If your data distribution shifts, use self-learning

Anonymous authors
Paper under double-blind review

Abstract

We demonstrate that self-learning techniques like entropy minimization and pseudo-labeling are simple and effective at improving performance of a deployed computer vision model under systematic domain shifts. We conduct a wide range of large-scale experiments and show consistent improvements irrespective of the model architecture, the pre-training technique or the type of distribution shift. At the same time, self-learning is simple to use in practice because it does not require knowledge or access to the original training data or scheme, is robust to hyperparameter choices, is straight-forward to implement and requires only a few adaptation epochs. This makes self-learning techniques highly attractive for any practitioner who applies machine learning algorithms in the real world. We present state-of-the-art adaptation results on CIFAR10-C (8.5% error), ImageNet-C (22.0% mCE), ImageNet-R (17.4% error) and ImageNet-A (14.8% error), theoretically study the dynamics of self-supervised adaptation methods and propose a new classification dataset (ImageNet-D) which is challenging even with adaptation.

1 Introduction

Deep Neural Networks (DNNs) can reach human-level performance in complex cognitive tasks (Brown et al., 2020; He et al., 2016a; Berner et al., 2019) if the distribution of the test data is sufficiently similar to the training data. However, DNNs are known to struggle if the distribution of the test data is shifted relatively to the training data (Geirhos et al., 2018; Dodge & Karam, 2017).

Two largely distinct communities aim to increase the performance of models under test-time distribution shifts: The *robustness community* generally considers ImageNet-scale datasets and evaluates models in an *ad-hoc* scenario. Models are trained on a clean source dataset like ImageNet (Deng et al., 2009), using heavy data augmentation (Hendrycks et al., 2020a; Rusak et al., 2020; Geirhos et al., 2019) and/or large-scale pre-training (Xie et al., 2020a; Mahajan et al., 2018). The trained models are not adapted in any way to test-time distribution shifts. This evaluation scenario is relevant for applications in which very different distribution shifts are encountered in an unpredictable order, and hence misses out on the gains of adaptation to unlabeled samples of the target distribution.

The *unsupervised domain adaptation (UDA) community* often considers smaller-scale datasets and assumes that both the source and the (unlabeled) target dataset are known. Models are trained on both datasets, e.g., with an adversarial objective (Ganin et al., 2016; Tzeng et al., 2017; Hoffman et al., 2018), before evaluation on the target domain data. This evaluation scenario provides optimal conditions for adaptation, but the reliance on the source dataset makes UDA more computationally expensive, more impractical and prevents the use of pre-trained models for which the source dataset is unknown or simply too large. We refer the reader to Farahani et al. (2021) for a review of UDA.

In this work, we consider the *source-free domain adaptation setting*, a middle ground between the classical ad-hoc robustness setting and UDA in which models can adapt to the target distribution but without using the source dataset (Kundu et al., 2020; Kim et al., 2021; Li et al., 2020; Liang et al., 2020). This evaluation scenario is interesting for many practitioners and applications as an extension of the ad-hoc robustness scenario. It evaluates the possible performance of a *deployed* model on a systematic, unseen distribution shift at inference time: an embedded computer vision system in an autonomous car should adapt

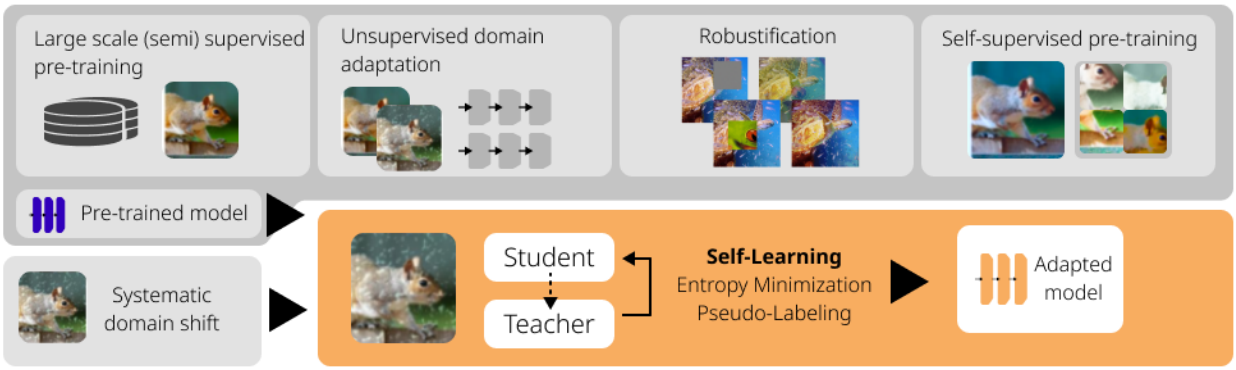

Figure 1: Robustness and adaptation to new datasets has traditionally been achieved by robust pre-training (with hand-selected/data-driven augmentation strategies, or additional data), unsupervised domain adaptation (with access to unlabeled samples from the test set), or, more recently, self-supervised learning methods. We show that on top of these different pre-training tasks, it is always possible (irrespective of architecture, model size or pre-training algorithm) to further adapt models to the target domain with simple self-learning techniques.

to changes without being trained on all available training data; an image-based quality control software may not necessarily open-source the images it has been trained on, but still has to be adapted to the lighting conditions at the operation location; a computer vision system in a hospital should perform robustly when tested on a scanner different from the one used for producing the training images—importantly, it might not be known at development time which scanner the vision system will be tested on, and it might be prohibited to share images from many hospitals to run UDA.

Can self-learning methods like *pseudo-labeling* and *entropy-minimization* also be used in this *source-free* domain adaptation setting? To answer this question, we perform an extensive study of several self-learning variants, and find consistent and substantial gains in test-time performance across several robustness and out-of-domain benchmarks and a wide range of models and pre-training methods, including models trained with UDA methods that do not use self-learning, see Figure 1. We also find that self-learning outperforms state-of-the-art source-free domain adaptation methods, namely Test-Time Training which is based on a self-supervised auxiliary objective and continual training (Sun et al., 2020), test-time entropy minimization (Wang et al., 2021), Meta Test-Time Training (Bartler et al., 2022), and (gradient-free) BatchNorm adaptation (Schneider et al., 2020; Nado et al., 2020). We perform a large number of ablations to study important design choices for self-learning methods in source-free domain adaptation. Furthermore, we show that a variant of pseudo-labeling with a robust loss function consistently outperforms entropy minimization on ImageNet-scale datasets.

We begin by positioning our work in the existing literature (§ 2) and proceed with an overview of various self-learning variants that have been applied over the past years, and propose a new technique for robust pseudo-labeling (§ 3). We then outline a rigorous experimental protocol that aims to highlight the strengths (and shortcomings) of various self-learning methods. We test various model architectures, with different pre-training schemes covering the most important models in unsupervised domain adaptation, robustness, and large-scale pre-training (§ 4). Using this protocol, we show the effectiveness of self-learning across architectures, models and pre-training schemes (§ 5). We proceed with an in-depth analysis of self-learning, both empirical (§ 6) and theoretical (§ 7). Since the outlined results on ImageNet-C (22.0% mCE), ImageNet-R (17.4% error) and ImageNet-A (14.8%) approach clean performance (11.6% error for our baseline), we propose ImageNet-D as a new benchmark, which we analyse in § 8. We conclude by proposing a set of best practices for evaluating test-time adaptation techniques in the future to ensure scientific rigor and to enable fair model and method comparisons (§ 9).

2 Related Work

Xie et al. (2020b) introduce "In-N-Out" which uses auxiliary information to boost both in- and out-of-distribution performance. AdaMatch (Berthelot et al., 2021) builds upon FixMatch (Sohn et al., 2020) and

can be used for the tasks of unsupervised domain adaptation, semi-supervised learning and semi-supervised domain adaptation as a general-purpose algorithm. Prabhu et al. (2021) propose SENTRY, an algorithm based on judging the predictive consistency of samples from the target domain under different image transformations. Zou et al. (2019) show that different types of confidence regularization can improve the performance of self-learning. A theoretically motivated framework for self-learning in domain adaptation based on consistency regularization has been proposed by Wei et al. (2020) and then extended by Cai et al. (2021).

The main differences from these works to ours are that they 1) utilize both source and target data during training (i.e., the classical UDA setup) whereas we only require access to unlabeled target data (source-free setup), 2) train their models from scratch whereas we adapt pretrained checkpoints to the unlabeled target data, and 3) are oftentimes more complicated (also in terms of the number of hyperparameters) than our approach due to using more than one term in the objective function. We would like to highlight that utilizing source data should always result in better performance compared to not using source data. Our contribution is to show that self-learning can still be very beneficial with a small compute budget and no access to source data. Our setup targets "deployed systems", e.g., a self-driving car or a detection algorithm in a production line which adapts to the distribution shift "on-the-fly" and cannot (or should not) be retrained from scratch for every new domain shift.

Our work is conceptually most similar to virtual adversarial domain adaptation in the fine-tuning phase of DIRT-T (Shu et al., 2018) and Test-time entropy minimization (TENT; Wang et al., 2021). In contrast to DIRT-T, our objective is simpler and we scale the approach to considerably larger datasets on ImageNet scale. TENT, on the other hand, only evaluated a single method (entropy minimization) on a single vanilla model (ResNet-50) on ImageNet-C (Hendrycks & Dietterich, 2019). We substantially expand this analysis to show that self-learning almost universally increases test-time performance under distribution shifts, regardless of the type of distribution shift, the model architecture, the pre-training method or the self-learning loss function.

Gulrajani & Lopez-Paz (2021) show that model selection for hyperparameter tuning is non-trivial for the task of domain generalization, and propose model selection criteria under which models should be selected for this task. Following their spirit, we identify a model selection criterion for test-time adaptation, and rigorously use it in all our experiments. We outperform state-of-the-art techniques which did not disclose their hyperparameter selection protocols.

Kumar et al. (2020) study the setting of self-learning for gradual domain adaptation. They find that self-learning works better if the data distribution changes slowly. The gradual domain adaptation setting differs from ours: instead of a gradual shift over time, we focus on a fixed shift at test time. Kumar et al. (2020) tested their method on small-scale datasets; building and evaluating ImageNet-scale datasets for the task of gradual domain adaptation is left for future work, and would not only require changes/adaptations to the self-learning method, but also to the evaluation datasets.

Azimi et al. (2022) show performance improvements when using test-time adaptation on video data. They show adaptation results for the popular test-time adaptation techniques of BN adaptation (Schneider et al., 2020), Test-Time Training (Sun et al., 2020) and test-time entropy minimization (Wang et al., 2021). Meta Test-Time Training (Bartler et al., 2022) combine meta-learning, self-supervision and test-time training to adapt a model trained on clean CIFAR10 to CIFAR10-C.

3 Self-learning for Test-Time Adaptation

Different variants of self-learning have been used in both unsupervised domain adaptation (French et al., 2018; Shu et al., 2018), self-supervised representation learning (Caron et al., 2021), and in semi-supervised learning (Xie et al., 2020a). In a typical self-learning setting, a *teacher* network $\mathbf{f}^t$ trained on the source domain predicts labels on the target domain. Then, a *student* model $\mathbf{f}^s$ is fine-tuned on the predicted labels.

In the following, let $\mathbf{f}^t(\mathbf{x})$ denote the logits for sample $\mathbf{x}$ and let $p^t(j|\mathbf{x}) \equiv \sigma_j(\mathbf{f}^t(\mathbf{x}))$ denote the probability for class j obtained from a softmax function $\sigma_j(\cdot)$. Similarly, $\mathbf{f}^s(\mathbf{x})$ and $p^s(j|\mathbf{x})$ denote the logits and probabilities for the student model $\mathbf{f}^s$. For all techniques, one can optionally only admit samples where the probability

$\max_j p^t(j|\mathbf{x})$ exceeds some threshold. We consider three popular variants of self-learning: Pseudo-labeling with hard or soft labels, as well as entropy minimization.

Hard Pseudo-Labeling (Lee, 2013; Galstyan & Cohen, 2007). We generate labels using the teacher and train the student on pseudo-labels i using the softmax cross-entropy loss,

$$\ell_H(\mathbf{x}) := -\log p^s(i|\mathbf{x}), \quad i = \text{argmax}_j\, p^t(j|\mathbf{x}) \tag{1}$$

Usually, only samples with a confidence above a certain threshold are considered for training the student. We test several thresholds but note that thresholding means discarding a potentially large portion of the data which leads to a performance decrease in itself. The teacher is updated after each epoch.

Soft Pseudo-Labeling (Lee, 2013; Galstyan & Cohen, 2007). In contrast to the hard pseudo-labeling variant, we here train the student on class probabilities predicted by the teacher,

$$\ell_S(\mathbf{x}) := -\sum_j p^t(j|\mathbf{x}) \log p^s(j|\mathbf{x}). \tag{2}$$

Soft pseudo-labeling is typically not used in conjunction with thresholding, since it already incorporates the certainty of the model. The teacher is updated after each epoch.

Entropy Minimization (ENT; Grandvalet & Bengio, 2004; Wang et al., 2021). This variant is similar to soft pseudo-labeling, but we no longer differentiate between a teacher and student network. It corresponds to an "instantaneous" update of the teacher. The training objective becomes

$$\ell_E(\mathbf{x}) := -\sum_j p^s(j|\mathbf{x}) \log p^s(j|\mathbf{x}). \tag{3}$$

Intuitively, self-learning with entropy minimization leads to a sharpening of the output distribution for each sample, making the model more confident in its predictions.

Robust Pseudo-Labeling (RPL). Virtually all introduced self-learning variants use the softmax cross-entropy classification objective. However, the softmax cross-entropy loss has been shown to be sensitive to label noise (Zhang & Sabuncu, 2018; Zhang et al., 2017). In the setting of domain adaptation, inaccuracies in the teacher predictions and, thus, the labels for the student, are inescapable, with severe repercussions for training stability and hyperparameter sensitivity as we show in the results.

As a straight-forward solution to this problem, we propose to replace the cross-entropy loss by a robust classification loss designed to withstand certain amounts of label noise (Ghosh et al., 2017; Song et al., 2020; Shu et al., 2020; Zhang & Sabuncu, 2018). A popular candidate is the *Generalized Cross Entropy (GCE)* loss which combines the noise-tolerant Mean Absolute Error (MAE) loss (Ghosh et al., 2017) with the CE loss. We only consider the hard labels and use the robust GCE loss as the training loss for the student,

$$i = \text{argmax}_j\, p^t(j|\mathbf{x}), \quad \ell_{GCE}(\mathbf{x}, i) := q^{-1}(1 - p^s(i|\mathbf{x})^q), \tag{4}$$

with $q \in (0, 1]$. For the limit case $q \to 0$, the GCE loss approaches the CE loss and for $q = 1$, the GCE loss is the MAE loss (Zhang & Sabuncu, 2018). We test updating the teacher both after every update step of the student (RPL) and once per epoch (RPL^{ep}).

4 Experiment design

Datasets. ImageNet-C (IN-C; Hendrycks & Dietterich, 2019) contains corrupted versions of the 50 000 images in the ImageNet validation set. There are fifteen test and four hold-out corruptions, and there are five severity levels for each corruption. The established metric to report model performance on IN-C is the mean Corruption Error (mCE) where the error is normalized by the AlexNet error, and averaged over all corruptions and severity levels, see Eq. 20, Appendix C.1. ImageNet-R (IN-R; Hendrycks et al., 2020a) contains 30 000 images with artistic renditions of 200 classes of the ImageNet dataset. ImageNet-A (IN-A; Hendrycks et al., 2019) is composed of 7500 unmodified real-world images on which standard ImageNet-trained ResNet50

Table 1: Self-learning decreases the error on ImageNet-scale robustness datasets. Robust pseudo-labeling generally outperforms entropy minimization.

mCE [%] on IN-C test (↘)	number of parameters	w/o adapt	w/ adapt (Δ) RPL	w/ adapt (Δ) ENT
ResNet50 vanilla (He et al., 2016b)	2.6×10^7	76.7	50.5 (-26.2)	51.6 (-25.1)
ResNet50 DAug+AM (Hendrycks et al., 2020a)	2.6×10^7	53.6	41.7 (-11.9)	42.6 (-11.0)
DenseNet161 vanilla (Huang et al., 2017)	2.8×10^7	66.4	47.0 (-19.4)	47.7 (-18.7)
ResNeXt101$_{32\times8d}$ vanilla (Xie et al., 2017)	8.8×10^7	66.6	43.2 (-23.4)	44.3 (-22.3)
ResNeXt101$_{32\times8d}$ DAug+AM (Hendrycks et al., 2020a)	8.8×10^7	44.5	34.8 (-9.7)	35.5 (-9.0)
ResNeXt101$_{32\times8d}$ IG-3.5B (Mahajan et al., 2018)	8.8×10^7	51.7	40.9 (-10.8)	40.8 (-10.9)
EfficientNet-L2 Noisy Student (Xie et al., 2020a)	4.8×10^8	28.3	**22.0** (-6.3)	23.0 (-5.3)
top1 error [%] on IN-R (↘)				
ResNet50 vanilla (He et al., 2016b)	2.6×10^7	63.8	54.1 (-9.7)	56.1 (-7.7)
EfficientNet-L2 Noisy Student (Xie et al., 2020a)	4.8×10^8	23.5	**17.4** (-6.1)	19.7 (-3.8)
top1 error [%] on ImageNet-A (↘)				
EfficientNet-L2 Noisy Student (Xie et al., 2020a)	4.8×10^8	16.5	**14.8** (-1.7)	15.5 (-1.0)

(He et al., 2016b) models yield chance level performance. CIFAR10 (Krizhevsky et al., 2009) and STL10 (Coates et al., 2011) are small-scale image recognition datasets with 10 classes each, and training sets of 50 000/5000 images and test sets of 10 000/8000 images, respectively. The digit datasets MNIST (Deng, 2012) and MNIST-M (Ganin et al., 2016) both have 60 000 training and 10 000 test images.

Adaptation parameters. In most of our experiments, we only adapt the affine scale and shift parameters γ and β following the batch normalization layers (Ioffe & Szegedy, 2015). We verify that this type of adaptation works better than full model adaptation for large models in an ablation study in Section 6.

Hyperparameters. The different self-learning variants have a range of hyperparameters such as the learning rate or the stopping criterion. Our goal is to give a realistic estimation on the performance to be expected in practice. To this end, we optimize hyperparameters for each variant of pseudo-labeling on a hold-out set of IN-C that contains four types of image corruptions ("speckle noise", "Gaussian blur", "saturate" and "spatter") with five different strengths each, following the procedure suggested in Hendrycks & Dietterich (2019). We refer to the hold-out set of IN-C as our *dev* set. On the small-scale datasets, we use the hold-out set of CIFAR10-C for hyperparameter tuning. On all other datasets, we use the hyperparameters obtained on the hold-out sets of IN-C (for large-scale datasets) or CIFAR10-C (on small-scale datasets).

Models for ImageNet-scale datasets. We consider five popular model architectures: ResNet50 (He et al., 2016b), DenseNet161 (Huang et al., 2017), ResNeXt101 (Xie et al., 2017), EfficientNet-L2 (Tan & Le, 2019), and the Vision Transformer (ViT; Dosovitskiy et al., 2021) (see Appendix B.1 for details on the used models). For ResNet50, DenseNet and ResNeXt101, we include a simple *vanilla* version trained on ImageNet only. For ResNet50 and ResNeXt101, we additionally include a state-of-the-art robust version trained with DeepAugment and Augmix (DAug+AM; Hendrycks et al., 2020a)[1]. For the ResNeXt model, we also include a version that was trained on 3.5 billion weakly labeled images (IG-3.5B; Mahajan et al., 2018). For EfficientNet-L2 we select the current state of the art on IN-C which was trained on 300 million images from JFT-300M (Chollet, 2017; Hinton et al., 2014) using a noisy student-teacher protocol (Xie et al., 2020a). Finally, for the ViT, we use the model pretrained with DINO (Caron et al., 2021). We validate the ImageNet and IN-C performance of all considered models and match the originally reported scores (Schneider et al., 2020). For EfficientNet-L2, we match ImageNet top-1 accuracy up to 0.1% points, and IN-C up to 0.6% points mCE.

Models for CIFAR10/ MNIST-scale datasets. For CIFAR10-C experiments, we use three WideResNets (WRN; Zagoruyko & Komodakis, 2016): the first one is trained on clean CIFAR10 and has a depth of 28 and a width of 10, the second one is trained with CIFAR10 with the AugMix protocol (Hendrycks et al., 2020b) and has a depth of 40 and a width of 2, and the third one has a depth of 26 layers, and is pre-trained on clean CIFAR10 using the default training code from `https://github.com/kuangliu/pytorch-cifar`.

[1]see leaderboard at `github.com/hendrycks/robustness`

Table 2: Self-learning decreases the error on small-scale datasets, for models pre-trained using data augmentation and unsupervised domain adaptation. Entropy minimization outperforms robust pseudo-labeling.

top1 error [%] on CIFAR10-C (↘)	number of parameters	w/o adapt	w/ adapt (Δ) RPL	w/ adapt (Δ) ENT
WRN-28-10 vanilla (Zagoruyko & Komodakis, 2016)	3.6×10^7	26.5	13.7 (-12.8)	13.3 (-13.2)
WRN-40-2 AM (Hendrycks et al., 2020b)	2.2×10^6	11.2	9.0 (-2.2)	8.5 (-2.7)
WRN-26-1-GN (Bartler et al., 2022)	1.5×10^6	18.6	18.0 (-0.6)	18.4 (0.2)
WRN-26-1-BN (Zagoruyko & Komodakis, 2016)	1.5×10^6	25.8	15.1 (-10.7)	13.1 (-12.7)
WRN-26-16 UDA-SS (Sun et al., 2019)	9.3×10^7	27.7	18.2 (-9.5)	16.7 (-11.0)
WRN-26-16 DANN (Ganin et al., 2016)	9.3×10^7	29.7	28.6 (-1.1)	28.5 (-1.2)
UDA CIFAR10→STL10, top1 error on target [%](↘)				
WRN-26-16 UDA-SS (Sun et al., 2019)	9.3×10^7	28.7	22.9 (-5.8)	21.8 (-6.9)
WRN-26-16 DANN (Ganin et al., 2016)	9.3×10^7	25.0	24.0 (-1.0)	23.9 (-1.1)
UDA MNIST→MNIST-M, top1 error on target [%](↘)				
WRN-26-16 UDA-SS (Sun et al., 2019)	9.3×10^7	4.8	2.4 (-2.4)	2.0 (-2.8)
WRN-26-2 DANN (Ganin et al., 2016)	1.5×10^6	11.4	5.2 (-6.2)	5.1 (-6.3)

The remaining small-scale models are trained with UDA methods. We propose to regard any UDA method which requires joint training with source and target data as a pre-training step, similar to regular pre-training on ImageNet, and use self-learning on top of the final checkpoint. We consider two popular UDA methods: self-supervised domain adaptation (UDA-SS; Sun et al., 2019) and Domain-Adversarial Training of Neural Networks (DANN; Ganin et al., 2016). In UDA-SS, the authors seek to align the representations of both domains by performing an auxiliary self-supervised task on both domains simultaneously. In all UDA-SS experiments, we use a WideResNet with a depth of 26 and a width of 16. In DANN, the authors learn a domain-invariant embedding by optimizing a minimax objective. For all DANN experiments except for MNIST→MNIST-M, we use the same WRN architecture as above. For the MNIST→MNIST-M experiment, the training with the larger model diverged and we used a smaller WideResNet version with a width of 2. We note that DANN training involves optimizing a minimax objective and is generally harder to tune.

5 Self-learning universally improves models

Self-learning is a powerful learning scheme, and in the following section we show that it allows to perform test-time adaptation on robustified models, models obtained with large-scale pre-training, as well as (already) domain adapted models across a wide range of datasets and distribution shifts. Our main results on large-scale and small-scale datasets are shown in Tables 1 and 2. These summary tables show final results, and all experiments use the hyperparameters we determined separately on the dev set. The self-learning loss function, i.e. soft- or hard-pseudo-labeling / entropy minimization / robust pseudo-labeling, is a hyperparameter itself, and thus, in Tables 1 and 2, we show the overall best results. Results for the other loss functions can be found in Section 6 and in Appendix C.

Self-learning successfully adapts ImageNet-scale models across different model architectures on IN-C, IN-A and IN-R (Table 1). We adapt the vanilla ResNet50, ResNeXt101 and DenseNet161 models to IN-C and decrease the mCE by over 19 percent points in all models. Further, self-learning works for models irrespective of their size: Self-learning substantially improves the performance of the ResNet50 and the ResNext101 trained with DAug+AM, on IN-C by 11.9 and 9.7 percent points, respectively. Finally, we further improve the current state of the art model on IN-C—the EfficientNet-L2 Noisy Student model—and report a new state-of-the-art result of 22% mCE (which corresponds to a top1 error of 17.1%) on this benchmark with test-time adaptation (compared to 28% mCE without adaptation).

Self-learning is not limited to the distribution shifts in IN-C like compression artefacts or blur. On IN-R, a dataset with renditions, self-learning improves both the vanilla ResNet50 and the EfficientNet-L2 model, the latter of which improves from 23.5% to a new state of the art of 17.4% top-1 error. For a vanilla ResNet50, we improve the top-1 error from 63.8% (Hendrycks et al., 2020a) to 54.1%. On IN-A, adapting the EfficientNet-L2 model using self-learning decreases the top-1 error from 16.5% (Xie et al., 2020a) to 14.8% top-1 error, again

Table 3: Unlike batch norm adaptation, self-learning adapts large-scale models trained on external data.

mCE, test [%] (↘)	w/o adapt	BN adapt	RPL
ResNeXt101 vanilla	66.6	56.8	43.2
ResNeXt101 IG-3.5B	51.7	51.8	**40.9**

Table 4: Properly tuned self-learning outperforms TENT and TTT.

mCE on IN-C test [%] (↘)	w/o adapt	BN adapt	TENT (ours)	RPL
ResNet50 vanilla	76.7	62.2	53.5 (51.6)	**50.5**
top1 error [%] on IN-C, sev. 5 (↘)	w/o adapt	BN adapt	TTT	RPL
ResNet18 vanilla	85.4	72.2	66.3	**61.9**

Table 5: Vision Transformers can be adapted with self-learning.

mCE on IN-C test [%] (↘)	w/o adapt	w/ adapt affine layers	w/ adapt bottleneck layers	w/ adapt lin. layers	w/ adapt all weights
ViT-S/16	62.3	51.8	46.8	45.2	**43.5**

constituting a new state of the art with test-time adaptation on this dataset. Self-learning can also be used in an online adaptation setting, where the model continually adapts to new samples on IN-C in Fig. 7(i) or IN-R Fig. 7(ii), Appendix C.10.

The improvements of self-learning are very stable: In Table 26, Appendix C.9, we show the averaged results across three different seeds for a ResNet50 model adapted with ENT and RPL. The unbiased std is roughly two orders of magnitude lower than any improvement we report in our paper, showcasing the robustness of our results to random initialization.

Self-learning improves robustified and domain adapted models on small-scale datasets (Table 2). We test common domain adaptation techniques like DANN (Ganin et al., 2016) and UDA-SS (Sun et al., 2019), and show that self-learning is effective at further tuning such models to the target domain. We suggest to view unsupervised source/target domain adaptation as a step comparable to pre-training under corruptions, rather than an adaptation technique specifically tuned to the target set—indeed, we can achieve error rates using, e.g., DANN + target adaptation previously only possible with source/target based pseudo-labeling, across different common domain adaptation benchmarks. Self-learning also decreases the error on CIFAR10-C of the Wide ResNet model trained with AugMix (AM, Hendrycks et al., 2020b) and reaches a new state of the art on CIFAR10-C of 8.5% top1 error with test-time adaptation.

Self-learning also improves large pre-trained models (Table 3). Unlike BatchNorm adaptation (Schneider et al., 2020), we show that self-learning transfers well to models pre-trained on a large amount of unlabeled data: self-learning decreases the mCE on IN-C of the ResNeXt101 trained on 3.5 billion weakly labeled samples (IG-3.5B, Mahajan et al., 2018) from 51.7% to 40.9%.

Self-learning outperforms previously published test-time adaptation approaches on IN-C (Table 4). The robustness benchmark IN-C has so far mostly been regarded in the ad-hoc evaluation setting as discussed in our introduction. Thus, there are only few published methods that report numbers for test-time adaptation: BatchNorm adaptation (Schneider et al., 2020), Test-Time Training (TTT, Sun et al., 2020), and TENT (Wang et al., 2021). In particular, note that TTT requires a special loss function at training time, while our approach is agnostic to the pre-training phase. Our self-learning results outperforms all three baselines (also after tuning TENT with our full experimental protocol). A detailed comparison to TTT is included in Appendix C.6. We also compared our approach to Meta Test-Time Training (MT3, Bartler et al., 2022), which combines meta-learning, self-supervision and test-time training for test-time adaptation. We find that both ENT and RPL perform better than MT3: using the same architecture as Bartler et al. (2022), our best error on CIFAR10-C is 18.0% compared to their best result of 24.4%. When exchanging GroupNorm layers (Wu & He, 2018) for BN layers, the error further reduces to 13.0%. We thus find that self-learning is more effective when adapting affine BN parameters instead of GN layers, which is consistent with the findings in Schneider et al. (2020). We included a detailed comparison to Bartler et al. (2022) in Appendix C.7.

Table 6: RPL (ENT) performs better on IN-C (CIFAR10-C).

	mCE, IN-C dev			err, C10-C
	ResNet50	ResNeXt-101	EffNet-L2	WRN-40
ENT	50.0 ± 0.04	43.0	22.2	**8.5**
RPL	**48.9** ± 0.02	**42.0**	**21.3**	9.0

Table 7: RPL performs best without a threshold.

threshold	0.0	0.5	0.9
mCE on IN-C dev [%]			
no adapt	69.5		
soft PL	60.1		
hard PL	53.8	51.9	52.4
RPL	**49.7**	49.9	51.8

Table 8: RPL performs best with instantaneous updates.

Update interval (RPL)	w/o adapt	none	epoch	instant
mCE, IN-C dev [%]	69.5	54.0	49.7	**49.2**

Self-supervised methods based on self-learning allow out-of-the-box test-time adaptation (Table 5). The recently published DINO method (Caron et al., 2021) is another variant of self-supervised learning that has proven to be effective for unsupervised representation learning. At the core, the method uses soft pseudo-labeling. Here, we test whether a model trained with DINO on the source dataset can be test-time adapted on IN-C using DINO to further improve out-of-distribution performance. We highlight that we specifically test the self-supervised DINO objective for its practicality as a test-time adaptation method, and did not switch the DINO objective for ENT or RPL to do test-time adaptation. Since the used model is a vision transformer model, we test different choices of adaptation parameters and find considerable performance improvements in all cases, yielding an mCE of 43.5% mCE at a parameter count comparable to a ResNet50 model. For adapting the affine layers, we follow Houlsby et al. (2019).

6 Understanding test-time adaptation with self-learning

In the following section, we show ablations and interesting insights of using self-learning for test-time adaptation. If not specified otherwise, all ablations are run on the hold-out corruptions of IN-C (our dev set) with a vanilla ResNet50.

Robust pseudo-labeling outperforms entropy minimization on large-scale datasets while the reverse is true on small-scale datasets (Table 6). We find that robust pseudo-labeling consistently improves over entropy minimization on IN-C, while entropy minimization performs better on smaller scale data (CIFAR10, STL10, MNIST). The finding highlights the importance of testing both algorithms on new datasets. The improvement is typically on the order of one percent point.

Robust pseudo-labeling allows usage of the full dataset without a threshold (Table 7). Classical hard labeling needs a confidence threshold (T) for best performance, thereby reducing the dataset size, while best performance for RPL is reached for full dataset training with a threshold T of 0.0. We corroborate the results of Wang et al. (2021) who showed that TENT outperforms hard labeling with a threshold on CIFAR10-C and CIFAR100-C.

Short update intervals are crucial for fast adaptation (Table 8). Having established that RPL generally performs better than soft- and hard-labeling, we vary the update interval for the teacher. We find that instant updates are most effective. In entropy minimization, the update interval is instant per default.

Table 9: RPL performs best when affine BN parameters are adapted.

Mechanism	w/o adapt	last layer	full model	affine
mCE, IN-C dev [%]	69.5	60.2	51.5	**48.9**
adapted parameters	0	2M	22.6M	5.3k

Table 10: Self-learning works better in models with BN layers compared to models with GN layers, although un-adapted models with GN are more stable under distribution shift compared to models with BN layers.

top1 error [%] on CIFAR10-C (↘)	number of parameters	w/o adapt	w/ adapt (Δ) RPL	w/ adapt (Δ) ENT
WRN-26-1-BN (Zagoruyko & Komodakis, 2016)	1.5×10^6	25.8	15.1 (-10.7)	13.1 (-12.7)
WRN-26-1-GN (Bartler et al., 2022)	1.5×10^6	18.6	18.4 (-0.2)	18.0 (-0.6)
mCE [%] on IN-C test (↘)				
ResNet50 BigTransfer (Kolesnikov et al., 2020)	2.6×10^7	55.0	54.4 (-0.6)	56.4 (+1.4)

Adaptation of only affine layers is important in CNNs (Table 9). On IN-C, adapting only the affine parameters after the normalization layers (i.e., the rescaling and shift parameters β and γ) works better on a ResNet50 architecture than adapting all parameters or only the last layer. We indicate the number of adapted parameters in brackets. Note that for Vision Transformers, full model adaptation works better than affine adaptation (see Table 5). We also noticed that on convolutional models with a smaller parameter count like ResNet18, full model adaptation is possible. Wang et al. (2021) also used the affine BN parameters for test-time adaptation with TENT. Here, we validate and corroborate this hyperparameter choice.

Affine BN parameters work better for test-time adaptation compared to GN parameters. (Table 10). Schneider et al. (2020) showed that models with batch normalization layers are less robust to distribution shift compared to models with group normalization (Wu & He, 2018) layers. However, after adapting BN statistics, the adapted model outperformed the non-adapted GN model. Here, we show that these results also hold for test-time adaptation when adapting a model with GN or BN layers. We show that a WideResNet-26-1 (WRN-26-1) vanilla model with BN layers pretrained on clean CIFAR10 has a much higher error on CIFAR10-C than the same model with GN layers, but it has a much lower error after adaptation. The full results for the WRN-26-1 model can be found in Appendix C.7. Further, we test the pretrained BigTransfer (Kolesnikov et al., 2020) models which have GN layers, and find only small improvements with RPL , and no improvements with ENT. There are no pretrained weights released for the BigTransfer models which have BN layers, thus, a comparison similar to the WRN-26-1 model is not possible. A more detailed discussion on our BigTransfer results as well as a hyperparameter selection study can be found in Appendix E.2.

Hyperparameters obtained on corruption datasets transfer well to real world datasets. When evaluating models, we select the hyperparameters discussed above (the learning rate and the epoch used for early stopping are the most critical ones) on the dev set (full results in Appendix C.2). We note that this technique transfers well to IN-R and -A, highlighting the practical value of corruption robustness datasets for adapting models on real distribution shifts.

Learning rate and number of training epochs are important hyperparameters We tune the learning rate as well as the number of training epochs for all models, except for the EfficientNet-L2 model where we only train for one epoch due to computational constraints. In Tables 14, 15, 16, and 17 in Appendix C.2, we show that both ENT and RPL collapse after a certain number of epochs, showing the inherent instability of pseudo-labeling. While Wang et al. (2021) trained their model only for one epoch with one learning rate, we rigorously perform a full hyperparameter selection on a hold-out set (our dev set), and use the optimal hyperparameters on all tested datasets. We believe that this kind of experimental rigor is essential in order to be able to properly compare methods.

Additional experiments and ablation studies, as well as detailed results for all models and datasets can be found in Appendix C. We discuss additional proof-of-concept implementations on the WILDS benchmark (Koh et al., 2021), BigTransfer (BiT; Chen et al., 2020) models and on self-learning based UDA models in Appendix E. On WILDS, self-learning is effective for the Camelyon17 task with a systematic shift between train, validation and test sets (each set is comprised of different hospitals), while self-learning fails to improve on tasks with mixed domains, such as on the RxRx1 and the FMoW tasks.These results support our claim that self-learning is effective, while showing the important limitation when applied to more diverse shifts.

7 A simple model of stability in self-learning

We observed that different self-learning schemes are optimal for small-scale vs. large-scale datasets and varying amount of classes. We reconsider the used loss functions, and unify them into

$$\begin{aligned} \ell(\mathbf{x}) &= -\sum_j \sigma_j\left(\frac{\mathbf{f}^t(\mathbf{x})}{\tau_t}\right)\log\left(\sigma_j\left(\frac{\mathbf{f}^s(\mathbf{x})}{\tau_s}\right)\right), \\ \mathbf{f}^t(\mathbf{x}) &= \begin{cases} \mathbf{f}(\mathbf{x}), & \text{entropy minimization} \\ \text{sg}(\mathbf{f}(\mathbf{x})), & \text{pseudo-labeling.} \end{cases} \end{aligned} \tag{5}$$

where we introduced student and teacher temperature τ_s and τ_t as parameters in the softmax function and the stop gradient operation sg. To study the learning dynamics, we consider a linear student network $\mathbf{f}^s = \mathbf{w}^s \in \mathbb{R}^d$ and a linear teacher network $\mathbf{f}^t = \mathbf{w}^t \in \mathbb{R}^d$ which are trained on N data points $\{\mathbf{x}_i\}_{i=1}^N$ with a binary cross-entropy loss function $\mathcal{L}$ defined as

$$\begin{aligned} &\mathcal{L} = -\sum_{i=1}^N \ell(\mathbf{x}_i) = -\sum_{i=1}^N \left(\sigma_t(\mathbf{x}_i^\top \mathbf{w}^t)\log\sigma_s(\mathbf{x}_i^\top \mathbf{w}^s) + \sigma_t(-\mathbf{x}_i^\top \mathbf{w}^t)\log\sigma_s(-\mathbf{x}_i^\top \mathbf{w}^s)\right), \\ &\text{where } \sigma_t(z) = \frac{1}{1+e^{-z/\tau_t}} \text{ and } \sigma_s(z) = \frac{1}{1+e^{-z/\tau_s}}. \end{aligned} \tag{6}$$

With stop gradient, student and teacher evolve in time according to

$$\dot{\mathbf{w}}^s = -\nabla_{\mathbf{w}^s}\mathcal{L}\left(\mathbf{w}^s, \mathbf{w}^t\right), \quad \dot{\mathbf{w}}^t = \alpha(\mathbf{w}^s - \mathbf{w}^t), \tag{7}$$

where α is the learning rate of the teacher. Without stop gradient, student and teacher are set equal to each other (following the instant updates we found to perform empirically best), and they evolve as

$$\dot{\mathbf{w}} = -\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}), \text{ where } \mathbf{w}^s = \mathbf{w}^t = \mathbf{w}. \tag{8}$$

We restrict the theoretical analysis to the time evolution of the components of $\mathbf{w}^{s,t}$ in direction of two data points $\mathbf{x}_k$ and $\mathbf{x}_l$, $y_k^{s,t} \equiv \mathbf{x}_k^\top \mathbf{w}^{s,t}$ and $y_l^{s,t} \equiv \mathbf{x}_l^\top \mathbf{w}^{s,t}$. All other components $y_i^{s,t}$ with $i \neq k, l$ are neglected to reduce the dimensionality of the equation system. It turns out that the resulting model captures the neural network dynamics quite well despite the drastic simplification of taking only two data points into account (see Figure 2 for a comparison of the model vs. self-learning on the CIFAR-C dataset). We obtain the dynamics:

$$\begin{aligned} \text{with stop gradient: } \dot{y}_k^s &= -\mathbf{x}_k^\top \nabla_{\mathbf{w}^s}\left(\ell(\mathbf{x}_k) + \ell(\mathbf{x}_l)\right), \quad \dot{y}_l^s = -\mathbf{x}_l^\top \nabla_{\mathbf{w}^s}\left(\ell(\mathbf{x}_k) + \ell(\mathbf{x}_l)\right), \\ \dot{y}_k^t &= \alpha(y_k^t - y_k^s), \quad \dot{y}_l^t = \alpha(y_l^t - y_l^s), \\ \text{without stop gradient: } \dot{y}_k &= -\mathbf{x}_k^\top \nabla_{\mathbf{w}}\left(\ell(\mathbf{x}_k) + \ell(\mathbf{x}_l)\right), \quad \dot{y}_l = -\mathbf{x}_l^\top \nabla_{\mathbf{w}}\left(\ell(\mathbf{x}_k) + \ell(\mathbf{x}_l)\right). \end{aligned} \tag{9}$$

With this setup in place, we can derive

Proposition 7.1 (Collapse in the two-point model)**.** *The student and teacher networks $\mathbf{w}_s$ and $\mathbf{w}_t$ trained with stop gradient do not collapse to the trivial representation $\forall \mathbf{x} : \mathbf{x}^\top \mathbf{w}^s = 0, \mathbf{x}^\top \mathbf{w}^t = 0$ if $\tau_s > \tau_t$. The network $\mathbf{w}$ trained without stop gradient does not collapse if $\tau_s > \tau_t/2$. Proof. see § A.1.* □

We validate the proposition on a simulated two datapoint toy dataset, as well as on the CIFAR-C dataset (Figure 2). In general, the size and location of the region where collapse is observed in the simulated model also depends on the initial conditions, the learning rate and the optimization procedure. An in depth discussion, as well as additional simulations are given in Appendix A.

Entropy minimization with standard temperatures ($\tau_s = \tau_t = 1$) and hard pseudo-labeling ($\tau_t \to 0$) are hence stable. The two-point learning dynamics vanish for soft pseudo-labeling with $\tau_s = \tau_t$, suggesting that one would have to analyze a more complex model with more data points. While this does not directly imply that the learning is unstable at this point, we empirically observe that both entropy minimization and hard labeling outperform soft-labeling in practice.

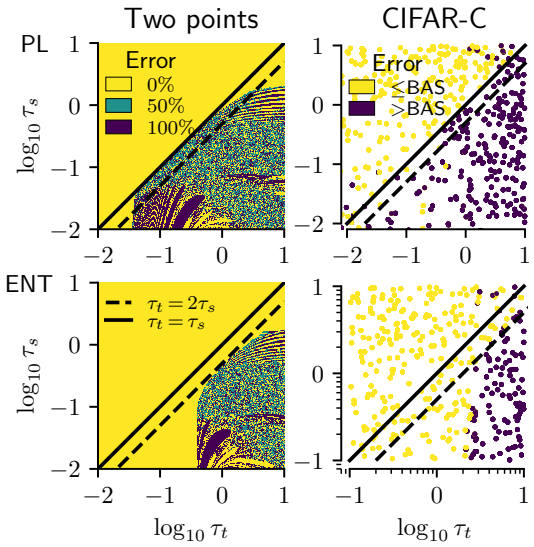

Figure 2: For the two point model, we show accuracy, and for the CIFAR10-C simulation, we show improvement (yellow) vs. degradation (purple) over the non-adapted baseline (BAS). An important convergence criterion for pseudo-labeling (top row) and entropy minimization (bottom row) is the ratio of student and teacher temperatures; it lies at $\tau_s = \tau_t$ for PL, and $2\tau_s = \tau_t$ for ENT. Despite the simplicity of the two-point model, the general convergence regions transfer to CIFAR10-C.

The finding aligns with empirical work: For instance, Caron et al. (2021) fixed τ_s and varied τ_t during training, and empirically found an upper bound for τ_t above which the training was no longer stable. It also aligns with our findings suggesting that hard-labeling tends to outperform soft-labeling approaches, and soft-labeling performs best when selecting lower teacher temperatures.

In practice, the result suggests that *student temperatures should always exceed the teacher temperatures for pseudo-labeling*, and *student temperatures should always exceed half the teacher temperature for entropy minimization*, which narrows the search space for hyperparameter optimization considerably.

8 Adapting models on a wider range of distribution shifts reveals limitations of robustification and adaptation methods

Robustness datasets on ImageNet-scale have so far been limited to a few selected domains (image corruptions in IN-C, image renditions in IN-R, difficult images for ResNet50 classifiers in IN-A). In order to test our approach on a wider range of complex distribution shifts, we re-purpose the dataset from the Visual Domain Adaptation Challenge 2019 (DomainNet; Saenko et al., 2019) as an additional robustness benchmark.

Table 11: Self-learning decreases the top1 error on IN-D domains with strong initial performance, but fails to improve performance on challenging domains.

domain	Real		Painting		Clipart		Sketch		Infograph		Quickdraw	
adapt model	w/o	w/	w/o	w/	w/o	w/	w/o	w/	w/o	w/	w/o	w/
EffNet-L2 Noisy Student	29.2	**27.9**	42.7	**40.9**	45.0	**37.9**	56.4	**51.5**	**77.9**	94.3	**98.4**	99.4
ResNet50 DAug+AM	39.2	36.5	58.7	53.4	68.4	57.0	75.2	61.3	88.1	83.2	98.2	99.1
ResNet50 vanilla	40.1	37.3	65.1	57.8	76.0	63.6	82.0	73.0	89.6	85.1	99.2	99.8

Creation of ImageNet-D The original DomainNet dataset comes with six image styles: "Clipart", "Real", "Infograph", "Painting", "Quickdraw" and "Sketch", and has 345 classes in total, out of which 164 overlap

with ImageNet. We map these 164 DomainNet classes to 463 ImageNet classes, e.g., for an image from the "bird" class in DomainNet, we accept all 39 bird classes in ImageNet as valid predictions. ImageNet also has ambiguous classes, e.g., it has separate classes for "cellular telephone" and "dial phone". For these cases, we accept both predictions as valid. In this sense, the mapping from DomainNet to ImageNet is a one-to-many mapping. We refer to the smaller version of DomainNet that is now compatible with ImageNet-trained models as ImageNet-D (IN-D). The benefit of IN-D over DomainNet is this re-mapping to ImageNet classes which allows robustness researchers to easily benchmark on this dataset, without the need of re-training a model (as is common in UDA). We show example images from IN-D in Table 11. The detailed evaluation protocol on IN-D, our label-mapping procedure from DomainNet to ImageNet along with justifications for our design choices and additional analysis are outlined in Appendix D.

The most similar robustness dataset to IN-D is IN-R which contains renditions of ImageNet classes, such as art, cartoons, deviantart, graffiti, embroidery, graphics and others. The benefit of IN-D over IN-R is that in IN-D, the images are separated according to the domain allowing for studying of systematic domain shifts, while in IN-R, the different domains are not distinguished. ImageNet-Sketch (Wang et al., 2019) is a dataset similar to the "Sketch" domain of IN-D.

More robust models perform better on IN-D. To test whether self-learning is helpful for more complex distribution shifts, we adapt a vanilla ResNet50, several robust IN-C models and the EfficientNet-L2 Noisy Student model on IN-D. We use the same hyperparameters we obtained on IN-C dev for all our IN-D experiments[2]. We show our main results in Table 11. Comparing the performance of the vanilla ResNet50 model to its robust DAug+AM variant, we find that the DAug+AM model performs better on all domains, with the most significant gains on the "Clipart", "Painting" and "Sketch" domains. We show detailed results for all domains and all tested models in Appendix D.3, along with results on IN-C and IN-R for comparison. We find that the best performing models on IN-D are also the strongest ones on IN-C and IN-R which indicates good generalization capabilities of the techniques combined for these models, given the large differences between the three considered datasets. The Spearman's rank correlation coefficient between IN-C and IN-D (averaged over all domains) is 0.54, and 0.73 between IN-R and IN-D. Thus, the errors on IN-R are strongly correlated to errors on IN-D which can be explained by the similarity of IN-D and IN-R. We show Spearman's rank correlation cofficients for the individual domains versus IN-C/IN-R in Fig. 9 in Appendix D.5, and find correlation values above 0.8 between IN-R and IN-D for all domains except for the "Real" domain where the coefficient is almost zero. Further, we find that even the best models perform 20 to 30 percentage points worse on IN-D compared to their performance on IN-C or IN-R, indicating that IN-D might be a more challenging benchmark.

All models struggle with some domains of IN-D. The EfficientNet-L2 Noisy Student model obtains the best results on most domains. However, we note that the overall error rates are surprisingly high compared to the model's strong performance on the other considered datasets (IN-A: 14.8% top-1 error, IN-R: 17.4% top-1 error, IN-C: 22.0% mCE). Even on the "Real" domain closest to clean ImageNet where the EfficientNet-L2 model has a top-1 error of 11.6%, the model only reaches a top-1 error of 29.2%. Self-learning decreases the top-1 error on all domains except for "Infograph" and "Quickdraw". We note that both domains have very high error rates from the beginning and thus hypothesize that the produced pseudo-labels are of low quality.

Error analysis on IN-D. We investigate the errors a ResNet50 model makes on IN-D by analyzing the most frequently predicted classes for different domains to reveal systematic errors indicative of the encountered distribution shifts. We find most errors interpretable: the classifier assigns the label "comic book" to images from the "Clipart" or "Painting" domains, "website" to images from the "Infograph" domain, and "envelope" to images from the "Sketch" domain. Thus, the classifier predicts the domain rather than the class. We find no systematic errors on the "Real" domain which is expected since this domain should be similar to ImageNet. Detailed results on the most frequently predicted classes for different domains can be found in Fig. 9, Appendix D.5.

IN-D should be used as an additional robustness benchmark. While the error rates on IN-C, -R and -A are at a well-acceptable level for our largest EfficientNet-L2 model after adaptation, IN-D performance is

[2]In regards to hyperparameter selection, we performed a control experiment where we selected hyperparameters with leave-one-out cross validation—this selection scheme actually performed worse than IN-C parameter selection (see Appendix D.2).

consistently worse for all models. We propose to move from isolated benchmark settings like IN-R (single domain) to benchmarks more common in domain adaptation (like DomainNet) and make IN-D publicly available as an easy to use dataset for this purpose.

9 Best practices and evaluation in test-time adaptation

Based on our results as well as our discussion on previous work, we arrive at several proposals on how test-time adaptation should be evaluated in future work to ensure scientific rigor:

1. **Cross-validation:** We propose using the hold-out set of IN-C for model selection of all relevant hyperparameters, and then using these hyperparameters for testing on different datasets.
2. **Comparison to simple baselines:** With proper hyperparameter tuning, very simple baselines can perform on par with sophisticated approaches. This insight is also discussed by Gulrajani & Lopez-Paz (2021) for the setting of domain generalization and by Rusak et al. (2020) for robustness to common corruptions.
3. **Using more robust models:** Test-time adaptation can further improve upon robust models, which were pre-trained with more data or with UDA, or using protocols to increase robustness. A test-time adaptation method will be much more relevant to practitioners if it can improve upon the most robust model they can find for their task.
4. **Important hyperparameters:** We identify several important hyperparameters which affect the final performance in a crucial way, and thus, should be tuned to ensure fair comparisons:
   - **Number of adaptation epochs and learning rate**: The final performance crucially depends on both of these parameters for all models and all methods that we have studied.
   - **Adaptation parameters** While adaptation of affine batch normalization parameters works best for adaptation of CNNs, full adaptation performs best for ViTs. Therefore, it is important to benchmark test-time adaptation for different model architectures and adaptation parameters.

10 Conclusion

We evaluated and analysed how self-learning, an essential component in many unsupervised domain adaptation and self-supervised pre-training techniques, can be applied for adaptation to both small and large-scale image recognition problems common in robustness research. We demonstrated new state-of-the-art adaptation results with the EfficientNet-L2 model on the benchmarks ImageNet-C, -R, and -A, and introduced a new benchmark dataset (ImageNet-D) which remains challenging even after adaptation. Our theoretical analysis shows the influence of the temperature parameter in the self-learning loss function on the training stability and provides guidelines how to choose a suitable value. Based on our extensive experiments, we formulate best practices for future research on test-time adaptation. Self-learning universally improves test-time performance under diverse, but systematic distribution shifts irrespective of the architecture or pre-training method, thus, we hope that our work encourages both researchers and practitioners to *use self-learning if their data distribution shifts.*

Reproducibility Statement We attempted to make our work as reproducible as possible: We mostly used pre-trained models which are publicly available and we denoted the URL addresses of all used checkpoints; for the checkpoints that were necessary to retrain, we report the Github directories with the source code and used an official or verified reference implementation when available. We report all used hyperparameters in the Appendix and will release our code upon acceptance of the paper.

Software and Data Code for reproducing results of this paper will be open sourced upon publication.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283, 2016. Cited on p. 41.

Azimi, F., Palacio, S., Raue, F., Hees, J., Bertinetto, L., and Dengel, A. Self-supervised test-time adaptation on video data. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3439–3448, 2022. Cited on p. 3.

Bartler, A., Bühler, A., Wiewel, F., Döbler, M., and Yang, B. Mt3: Meta test-time training for self-supervised test-time adaption. In *International Conference on Artificial Intelligence and Statistics*, pp. 3080–3090. PMLR, 2022. Cited on pp. 2, 3, 6, 7, 9, and 30

Berner, C., Brockman, G., Chan, B., Cheung, V., Debiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., et al. Dota 2 with large scale deep reinforcement learning. *ArXiv preprint*, abs/1912.06680, 2019. URL `https://arxiv.org/abs/1912.06680`. Cited on p. 1.

Berthelot, D., Roelofs, R., Sohn, K., Carlini, N., and Kurakin, A. Adamatch: A unified approach to semi-supervised learning and domain adaptation, 2021. Cited on p. 2.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html`. Cited on p. 1.

Cai, T., Gao, R., Lee, J. D., and Lei, Q. A theory of label propagation for subpopulation shift. *arXiv preprint arXiv:2102.11203*, 2021. Cited on p. 3.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. Emerging properties in self-supervised vision transformers. *ArXiv preprint*, abs/2104.14294, 2021. URL `https://arxiv.org/abs/2104.14294`. Cited on pp. 3, 5, 8, 11, and 25

Chen, T., Kornblith, S., Swersky, K., Norouzi, M., and Hinton, G. E. Big self-supervised models are strong semi-supervised learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/fcbc95ccdd551da181207c0c1400c655-Abstract.html`. Cited on p. 9.

Chollet, F. Xception: Deep learning with depthwise separable convolutions. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 1800–1807. IEEE Computer Society, 2017. doi: 10.1109/CVPR.2017.195. URL `https://doi.org/10.1109/CVPR.2017.195`. Cited on pp. 5 and 24

Coates, A., Ng, A., and Lee, H. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011. Cited on p. 5.

Croce, F., Andriushchenko, M., Sehwag, V., Debenedetti, E., Flammarion, N., Chiang, M., Mittal, P., and Hein, M. Robustbench: a standardized adversarial robustness benchmark. *ArXiv preprint*, abs/2010.09670, 2020. URL `https://arxiv.org/abs/2010.09670`. Cited on p. 25.

Deng, J., Dong, W., Socher, R., Li, L., Li, K., and Li, F. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pp. 248–255. IEEE Computer Society, 2009. doi: 10.1109/CVPR.2009.5206848. URL `https://doi.org/10.1109/CVPR.2009.5206848`. Cited on p. 1.

Deng, L. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. Cited on p. 5.

Dodge, S. F. and Karam, L. J. A study and comparison of human and deep learning recognition performance under visual distortions. In *International Conference on Computer Communications and Networks, ICCCN 2017*, 2017. Cited on p. 1.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of the 36th International Conference on Machine Learning*, 2021. Cited on p. 5.

Farahani, A., Voghoei, S., Rasheed, K., and Arabnia, H. R. A brief review of domain adaptation. *Advances in data science and information engineering*, pp. 877–894, 2021. Cited on p. 1.

French, G., Mackiewicz, M., and Fisher, M. H. Self-ensembling for visual domain adaptation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL `https://openreview.net/forum?id=rkpoTaxA-`. Cited on pp. 3, 39, and 41

Galstyan, A. and Cohen, P. R. Empirical comparison of hard and soft label propagation for relational classification. In *17th international conference on Inductive logic programming*, 2007. Cited on p. 4.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1): 2096–2030, 2016. Cited on pp. 1, 5, 6, and 7

Geirhos, R., Temme, C. R. M., Rauber, J., Schütt, H. H., Bethge, M., and Wichmann, F. A. Generalisation in humans and deep neural networks. In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 7549–7561, 2018. URL `https://proceedings.neurips.cc/paper/2018/hash/0937fb5864ed06ffb59ae5f9b5ed67a9-Abstract.html`. Cited on p. 1.

Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A., and Brendel, W. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL `https://openreview.net/forum?id=Bygh9j09KX`. Cited on pp. 1 and 33

Ghosh, A., Kumar, H., and Sastry, P. S. Robust loss functions under label noise for deep neural networks. In Singh, S. P. and Markovitch, S. (eds.), *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pp. 1919–1925. AAAI Press, 2017. URL `http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14759`. Cited on p. 4.

Grandvalet, Y. and Bengio, Y. Semi-supervised learning by entropy minimization. In *Advances in Neural Information Processing Systems 17 [Neural Information Processing Systems, NIPS 2004, December 13-18, 2004, Vancouver, British Columbia, Canada]*, pp. 529–536, 2004. URL `https://proceedings.neurips.cc/paper/2004/hash/96f2b50b5d3613adf9c27049b2a888c7-Abstract.html`. Cited on p. 4.

Gulrajani, I. and Lopez-Paz, D. In search of lost domain generalization. In *Proceedings of the 36th International Conference on Machine Learning*, 2021. Cited on pp. 3 and 13

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 770–778. IEEE Computer Society, 2016a. doi: 10.1109/CVPR.2016.90. URL `https://doi.org/10.1109/CVPR.2016.90`. Cited on p. 1.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 770–778. IEEE Computer Society, 2016b. doi: 10.1109/CVPR.2016.90. URL `https://doi.org/10.1109/CVPR.2016.90`. Cited on pp. 5 and 25

Hendrycks, D. and Dietterich, T. G. Benchmarking neural network robustness to common corruptions and perturbations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=HJz6tiCqYm. Cited on pp. 3, 4, 5, and 32

Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., and Song, D. Natural adversarial examples. *ArXiv preprint*, abs/1907.07174, 2019. URL https://arxiv.org/abs/1907.07174. Cited on p. 4.

Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. *ArXiv preprint*, abs/2006.16241, 2020a. URL https://arxiv.org/abs/2006.16241. Cited on pp. 1, 4, 5, 6, 24, 25, and 33

Hendrycks, D., Mu, N., Cubuk, E. D., Zoph, B., Gilmer, J., and Lakshminarayanan, B. Augmix: A simple data processing method to improve robustness and uncertainty. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020b. URL https://openreview.net/forum?id=S1gmrxHFvB. Cited on pp. 5, 6, 7, 25, and 33

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. In *NIPS Deep Learning Workshop*, 2014. Cited on pp. 5 and 24

Hoffman, J., Tzeng, E., Park, T., Zhu, J.-Y., Isola, P., Saenko, K., Efros, A., and Darrell, T. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pp. 1989–1998. Pmlr, 2018. Cited on p. 1.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, 2019. Cited on p. 8.

Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 2261–2269. IEEE Computer Society, 2017. doi: 10.1109/CVPR.2017.243. URL https://doi.org/10.1109/CVPR.2017.243. Cited on pp. 5, 25, and 38

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Bach, F. R. and Blei, D. M. (eds.), *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 448–456. JMLR.org, 2015. URL http://proceedings.mlr.press/v37/ioffe15.html. Cited on p. 5.

Kim, Y., Cho, D., Han, K., Panda, P., and Hong, S. Domain adaptation without source data. *IEEE Transactions on Artificial Intelligence*, 2021. Cited on p. 1.

Koh, P. W., Sagawa, S., Marklund, H., Xie, S. M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., Phillips, R. L., Gao, I., Lee, T., David, E., Stavness, I., Guo, W., Earnshaw, B. A., Haque, I. S., Beery, S., Leskovec, J., Kundaje, A., Pierson, E., Levine, S., Finn, C., and Liang, P. WILDS: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning (ICML)*, 2021. Cited on pp. 9, 38, and 41

Kolesnikov, A., Beyer, L., Zhai, X., Puigcerver, J., Yung, J., Gelly, S., and Houlsby, N. Big transfer (bit): General visual representation learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pp. 491–507. Springer, 2020. Cited on pp. 9 and 39

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009. Cited on p. 5.

Kumar, A., Ma, T., and Liang, P. Understanding self-training for gradual domain adaptation. In *International Conference on Machine Learning*, pp. 5468–5479. PMLR, 2020. Cited on p. 3.

Kundu, J. N., Venkat, N., Babu, R. V., et al. Universal source-free domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4544–4553, 2020. Cited on p. 1.

Lee, D.-H. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML Workshop : Challenges in Representation Learning (WREPL)*, 2013. Cited on p. 4.

Li, R., Jiao, Q., Cao, W., Wong, H.-S., and Wu, S. Model adaptation: Unsupervised domain adaptation without source data. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. Cited on p. 1.

Liang, J., Hu, D., and Feng, J. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International Conference on Machine Learning*, 2020. Cited on p. 1.

Mahajan, D., Girshick, R., Ramanathan, V., He, K., Paluri, M., Li, Y., Bharambe, A., and van der Maaten, L. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. Cited on pp. 1, 5, 7, 24, and 25

Marcel, S. and Rodriguez, Y. Torchvision the machine-vision package of torch. In *ACM International Conference on Multimedia*, 2010. Cited on pp. 25 and 41

Merkel, D. Docker: Lightweight linux containers for consistent development and deployment. *Linux J.*, 2014 (239), 2014. ISSN 1075-3583. Cited on p. 41.

Nado, Z., Padhy, S., Sculley, D., D'Amour, A., Lakshminarayanan, B., and Snoek, J. Evaluating prediction-time batch normalization for robustness under covariate shift. *ArXiv preprint*, abs/2006.10963, 2020. URL `https://arxiv.org/abs/2006.10963`. Cited on p. 2.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017. Cited on p. 41.

Prabhu, V., Khare, S., Kartik, D., and Hoffman, J. Sentry: Selective entropy optimization via committee consistency for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8558–8567, 2021. Cited on p. 3.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763. PMLR, 2021. Cited on p. 32.

Recht, B., Roelofs, R., Schmidt, L., and Shankar, V. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, pp. 5389–5400. PMLR, 2019. Cited on p. 32.

Rusak, E., Schott, L., Zimmermann, R., Bitterwolf, J., Bringmann, O., Bethge, M., and Brendel, W. Increasing the robustness of dnns against image corruptions by playing the game of noise. *ArXiv preprint*, abs/2001.06057, 2020. URL `https://arxiv.org/abs/2001.06057`. Cited on pp. 1, 13, and 33

Saenko, K., Peng, X., Usman, B., Saito, K., and Hu, P. *Visual Domain Adaptation Challenge (VisDA-2019)*, 2019. URL `http://ai.bu.edu/visda-2019/`. Cited on p. 11.

Schneider, S., Rusak, E., Eck, L., Bringmann, O., Brendel, W., and Bethge, M. Improving robustness against common corruptions by covariate shift adaptation. In *Advances in neural information processing systems*, 2020. Cited on pp. 2, 3, 5, 7, 9, 24, 28, and 30

Shu, J., Zhao, Q., Chen, K., Xu, Z., and Meng, D. Learning adaptive loss for robust learning with noisy labels. *ArXiv preprint*, abs/2002.06482, 2020. URL `https://arxiv.org/abs/2002.06482`. Cited on p. 4.

Shu, R., Bui, H. H., Narui, H., and Ermon, S. A DIRT-T approach to unsupervised domain adaptation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL `https://openreview.net/forum?id=H1q-TM-AW`. Cited on pp. 3 and 40

Sohn, K., Berthelot, D., Li, C.-L., Zhang, Z., Carlini, N., Cubuk, E. D., Kurakin, A., Zhang, H., and Raffel, C. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *NeurIPS*, 2020. Cited on p. 2.

Song, H., Kim, M., Park, D., and Lee, J.-G. Learning from noisy labels with deep neural networks: A survey. *ArXiv preprint*, abs/2007.08199, 2020. URL `https://arxiv.org/abs/2007.08199`. Cited on p. 4.

Sun, Y., Tzeng, E., Darrell, T., and Efros, A. A. Unsupervised domain adaptation through self-supervision. *ArXiv preprint*, abs/1909.11825, 2019. URL `https://arxiv.org/abs/1909.11825`. Cited on pp. 6 and 7

Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A., and Hardt, M. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pp. 9229–9248. PMLR, 2020. Cited on pp. 2, 3, 7, and 29

Tan, M. and Le, Q. V. Efficientnet: Rethinking model scaling for convolutional neural networks. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 6105–6114. PMLR, 2019. URL `http://proceedings.mlr.press/v97/tan19a.html`. Cited on pp. 5 and 25

Tange, O. Gnu parallel - the command-line power tool. *;login: The USENIX Magazine*, 36(1):42–47, 2011. URL `http://www.gnu.org/s/parallel`. Cited on p. 41.

Taori, R., Dave, A., Shankar, V., Carlini, N., Recht, B., and Schmidt, L. Measuring robustness to natural distribution shifts in image classification. *Advances in Neural Information Processing Systems*, 33:18583–18599, 2020. Cited on p. 32.

Tzeng, E., Hoffman, J., Saenko, K., and Darrell, T. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7167–7176, 2017. Cited on p. 1.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Jarrod Millman, K., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C., Polat, İ., Feng, Y., Moore, E. W., Vand erPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and Contributors, S. . . SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: https://doi.org/10.1038/s41592-019-0686-2. Cited on p. 41.

Wang, D., Shelhamer, E., Liu, S., Olshausen, B., and Darrell, T. Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations (ICLR)*, 2021. Cited on pp. 2, 3, 4, 7, 8, and 9

Wang, H., Ge, S., Lipton, Z., and Xing, E. P. Learning robust global representations by penalizing local predictive power. In *Advances in Neural Information Processing Systems*, pp. 10506–10518, 2019. Cited on p. 12.

Wei, C., Shen, K., Chen, Y., and Ma, T. Theoretical analysis of self-training with deep networks on unlabeled data. In *ICLR*, 2020. Cited on p. 3.

Wightman, R. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019. Cited on pp. 39 and 41

Wu, Y. and He, K. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. Cited on pp. 7 and 9

Xie, Q., Luong, M., Hovy, E. H., and Le, Q. V. Self-training with noisy student improves imagenet classification. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pp. 10684–10695. IEEE, 2020a. doi: 10.1109/CVPR42600.2020.01070. URL `https://doi.org/10.1109/CVPR42600.2020.01070`. Cited on pp. 1, 3, 5, 6, 24, 25, and 28

Xie, S., Girshick, R. B., Dollár, P., Tu, Z., and He, K. Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 5987–5995. IEEE Computer Society, 2017. doi: 10.1109/CVPR.2017.634. URL https://doi.org/10.1109/CVPR.2017.634. Cited on pp. 5 and 25

Xie, S. M., Kumar, A., Jones, R., Khani, F., Ma, T., and Liang, P. In-n-out: Pre-training and self-training using auxiliary information for out-of-distribution robustness. *arXiv preprint arXiv:2012.04550*, 2020b. Cited on p. 2.

Zagoruyko, S. and Komodakis, N. Wide residual networks. In Wilson, R. C., Hancock, E. R., and Smith, W. A. P. (eds.), *Proceedings of the British Machine Vision Conference 2016, BMVC 2016, York, UK, September 19-22, 2016*. BMVA Press, 2016. URL http://www.bmva.org/bmvc/2016/papers/paper087/index.html. Cited on pp. 5, 6, and 9

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Sy8gdB9xx. Cited on p. 4.

Zhang, M., Levine, S., and Finn, C. Memo: Test time robustness via adaptation and augmentation. *arXiv preprint arXiv:2110.09506*, 2021. Cited on p. 31.

Zhang, Z. and Sabuncu, M. R. Generalized cross entropy loss for training deep neural networks with noisy labels. In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 8792–8802, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/f2925f97bc13ad2852a7a551802feea0-Abstract.html. Cited on p. 4.

Zou, Y., Yu, Z., Liu, X., Kumar, B., and Wang, J. Confidence regularized self-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5982–5991, 2019. Cited on p. 3.