The Entropy Enigma: Success and Failure of Entropy Minimization

Ori Press [1] **Ravid Shwartz-Ziv** [2] **Yann LeCun** [2 3] **Matthias Bethge** [1]

Abstract

Entropy minimization (EM) is frequently used to increase the accuracy of classification models when they're faced with new data at test time. EM is a self-supervised learning method that optimizes classifiers to assign even higher probabilities to their top predicted classes. In this paper, we analyze why EM works when adapting a model for a few steps and why it eventually fails after adapting for many steps. We show that, at first, EM causes the model to embed test images close to training images, thereby increasing model accuracy. After many steps of optimization, EM makes the model embed test images far away from the embeddings of training images, which results in a degradation of accuracy. Building upon our insights, we present a method for solving a practical problem: estimating a model's accuracy on a given arbitrary dataset without having access to its labels. Our method estimates accuracy by looking at how the embeddings of input images change as the model is optimized to minimize entropy. Experiments on 23 challenging datasets show that our method sets the SoTA with a mean absolute error of 5.75%, an improvement of 29.62% over the previous SoTA on this task. Our code is available at: `https://github.com/oripress/EntropyEnigma`

1. Introduction

Practitioners commonly employ model adaptation strategies to enhance classifier performance on real-world data, which often differs significantly from training data. Unsupervised losses play a crucial role in adapting models to images corrupted by noise, such as snow or motion blur, or images from domains not seen in training, such as paintings or computer rendered images. Entropy minimization (EM) is a Test Time Adaptation (TTA) method that can improve the accuracy of a model on new datasets, without the need for additional labeled training data. EM adapts classifiers by iteratively increasing the probabilities assigned to the most likely classes while diminishing those of the others, and is an integral part of many recent TTA methods (Wang et al., 2020; Mummadi et al., 2021; Rusak et al., 2022b; Goyal et al., 2022; Niu et al., 2022; Cho et al., 2023; Niu et al., 2023; Press et al., 2023; Döbler et al., 2024; Marsden et al., 2024). In this paper, we analyze EM to understand how it works, when and why it fails, and how to use it to predict model accuracy.

The initial intuition behind using entropy minimization, given by Wang et al. (2020) was based on the observation that models tend to be more accurate on images for which they make predictions with higher confidence. The logical extension of this observation was to encourage models to bolster their confidence on such images. However, our analysis reveals this intuition to be only partly true. Remarkably, even when we construct datasets by excluding samples initially classified correctly — effectively creating datasets with a 100% classification error rate at the start — entropy minimization performance remains largely intact.

Our analysis uncovers that during entropy minimization, embeddings of images from the input dataset tend to form distinct clusters. The distances between samples within each cluster diminish, creating more defined groupings, while the centers of these clusters gradually move apart, a phenomenon akin to neural collapse (Papyan et al., 2020; Han et al., 2021; Ben-Shaul et al., 2023). At first, embeddings of the input images not only cluster, but also stay close to the embeddings of original training images. Only after numerous optimization steps do these embeddings begin to diverge, distancing themselves from the embeddings of the clean training data (Fig. 1). We show this divergence to be intricately tied to a reduction in the model's accuracy.

Drawing from our insights, we present a method designed to estimate the accuracy of a given model on any dataset, without labels. This task is notably difficult, because in some cases in-distribution accuracy is tied to out-of-distribution (OOD) accuracy (Miller et al., 2021), while in other cases it is not (Teney et al., 2022). Our approach, termed Weighted Flips (WF), works in conjunction with TTA methods as they

[1]University of Tübingen, Tübingen AI Center, Germany [2]New York University [3]Meta AI, FAIR. Correspondence to: <ori.press@bethgelab.org>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

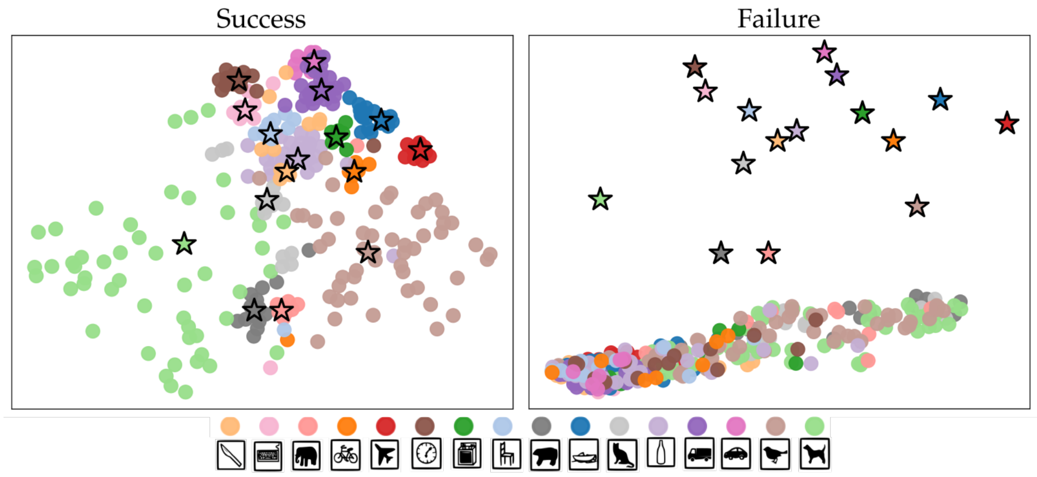

Figure 1. **Understanding the successes and failures of EM through clustering embedding dynamics.** After a few iterations of adaptation (left), EM improves the accuracy of pretrained classifiers by embedding the input test data near mean embeddings of classes from the training data, marked by stars. Eventually, after many iterations (right), EM fails, because it embeds input test data far from where training data is embedded. We show the t-SNE embeddings of 16-class-Imagenet (Geirhos et al., 2018), throughout adaptation to Gaussian Noise 3 (Hendrycks & Dietterich, 2019).

adapt to input data, with minimal added overhead. Using approximations of cluster consistency, WF estimates the accuracy of the network by measuring how the predictions of a fixed set of images change: the more they change, the lower the consistency of the clusters and the lower the predicted accuracy. We validate the efficacy of our method across an extensive array of 23 ImageNet-scale (Deng et al., 2009) datasets, encompassing diverse challenges, such as random adversarial noises, hard images, and datasets featuring OOD classes. WF surpasses the prior state-of-the-art methods by a substantial margin of 29.62%, setting a new benchmark in the accuracy estimation domain.

2. The Mystery of Entropy Minimization

EM has been validated as effective in semi-supervised settings, with pioneering work by (Grandvalet & Bengio, 2004) and subsequent advancements, such as Tent (Wang et al., 2020), which demonstrated EM's ability to enhance the accuracy of pre-trained classifiers on unlabeled ImageNet-scale (Deng et al., 2009) datasets. EM operates by iteratively optimizing the model to minimize the entropy of the output classification probabilities, denoted by $H(\hat{y}) = -\sum_c p(\hat{y}_c) \log p(\hat{y}_c)$, where $\hat{y}$ is the logits vector and $p(\hat{y}_c)$ is the probability assigned to class c. This approach inherently boosts the likelihood of the most probable classes while diminishing that of the others. Wang et al. (2020) observed a correlation between lower output entropy and accuracy, indicating that images with low entropy outputs are more likely to be classified correctly. Subsequent studies, including (Niu et al., 2022; Press et al., 2023; Marsden et al., 2024), have built on this foundation, assigning more weight to lower-entropy samples, and even ignoring high-entropy samples entirely.

To assess the influence of correctly classified images on EM's effectiveness, we tested the effects of omitting images that were initially correctly classified by the model. If such images are pivotal in EM's ability to enhance classifier performance, we expect a notable decline in the EM efficacy.

For this purpose, we utilized ImageNet-C (Hendrycks & Dietterich, 2019) Gaussian Noise level 3, dividing it into training and holdout sets. The training set was replicated seven times, systematically omitting images for which the ground truth label lay somewhere in the pre-trained model's top-k predictions, for ($k \in [1, 2, 3, 5, 10, 20, 50]$). Concretely, for $k = 1$, all accurately classified images were excluded, and for $k = 2$, images whose label ranked within the top two predictions were removed, and so forth. Each altered training set was used to adapt a Tented model. The model's accuracy was then evaluated on the holdout set, with evaluations every ten iterations, spanning a total of 1,000 iterations.

The experiment results (see Figure 2) are revealing, underscoring the robustness of EM. Notably, EM's effectiveness endures even when images initially classified correctly are excluded.

For instance, removing all initially correctly classified images before adaptation produces an increase in accuracy comparable to not removing any images, with gains of 10.50% and 12.38%, respectively. Even more remarkable is the persistence of this trend: with $k = 10$, the model still

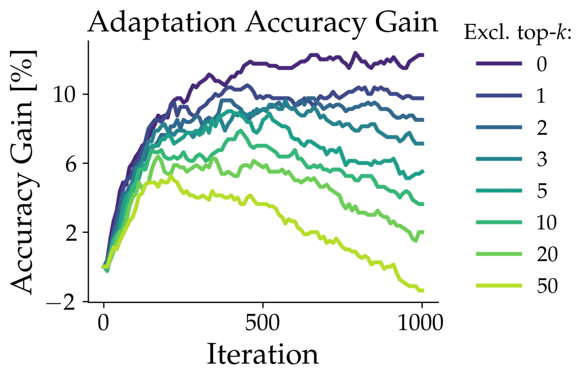

Figure 2. **EM remains effective even when initially correctly classified images are excluded**. Accuracy gain per iteration on a holdout set, as Tent adapts to its inputs. Surprisingly, the performance gain on the holdout set is high, even when we exclude top-k samples from the training set. When top-$k = 0$, no images are excluded.

registers a notable accuracy improvement of 7.88%. This observation is particularly striking given the nature of the excluded images – they are not just numerous, but also represent the highest quality, being those the network is most certain about. Specifically, images excluded at $k = 1$, which constitute 45% of the dataset, have an average entropy of 1.85, markedly lower than the original dataset's average entropy of 2.84.

Additionally, we also tested the effects of removing images according on their initial entropy level, and found similar results (see Appendix G). These findings intriguingly suggest that the model's accuracy and entropy on individual images may not be as pivotal to EM's success in enhancing classifier performance as previously thought. It reveals a nuanced dimension of EM's functionality and hints at the presence of deeper mechanisms, which we will investigate next.

3. Phases of Entropy Minimization: Clustering Dynamics and Embedding Alignment

We analyze the evolution of input data embeddings as EM progresses through its iterations. At first, EM causes the model to increase in accuracy, which we refer to as the first phase, followed by a decrease in accuracy, which we refer to as the second phase. The number of EM iterations needed for the model to reach its maximum accuracy (the end of the first phase, and the beginning of the second) is varied and depends on the input data.

In the first phase, these embeddings align closely with the embeddings of samples from the original training distribution. However, in the second phase, this alignment starts to deteriorate; the embeddings drift progressively further from the training distribution, disrupting the initial alignment, as conceptually depicted in Figure 3.

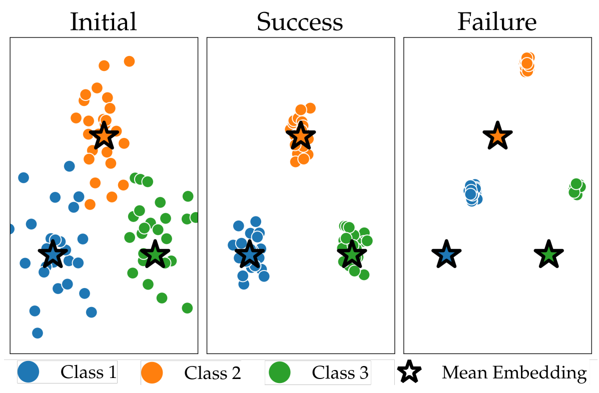

Figure 3. **The two-phase clustring paradigm explains EM behavior**. Intuitive visualization of EM's phases. In the first phase (success), input test data becomes more clustered, aligning closely with the mean embeddings of corresponding classes from the training data (the colored stars). In the second phase (failure), these clusters diverge from the mean embeddings.

To examine the clustering process across the two phases of the EM, we focus on two measures: (1) the quality of the clusters and (2) their alignment with the original training data distribution. For evaluating cluster quality, we ran k-means on the embeddings and computed the Silhouette score (Rousseeuw, 1987), a widely recognized metric for measuring cluster quality. The Silhouette score gauges how closely an embedding corresponds to its own cluster in contrast to neighboring clusters, with a high score indicating distinct and well-separated clusters.

To quantify the alignment between clusters and embeddings of the original training distribution, we looked at mean embeddings for the classes in the ImageNet validation set, alongside the centroids of clusters found by k-means. We use the Hungarian method (Kuhn, 1955) to find a matching between mean class embeddings and centroids, which minimizes the average distance between each assigned pair of (class embedding, centroid). Henceforth, we refer to this average of distances as "Shift distance".

As ImageNet contains many similar fine-grained classes, we restrict our analyses to the 16 classes outlined in (Geirhos et al., 2018), which represent approximately 20% of the total images. Consequently, we use $k = 16$ when we cluster the embeddings using k-means. This focused approach allowed for a detailed and controlled examination of clustering behaviors within the framework of EM.

We now examine changes in the Silhouette score and Shift distance as Tent adapts to the input data, over 50,000 iterations using a ResNet-50 (He et al., 2016). Figure 4 showcases the comparative Silhouette scores and Shift dis-

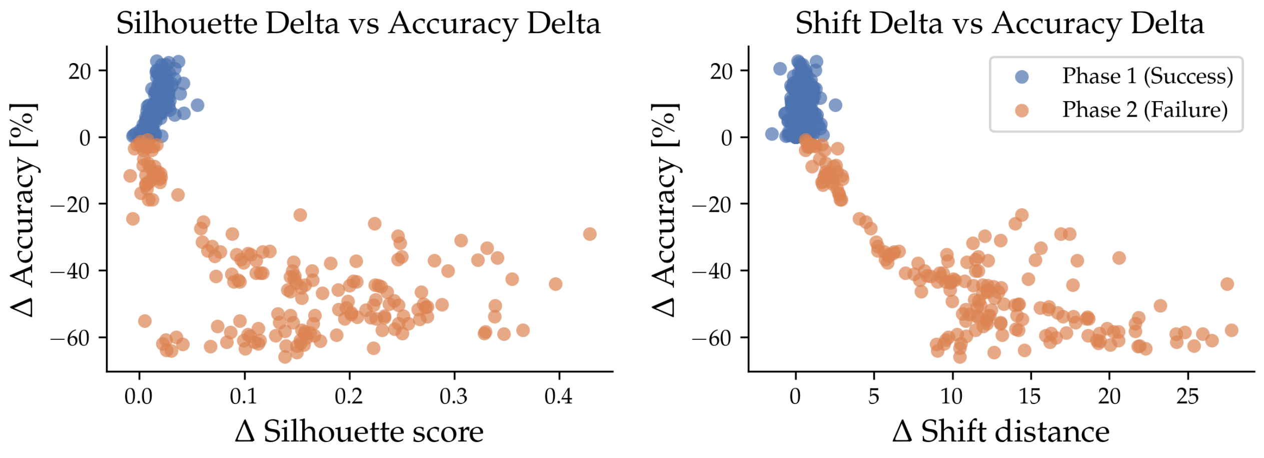

Figure 4. **Two-phase behavior during the EM adaption predicts accuracy.** Differences in Silhouette score, Shift distance, and accuracy for Tent adaptation. Each point corresponds to a test dataset; each dataset appears twice: once in blue, corresponding to phase 1 (success, Δ Acc ≥ 0), and once in orange, corresponding to phase 2 (failure, Δ Acc < 0). **Left:** In both phases, and across almost all datasets, the Silhouette score of embeddings increases, corresponding to a better-clustered embedding space. **Right:** In the first phase, input data embeddings are kept close to training image embeddings, while in the second phase, they drift away, exhibiting large Shift distance changes. The datasets used are IN-C, IN-$\overline{\text{C}}$ and IN-3DCC.

tances for both phases, incorporating findings from three diverse datasets: IN-C (Hendrycks & Dietterich, 2019), IN-$\overline{\text{C}}$ (Mintun et al., 2021), and IN-3DCC (Kar et al., 2022).

Our findings distill into two primary insights: **First**, a positive change in Silhouette score, indicative of enhanced clustering, is observed in both phases for more than 98% of cases. Notably, during the initial phase, a positive correlation exists between changes in Silhouette score and accuracy ($\rho = 0.70$, significant at $\alpha = 0.05$). **Second**, Shift distances minimally change (and sometimes diminish, signifying closer proximity to training data embeddings) in the first phase, they notably grow larger in the second phase. During this latter phase, a substantial negative correlation emerges between changes in Shift distance and accuracy ($\rho = -0.79$, significant at $\alpha = 0.05$).

Synthesizing these results reveals a nuanced picture: EM bolsters accuracy by clustering the embedded data into more concentrated clusters. This strategy remains efficacious as long as these embeddings align closely with the embeddings of the training data. However, as input data embeddings diverge from the training distribution, the classifier's accuracy diminishes. This intricate interplay offers a deeper understanding of EM's operation and its dependency on the spatial dynamics of data embeddings. We discuss the connection between EM and clustering in more detail in Appendix A.

4. Estimating Dataset Accuracy

Leveraging our understanding of EM, we tackle a critical challenge in TTA settings: estimating the accuracy of a classification model on a given dataset. Ideally, one might resort to the metrics used in this paper, namely Silhouette score or Shift distance, for this purpose. However, these metrics encounter practical hurdles: the Silhouette score depends on clustering, which varies across datasets due to differences in class distributions or the total number of classes, and calculating the Shift distance is impossible, as accessing the training data (in order to calculate mean embedding vectors per class) is forbidden in most TTA settings (Wang et al., 2020; Niu et al., 2022; Yuan et al., 2023).

4.1. Label Flipping

Due to the difficulties of measuring these scores in practice, we take a different approach. We look at the number of images for which the model's prediction changes somewhere between the initial and the final iteration of the EM ("label flips"). According to our hypothesis, the number of label flips is correlated with the pre-trained model's accuracy on the dataset. Our reasoning is as follows: there exists a tight correlation between accuracy and Silhouette score at iteration 0 — the higher the accuracy, the better clustered the input data, shown in Figure 5. Therefore, we do not expect EM, which works by clustering its inputs, to significantly change an already well-clustered set of embeddings. It follows that there will likely be only a few label flips. Conversely, given a dataset with a low accuracy, its

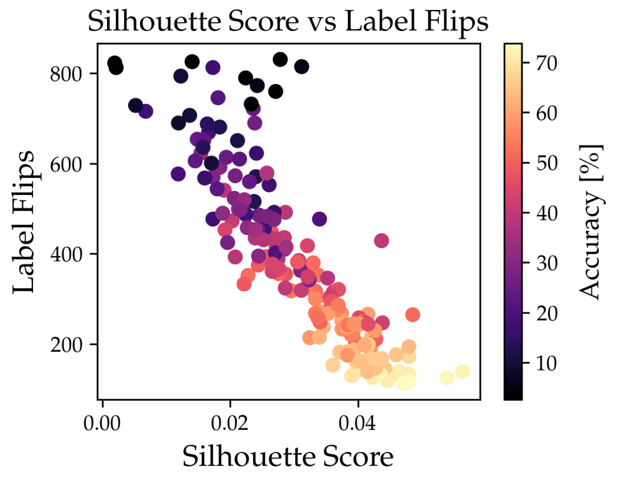

Figure 5. **Label flips are strongly correlated with Silhouette score**. Silhouette score at the initial iteration and the total number of label flips at the final iteration are correlated for datasets in IN-C, IN-$\overline{\text{C}}$, and IN-3DCC. Both metrics are correlated with accuracy, but measuring label flips is easier and more practical.

image embeddings will likely be badly clustered initially, which leads EM to change them significantly, resulting in many label flips.

We demonstrate the validity of this reasoning by adapting the state-of-the-art TTA method, Rdumb (Press et al., 2023), to IN-C, IN-$\overline{\text{C}}$, and IN-3DCC. Initially, we used the pre-trained model to classify 1,000 input images and then recorded the total number of label flips after adaptation. The model is adapted for 1,000 iterations because Rdumb resets itself every 1,000 iterations. We find a strong correlation between accuracy and label flips, seen in Figure 5.

4.2. Weighted Flips

We now describe the Weighted Flips (WF) method of converting the count of label flips into a dataset accuracy estimate. Instead of just counting the number of flips, we additionally consider the classifier's initial confidence in its predictions for each image; images initially classified with high confidence that later flip should contribute more significantly than those with lower initial confidence. We then compute the WF as:

$$WF = \sum_i 1_{\{flip\}}(i) \cdot c_i$$

where $1_{flip}(i)$ is 1 if image i's label flipped and 0 otherwise, and c_i is the confidence percentile of image i. Utilizing pairs of weighted flips and accuracy $((\text{WF}, \text{accuracy})_k)$ from IN-Validation and ImageNet-C holdout noises, we interpolate the weighted-flips-to-accuracy function, f (refer to Figure 6). To estimate the accuracy of a model on an unfamiliar dataset, we adapt the model to it using RDumb (for details, see Appendix J), measuring flips on the first 1,000 input images. After adaptation, we count and weigh the flips, estimating the model's accuracy as $f(\text{WF})$. Importantly, WF is versatile and can work with a range of TTA methods (see Appendix E), and f can be interpolated in a variety of different ways (see Appendix B). In Appendix D.1, D.2, we present ablation studies on the effects of varying end iterations and holdout set sizes on performance.

4.3. Experimental Setting

Accuracy estimation methods must yield robust estimates across diverse and challenging datasets to be considered reliable. In our evaluation, we probe the effectiveness of our proposed method using an extensive selection of popular ImageNet-scale classification datasets. This includes all classification datasets from the Shift-Happens benchmark[1]. Our chosen datasets encompass a wide spectrum, from various types of noise (IN-C, IN-$\overline{\text{C}}$, IN-3DCC, CCC) and domain shifts (IN-R, IN-V2, IN-D), to adversarial noises (Patch-IN, BG Challenge, IN-Obfuscations), and even images featuring classes not present in ImageNet (NINCO).

Several datasets provide multiple splits of a similar nature, the results of which we average, except for ImageNet-D (Rusak et al., 2022a), which encompasses a variety of distinct domains. The CCC dataset (Press et al., 2023) is particularly expansive, containing 27 splits with 7.5M images each; for practicality, we only include the initial 25k images from each split in our analysis. Altogether, our evaluation spans 326 individual dataset splits.

We briefly describe the other methods tested alongside ours:

- AC (Hendrycks & Gimpel, 2016): Computes the dataset-wide average confidence for the top-predicted class in each image.
- DoC (Guillory et al., 2021): Builds upon AC by assessing the variance in mean confidence between the validation and OOD sets, demonstrating consistent enhancements in performance.
- ATC (Garg et al., 2022): Estimates accuracy by determining the fraction of unlabeled data samples where the model's confidence exceeds a learned threshold.
- COT (Lu et al., 2023): Estimates accuracy by applying Optimal Transport to quantify the disparity between OOD and in-distribution model outputs.

[1] https://github.com/shift-happens-benchmark/icml-2022

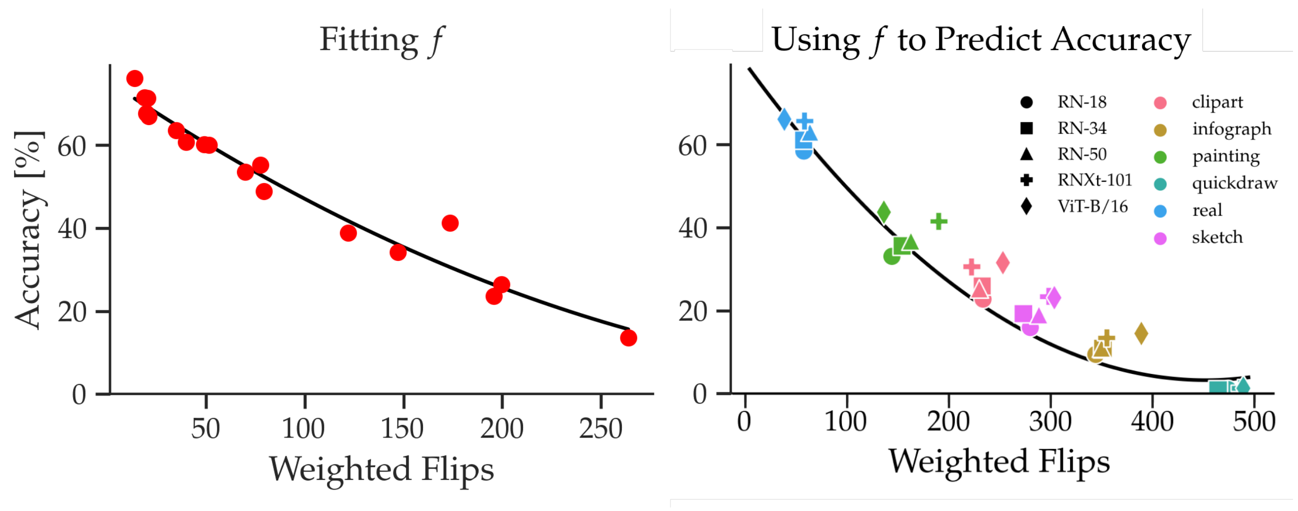

Figure 6. **Fitting and accuracy prediction using the WF method. Left:** Fitting f: using the noises in IN-C Holdout and ImageNet-Validation, we fit pairs of (weighted flips, accuracy), shown in red. The black curve shows the function resulting from interpolating the points, $f(x) = 0.00036x^2 - 0.32x + 75.66$. **Right:** With our weighted-flips-to-accuracy function f, we can estimate the accuracy of a model across the six splits from (Rusak et al., 2022a). We use the same f function and show that it works across different architectures, without refitting.

4.4. Results

Looking at Table 1 reveals that our WF method consistently stands out as the best estimator across a broad spectrum of ImageNet-scale datasets. WF sets a new benchmark by achieving an average estimation error of just 5.75%, significantly outperforming the nearest competitor, COT, reducing the relative error by 29.62%. This exemplary performance of WF is not limited to average cases; even in the most challenging scenarios of worst-case performance, WF maintains its superiority, cutting the error by 29.74% compared to COT. Furthermore, WF demonstrates remarkable consistency as an estimator. In 18 of the 23 datasets evaluated, it either leads the pack or comes a close second. This is in stark contrast to the performance of COT, which, despite being second-best, only achieves top-two rankings in 12 datasets. The persistent effectiveness of WF across diverse conditions underscores its reliability and superiority in accuracy estimation.

Practicality of WF: Beyond its top-tier performance, WF stands out for its practicality. It operates concurrently with the EM process, requiring only three parameters that define the weighted-flips-to-accuracy function, f. This process adds minimal computational overhead, requiring only 20 additional forward passes for every 1,000 Rdumb iteration steps. Lastly, WF is effective even when only a small number of samples are available, see Appendix C.

Versatility across Models and Architectures: To demonstrate the adaptability of the WF method, we tested it across various models and architectures, employing the **same** weighted-flips-to-accuracy function, f, used in our primary experiments (Table 1). Testing encompassed different ResNet variants, including models enhanced with noise augmentation techniques, such as ANT (Rusak et al., 2020), AugMix (Hendrycks et al., 2019), and DeepAugment (Hendrycks et al., 2021a). Additionally, we evaluated a ResNext-101 (Xie et al., 2017), ViTB-16 (Dosovitskiy et al., 2010), and MaxViT-T (Tu et al., 2022). The mean absolute errors between estimated and actual accuracies are reported in Table 2. Remarkably, 5 of the 8 models tested achieved a lower mean absolute error than the baseline model, RN-50, showing that f maintains its efficacy across different model architectures. When f is refitted on the architecture that WF is evaluated on, performance improves (see Appendix F).

Robustness to Dataset Choice: In Table 1, we derived the weighted-flips-to-accuracy function f using IN-C holdout and ImageNet validation noises. We further validated the robustness of the WF method by fitting f using a subset of the 23 datasets and then assessing its performance on the remaining datasets. As an added challenge, we excluded datasets used in the original configuration: IN-C Holdout and ImageNet-Validation. For each subset size, we repeated the fitting and evaluation process 50 times. The results, plotted in Figure 7, illustrate that the WF method consistently outperforms COT across almost all subset sizes, reinforcing its resilience and reliability across a broad spectrum of datasets.

Table 1. Mean Absolute Error between estimated accuracy, and true accuracy on a ResNet-50 model, for 4 estimation methods (AC, DoC, ATC, COT) (Hendrycks & Gimpel, 2016; Guillory et al., 2021; Garg et al., 2022; Lu et al., 2023), and ours. Our method (WF) is consistently either best or second best, with the best average and worst-case performance across many different OOD datasets. **Best** results are in bold; second best are underlined, $\{.\}$ indicates how many splits are in each dataset, when there are more than 1.

Datasets	AC	DoC	ATC	COT	WF (ours)
Noises					
IN-C $\{75\}$ (Hendrycks & Dietterich, 2019)	10.06	6.61	7.44	**2.23**	4.79
IN-$\overline{\text{C}}$ $\{50\}$ (Mintun et al., 2021)	19.48	15.96	12.16	**3.17**	7.35
IN-3DCC $\{60\}$ (Kar et al., 2022)	11.83	3.44	8.15	**3.02**	3.66
CCC $\{27\}$ (Press et al., 2023)	15.51	11.95	6.05	**2.04**	2.80
Domain Shifts					
Stylized (Geirhos et al., 2019)	31.63	28.08	7.36	12.18	**3.81**
IN-V2 $\{3\}$ (Recht et al., 2019)	5.58	2.41	**0.45**	2.68	4.70
IN-Sketch (Wang et al., 2019)	22.34	18.78	**0.15**	4.23	1.71
IN-R (Hendrycks et al., 2021a)	23.21	19.65	**0.37**	2.44	1.88
IN-D (Rusak et al., 2022a)					
→ Real	10.56	7.00	**1.35**	27.54	3.18
→ Painting	17.40	13.85	3.27	7.49	**2.12**
→ Clipart	21.27	17.72	**1.62**	4.52	3.37
→ Sketch	24.43	20.87	**0.61**	0.71	5.44
→ Infograph	54.12	50.57	36.26	**3.44**	3.63
→ Quickdraw	32.67	29.11	4.13	**1.60**	2.57
Cartoon & Drawing $\{2\}$ (Salvador & Oberman, 2022)	15.69	12.13	4.42	**1.62**	13.25
Adversarial Noises					
BG Challenge $\{8\}$ (Xiao et al., 2020)	10.54	7.37	**4.88**	19.68	6.92
IN-A (Hendrycks et al., 2021b)	45.12	41.57	**20.51**	30.38	21.61
IN-C Patch $\{75\}$ (Gu et al., 2022)	4.37	**0.16**	4.42	2.57	1.60
IN-Hard (Taesiri et al., 2023)	29.71	26.15	6.73	15.33	**3.64**
Patch-IN $\{10\}$ (Pintor et al., 2023)	8.06	**5.11**	**5.11**	10.13	8.87
IN-Obfuscations $\{3\}$ (Stimberg et al., 2023)	99.90	96.34	99.90	**0.12**	4.58
OOD/Other					
ObjectNet (Barbu et al., 2019)	34.59	31.03	9.43	10.40	**2.74**
NINCO (Bitterwolf et al., 2023)	50.29	46.74	26.97	20.28	**18.07**
Average	26.02	22.29	11.81	8.17	**5.75**
Worst Case	99.90	96.34	99.90	30.38	**21.61**
Average (Worst Case Excluded)	22.66	18.92	7.81	7.16	**5.03**

5. Related Work

To the best of our knowledge, the first time EM was shown to be useful for improving a classifier's accuracy was in (Grandvalet & Bengio, 2004). They showed how EM can be applied to a logistic regressor, and found it to be beneficial in cases where the data was corrupted by outliers. Following this, Lee et al. (2013) proposed pseudo labeling as a means of improving classification accuracy on MNIST. Interestingly, t-SNE is used to show that pseudo labeling works partly by encouraging the model's embeddings to be better clustered, and away from the decision boundaries of the model. Moreoever, it is stated that pseudo labeling is equivalent to entropy regularization (Grandvalet & Bengio, 2004). Although this might be true in the settings considered then, pseudo labeling was shown to be less effective (and thus not equivalent) on larger-scale datasets, by Tent. Unlike previous work, we demonstrate that EM clusters by measuring the Silhouette score of the clusters themselves, allowing us to empirically evaluate ImageNet scale datasets. Additionally, we show what happens when EM fails, which is not discussed in prior work, with the exception of (Oliver et al., 2018), which shows how EM fails to adapt to a toy "two moons" dataset, because the model increases the magnitude of its output logits. This isn't the case in most TTA settings, as the final layer of the model isn't trained.

Minimizing entropy at test time was popularized by Tent

Table 2. Mean Absolute Error between estimated accuracy and true accuracy, across different architectures. Using the same weighted-flips to-accuracy function, f, works across different architectures and models, without need for finetuning. For each model and dataset, the task is to estimate the accuracy of that model on the dataset. **Best** results are in bold; second best are underlined. AugMix: ◇ ANT: ‡ DeepAugment: ♠ (Hendrycks et al., 2019; Rusak et al., 2020; Hendrycks et al., 2021a)

Datasets	RN-50	RN-18	RN-34	RN-50 ‡	RN-50 ◇	RN-50 ◇♠	RNXt-101	RNXt-101 ♠	ViT-B/16	MaxViT-T
IN-C	4.79	7.21	6.04	5.39	5.02	4.81	5.35	**4.12**	8.34	6.73
IN-$\overline{C}$	7.35	7.90	6.77	6.84	6.60	6.48	**5.60**	5.63	6.59	4.97
IN-3DCC	3.66	3.58	3.89	3.20	3.07	**2.98**	7.23	4.37	7.19	6.79
IN-V2	4.70	4.11	**3.37**	3.67	5.06	5.00	6.47	5.54	4.44	6.08
IN-D ↴										
Real	3.18	2.83	**0.38**	2.72	6.59	3.30	4.24	0.61	1.02	3.40
Painting	2.12	5.36	0.78	**0.59**	7.62	2.51	12.02	1.12	3.02	0.60
Clipart	3.37	1.59	4.42	0.32	6.19	2.19	7.24	**0.53**	12.82	4.52
Sketch	5.44	**1.53**	3.93	6.18	9.73	3.60	10.75	1.89	11.04	10.88
Infograph	3.63	1.76	3.67	3.74	6.78	**0.28**	6.37	2.34	9.27	9.13
Quickdraw	2.57	2.34	2.24	2.20	2.53	1.27	2.27	1.21	2.36	2.31
Average	4.08	3.82	3.55	3.49	5.92	3.10	6.76	**2.74**	6.61	5.54

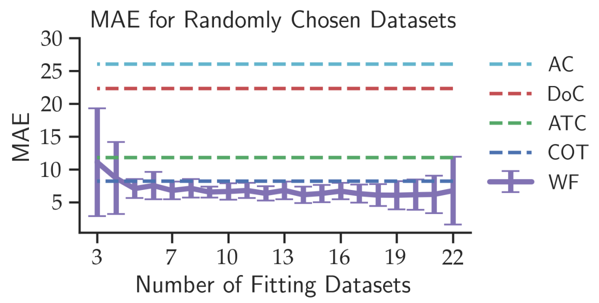

Figure 7. **WF outperforms other methods across almost all subset sizes**. Mean Absolute Error of WF when using a weighted-flips-to-accuracy function f to fit on random subsets of the 23 datasets in Table 1. For each point on the x-axis, we sample 50 fitting datasets for WF, and plot the average and the standard deviation of the MAE. For the other methods, we plot average MAE across all datasets

.

(Wang et al., 2020), which demonstrated the effectiveness of EM on large-scale datasets, such as ImageNet-C.

Entropy minimization is ideal for domain adaptation: it can be used on a trained model, without retraining, and doesn't require balancing a proxy loss with a classification loss, as in (Gidaris et al., 2018; Sun et al., 2020; Gandelsman et al., 2022).

Though many prior works use losses that are based on entropy (Wang et al., 2020; Rusak et al., 2022b; Goyal et al., 2022; Mummadi et al., 2021; Wang et al., 2022; Niu et al., 2022; Cho et al., 2023; Press et al., 2023; Niu et al., 2023; Döbler et al., 2024; Marsden et al., 2024), little is known as to *why* it works. Additionally, entropy minimization, when used in TTA settings, is effective for only a limited number of iterations, before the classifier degrades to chance accuracy, shown in (Press et al., 2023). Interestingly, this degradation of accuracy, named "collapse", differs from classical definitions of catastrophic forgetting in continual learning (De Lange et al., 2021), in that the task itself does not change.

A plethora of methods have been used for adapting a trained classifier to out-of-domain data: from using an auxiliary loss to help learn the test domain (Sun et al., 2019; 2020; Gandelsman et al., 2022) through simply re-estimating the mean and variance statistics (Schneider et al., 2020; Nado et al., 2020) to using image augmentations (Wang et al., 2022; Song et al., 2023; Chakrabarty et al., 2023). However, for their simplicity and success, entropy minimization-based methods are still the most widely used and successful in settings most relevant to this work.

Works that follow Tent improve EM by modifying the loss to be more robust to label noise (Rusak et al., 2022b) or smoother (Mummadi et al., 2021), or by adjusting the temperature of the output distribution (Goyal et al., 2022). While testing on long sequences of images, both (Wang et al., 2022) and (Niu et al., 2022) show that Tent degrades in accuracy, the more iterations it does. (Press et al., 2023) show that this is in fact true for all TTA methods apart from EATA (Niu et al., 2022), which uses an L2 regularizer to constrain the adapting model's weights to be close to those of the pretrained model. (Niu et al., 2023) study the effects of batch size, label shifts and other factors on adaptation; they propose a method to stabilize adaptation. Similarly, (Döbler et al., 2024) also test entropy minimization-based methods in real-world conditions, and propose a new method based on a diversity and a weighted entropy loss. Entropy has also been used in semi-supervised settings: (Sohn et al., 2020) propose augmentation and an entropy loss to train a classifier when only a few labels are available.

Analyzing which labels flip during training has been studied in (Toneva et al., 2018), which explored which samples are forgotten during training. Another work, (Deng et al., 2022) looked at how to reduce the amount of times a label flips during training. The agreement/disagreement between different models on ID data was shown to be linearly correlated to OOD accuracy and has been recently used to estimate accuracy in (Miller et al., 2021; Jiang et al., 2021; Baek et al., 2022; Kim et al., 2023). These works are beyond the scope of this work, as they require access to multiple models and ID data, which is disallowed in most TTA settings (Wang et al., 2020; Niu et al., 2022; Yuan et al., 2023).

6. Conclusion

While EM is a cornerstone in many TTA methods, the mechanics of its success have remained enigmatic. This study sheds light on the transformative journey of input data embeddings under the EM adaption. It reveals a biphasic clustering process, where alignment with the training data's embedding clusters bolsters accuracy, followed by a subsequent phase where excessive divergence diminishes it.

Our work goes beyond deciphering the mystery behind entropy minimization; it also utilizes this knowledge to significantly refine the precision of model accuracy predictions in TTA contexts. This dual achievement underscores the potential of deep analytical approaches in enhancing the efficacy and applicability of machine learning models.

Acknowledgements

We thank Ofir Press for helpful insights and feedback.

The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Ori Press. Matthias Bethge is a member of the Machine Learning Cluster of Excellence, funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC number 2064/1 – Project number 390727645 and acknowledges support by the German Research Foundation (DFG): SFB 1233, Robust Vision: Inference Principles and Neural Mechanisms, TP 4, Project No: 276693517. This work was supported by the Tübingen AI Center. The authors declare no conflicts of interests.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel needs to be highlighted here specifically.

References

Amini, M.-R. and Gallinari, P. Semi-supervised logistic regression. In *ECAI*, volume 2, pp. 11, 2002.

Baek, C., Jiang, Y., Raghunathan, A., and Kolter, J. Z. Agreement-on-the-line: Predicting the performance of neural networks under distribution shift. *Advances in Neural Information Processing Systems*, 35:19274–19289, 2022.

Barbu, A., Mayo, D., Alverio, J., Luo, W., Wang, C., Gutfreund, D., Tenenbaum, J., and Katz, B. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. *Advances in neural information processing systems*, 32, 2019.

Ben-Shaul, I., Shwartz-Ziv, R., Galanti, T., Dekel, S., and LeCun, Y. Reverse engineering self-supervised learning. *arXiv preprint arXiv:2305.15614*, 2023.

Bitterwolf, J., Müller, M., and Hein, M. In or out? fixing imagenet out-of-distribution detection evaluation. *arXiv preprint arXiv:2306.00826*, 2023.

Chakrabarty, G., Sreenivas, M., and Biswas, S. Santa: Source anchoring network and target alignment for continual test time adaptation. *Transactions on Machine Learning Research*, 2023.

Cho, Y., Kim, Y., and Lee, D. Beyond entropy: Style transfer guided single image continual test-time adaptation. *arXiv preprint arXiv:2311.18270*, 2023.

De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., and Tuytelaars, T. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society: series B (methodological)*, 39(1):1–22, 1977.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Deng, X., Xiao, Y., Long, B., and Zhang, Z. Reducing flipping errors in deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 6506–6514, 2022.

Döbler, M., Marencke, F., Marsden, R. A., and Yang, B. Diversity-aware buffer for coping with temporally correlated data streams in online test-time adaptation. *arXiv preprint arXiv:2401.00989*, 2024.

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. arxiv 2020. *arXiv preprint arXiv:2010.11929*, 2010.

Gandelsman, Y., Sun, Y., Chen, X., and Efros, A. Test-time training with masked autoencoders. *Advances in Neural Information Processing Systems*, 35:29374–29385, 2022.

Garg, S., Balakrishnan, S., Lipton, Z. C., Neyshabur, B., and Sedghi, H. Leveraging unlabeled data to predict out-of-distribution performance. *arXiv preprint arXiv:2201.04234*, 2022.

Geirhos, R., Temme, C. R., Rauber, J., Schütt, H. H., Bethge, M., and Wichmann, F. A. Generalisation in humans and deep neural networks. *Advances in neural information processing systems*, 31, 2018.

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 *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bygh9j09KX.

Gidaris, S., Singh, P., and Komodakis, N. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

Goyal, S., Sun, M., Raghunathan, A., and Kolter, J. Z. Test time adaptation via conjugate pseudo-labels. *Advances in Neural Information Processing Systems*, 35:6204–6218, 2022.

Grandvalet, Y. and Bengio, Y. Semi-supervised learning by entropy minimization. *Advances in neural information processing systems*, 17, 2004.

Gu, J., Tresp, V., and Qin, Y. Evaluating model robustness to patch perturbations. In *ICML 2022 Shift Happens Workshop*, 2022.

Guillory, D., Shankar, V., Ebrahimi, S., Darrell, T., and Schmidt, L. Predicting with confidence on unseen distributions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1134–1144, 2021.

Han, X., Papyan, V., and Donoho, D. L. Neural collapse under mse loss: Proximity to and dynamics on the central path. *arXiv preprint arXiv:2106.02073*, 2021.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Hendrycks, D. and Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

Hendrycks, D. and Gimpel, K. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.

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. *arXiv preprint arXiv:1912.02781*, 2019.

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. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8340–8349, 2021a.

Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., and Song, D. Natural adversarial examples. *CVPR*, 2021b.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. pmlr, 2015.

Jiang, Y., Nagarajan, V., Baek, C., and Kolter, J. Z. Assessing generalization of sgd via disagreement. *arXiv preprint arXiv:2106.13799*, 2021.

Kar, O. F., Yeo, T., Atanov, A., and Zamir, A. 3d common corruptions and data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18963–18974, 2022.

Kim, E., Sun, M., Raghunathan, A., and Kolter, Z. Reliable test-time adaptation via agreement-on-the-line. *arXiv preprint arXiv:2310.04941*, 2023.

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

Kuhn, H. W. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

Kullback, S. and Leibler, R. A. On information and sufficiency. *The annals of mathematical statistics*, 22(1): 79–86, 1951.

Lee, D.-H. et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, pp. 896. Atlanta, 2013.

Lu, Y., Wang, Z., Zhai, R., Kolouri, S., Campbell, J., and Sycara, K. Predicting out-of-distribution error with confidence optimal transport. *arXiv preprint arXiv:2302.05018*, 2023.

Marsden, R. A., Döbler, M., and Yang, B. Universal test-time adaptation through weight ensembling, diversity weighting, and prior correction. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2555–2565, 2024.

Miller, J. P., Taori, R., Raghunathan, A., Sagawa, S., Koh, P. W., Shankar, V., Liang, P., Carmon, Y., and Schmidt, L. Accuracy on the line: on the strong correlation between out-of-distribution and in-distribution generalization. In *International Conference on Machine Learning*, pp. 7721–7735. PMLR, 2021.

Mintun, E., Kirillov, A., and Xie, S. On interaction between augmentations and corruptions in natural corruption robustness. *Advances in Neural Information Processing Systems*, 34:3571–3583, 2021.

Mummadi, C. K., Hutmacher, R., Rambach, K., Levinkov, E., Brox, T., and Metzen, J. H. Test-time adaptation to distribution shift by confidence maximization and input transformation. *arXiv preprint arXiv:2106.14999*, 2021.

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 arXiv:2006.10963*, 2020.

Niu, S., Wu, J., Zhang, Y., Chen, Y., Zheng, S., Zhao, P., and Tan, M. Efficient test-time model adaptation without forgetting. In *International conference on machine learning*, pp. 16888–16905. PMLR, 2022.

Niu, S., Wu, J., Zhang, Y., Wen, Z., Chen, Y., Zhao, P., and Tan, M. Towards stable test-time adaptation in dynamic wild world. *arXiv preprint arXiv:2302.12400*, 2023.

Oliver, A., Odena, A., Raffel, C., Cubuk, E., and Goodfellow, I. Realistic evaluation of semi-supervised learning algortihms. In *International conference on learning representations*, pp. 1–15, 2018.

Papyan, V., Han, X., and Donoho, D. L. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40):24652–24663, 2020.

Pintor, M., Angioni, D., Sotgiu, A., Demetrio, L., Demontis, A., Biggio, B., and Roli, F. Imagenet-patch: A dataset for benchmarking machine learning robustness against adversarial patches. *Pattern Recognition*, 134:109064, 2023.

Poland, W. B. and Shachter, R. D. Mixtures of gaussians and minimum relative entropy techniques for modeling continuous uncertainties. In *Uncertainty in Artificial Intelligence*, pp. 183–190. Elsevier, 1993.

Press, O., Schneider, S., Kümmerer, M., and Bethge, M. Rdumb: A simple approach that questions our progress in continual test-time adaptation. *Advances in Neural Information Processing Systems*, 36, 2023.

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.

Rousseeuw, P. J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

Rusak, E., Schott, L., Zimmermann, R. S., Bitterwolf, J., Bringmann, O., Bethge, M., and Brendel, W. A simple way to make neural networks robust against diverse image corruptions. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pp. 53–69. Springer, 2020.

Rusak, E., Schneider, S., Gehler, P. V., Bringmann, O., Brendel, W., and Bethge, M. Imagenet-d: A new challenging robustness dataset inspired by domain adaptation. In *ICML 2022 Shift Happens Workshop*, 2022a.

Rusak, E., Schneider, S., Pachitariu, G., Eck, L., Gehler, P. V., Bringmann, O., Brendel, W., and Bethge, M. If your data distribution shifts, use self-learning. *Transactions on Machine Learning Research*, 2022b.

Salvador, T. and Oberman, A. M. Imagenet-cartoon and imagenet-drawing: two domain shift datasets for imagenet. In *ICML 2022 Shift Happens Workshop*, 2022.

Schneider, S., Rusak, E., Eck, L., Bringmann, O., Brendel, W., and Bethge, M. Improving robustness against common corruptions by covariate shift adaptation. *Advances in neural information processing systems*, 33: 11539–11551, 2020.

Sohn, K., Berthelot, D., Carlini, N., Zhang, Z., Zhang, H., Raffel, C. A., Cubuk, E. D., Kurakin, A., and Li, C.-L. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in neural information processing systems*, 33:596–608, 2020.

Song, J., Lee, J., Kweon, I. S., and Choi, S. Ecotta: Memory-efficient continual test-time adaptation via self-distilled regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11920–11929, 2023.

Stimberg, F., Chakrabarti, A., Lu, C.-T., Hazimeh, H., Stretcu, O., Qiao, W., Liu, Y., Kaya, M., Rashtchian, C., Fuxman, A., et al. Benchmarking robustness to adversarial image obfuscations. *arXiv preprint arXiv:2301.12993*, 2023.

Sun, Y., Tzeng, E., Darrell, T., and Efros, A. A. Unsupervised domain adaptation through self-supervision. *arXiv preprint arXiv:1909.11825*, 2019.

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.

Taesiri, M. R., Nguyen, G., Habchi, S., Bezemer, C.-P., and Nguyen, A. Zoom is what you need: An empirical study of the power of zoom and spatial biases in image classification. *arXiv preprint arXiv:2304.05538*, 2023.

Teney, D., Lin, Y., Oh, S. J., and Abbasnejad, E. Id and ood performance are sometimes inversely correlated on real-world datasets. *arXiv preprint arXiv:2209.00613*, 2022.

Toneva, M., Sordoni, A., Combes, R. T. d., Trischler, A., Bengio, Y., and Gordon, G. J. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*, 2018.

Tu, Z., Talebi, H., Zhang, H., Yang, F., Milanfar, P., Bovik, A., and Li, Y. Maxvit: Multi-axis vision transformer. In *European conference on computer vision*, pp. 459–479. Springer, 2022.

Wang, D., Shelhamer, E., Liu, S., Olshausen, B., and Darrell, T. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*, 2020.

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.

Wang, Q., Fink, O., Van Gool, L., and Dai, D. Continual test-time domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7201–7211, 2022.

Xiao, K., Engstrom, L., Ilyas, A., and Madry, A. Noise or signal: The role of image backgrounds in object recognition. *ArXiv preprint arXiv:2006.09994*, 2020.

Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500, 2017.

Yuan, L., Xie, B., and Li, S. Robust test-time adaptation in dynamic scenarios. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15922–15932, 2023.

A. The Relationship between Entropy Minimization and Clustering

In this section, we explain the connection between entropy minimization and the Expectation-Maximization algorithm (Dempster et al., 1977) with a mixture of Gaussians and show how the iterative entropy minimization objective leads to a clustering process similar to the Expectation-Maximization algorithm.

In the Expectation-Maximization algorithm for clustering, the latent variables represent the cluster assignments, and the algorithm alternates between estimating the cluster assignments (E-step) and updating the cluster parameters (M-step). The convergence of the EM algorithm in this setting has been formally established (Dempster et al., 1977).

Poland & Shachter (1993) showed that for a random variable X with a given distribution and the mixture of random variables Y that derive from it, the objective of minimizing the "relative entropy" between X and Y generalizes the objective of the Expectation Maximization algorithm: to maximize the likelihood of the observations x drawn from Y's distribution.

In our setting, the iterative entropy minimization process corresponds to the Expectation Maximization algorithm, as iterative entropy minimization can also be seen as a form of "self-training" with minimization of the relative entropy (the DKL (Kullback & Leibler, 1951)) of the pseudo-labels (the model's predictions) (Grandvalet & Bengio, 2004). The forward pass of our training process serves two purposes: (1) it sets the "observations", which are the model's predictions, and (2) it acts as the E-step of the algorithm, estimating the distribution given the model parameters (the clustering assignment). The backpropagation step, which updates the model parameters (the cluster parameters), serves as the M-step and maximizes the likelihood under the current pseudo-label estimates (Amini & Gallinari, 2002). It is important to note that in our setting, the entropy minimization procedure involves changing both X and Y in each iteration, which may be different from the original Expectation Maximization algorithm.

Using these insights, we can provide a better explanation for the two-phase clustering phenomenon observed in our experiments. In the initial "success" phase, where the change in the embeddings is relatively small during the process, the entropy minimization effectively performs Expectation Maximization unsupervised clustering in the model's embedding space, guided by the smart initialization provided by the pre-trained model. The E-step estimates the pseudo-labels based on the current embedding structure, while the M-step updates the model to refine the embeddings and increase intra-cluster similarity. This process leads to the formation of well-separated clusters, as reflected by the increasing Silhouette score.

However, as the Expectation Maximization algorithm continues over many iterations in the "failure" phase or if there is bad initialization, it starts to overfit the model to the specific characteristics of the "new" test data. Unlike the regular Expectation Maximization algorithm, in our case, the data distribution (the observations) changes over time, which leads to a drift in the embeddings away from the initialized representations learned from the training data. This overfitting effect, which might even converge to a global minimum, is captured by the increasing Shift distance between the test data embeddings and the training class embeddings.

To support this explanation, we also provide visualizations of the prediction space to illustrate the clustering process and the eventual drift from the training embeddings. We used a mixture of Gaussians, and trained a GMM with the Expectation Maximization algorithm using maximum likelihood, where the means are initialized based on random samples. The covariance is used as the identity matrix, with the input samples being trainable and optimizing their location. In Figure 8, each dot represents a sample colored by its original class, where the Xs are the centroids at each iteration. As we can see, with the "smart initialization" of the cluster centers, the points converge to the "right" clusters based on the original cluster centers. However, when we start the cluster centers with some shift, namely there is "wrong" initialization, the clusters start with good clustering but then converge to wrong solutions where they mix points with different classes.

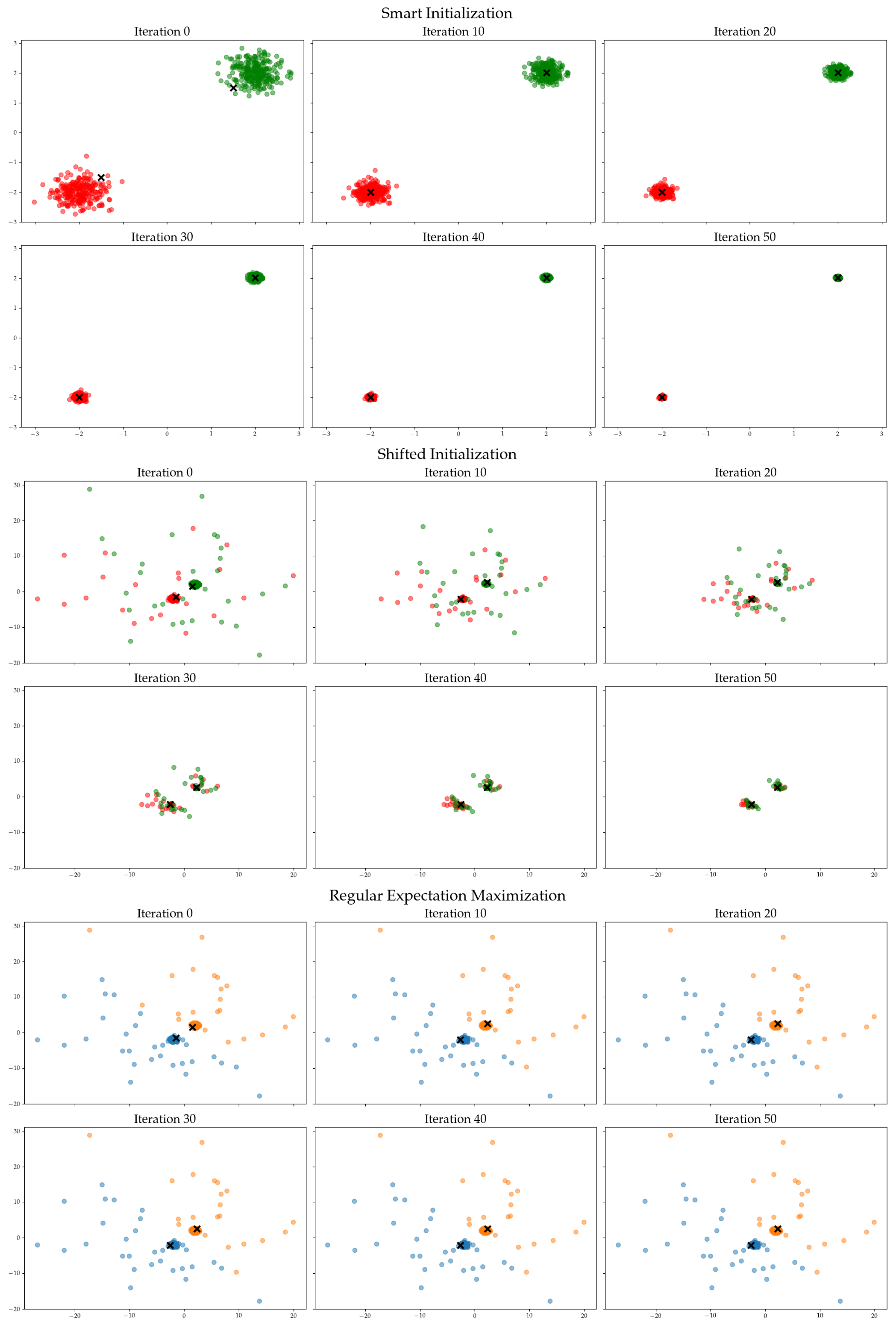

Figure 8. **Top:** With the “smart initialization” of the cluster centers, the points converge to the “right” clusters based on the original cluster centers. **Middle:** When we start the cluster centers with some shift, namely there is “wrong” initialization, the clusters start with good clustering but then converge to wrong solutions where they mix points with different classes. **Bottom:** For reference, we also show the regular Expectation Maximization algorithm on the shifted dataset. The X’s represent cluster centroids at each iteration.

B. Different Parameterizations of f

In this section, we test the different ways of parameterizing the weighted-flips-to-accuracy function, f. Firstly, we look at the effects of not weighing each flip, and then we look at linear and cubic interpolations between flips and accuracy (as opposed to quadratic interpolation, used in the rest of the paper). Our results in Table 3 show that the optimal f is a weighted and interpolated quadratically, with the other variations not far behind. Importantly, all variations of f perform better than the second best performing method, COT (Lu et al., 2023).

Table 3. Mean Absolute Error between estimated accuracy, and true accuracy on a ResNet-50 model, for weighted and unweighted flips-to-accuracy functions, that are either linear, quadratic, or cubic interpolations of points.

Datasets	Unweighted Linear	Unweighted Quadratic	Weighted Linear	Weighted Quadratic	Weighted Cubic
Noises					
IN-C {75} (Hendrycks & Dietterich, 2019)	<u>4.95</u>	5.04	5.94	**4.79**	5.23
IN-$\overline{\text{C}}$ {50} (Mintun et al., 2021)	<u>7.19</u>	7.36	7.94	7.35	**7.01**
IN-3DCC {60} (Kar et al., 2022)	<u>4.10</u>	4.12	4.33	**3.66**	4.25
CCC {27} (Press et al., 2023)	<u>2.97</u>	3.22	4.8	**2.80**	4.34
Domain Shifts					
Stylized (Geirhos et al., 2019)	<u>7.12</u>	<u>7.12</u>	<u>7.12</u>	**3.81**	<u>7.12</u>
IN-V2 {3} (Recht et al., 2019)	**3.55**	<u>3.71</u>	5.42	4.70	4.03
IN-Sketch (Wang et al., 2019)	<u>1.11</u>	1.32	2.64	4.23	**0.23**
IN-R (Hendrycks et al., 2021a)	<u>1.43</u>	1.67	3.01	1.88	**0.52**
IN-D (Rusak et al., 2022a)					
→ Real	3.39	<u>3.16</u>	**2.04**	3.18	4.70
→ Painting	2.07	1.94	**0.34**	2.20	<u>0.85</u>
→ Clipart	<u>2.78</u>	3.08	5.12	3.37	**2.44**
→ Sketch	<u>6.12</u>	6.95	12.89	**5.44**	12.38
→ Infograph	<u>7.28</u>	8.76	10.35	**3.63**	10.35
→ Quickdraw	**0.79**	**0.79**	**0.79**	<u>2.57</u>	**0.79**
Cartoon & Drawing {2} (Salvador & Oberman, 2022)	13.60	13.76	14.34	<u>13.25</u>	**12.96**
Adversarial Noises					
BG Challenge {8} (Xiao et al., 2020)	<u>7.19</u>	7.36	7.33	<u>6.92</u>	8.50
IN-A (Hendrycks et al., 2021b)	23.70	23.53	**20.39**	<u>21.61</u>	22.91
IN-C Patch {75} (Gu et al., 2022)	1.95	2.00	2.42	<u>1.60</u>	**1.48**
IN-Hard (Taesiri et al., 2023)	5.27	4.92	**0.72**	3.64	<u>3.49</u>
Patch-IN {10} (Pintor et al., 2023)	**7.42**	<u>7.55</u>	9.02	8.87	7.98
IN-Obfuscations {3} (Stimberg et al., 2023)	<u>0.20</u>	**0.10**	**0.10**	4.58	**0.10**
OOD/Other					
ObjectNet (Barbu et al., 2019)	<u>6.81</u>	<u>6.81</u>	<u>6.81</u>	**2.74**	<u>6.81</u>
NINCO (Bitterwolf et al., 2023)	20.20	19.85	**14.98**	18.07	<u>17.73</u>
Average	<u>6.14</u>	6.27	6.47	**5.75**	6.36
Worst Case	23.70	23.53	**20.39**	<u>21.61</u>	22.91
Average (Worst Case Excluded)	<u>5.34</u>	5.48	5.84	**5.03**	5.60

C. WF with Limited Data

To further test WF's ability in a challenging setting, we look at how it performs under memory and data constraints. To this end, we test WF in the following scenarios: (1) WF is only allowed to store 100 samples for calculating flips, and (2) when whole dataset is limited to 100 samples for flip calculation and 1,000 samples for adaptation). We note that previous work assumes the existence of at least 2,000 test samples (Niu et al., 2022). In both cases, we use the original weighted-flips-to-accuracy function, f, by multiplying the the weighted flips calculated on 100 samples by 10, and plugging the output into f. Even with only using 100 samples, WF is able to best the original implementation by a bit. Surprisingly, even with limited data and memory, WF manages to remain competitive with unconstrained methods, and is significantly ahead of COT, when it is constrained in a similar manner.

Table 4. WF is effective in memory constrained settings. Without finetuning or refitting f, WF beats the original implementation, when only using 100 samples to calculate weighted flips (WF limited mem). In the limited memory/data setting, WF gets access to only 1000 samples in total, 100 of which are used for flip calculations. In this setting, COT gets access to 1,000 input samples and 1,000 in distribution samples. **Best** results are in bold; second best are underlined, $\{.\}$ indicates how many splits are in each dataset, when there are more than 1.

Datasets	COT original	WF original	WF limited mem	COT limited mem/data	WF limited mem/data
Noises					
IN-C $\{75\}$ (Hendrycks & Dietterich, 2019)	**2.23**	4.79	7.52	36.67	6.52
IN-$\bar{\text{C}}$ $\{50\}$ (Mintun et al., 2021)	**3.17**	7.35	8.34	40.55	4.60
IN-3DCC $\{60\}$ (Kar et al., 2022)	**3.02**	3.66	3.97	34.44	4.31
CCC $\{27\}$ (Press et al., 2023)	2.04	2.80	3.71	26.67	4.92
Domain Shifts					
Stylized (Geirhos et al., 2019)	12.18	3.81	3.37	38.84	**2.50**
IN-V2 $\{3\}$ (Recht et al., 2019)	**2.68**	4.70	4.00	43.96	3.80
IN-Sketch (Wang et al., 2019)	4.23	1.71	**1.68**	12.46	3.39
IN-R (Hendrycks et al., 2021a)	2.44	**1.88**	3.03	14.99	12.03
IN-D (Rusak et al., 2022a)					
→ Real	27.54	3.18	**1.73**	41.52	6.51
→ Painting	7.49	2.12	**0.71**	26.21	18.44
→ Clipart	4.52	**3.37**	5.91	15.98	8.10
→ Sketch	**0.71**	5.44	6.30	12.65	4.50
→ Infograph	3.44	3.63	**1.24**	4.57	2.51
→ Quickdraw	1.60	2.57	2.80	**0.06**	2.46
Cartoon & Drawing $\{2\}$ (Salvador & Oberman, 2022)	**1.62**	13.25	16.48	33.25	13.44
Adversarial Noises					
BG Challenge $\{8\}$ (Xiao et al., 2020)	19.68	6.92	**5.84**	32.84	10.15
IN-A (Hendrycks et al., 2021b)	30.38	21.61	16.75	**15.30**	29.15
IN-C Patch $\{75\}$ (Gu et al., 2022)	2.57	**1.60**	1.98	47.03	1.92
IN-Hard (Taesiri et al., 2023)	15.33	3.64	**0.65**	5.83	14.73
Patch-IN $\{10\}$ (Pintor et al., 2023)	10.13	**8.87**	9.09	49.68	9.81
IN-Obfuscations $\{3\}$ (Stimberg et al., 2023)	0.12	4.58	4.67	**0.09**	8.93
OOD/Other					
ObjectNet (Barbu et al., 2019)	10.40	2.74	**0.29**	2.44	2.74
NINCO (Bitterwolf et al., 2023)	20.28	18.07	20.24	**13.05**	35.68
Average	8.17	5.75	**5.67**	23.87	9.40
Worst Case	30.38	21.61	**20.24**	49.68	35.68
Average (Worst Case Excluded)	7.16	5.03	**5.00**	22.70	8.21

D. Weighted Flips Ablations

D.1. Stopping Iteration Ablations

WF measures the amount of weighted flips from iteration 0 to iteration 1,000. This is done because RDumb resets the model to its pretrained state every 1,000 iterations, in order to avoid collapse (Press et al., 2023). Here, we look at how measuring weighted flips before iteration 1,000 affects the performance of WF. Interestingly, using 500 iterations increases performance by a relative 26.89% as opposed to the 1,000 iterations used in the rest of the paper.

	Stopping Iteration				
Datasets	1000	500	250	100	50
IN-C	4.79	4.88	4.99	5.48	5.67
IN-$\overline{C}$	7.35	7.48	7.84	8.96	10.10
IN-3DCC	3.66	3.32	3.37	3.00	3.06
IN-V2	4.70	4.59	4.93	5.11	5.49
IN-D ↴					
Real	3.18	0.36	0.96	2.82	5.01
Painting	2.12	1.21	4.02	11.28	14.83
Clipart	3.37	0.31	3.75	9.30	14.85
Sketch	5.44	2.49	0.33	5.61	9.26
Infograph	3.63	3.46	0.09	4.37	7.53
Quickdraw	2.57	1.73	7.55	17.54	25.35
Average	4.08	**2.98**	3.78	7.35	10.12

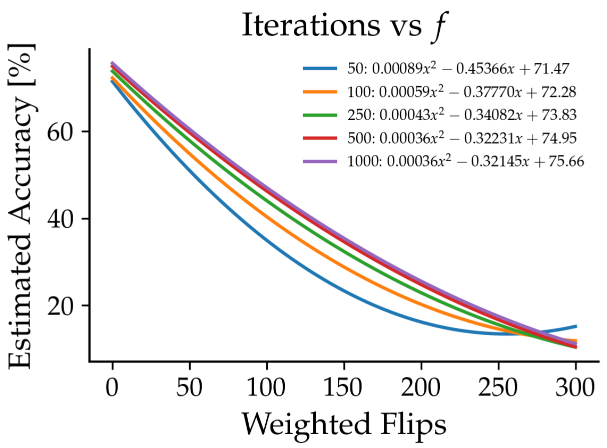

Figure 9. **Left:** Mean Absolute Error between estimated accuracy and true accuracy, when measuring weighted flips between iteration 0 and various stopping iterations. **Right:** For different stopping iterations, interpolating between the points in the holdout set yields different weighted-flips-to-accuracy functions.

D.2. Holdout Set Size Ablations

	Holdout Set Size				
Datasets	1000	500	250	100	50
IN-C	4.79	4.77	5.52	6.07	7.62
IN-$\overline{C}$	7.35	5.91	6.74	7.06	8.40
IN-3DCC	3.66	5.10	4.34	5.17	5.22
IN-V2	4.70	5.13	5.48	3.47	9.10
IN-D ↴					
Real	3.18	3.50	1.78	0.57	4.21
Painting	2.12	5.38	1.92	6.93	8.57
Clipart	3.37	6.69	7.96	8.79	7.70
Sketch	5.44	10.23	7.39	4.87	4.10
Infograph	3.63	6.15	0.88	3.05	3.77
Quickdraw	2.57	0.71	9.87	38.86	31.66
Average	**4.08**	5.36	5.19	8.48	9.04

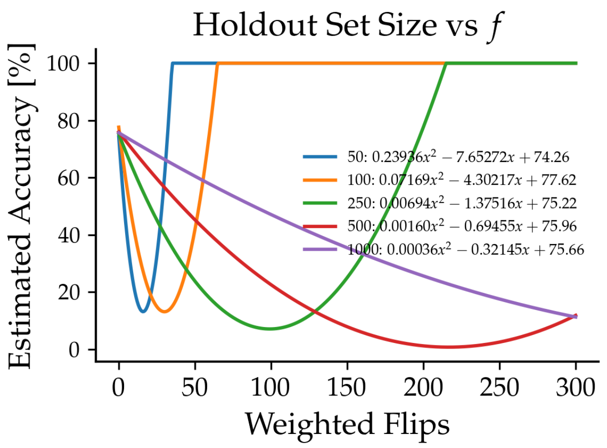

Figure 10. **Left:** Mean Absolute Error between estimated accuracy and true accuracy, when measuring weighted flips on sets of images of different sizes. **Right:** For different holdout set sizes, interpolating between the points in the holdout set yields different weighted-flips-to-accuracy functions. f can only output values that are between 0 and 100.

E. WF with other TTA Methods

WF esimates the accuracy of a dataset as RDumb (Press et al., 2023) is used to adapt to it. In this section, we show that WF work with a variety of different EM methods. To further showcase the versatility of WF, we do not finetune any method, and use the original weighted-flips-to-accuracy function f, for all experiments in Table 5.

Table 5. Mean Absolute Error between estimated accuracy and true accuracy, when adapting to data using a ResNet-50 backbone and different TTA methods: Tent (Wang et al., 2020), RPL (Rusak et al., 2022b), and CPL (Goyal et al., 2022). In all cases, the original weighted-flips-to-accuracy function f is used, highlighting the versatility of WF.

Datasets	RDumb	Tent	RPL	CPL
IN-C	4.79	6.75	6.85	5.14
IN-$\overline{\text{C}}$	7.35	9.68	7.20	7.44
IN-3DCC	3.66	2.92	3.99	3.72
IN-V2	4.70	3.80	3.82	4.42
IN-D ↴				
Real	3.18	5.15	5.15	0.31
Painting	2.12	7.59	7.59	0.03
Clipart	3.37	6.98	7.11	2.57
Sketch	5.44	3.30	3.52	3.86
Infograph	3.63	2.29	2.24	3.40
Quickdraw	2.57	2.24	2.24	2.37
Average	4.08	5.07	4.97	**3.33**

F. Additional Vision Transformer Experiments

To further analyze WF and the second best method, COT, we add additionally analysis using a ViT-B/16 model. The task is to estimate the accuracy of a ViT-B/16 on a variety of datasets. We compare between using the original weighted-flips-accuracy function, f, which was interpolated using data from a ResNet-50, and interpolating the function using ViT-B/16 data points. In both cases, the datasets used to interpolate are the same. Additionally, we compare to COT on this task.

Table 6. Mean Absolute Error between estimated accuracy and true accuracy, when estimating the accuracy of a ViT-B/16 on different datasets.

Datasets	WF	WF (new f)	COT
IN-C	8.34	1.64	22.24
IN-$\overline{\text{C}}$	6.59	1.48	25.37
IN-3DCC	7.19	1.87	18.43
IN-V2	4.44	3.63	21.29
IN-D ↴			
Real	1.02	7.27	37.21
Painting	3.02	7.06	19.13
Clipart	12.82	0.25	13.58
Sketch	11.04	1.90	5.74
Infograph	9.27	3.34	1.16
Quickdraw	2.36	13.04	0.28
Average	6.61	**4.15**	16.44

G. Omitting Samples by Top-k Accuracy/Entropy Level

In addition to removing samples by Top-k accuracy, we also analyze the effects of removing samples according to their initial entropy level. We find that both experiments exhibit similar behaviour: it is possible to remove many Top-k/low entropy samples, without significantly affecting the accuracy gain of Tent (on a holdout set of Gaussian Noise 3).

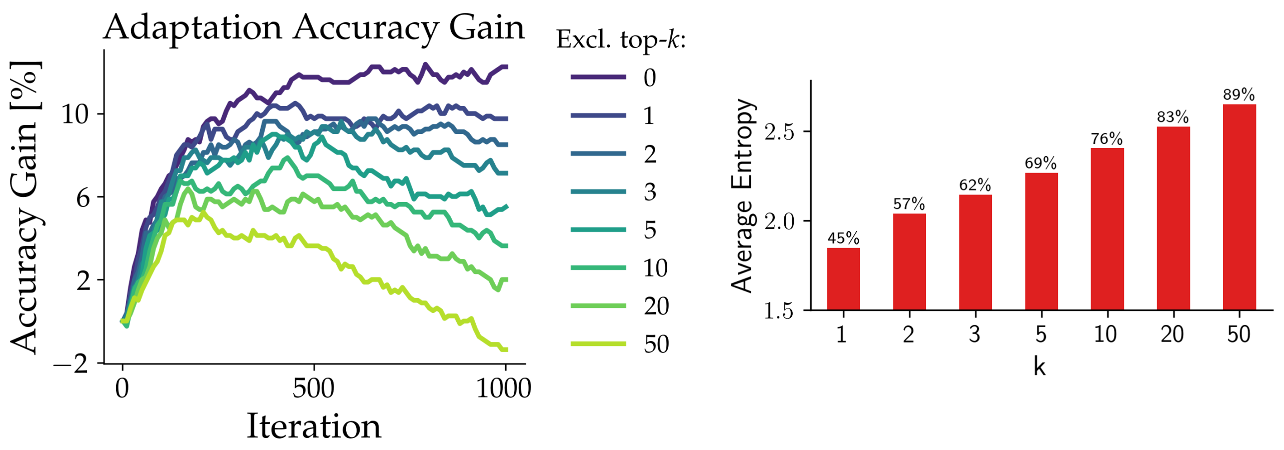

Figure 11. **Left:** Average entropy across top-k samples for different values of k. The percentages shown are the fraction of images out of the whole dataset. The original dataset, Gaussian Noise 3, has an average entropy of 2.84. **Right:** The relative size of the datasets, when top-k samples are removed.

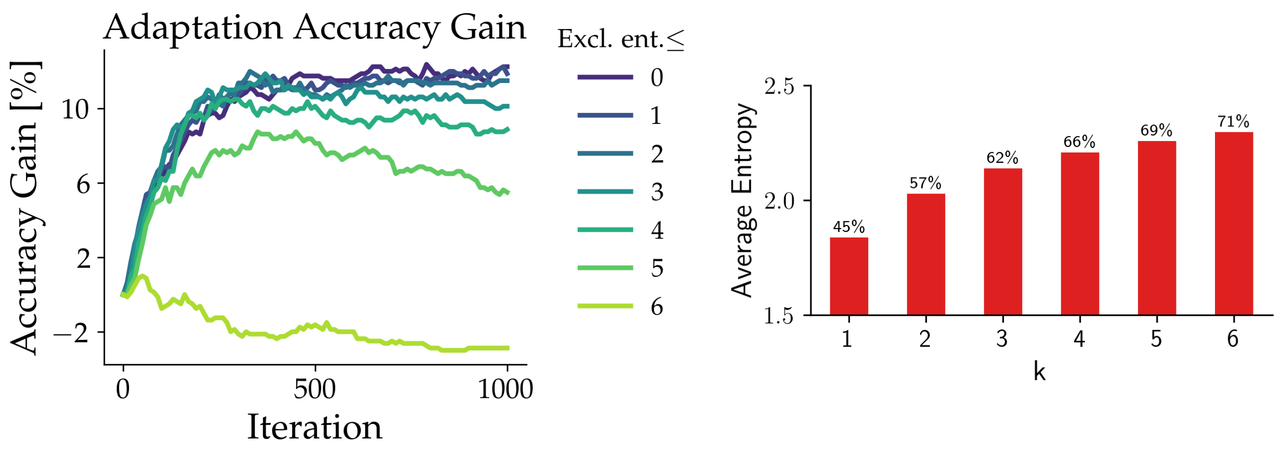

Figure 12. **Left:** Accuracy gain per iteration on a holdout set, as Tent adapts to its inputs. Each line corresponds to a different experiment where we remove samples based on their initial entropy level. Similarly to Figure 2, it's possible to remove low entropy samples while barely hurting performance. When entropy ≤ 0, no images are excluded. **Right:** The relative size of the datasets and their average entropy, when samples with a entropy level $\leq k$ are removed.

H. Silhouette score, Shift distance, and Accuracy Throughout Entropy Minimization

In Figure 4, we looked at the changes of Silhouette scores/Shift distances for each phase in EM. Here, we show how these scores, along with accuracy, change in every iteration of Tent. For each one of the datasets analyzed, we group noises based on severity level, and plot their averages and standard deviations, for every iteration.

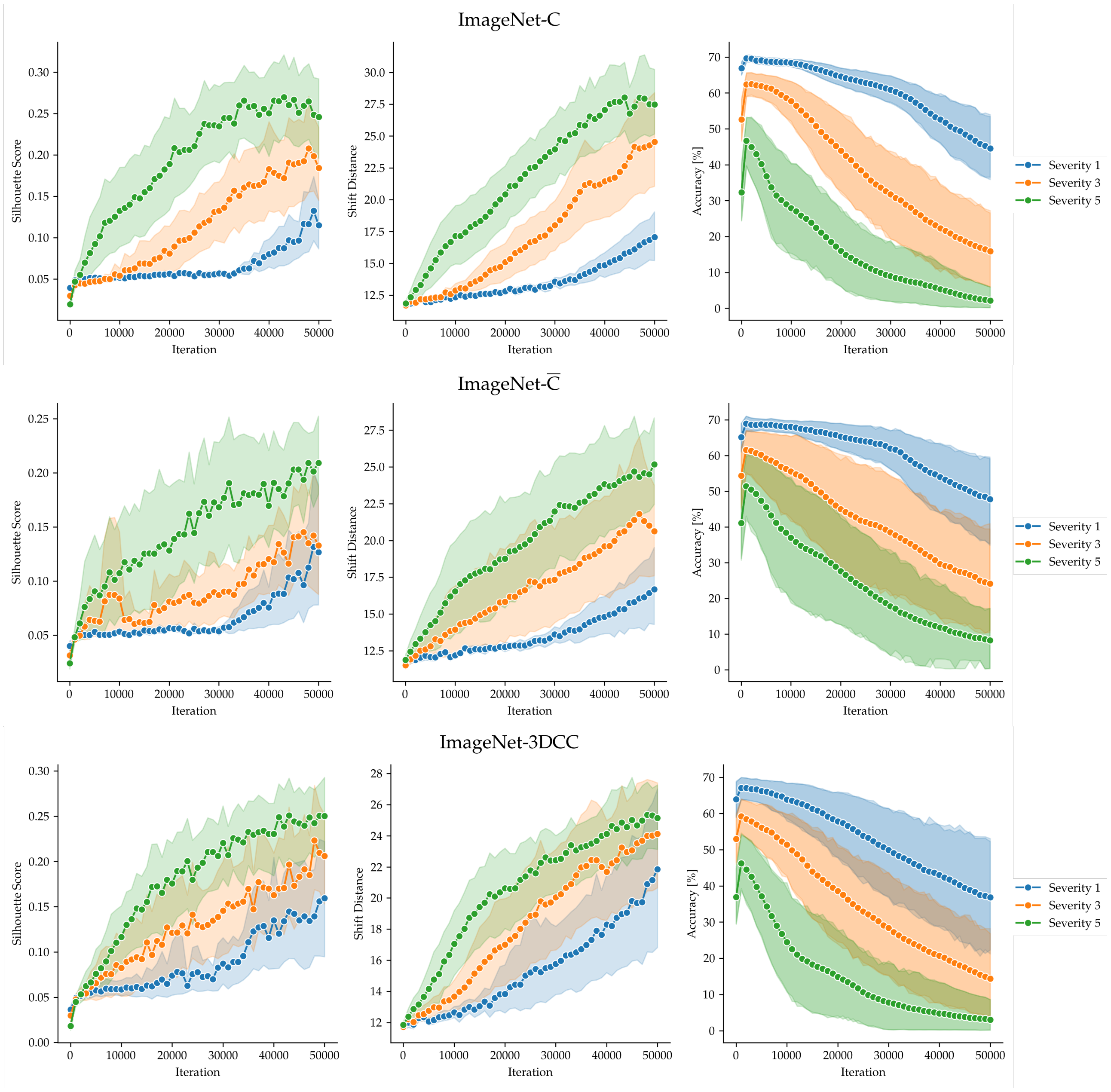

Figure 13. Changes in Silhouette scores, Shift distances, and Accuracies as Tent adapts to its inputs. We group together noises by severity level, and average the data for every iteration.

I. WF on CIFAR10/100

WF is not as effecitve on CIFAR10 (Krizhevsky et al., 2009) as it is on ImageNet (Deng et al., 2009) scale datasets. CIFAR10 is an outlier in entropy minimization: for example, Press et al. (2023) showed that Tent doesn't degrade in accuracy, even after 100 million CIFAR10 images seen. We nonetheless run our method on CIFAR10. On average, we see only 0-5 label flips per dataset on C10-C. This is far from what we see ImageNet-scale datasets we tested.

Like in the paper, we interpolate a weighted-flips-to-accuracy function f on the holdout set and get:

$$f(x) = -249.36x^2 - 87.39x + 77.01$$

which has a MAE of 16.64 on the C10 validation set.

We repeat this for CIFAR100 and get:

$$f(x) = 0.000322x^2 - 0.287x + 99.54$$

which has a MAE of 9.10 on the C100 validation set.

Apart from refitting f, we did not tune any other parameter in these two experiments.

J. RDumb

WF uses RDumb (Press et al., 2023) to estimate accuracy. We go over the implementation of the method in brief. RDumb is based on ETA (Niu et al., 2022), wherein the model is reset to its pretrained state every 1,000 iterations. Rdumb optimizes the BatchNorm (Ioffe & Szegedy, 2015) parameters, Θ of a given classifier f.

The loss optimized is entropy, with two filtration steps: the first, in which samples with high entropy are filtered out, and the second, in which samples that produce logits similar to previous samples are filtered out.

For a sample x, the first filtration is given by:

$$S^{ent}(x) = \frac{1}{\exp[E(x;\Theta) - E_0]} \cdot \mathbb{I}_{E(x;\Theta)<E_0}(x),$$

with $E_0 = 0.4 \times \ln 10^3$.

The second filtration is given by:

$$S^{div}(x) = \mathbb{I}_{\{cos(f_o(x), m^{-1})<\epsilon\}}(x)$$

where $cos()$ is the cosine similarity, and m^t is an exponential moving average of the logits of previously seen samples at iteration t:

$$m^t = \begin{cases} y^1, & \text{if } t = 1 \\ \alpha y^t + (1-\alpha)m^{t-1}, & \text{if } t > 1 \end{cases}$$

and y^t is the average model prediction on a batch of inputs at step t, and $\alpha = 0.9$.

Put together with entropy minimization, the optimization formula becomes:

$$\min_{\hat{\Theta}} -S^{ent}(x) \cdot S^{div}(x) \sum_{y \in C} f_\Theta(y|x) \log f_\Theta(y|x)$$

RDumb uses a SGD with a learning rate of 2.5×10^{-4}, and a batch size of 64, and is reset to its pre-trained state every 1,000 iterations.

K. Software Licenses

- ImageNet-C (Hendrycks & Dietterich, 2019)Apache License 2.0
  `https://github.com/hendrycks/robustness`
- ImageNet-R (Hendrycks et al., 2021a) MIT License
  `https://github.com/hendrycks/imagenet-r`
- ImageNet-3D-CC (Kar et al., 2022): CC-BY-NC 4.0 License
  `https://github.com/EPFL-VILAB/3DCommonCorruptions`
- ImageNet-$\overline{\text{C}}$ (Mintun et al., 2021): MIT License
  `https://github.com/facebookresearch/augmentation-corruption`
- ImageNet-V2 (Recht et al., 2019): MIT License
  `https://github.com/modestyachts/ImageNetV2`
- Backgrounds Challenge (Xiao et al., 2020):
  `https://github.com/MadryLab/backgrounds_challenge`
- CCC (Press et al., 2023): MIT License
  `https://github.com/oripress/CCC`
- Stylized ImageNet (Geirhos et al., 2019): MIT License
  `https://github.com/rgeirhos/Stylized-ImageNet`
- NINCO (Bitterwolf et al., 2023): MIT License `https://github.com/j-cb/NINCO`
- ImageNet-D (Rusak et al., 2022a): Apache License 2.0
  `https://github.com/bethgelab/robustness`
- ObjectNet (Barbu et al., 2019): MIT License `https://objectnet.dev/`
- Shift Happens Benchmark: Apache License 2.0 `https://github.com/shift-happens-benchmark/icml-2022`