44.76	3.07	73.85	-59.55
	TTT (standard)* [10]	-	-	15.90	0.80
	TTT (standard) [10]	31.50	2.36	16.00	0.60
	TTT-online [10]	27.23	6.64	14.40	2.30
	PAD (Ours)	25.32	9.40	4.80	14.05

* Indicates the results reported on their paper, † indicates that they use a larger batch size (128), which is an advantage (not entirely fair comparison). GN stands for group normalization. Our method's test set error for the original CIFAR-10 is 9.21%.

CIFAR-10.1 It is surprising that our method achieves a much lower error rate than that on the original test set (9.21%), outperforming other methods by a large margin, even if our baseline is lower than that for other methods, indicating our method's adaptation to this new test set is quite successful. This is significant as this domain shift largely remains unknown, and TTT only partly managed to fill the gap between the performance. TENT and Norm lead to model collapse on this new dataset, even if their baseline is better.

4.2 Ablation Study

Table 2: Performance contribution of each part in PAD on CIFAR-10.1.

Methods	Error (%)	Improvement
Baseline	18.85	
+ voting	15.10	3.75
+ $\mathcal{L}_{\mathrm{proto}}$	10.90	4.20
+ $\mathcal{L}_{\mathrm{aux}}$ (PAD)	4.80	6.10

For the ablation study, we use CIFAR-10.1, and the results are available in Table 2. We find that every component contributes to our final result. Crucially, the regularization with auxiliary data contributes the most improvement on top of other components. Surprisingly, we find voting over augmentations reasonably effective (3.75% improvement), which is more than that for TTT-online, showing the potential of exploiting the training augmentations during test-time adaptation.

5 Discussion

The selection for auxiliary data is arbitrary in our experiments, and we have not tried other auxiliary datasets yet, leaving for future work to investigate the influence for different auxiliary data.

Why do we use the L1 loss for prototypical loss $\mathcal{L}_p$ instead of L2? We find L1 loss slightly outperforms L2 loss without auxiliary data.

TTA is also important in other tasks, like segmentation and detection, as domain shift is common in real applications, and sometimes we may not be aware of it. More generally, we hope this work can encourage researchers to bring down the barrier between training and testing in real deployments, as test-time adaptation models have larger capacities than typical domain adaptation models as they can learn from test data.

6 Conclusion

In this paper, we investigate Domain-agnostic Test-time Adaptation, where source data is not accessible during the adaptation process. We propose a framework to perform prototypical training with auxiliary data (PAD). Experiment results on two datasets, together with the ablation study, demonstrate the superiority of PAD over previous best methods.

References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012. 1

[2] 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. 1

[3] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 1

[4] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528. IEEE, 2011. 1

[5] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014. 1

[6] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176, 2017. 1

[7] 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. 1

[8] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pages 97–105. PMLR, 2015. 1

[9] 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. 1

[10] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International Conference on Machine Learning*, pages 9229–9248. PMLR, 2020. 1, 2, 5

[11] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*, 2020. 1, 2, 3, 5

[12] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018. 1

[13] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015. 1

[14] Xiangyu Yue, Zangwei Zheng, Shanghang Zhang, Yang Gao, Trevor Darrell, Kurt Keutzer, and Alberto Sangiovanni Vincentelli. Prototypical cross-domain self-supervised learning for few-shot unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13834–13844, 2021. 3

[15] Muhammad Awais, Fengwei Zhou, Hang Xu, Lanqing Hong, Ping Luo, Sung-Ho Bae, and Zhenguo Li. Adversarial robustness for unsupervised domain adaptation. *arXiv preprint arXiv:2109.00946*, 2021. 4

[16] Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. *Advances in Neural Information Processing Systems*, 33, 2020. 4, 5

[17] Zachary Nado, Shreyas Padhy, D Sculley, Alexander D'Amour, Balaji Lakshminarayanan, and Jasper Snoek. Evaluating prediction-time batch normalization for robustness under covariate shift. *arXiv preprint arXiv:2006.10963*, 2020. 4, 5

[18] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do cifar-10 classifiers generalize to cifar-10? *arXiv preprint arXiv:1806.00451*, 2018. 8

Appendix

A Problem Setting

Table 3: Comparison for data usage for updating model for different settings.

Setting	Training		Testing	
	Source data	Target data	Source data	Target data
Fine-tuning	x_s, y_s	x_t, y_t	-	-
Domain adaptation	x_s, y_s	x_t	-	-
Test-time training	x_s, y_s	-	x_s, y_s	x_t
Test-time adaptation	x_s, y_s	-	-	x_t
Domain-agnostic test-time adaptation	x_s, y_s	-	-	$x_t^1; x_t^2; \dots$

x_s and y_s stands for source data and label, x_t and y_t stands for target data and label, respectively. Different superscript for x_t like x_t^1, x_t^2 stands for different target domain data.

A.1 Test-time Adaptation

Adaptation addresses generalization from source to target. A model $f_\theta(x)$ with parameters theta trained on source data x_s and label y_s may generalize poorly when tested on x_t because of domain shift. Test-time adaptation only needs the model $f_\theta(x)$ and unlabeled target data x_t but not the source data x_s during inference.

A.2 Domain-agnostic Test-time Adaptation

In usual test-time adaptation, the test data are usually from a single domain. For example, the source domain is CIFAR-10, and the target domain is Gaussian noise in CIFAR-10-C. However, in real applications, we may not have such prior for the test data, and the data may come from different users, and every user's data may be a domain. Such data can be mixed together. Sometimes we can use users' identifiers as domain labels, but every users' data can also come from different domains, like from different places and times, making such domain labels useless.

In domain-agnostic test-time adaptation, the test data are from different domains, but the domain label is unavailable. There is no prior on which domain the next test image will be, making it even more challenging. Take our newly proposed Mixed CIFAR-10-C, for example (details to be explained in the Experiments), the model is solely trained on the training set of CIFAR-10, but it is tested on a mixture of 15 kinds of different corruptions on the test set for CIFAR-10.

B Experiment details

We optimize selectively in two aspects: selective in data and selective in certain layers for optimization. For data, we only optimize the model for the samples that have consistent predictions for the original version and augmented version, and the number of votes for the voting prediction is more than half. For different layers, we only optimize the first several layers, as shallow features are usually more low level and more related to domain-specific features.

Our implementation is in PyTorch. The number of augmentations for voting is set to 32. For better stability, we not only use features from the same layer as the input for the cosine classifier h but also features from other layers determined by the layer group division for the architecture (but not all layers) for the auxiliary data regularization loss.

CIFAR-10 has 10 classes, a training set of 50,000, and a test set of 10,000. CIFAR-10-C is generated by adding different 15 types of corruptions on the test set for CIFAR-10-C. Mixed CIFAR-10-C is derived from CIFAR-10-C by randomizing corruptions types. We split 150000 images in Mixed CIFAR-10-C to 15 splits as in CIFAR-10-C and keep the label sequence the same as CIFAR-10-C for better comparison. We use level 5 corruption for the most domain shift.

CIFAR-10.1 [18] is a new test set of size 2000 modeled after CIFAR-10, with the exact same classes and image dimensionality, following the dataset creation process documented by the original CIFAR-

10 paper as closely as possible. The domain shift between the original test set and CIFAR-10.1 remains unknown as there is no visual difference for humans, but the performance drop for models trained on CIFAR-10 is significant.

Tiny-imagenet-200 is employed as auxiliary data, and the batch size for auxiliary data is set to 512. Specifically, we use its training set, which contains 200 classes, and each class has 50000 images.

C Future Work

We will try different kinds of auxiliary data to see the robustness of our method on different kinds of auxiliary data. Also, we believe it will be very interesting to look into the circumstance where auxiliary data are a subset of training data (for a slightly different setting where only a small subset of training data are available during testing, this can be the case when a few data are allowed to distribute to the inference server) or auxiliary data are from a different domain (for example, training data are Clip art but auxiliary data are real images).

We will also compare our method against the method that adds loss to minimize the change of parameters.