Robust Weight Signatures: Gaining Robustness as Easy as Patching Weights?

Ruisi Cai [1] **Zhenyu Zhang** [1] **Zhangyang Wang** [1]
[1]VITA Group, University of Texas at Austin
https://github.com/VITA-Group/Robust_Weight_Signatures

Abstract

Given a robust model trained to be resilient to one or multiple types of distribution shifts (e.g., natural image corruptions), *how is that "robustness" encoded in the model weights, and how easily can it be disentangled and/or "zero-shot" transferred to some other models*? This paper empirically suggests a surprisingly simple answer: **linearly - by straightforward model weight arithmetic**! We start by drawing several key observations: (i) assuming that we train the same model architecture on both a clean dataset and its corrupted version, a comparison between the two resultant models shows their weights to mostly differ in shallow layers; (ii) the weight difference after projection, which we call *"Robust Weight Signature"* (**RWS**), appears to be discriminative and indicative of different corruption types; (iii) perhaps most strikingly, for the same corruption type, the RWSs obtained by one model architecture are highly consistent and transferable across different datasets.

Based on those RWS observations, we propose a minimalistic model robustness "patching" framework that carries a model trained on clean data together with its pre-extracted RWSs. In this way, injecting certain robustness to the model is reduced to directly adding the corresponding RWS to its weight. We experimentally verify our proposed framework to be remarkably (1) **lightweight**. since RWSs concentrate on the shallowest few layers and we further show they can be painlessly quantized, storing an RWS is up to 13 $\times$ more compact than storing the full weight copy; (2) **in-situ adjustable**. RWSs can be appended as needed and later taken off to restore the intact clean model. We further demonstrate one can linearly re-scale the RWS to control the patched robustness strength; (3) **composable**. Multiple RWSs can be added simultaneously to patch more comprehensive robustness at once; and (4) **transferable**. Even when the clean model backbone is continually adapted or updated, RWSs remain as effective patches due to their outstanding cross-dataset transferability.

[1]Department of Electrical and Computer Engineering, University of Texas at Austin. Correspondence to: Zhangyang Wang <atlaswang@utexas.edu>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

1. Introduction

1.1. Background and Related Work

The robustness and safety of machine learning models have become prevailing concerns for practitioners. Among many other possible forms of safety risks such as adversarial attacks (Madry et al., 2017; Zhang et al., 2019) and backdoor attacks (Goldblum et al., 2022), one concern of particular significance is the model's resilience against various distribution shifts from training data (Koh et al., 2021). For example, a computer vision model for autonomous driving or video surveillance could be trained on relatively "clean" and constrained data to achieve high performance on standard benchmarks. However, they are vulnerable to unforeseen distributional changes including natural corruptions (e.g., due to camera noise, motion blur, adverse weather), sensory perturbations (e.g., sensor transient error, electromagnetic interference), and larger domain shift forms (e.g., summer $\rightarrow$ winter, daytime $\rightarrow$ night) - hence jeopardizing their trustworthiness and safe deployment.

Many solutions have since been examined to strengthen the models' robustness against unforeseen distribution shifts, in particular natural image corruptions - which would be the focus of this paper. Examples include data augmentation (Hendrycks et al., 2021; 2019b; Rusak et al., 2020; Wang et al., 2021), stability-aware training (Hein & Andriushchenko, 2017; Zheng et al., 2016), leveraging pretrained models (Hendrycks et al., 2019a; Chen et al., 2020; Jiang et al., 2020; Sun et al., 2021; Wortsman et al., 2022b) or training on larger and more diverse datasets (Taori et al., 2020; Nguyen et al., 2022). Among them, data augmentation is perhaps the most popular practice, as it is easy to implement and plug in. It also remains as the most empirically effective approach to gain comprehensive robustness to various natural corruptions (Hendrycks et al., 2019b; Wang et al., 2021), though at the cost of training time overhead.

Further complicating the problem is the inherent "trade-off" between model standard accuracy and robustness, informally: *the more "comprehensive" robustness that a model strives to cover, the less "focused" it can fit the standard clean data distribution*. Firstly observed in (Tsipras et al., 2019), the authors pointed out that adversarial training (AT) (Madry et al., 2018), which utilizes adversarial samples as a special data augmentation method, has also shown to improve model robustness yet sacrificing the standard accuracy on clean images. The same trade-off observation holds generally true for other data augmentation and stability-aware training methods (Wang et al., 2021), essentially reflecting the "bias-variance" trade-off. Practically, most defense methods determine their accuracy-robustness trade-off by some empirically hyper-parameter pre-chosen at training, such as the coefficient weight between the standard and robust classification losses for AT, or the strength of data augmentations. Such methods will hence "pre-fix" achievable standard and robust accuracies at training time, leaving no flexibility to adjust for testing even if there is a demand.

In practical AI platforms especially at the edge, the desired trade-off between standard and robust accuracies often varies adaptively depending on contexts, which are not always met by the pre-fixed "default" settings. For example, an autonomous agent might perceive using its "standard" mode for normal-environment operations (most of the time), but switch to behaving more conservatively such as when placed in less familiar or adverse environments. Re-training the model, especially robustly, is notoriously resource-consuming and impossible for in-situ adjustment. Hence practitioners look for convenient means to explore and flexibly calibrate the accuracy-robustness trade-off at the testing time. Test-time adaption (Fleuret et al., 2021; Croce et al., 2022) or ensembling (Liu et al., 2018) methods, though effective, will turn impractical when memory and storage are in limited supply or the inference latency is sensitive. The recent "once-for-all" AT methods (Wang et al., 2020; Kundu et al., 2023) enable the network to adjust to different input distributions nearly free of overheads, by input-conditioning. However, all aforementioned methods would compromise the achievable clean accuracy more or less, in exchange for encapsulating more robustness in the same model. Also most of them focus on adversarial attacks.

1.2. Our Aim and Contributions

This paper targets the "in-situ" adaptive robustness similarly as defined in (Wang et al., 2020; Kundu et al., 2023), i.e., to painlessly calibrate on the accuracy-robustness trade-off at the test time, with minimal overhead in latency or memory. Our **problem setting and goal** will yet differ in (1) focusing on the comprehensive robustness against unforeseen natural image corruptions (**not** adversarial attacks); and (2) **not** impairing the standard accuracy on clean test images at all.

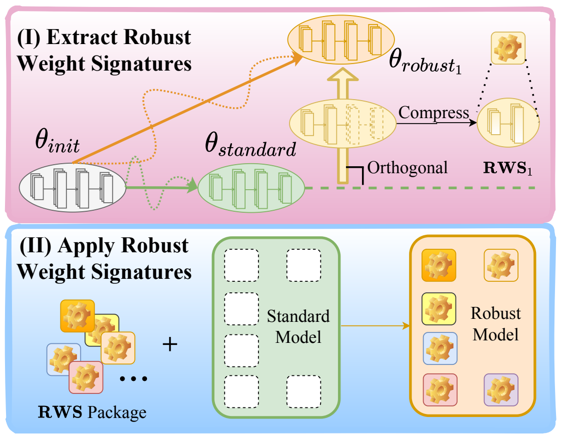

Figure 1. Overview of our pipeline: Step (I): Extract Robust Weight Signatures (RWSs) by comparing the difference between robust models and standard models of shallow layers in the weight space. Step (II): Patch non-robust models by RWSs as needed.

Our proposal is a minimalistic model robustness "patching" framework that differs remarkably from the aforementioned efforts. We are inspired by the recent findings on the linear interpolatability between model weights (such as pre-trained and fine-tuned) (Wortsman et al., 2022a;b; Li et al., 2022; Ilharco et al., 2022a). Contrary to the common wisdom of model output ensembling, (Wortsman et al., 2022a) pioneered averaging the weights of multiple fine-tuned models directly, without incurring any additional inference or memory costs, that yield significantly improved "zero-shot" and out-of-distribution generalization performance. Most relevantly, (Ilharco et al., 2022a) demonstrated that such model weight "arithmetic" can go beyond averaging: by computing the weight difference between a pre-trained model and its downstream-task fine-tuned version (called a "task vector"); the resulting task vectors are noted to meaningfully steer the behavior of neural networks: they can be modified and combined together through arithmetic operations such as negation and addition, e.g., adding multiple task vectors together can improve performance on multiple tasks at once.

In view of those, we ask: in a robust model, *how is that "robustness" encoded in the model weight space, and how can it be decoded, combined, or transferred? How would that further help our in-situ adaptive robustness goal?* Given a standard model (trained on clean data) and its robust counterpart (trained on corrupted versions of the same dataset), we compare their difference to extract the *"Robust Weight Signature"* (**RWS**). It turns out that RWS lends a surprisingly

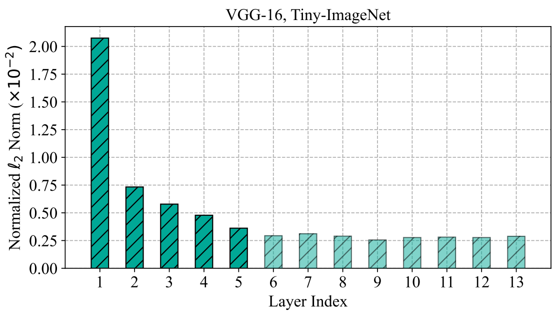

Figure 2. On TinyImageNet and VGG-16, we visualize ℓ_2 norms of RWSs for each convolutional layer, indicating the non-robust models and robust models mainly differ in the shallow layers.

effective, elegantly simple and flexible means to achieve in-situ robustness, due to the following novel findings:

- Assuming a model trained on clean data together with its pre-extracted RWSs to multiple image corruption types, "patching" certain robustness to the model is reduced to directly adding the corresponding RWS to its weight. Any appended RWS can be later taken off to switch back to the intact clean model: hence there is no compromise of standard accuracy.

- RWSs are highly compressible since their large-magnitude elements are dominantly in the lower layers. We further show them to be robust to quantization as well. Hence storing an RWS is up to 13× more compact than the full weight copy, mitigating the storage burden of carrying multiple pre-trained models.

- RWSs are extraordinarily controllable and combinable: one can linearly re-scale an RWS to control the patched robustness strength. Multiple RWSs can be added simultaneously to patch more comprehensive robustness at once. Essentially, we demonstrate the task "arithmetic" claims in (Ilharco et al., 2022a) to be generally valid for multiple robustness types as well.

- Lastly and uniquely, we find that an RWS is not tied with the standard model where it is subtracted. That is, when the standard model is updated, continually adapted, or even completely re-trained on a different dataset, the RWS seems to be the same applicable to the new model (same architecture). Such outstanding cross-data transferability decouples the standard model updating and robustness preservation, potentially saving training costs and boosting weight re-usability.

In what follows, we accompany our claims with experimental results, showing that RWSs extensively improve model robustness to various natural image corruptions in a plug-and-play manner, while demonstrating to be **lightweight**, **in-situ adjustable**, **composable**, and **transferrable**.

2. Robust Weight Signatures: Concept Proofs

2.1. Definition and Notations

We compare non-robust and robust models in the weight space, to investigate how robustness is encoded. To begin with, model providers train a standard model $\theta_{\text{std}} \in \mathbb{R}^d$ on a clean dataset, and multiple robust counterparts $\theta_r^c \in \mathbb{R}^d$, each with corruption type c, from the same initialization $\theta_{\text{init}} \in \mathbb{R}^d$ used by θ_{std}. We denote $\theta_{\text{std}} - \theta_{\text{init}}$ as the *base direction* v_{base}, which contains knowledge of fitting standard dataset. For each corruption type c, we similarly compute the *robustifying direction* $v_c = \theta_r^c - \theta_{\text{init}}$, which is assumed to contain the knowledge of resilience against corruption c. We then disentangle the robustness-part knowledge with the standard dataset-fitting knowledge, by subtracting v_c's **projection** on v_{base}, from v_c itself. We refer to the obtained residual vector as a robust weight signature (**RWS**):

$$\mathbf{RWS}_c = v_c - P_{v_{base}}(v_c), \tag{1}$$

where $P_{v_{base}}(v_c)$ denotes the projection operator from v_c onto the column space of v_{base}, implemented by matrix pseudoinverse. The process of extracting RWS is also illustrated in the upper Figure 1.

Note that the projection-residual idea implies the (somewhat gross) assumption that the "standard fitting knowledge" and "robustness knowledge" are encoded **nearly orthogonally** in the robust model weights. The use of projection $P_{v_{base}}$ also **goes beyond** the vanilla weight arithmetic regime in (Ilharco et al., 2022a) who simply subtract one weight from the other: we also tried the same and find it unable to extract effective RWSs (especially poor in composition). We conjecture that is because the weight gap between {pre-trained, fine-tuned} models (Ilharco et al., 2022a) is either smaller or more linearly connected, compared to the weight gap between {standard, robust} models in our case. We leave the verification of these two open thoughts as future work.

Besides the above, we point out two other substantial differences between our RWSs and "task vectors" in (Ilharco et al., 2022a). Firstly, RWSs exhibit a compact and compressible structure (Sec. 2.2) which was not observed in task arithmetic (Ilharco et al., 2022a). Secondly, we observe RWSs to be consistent and transferrable across datasets (Sec 2.4), echoing our conjecture that robustness is perhaps encoded relatively independently of standard dataset content: the finding has no counterpart in (Ilharco et al., 2022a) either.

Experimental Details. We use three datasets, CIFAR-10, CIFAR-100 (Krizhevsky et al., 2009) and Tiny-ImageNet (mnmoustafa, 2017), with two model architectures, VGG-16 (Simonyan & Zisserman, 2014) and ResNet-50 (He et al., 2016). By default, we obtain a robust model, by training with the corresponding type of data augmentations applied to the clean dataset. All VGG-16 models use a learning rate

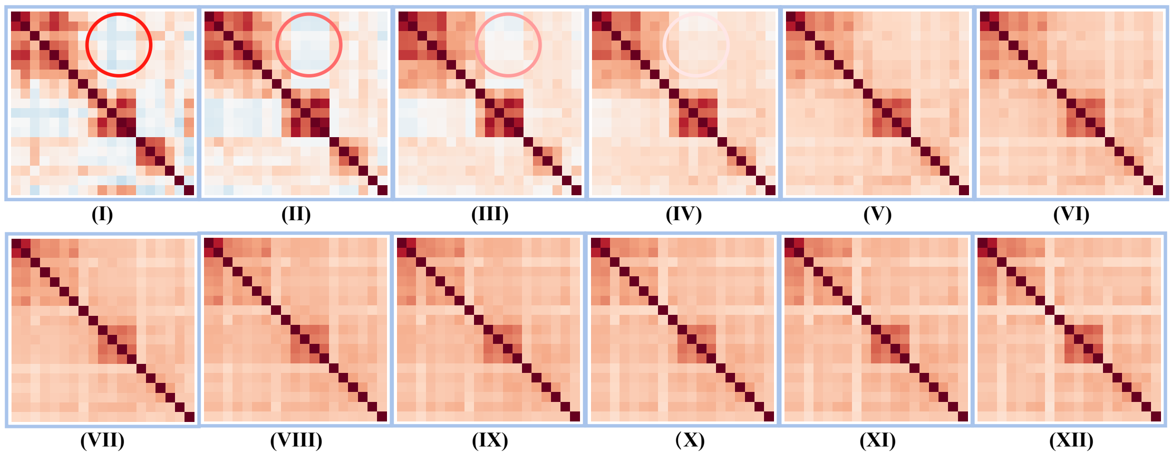

Figure 3. Based on TinyImageNet and VGG-16, we visualize cosine similarities between different types of corruptions, at different layers. Lighter colors indicate smaller cosine similarities. The roman numbers refer to the corresponding layer indexs. RWSs of different corruption types are significantly more diverse in shallow layers.

of 0.01, while all ResNet-50 models use 0.001. We follow the corruption types in (Hendrycks & Dietterich, 2019b) and the corruption severity levels are set to be 5 (strongest) for all experiments by default. Intentionally, neither adversarial training nor more compositional augmentation was involved, because we want to "purify" each RWS to cater to one corruption type, facilitating our later experiments to demonstrate their controllability and composition.

For the choice of common initialization θ_{init}, we found that the same random initialization did not suffice to manifest the RWS phenomenon. That is understandable: compared to fine-tuning the same pre-trained model (Ilharco et al., 2022a), two models trained (standard or robustly) from scratch could be far away in their weight space, even using the same initialization and dataset, due to many randomness factors in the much longer training process. To "anchor" the standard and robust model weights to be meaningfully close for RWS extraction, we explored two strategies: (1) use an ImageNet pre-trained model[1] as θ_{init}, and train both standard and robust models from there; (2) first train a standard model from scratch, and use it as θ_{init} to train all other robust models from. Both are found to expose RWSs much better and more stably, and we report results from the first option by default due to its superior cross-dataset transferrability.

2.2. RWSs are Concentrated in Shallow Layers

Intuitively, many image corruption artifacts interfere with the low-level features, inviting the natural guess: *whether the corruption fragility of standard models, and correspondingly the robustness to them encoded by robust models, are mainly encoded in shallow layers*. Prior works have presented relevant findings. For example, (Huang et al., 2021) observed that more parameters can improve robustness only when added to the shallow layers. We experimentally validate the hypothesis to be explicitly true.

Figures 2 and 3 visualize the norms (normalized to the same layer's standard weight norm, averaged across all corruption types) and diversities (cosine similarity across different corruption types) of RWSs from each layer. Overall, RWSs at shallower layers are (i) significantly larger in norm. For example, the first five layers occupy more than 65% of total norm energy for RWSs extracted from VGG-16 on Tiny-Imagenet; (ii) significantly more diverse and discriminative between corruption types. Both observations imply that RWSs are more "informative" in shallow layers.

In all experiments hereinafter, we use RWSs in the **shallowest five layers** by default. This also leads us to aggressively compress RWSs in Sec 3.1 for lightweight patching.

2.3. RWSs Recover Corruption Relationships

We now take a deeper dive from Figure 3, noting that different natural corruption types are not irrelevant. Instead, corruptions are roughly categorized into four groups: noise, blur, weather and digital (Hendrycks & Dietterich, 2019b). (Yin et al., 2019) also found that different corruptions related to different frequency domains. Corruptions within the same group category or frequency range are more similar, and the robustness against one corruption tends to help defend its similar ones too. On the contrary, different categories of corruptions may even offset each other's robustness.

Interestingly, RWSs successfully recover the relations of different corruption types. Figure 4 (left) visualizes the cosine similarities between RWS of different corruption types, which reflect the grouping identified in (Hendrycks & Dietterich, 2019b). We also visualize the robust accuracy gains to all other corruptions, when a robust model is trained

[1] https://pytorch.org/vision/stable/models.html

solely on one corruption type and then directly tested on other types: the results in Figure 4 (right) echo the former.

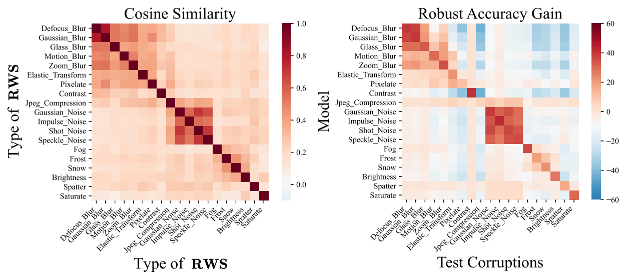

Figure 4. Left: cosine similarities between RWSs of different corruption types. Right: the robust accuracy gains to all other corruptions, between a robust model trained solely on one corruption type and then tested on other types, and a standard model directly applied. For example, each element in the row 'defocus blur' denotes the robust model trained with defocus blur and tested on other corruption types (column) - how much accuracy improvement or loss it will exhibit compared to the standard model. We use TinyImageNet with VGG-16.

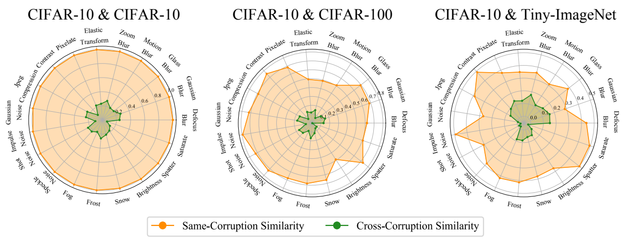

Figure 5. Same-corruption (orange circles) and cross-corruption cosine similarities (green circles) between RWSs extracted from: (left) CIFAR-10 & CIFAR-10; (middle) CIFAR-10 & CIFAR-100; (right) CIFAR-10 & TinyImageNet. The same-corruption similarity is computed between RWSs found on two datasets but of the same corruption type. The cross-corruption similarity is computed as the average of cosine similarities between the current corruption type's RWS, and every other type's RWS.

2.4. RWSs are Relatively Consistent across Datasets

Now one more step further: we compare RWSs generated from different datasets. We are hopeful because of the (gross) assumption made back in Sec. 2.1: the standard fitting and the robustness are relatively independent in weights. Figure 5 partially confirms our hypothesis that RWS found from different datasets are relatively consistent.

We first notice the same-corruption RWS cosine similarities across datasets to be consistently high. For example, between CIFAR-10 and CIFAR-100 (Figure 5 middle subfigure), all same-corruption similarities are larger than 0.5, and some reach 0.8. Even comparing CIFAR-10 and TinyImageNet (right) whose dataset statistics vary a lot, the same-corruption similarities are still all above 0.3 and sometimes reach 0.5. That implies the potential existence of "universal model robustifying directions" which is even agnostic to standard model weights.

On the other hand, the cross-corruption similarities remain consistent in the value range (most between 0.1 and 0.2), and more importantly, seem to preserve the relative similarity "ranking" to some extent. For example, 'Impulse noise' and 'Saturate' have constantly the lowest cross-corruption similarities with others, while 'Zoom blur' 'Forst' 'Defocus Blur' 'Gaussian Blur' and 'JPEG compression" are some of the consistent top rankers. We should note that this ranking consistency is imperfect: for example, 'Gaussian Noise' is a high-ranker in the left and middle subfigures, but low on the right; while 'Contrast' makes a vice versa case.

3. An In-Situ Robustness Patching Framework

Back to the main problem: *how to achieve in-situ robustness?* The aforementioned characteristics indicate the RWS involves discriminative and generalizable robust features, lending itself a promising option for direct weight patching on non-robust standard models. Given a standard model (trained on clean data) and its multiple robust counterparts (each trained on a corrupted version of the same dataset), we can store a single standard model and multiple RWSs. When needed, the robustness patching could be done immediately, by adding an RWS on standard model weight to create a *patched* model θ^c_{patch} with extra robustness on corruption c:

$$\theta^c_{\text{patch}} = \theta_{\text{std}} + \alpha * \mathbf{RWS}_c \tag{2}$$

The appended RWS can be taken off any time to switch back to the intact standard model: hence there is no compromise of standard accuracy. α is a coefficient to adjust the "strength" of the added robustness (Sec. 3.2), and the above equation could be extended to the weighted composition of multiple θ^c_{patch}s with different corruptions c (Sec. 3.3).

More Related Work on "Patching" We shall credit existing literature that has studied model patching or similar notions. In general, many efforts have been invested to efficiently for altering a model's behavior with post-training interventions, but without re-training. This stream of work may bear various names, such as patching (Goel et al., 2020; Ilharco et al., 2022b; Murty et al., 2022), editing (Mitchell et al., 2021; 2022; Santurkar et al., 2021), aligning (Askell et al., 2021; Kasirzadeh & Gabriel, 2022; Ouyang et al., 2022), debugging (Geva et al., 2022; Ribeiro & Lundberg, 2022; Ilyas et al., 2022), steering (Subramani et al., 2022), or reprogramming (Elsayed et al., 2018; Tsai et al., 2020; Hambardzumyan et al., 2021; Zhang et al., 2022). Those can operate on input, output, or weight levels, and many will take extra training or optimization steps. Several of them explored weight interpolation between a pre-trained model and its fine-tuned version, to either improve the fine-tuned model's distributional shift robustness (Wortsman et al., 2022b), or learn new specific tasks better without affecting other learned tasks (Ilharco et al., 2022b).

The most relevant work to us is the task vector arithmetic (Ilharco et al., 2022a), which uniquely adds, scales, deletes or composes model capabilities, by applying vectors in the weight space of pre-trained models. Their method is modular and efficient by re-using fine-tuned models, and does not modify the standard fine-tuning procedure. However, (Ilharco et al., 2022a) as well as (Wortsman et al., 2022b; Ilharco et al., 2022b) focus on the fine-tuning setting and rely on large pre-trained models, while ours dig into a brand-new context. In Sec. 2.1, we have also explained a few more differences between RWSs and task vectors in (Ilharco et al., 2022a), in both methodology and key findings.

Next, we present a series of experiments to demonstrate the key advantages of our patching, namely, **lightweight**, **in-situ adjustable**, **composable**, and **transferrable**.

3.1. Lightweight

The first sanity check question is: why not store multiple robust models directly, but their weight differences? The answer: those differences are much more compressible and incur much less storage overhead. The storage efficiency of RWS is achieved by two aspects: (1) as analyzed in Sec 2.2, we only use RWSs shallow layers, which usually contain much fewer parameters than latter layers; (2) we verify that RWS can further be compressed by quantization.

We follow the same setting in Sec 2.1 to construct RWSs and then follow Equation 2 to robustify the model. We set α as 1 by default. Our results are presented in Table 1. We provide several RWS options for patching standard models, including: (1) $\textbf{RWS}_{\text{full}}$: RWSs from all layers are used. (2) $\textbf{RWS}_{\text{shallow}}$: RWSs are only kept from the shallowest five layers (default). (3) $\textbf{RWS}_{\text{shallow,16bit}}$: $\textbf{RWS}_{\text{shallow}}$ further quantized to 16 bit. (4) $\textbf{RWS}_{\text{shallow,8bit}}$: $\textbf{RWS}_{\text{shallow}}$ further quantized to 8 bit. We also include three baselines: 'Standard' is the model trained on clean data only; 'Data Augmentation' is the robust model trained with all 19 corruption types seen as training data augmentations; and 'All Models' denotes the ensemble option, i.e., storing the standard model as well as 19 robust models (each dedicatedly trained with one corruption type) together. Note that 'All Models' baseline assumes always using the right dedicated model in each situation (clean, or one of the 19 corrupted). Hence it effectively makes the performance "upper bound" for all methods, though at the heaviest storage overhead. Meanwhile, 'Data Augmentation' substantially boosts the corruption robustness without any storage overhead, but sacrifices the clean data performance meanwhile.

All RWS variations show significant effectiveness in robustifying standard models while retaining/recovering the standard accuracy when taking off RWSs. $\textbf{RWS}_{\text{shallow,16bit}}$ achieve a decent trade-off between storage cost and robustness; with only 20% $\sim$ 40% storage cost increment than 'Standard' or 'Data Augmention', the method is able to (1) improve 30% $\sim$ 88% averaged robustness gain across four cases, compared to the standard baseline; and (3) consistently outperform the "Data Augmentation' baseline in achievable TA-RA trade-offs, $\textbf{RWS}_{\text{shallow,8bit}}$ further boosts the storage efficiency with small RA losses from 16-bit (especially, negligible on ResNet-50 + Tiny-ImageNet). More detailed comparisons and baseline results are in Appendix.

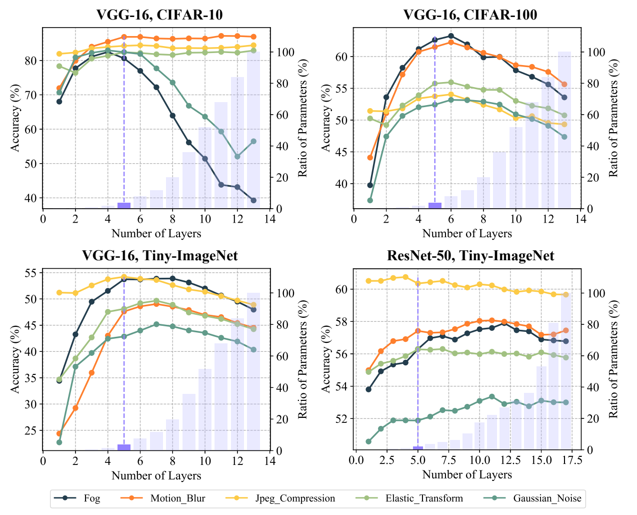

Figure 6. Robustness trends (we select a few representative corruptions types) when altering the number of shallowest layers used for RWS construction. We also plot the used parameter ratios.

We further alter the number of layers used for constructing RWS and plot the average robust accuracy of patched models (left bar of each subfigure), accompanied by the corresponding ratio of parameters participated (right bar). Figure 6 shows that: (1) the robustness of patched VGG models does not benefit from using more layers in extracting RWSs, and actually will be "backfired" when more latter layers are included; (2) ResNet models also see saturation effects on the robustness of most corruption types, after more than 5 layers are used. Both imply the high-level features have little to do with robustness encoding, and justify our design choice of using only the shallowest few layers.

3.2. In-Situ Adjustable

RWSs can be not only applied to patch a single robust model per corruption: they can even adapt to any corruption levels (e.g. different visibility in the fog weather) by linearly re-scaling, easily achieving the smooth trade-off between standard and robustness performances in one same model (note this is different from adding/taking off an RWS, which is essentially switching between two models). This can be achieved by adjusting the coefficient $\alpha \in [0, 1]$ in Equation 2: essentially, that is interpolating the (shallow layers')

Table 1. Comparison of RWS-based methods and other options. We consider 19 corruption types in (Hendrycks & Dietterich, 2019b). "RA" refers to averaged accuracy on all kinds of corrupted data, while "TA" refers to the test accuracy in the standard setting. N_{param} denotes the total model storage size (in MBs). Note that in the RWS-based pipeline, we report "TA" for the model when the RWSs are taken off (hence fully recovering the standard model); and report "RA" when the corresponding RWS is patched per corruption. This is an ideal case of using RWS patching - and its rationale and limitations will be both discussed in Sec. 4. Similarly, as a fair comparison, the 'All Models' baseline assumes we always use the right dedicated standard/robust model in each clean/corrupted situation, too.

Methods	CIFAR-10			CIFAR-100			Tiny-ImageNet					
	VGG-16			VGG-16			VGG-16			ResNet-50		
	N_{param} (MB)	TA (%)	RA (%)	N_{param} (MB)	TA (%)	RA (%)	N_{param} (MB)	TA (%)	RA (%)	N_{param} (MB)	TA (%)	RA (%)
Standard	58.8	92.59	65.44	58.8	71.44	36.76	59.2	61.28	23.58	95.7	65.72	29.65
Data Augmentation	58.8	89.58	84.34	58.8	67.34	56.95	59.2	52.11	43.64	95.7	59.17	47.96
All Models	1177.6 (20×)	92.59	88.97	1177.6 (20×)	71.44	64.72	1184.0 (20×)	61.28	51.55	1913.6 (20×)	65.72	55.97
Standard+**RWS**$_{\text{Full}}$	1177.6 (20×)	92.59	75.35	1177.6 (20×)	71.44	52.58	1184.0 (20×)	61.28	43.63	1913.6 (20×)	65.72	53.64
Standard+**RWS**$_{\text{Shallow}}$	101.0 (1.7×)	92.59	84.86	101.0 (1.7×)	71.44	58.78	101.4 (1.7×)	61.28	44.66	131.4 (1.4×)	65.72	52.84
Standard+**RWS**$_{\text{Shallow,16bits}}$	**79.9 (1.4×)**	**92.59**	**84.76**	**79.9** (1.4×)	**71.44**	**58.62**	**80.3** (1.4×)	**61.28**	**44.25**	**113.6 (1.2×)**	**65.72**	**52.81**
Standard+**RWS**$_{\text{Shallow,8bits}}$	69.4 (1.2×)	92.59	82.99	69.4 (1.2×)	71.44	53.52	69.7 (1.2×)	61.28	39.40	104.7 (1.1×)	65.72	52.79

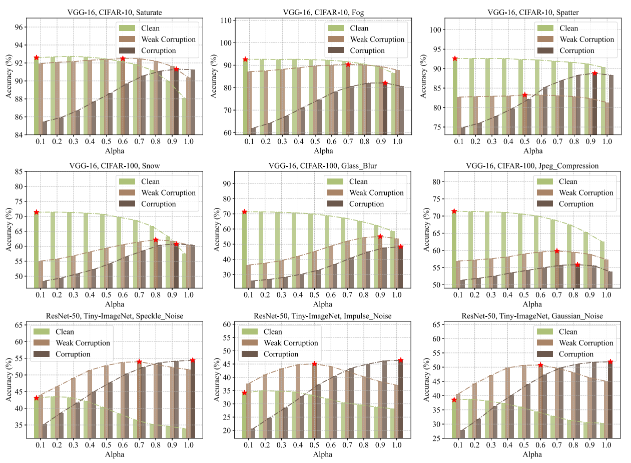

Figure 7. Effect of α on the robustness of the patched model under different types and severity levels of corrupted data. Green bars, brown bars, dark brown bars represent clean accuracy, robust accuracy under the corruption of severity level 3, and robust accuracy under the corruption of severity level 5, respectively.

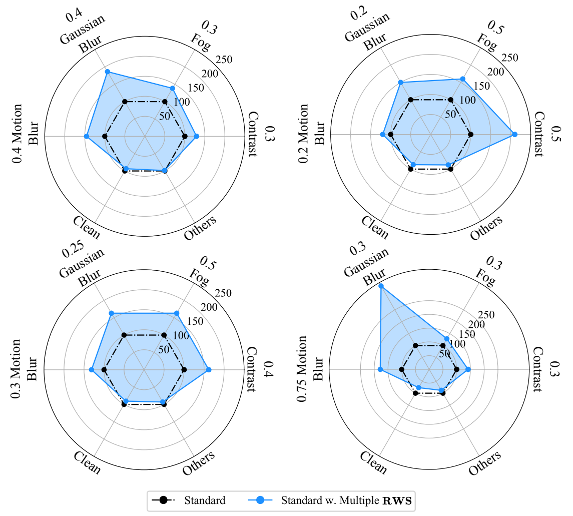

Figure 8. Visualization of robustness improvements when adding multiple RWSs together. On VGG-16 and Tiny-ImageNet, we add 4 different RWSs of "motion blur", "gaussian blur", "fog" and "contrast" together. By changing their coefficients, we can obtain robust models with different specialties, with minimal loss of clean accuracy and robustness on other corruption types.

weights between the standard and robust models.

To validate, we test the patched model on corrupted data with different severity levels (as defined in (Hendrycks & Dietterich, 2019a)). As in Figure 7, for instance on CIFAR-10 and VGG-16, patched models always achieve the best performance at the severity level 5 (strongest corruptions) when $\alpha = 0.9$ or 1. When the severity level is set to 3, the patched model performs the best when $\alpha = 0.6$. Meanwhile, the standard accuracy gracefully decays as α increases.

3.3. Composable

Usually, images do not just suffer from a single type of corruption. To resist compound natural corruptions, RWSs can also be linearly composed to form a model of multi-corruption robustness, by extending Equation 2 to adding multiple RWSs, each with their own αs. The previous in-situ adjustment could also be seen as a special case. We can even control the linear combination coefficient to obtain models with different "robustness specialties". Figure 8 shows that by composing RWSs with different coefficients, one can construct a wide range of models with different strengths at simultaneously tackling diverse corruptions. That leads us to an "infinite pool" of possible models, by just re-composing a small pool of RWSs and no re-training.

The composable property of RWSs reminds the weight interpolation between two different models (Izmailov et al., 2018; Zhao et al., 2020; Ilharco et al., 2022b; Wortsman et al., 2022a;b; Choshen et al., 2022), yet composing natural corruption robustness seems a new theme. Note that though, the construction of RWSs needs to first remove the robust weight's projection onto the standard weight column space,

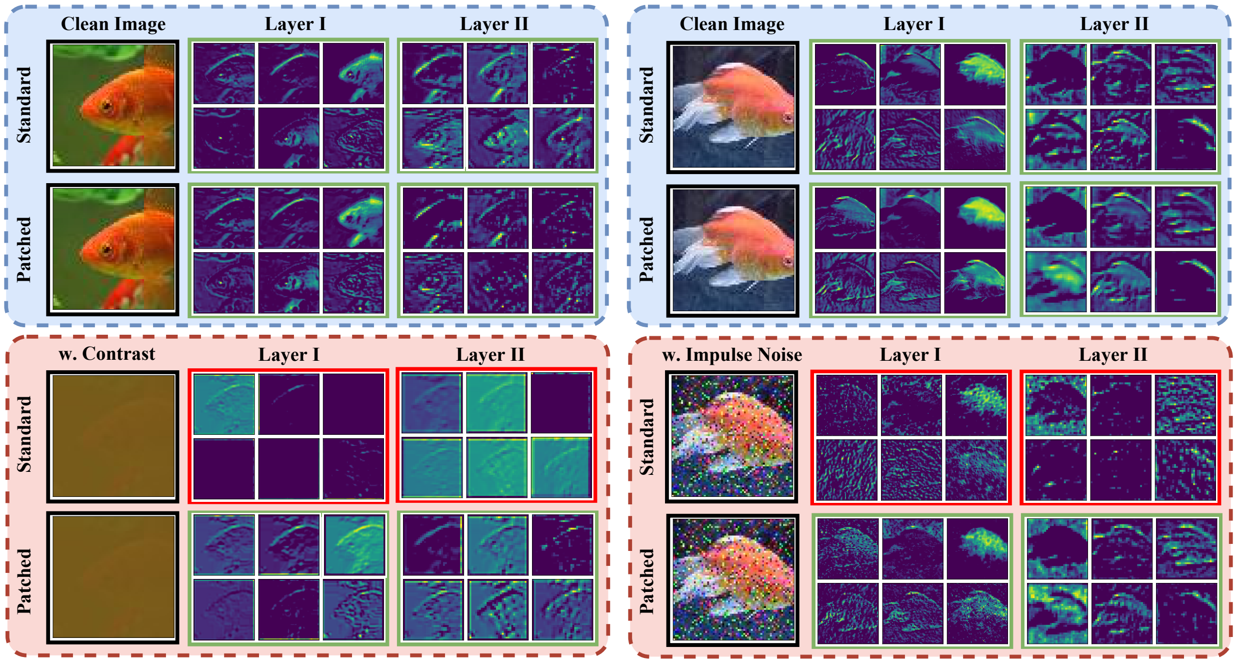

Figure 9. The comparison of the patched model and the standard model's feature maps given the same input sample shows our RWS patching method meaningfully equips the model with resistance to different types of corruptions.

Table 2. Robustness gains when applying RWSs extracted from small datasets (CIFAR-10, CIFAR-100, Tiny-ImageNet) to full ImageNet models. Robust accuracy is evaluated on ImageNet and averaged on all kinds of corrupted data. All models use the VGG-16 architecture. $\mathrm{DataAugmentation}^1$ use the same FLOPS as the overhead of extracting RWS based on CIFAR-10 models, while $\mathrm{DataAugmentation}^2$ use the same FLOPS as the RWS extraction based on TinyImageNet models.

Method	Robust Accuracy (%)
Standard	11.14
$\mathrm{DataAugmentation}^1$	11.01 (↓ 0.13)
$\mathrm{DataAugmentation}^2$	14.52 (↑ 3.38)
Standard + $\mathbf{RWS}_{\mathrm{CIFAR-10}}$	13.47 (↑ 2.33)
Standard + $\mathbf{RWS}_{\mathrm{CIFAR-100}}$	14.55 (↑ 3.41)
Standard + $\mathbf{RWS}_{\mathrm{Tiny-Imagenet}}$	**17.53** (↑ **6.39**)

hence composing RWSs does not naively equal interpolating their source robust model weights.

3.4. Transferable

Lastly, the cross-dataset consistency of RWSs as analyzed in Sec 2.4 motivates us to study if an RWS found from one dataset can be reused for the same architecture trained on a different dataset, to transfer robustness to the latter "for free". Table 3 confirms this possibility. Despite the training data domain shift of the standard model, RWSs stay effective for patching robustness in a "zero-shot" manner. Unsurprisingly also, smaller gaps will render the RWS transfer more effective. For example, the robustness gain of CIFAR-100 by patching CIFAR-10 RWSs is clearly larger than the Tiny-ImageNet gain by patching the same.

In addition, the strong transferability implies the tantalizing possibility of gaining robustness efficiently by "transferring" RWSs from small to large datasets. Specifically, direct robust training with data augmentations on large datasets such as ImageNet can be resource-demanding. Instead, one can first extract RWSs by robust-training over smaller datasets (e.g. CIFAR-10 or TinyImageNet), and subsequently, transfer them to "patching" the same model architecture standard-trained on the target large dataset. The results, as presented in Table 2, demonstrate that the "out of the box" application of RWS can lead to significant gains in ImageNet robustness, especially when RWS is obtained from TinyImageNet (whose distribution is the most similar to ImageNet).

Table 3. Transferring RWSs across datasets. For three non-robust models trained on CIFAR-10, CIFAR-100 and Tiny-ImageNet (by columns), we patch RWSs generated from CIFAR-10, CIFAR-100, and Tiny-ImageNet (by rows), respectively, for injecting zero-shot robustness. The robust accuracies are averaged across 19 corruptions/19 RWSs. We use the VGG-16 model here.

Methods	Robust Accuracy (%)		
	CIFAR-10	CIFAR-100	Tiny-ImageNet
Standard	65.44	36.76	23.58
Standard + $\mathbf{RWS}_{\mathrm{CIFAR-10}}$	84.76 (↑ 19.32)	55.65 (↑ 18.89)	32.23 (↑ 8.65)
Standard + $\mathbf{RWS}_{\mathrm{CIFAR-100}}$	83.01 (↑ 17.57)	58.62 (↑ 21.86)	33.17 (↑ 9.59)
Standard + $\mathbf{RWS}_{\mathrm{Tiny-ImageNet}}$	71.43 (↑ 5.99)	44.94 (↑ 8.18)	44.25 (↑ 20.67)

3.5. Feature Map Visualization

Besides, we compare feature maps of standard and patched models in Figure 9, to understand what information is actually patched. Using TinyImageNet and VGG-16, we visualize feature maps after the second and third convolutional layers (denoted as "Layer I" and "Layer II", respectively). The visualizations show that RWSs bring in meaningful feature adjustments to be resilient to corruption types. For example, to tackle reduced contrast, the patched model becomes more sensitive to edges, while the model patched for impulse noise picks up less high-frequency outlier features.

4. Conclusion and Limitations

Our work is dedicated to investigating how natural corruption "robustness" is encoded in weights and how to disentangle/transfer them. We introduce *"Robust Weight Signature"*(**RWS**), which nontrivially generalizes the prior wisdom in model weight interpolation and arithmetic, to analyzing standard/robust models, with both methodological innovations and new key findings. RWSs lead to a powerful in-situ model patching framework to easily achieve on-demand robustness towards a wide range of corruptions.

Current RWS patching faces one limitation that we must point out: in Table 1, the superior TA/RA trade-offs achieved by RWS methods are based on the perfect "oracle" knowledge: (1) the type of corruption is being handled, i.e., when to add or take off the "correct" RWSs. (2) the corruption severity is being handled, i.e., the choice of hyperparameter α when applying RWSs. This assumption is in line with the "once-for-all" AT methods (Wang et al., 2020; Kundu et al., 2023), which requires a human oracle to control a test-time hyperparameter to implicitly state the desired RA-TA trade-offs in contexts. Practically, that can be implemented by referring to environment sensors or other domain classification change or detection methods. We also ensure our fair comparison with "All Models" baseline in Table 1 by using the same ideal oracle.

One future work of immediate interest would be to examine RWS patching under a practical imperfect oracle (e.g., a trained corruption domain classifier that might predict incorrectly, hence applying inexact RWSs). We hypothesize the overall performance drop will be mild though, since an RWS trained for one corruption type can boost robustness against other "similar" corruptions too (see Sec. 2.3).

References

Askell, A., Bai, Y., Chen, A., Drain, D., Ganguli, D., Henighan, T., Jones, A., Joseph, N., Mann, B., DasSarma, N., et al. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*, 2021.

Chen, T., Liu, S., Chang, S., Cheng, Y., Amini, L., and Wang, Z. Adversarial robustness: From self-supervised pre-training to fine-tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 699–708, 2020.

Choshen, L., Venezian, E., Slonim, N., and Katz, Y. Fusing finetuned models for better pretraining. *arXiv preprint arXiv:2204.03044*, 2022.

Croce, F., Gowal, S., Brunner, T., Shelhamer, E., Hein, M., and Cemgil, T. Evaluating the adversarial robustness of adaptive test-time defenses. *arXiv preprint arXiv:2202.13711*, 2022.

Elsayed, G. F., Goodfellow, I., and Sohl-Dickstein, J. Adversarial reprogramming of neural networks. *arXiv preprint arXiv:1806.11146*, 2018.

Fleuret, F. et al. Test time adaptation through perturbation robustness. In *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2021.

Geva, M., Caciularu, A., Dar, G., Roit, P., Sadde, S., Shlain, M., Tamir, B., and Goldberg, Y. Lm-debugger: An interactive tool for inspection and intervention in transformer-based language models. *arXiv preprint arXiv:2204.12130*, 2022.

Goel, K., Gu, A., Li, Y., and Ré, C. Model patching: Closing the subgroup performance gap with data augmentation. *arXiv preprint arXiv:2008.06775*, 2020.

Goldblum, M., Tsipras, D., Xie, C., Chen, X., Schwarzschild, A., Song, D., Madry, A., Li, B., and Goldstein, T. Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

Hambardzumyan, K., Khachatrian, H., and May, J. Warp: Word-level adversarial reprogramming. *arXiv preprint arXiv:2101.00121*, 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.

Hein, M. and Andriushchenko, M. Formal guarantees on the robustness of a classifier against adversarial manipulation. *Advances in neural information processing systems*, 30, 2017.

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

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

Hendrycks, D., Lee, K., and Mazeika, M. Using pre-training can improve model robustness and uncertainty. In *International Conference on Machine Learning*, pp. 2712–2721. PMLR, 2019a.

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 *International Conference on Learning Representations*, 2019b.

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, 2021.

Huang, H., Wang, Y., Erfani, S., Gu, Q., Bailey, J., and Ma, X. Exploring architectural ingredients of adversarially robust deep neural networks. *Advances in Neural Information Processing Systems*, 34:5545–5559, 2021.

Ilharco, G., Ribeiro, M. T., Wortsman, M., Gururangan, S., Schmidt, L., Hajishirzi, H., and Farhadi, A. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022a.

Ilharco, G., Wortsman, M., Gadre, S. Y., Song, S., Hajishirzi, H., Kornblith, S., Farhadi, A., and Schmidt, L. Patching open-vocabulary models by interpolating weights. *arXiv preprint arXiv:2208.05592*, 2022b.

Ilyas, A., Park, S. M., Engstrom, L., Leclerc, G., and Madry, A. Datamodels: Understanding predictions with data and data with predictions. In *International Conference on Machine Learning*, pp. 9525–9587. PMLR, 2022.

Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., and Wilson, A. G. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

Jiang, Z., Chen, T., Chen, T., and Wang, Z. Robust pre-training by adversarial contrastive learning. *Advances in Neural Information Processing Systems*, 33:16199–16210, 2020.

Kasirzadeh, A. and Gabriel, I. In conversation with artificial intelligence: aligning language models with human values. *arXiv preprint arXiv:2209.00731*, 2022.

Koh, P. W., Sagawa, S., Marklund, H., Xie, S. M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., Phillips, R. L., Gao, I., et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pp. 5637–5664. PMLR, 2021.

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

Kundu, S., Sundaresan, S., Pedram, M., and Beerel, P. A. Float: Fast learnable once-for-all adversarial training for tunable trade-off between accuracy and robustness. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2349–2358, 2023.

Li, M., Gururangan, S., Dettmers, T., Lewis, M., Althoff, T., Smith, N. A., and Zettlemoyer, L. Branch-train-merge: Embarrassingly parallel training of expert language models. *arXiv preprint arXiv:2208.03306*, 2022.

Liu, X., Cheng, M., Zhang, H., and Hsieh, C.-J. Towards robust neural networks via random self-ensemble. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 369–385, 2018.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

Mitchell, E., Lin, C., Bosselut, A., Finn, C., and Manning, C. D. Fast model editing at scale. *arXiv preprint arXiv:2110.11309*, 2021.

Mitchell, E., Lin, C., Bosselut, A., Manning, C. D., and Finn, C. Memory-based model editing at scale. In *International Conference on Machine Learning*, pp. 15817–15831. PMLR, 2022.

mnmoustafa, M. A. Tiny imagenet, 2017. URL https://kaggle.com/competitions/tiny-imagenet.

Murty, S., Manning, C. D., Lundberg, S., and Ribeiro, M. T. Fixing model bugs with natural language patches. *arXiv preprint arXiv:2211.03318*, 2022.

Nguyen, T., Ilharco, G., Wortsman, M., Oh, S., and Schmidt, L. Quality not quantity: On the interaction between dataset design and robustness of clip. *arXiv preprint arXiv:2208.05516*, 2022.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022.

Ribeiro, M. T. and Lundberg, S. Adaptive testing and debugging of nlp models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3253–3267, 2022.

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 *European Conference on Computer Vision*, pp. 53–69. Springer, 2020.

Santurkar, S., Tsipras, D., Elango, M., Bau, D., Torralba, A., and Madry, A. Editing a classifier by rewriting its prediction rules. *Advances in Neural Information Processing Systems*, 34:23359–23373, 2021.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Subramani, N., Suresh, N., and Peters, M. E. Extracting latent steering vectors from pretrained language models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 566–581, 2022.

Sun, J., Cao, Y., Choy, C., Yu, Z., Xiao, C., Anandkumar, A., and Mao, Z. M. Improving adversarial robustness in 3d point cloud classification via self-supervisions. In *International Conference on Machine Learning Workshop (ICMLW)*, volume 1, 2021.

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.

Tsai, Y.-Y., Chen, P.-Y., and Ho, T.-Y. Transfer learning without knowing: Reprogramming black-box machine learning models with scarce data and limited resources. In *International Conference on Machine Learning*, pp. 9614–9624. PMLR, 2020.

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, number 2019, 2019.

Wang, H., Chen, T., Gui, S., Hu, T., Liu, J., and Wang, Z. Once-for-all adversarial training: In-situ tradeoff between robustness and accuracy for free. *Advances in Neural Information Processing Systems*, 33:7449–7461, 2020.

Wang, H., Xiao, C., Kossaifi, J., Yu, Z., Anandkumar, A., and Wang, Z. Augmax: Adversarial composition of random augmentations for robust training. *Advances in neural information processing systems*, 34:237–250, 2021.

Wortsman, M., Ilharco, G., Gadre, S. Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A. S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S., et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning*, pp. 23965–23998. PMLR, 2022a.

Wortsman, M., Ilharco, G., Kim, J. W., Li, M., Kornblith, S., Roelofs, R., Lopes, R. G., Hajishirzi, H., Farhadi, A., Namkoong, H., et al. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7959–7971, 2022b.

Wu, H., Judd, P., Zhang, X., Isaev, M., and Micikevicius, P. Integer quantization for deep learning inference: Principles and empirical evaluation. *arXiv preprint arXiv:2004.09602*, 2020.

Yin, D., Gontijo Lopes, R., Shlens, J., Cubuk, E. D., and Gilmer, J. A fourier perspective on model robustness in computer vision. *Advances in Neural Information Processing Systems*, 32, 2019.

Zhang, G., Zhang, Y., Zhang, Y., Fan, W., Li, Q., Liu, S., and Chang, S. Fairness reprogramming. *arXiv preprint arXiv:2209.10222*, 2022.

Zhang, H., Yu, Y., Jiao, J., Xing, E., El Ghaoui, L., and Jordan, M. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, pp. 7472–7482. PMLR, 2019.

Zhao, P., Chen, P.-Y., Das, P., Ramamurthy, K. N., and Lin, X. Bridging mode connectivity in loss landscapes and adversarial robustness. *arXiv preprint arXiv:2005.00060*, 2020.

Zheng, S., Song, Y., Leung, T., and Goodfellow, I. Improving the robustness of deep neural networks via stability training. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pp. 4480–4488, 2016.

A. More Experimental Results

A.1. Detailed Results on All 19 Corruption Types

To supplement Table 1, we provide the detailed experimental results of all corruption types in Table 4, which shows our consistent improvements in the RAs of all corruption types. We use **RWS**$_{\text{shallow,16bit}}$ for model patching: RWS are constructed by the shallowest five layers with 16-bit quantization.

Table 4. Detailed experimental results showing robustness improvements on all 19 corruption types in (Hendrycks & Dietterich, 2019b).

Corruptions	CIFAR-10			CIFAR-100			Tiny-ImageNet					
	VGG-16			VGG-16			VGG-16			ResNet-50		
	w.o. RWS (%)	w. RWS (%)	Diff.	w.o. RWS (%)	w. RWS (%)	Diff.	w.o. RWS (%)	w. RWS (%)	Diff.	w.o. RWS (%)	w. RWS (%)	Diff.
Brightness	88.41	90.87	↑ 2.46	60.85	67.06	↑ 6.21	30.33	49.41	↑ 19.08	37.96	56.61	↑ 18.65
Contrast	41.29	78.77	↑ 37.48	16.11	60.72	↑ 44.61	1.88	19.30	↑ 17.42	1.88	28.55	↑ 26.67
Defocus Blur	66.92	87.14	↑ 20.22	37.55	61.51	↑ 23.96	7.01	34.91	↑ 27.90	26.09	51.03	↑ 24.94
elastic Transform	78.38	81.93	↑ 3.55	49.74	56.01	↑ 6.27	35.06	48.15	↑ 13.09	41.60	56.12	↑ 14.52
Fog	61.40	80.57	↑ 19.17	30.18	62.33	↑ 32.15	27.59	53.51	↑ 25.92	20.51	56.23	↑ 35.72
Frost	70.65	84.89	↑ 14.24	40.73	56.80	↑ 16.07	38.40	51.22	↑ 12.82	42.23	56.47	↑ 14.24
Gaussian Blur	56.63	86.71	↑ 30.08	30.80	60.91	↑ 30.11	8.82	40.75	↑ 31.93	29.28	54.07	↑ 24.79
Gaussian Noise	50.92	82.08	↑ 31.16	24.79	52.83	↑ 28.04	12.37	42.38	↑ 30.01	15.29	51.86	↑ 36.57
Glass Blur	56.36	78.55	↑ 22.19	25.54	48.48	↑ 22.94	5.92	20.89	↑ 14.97	15.39	39.11	↑ 23.72
Impulse Noise	39.82	77.24	↑ 37.42	12.01	49.35	↑ 37.34	6.36	37.77	↑ 31.41	7.98	46.22	↑ 38.24
Jpeg Compression	80.72	84.26	↑ 3.54	51.18	53.71	↑ 2.53	51.23	54.16	↑ 2.93	57.80	60.47	↑ 2.67
Motion Blur	68.98	86.88	↑ 17.90	40.88	61.42	↑ 20.54	23.63	47.35	↑ 23.72	34.59	57.58	↑ 22.99
Pixelate	63.92	87.24	↑ 23.32	36.09	62.05	↑ 25.96	48.84	53.41	↑ 4.57	53.48	60.51	↑ 7.03
Saturate	85.27	91.30	↑ 6.03	53.55	66.47	↑ 12.92	23.97	46.37	↑ 22.40	29.70	52.73	↑ 23.03
Shot Noise	54.38	84.97	↑ 30.59	27.57	54.81	↑ 27.24	16.17	46.38	↑ 30.21	18.74	53.98	↑ 35.24
Snow	78.42	87.49	↑ 9.07	48.18	59.75	↑ 11.57	34.92	50.94	↑ 16.02	37.98	56.06	↑ 18.08
Spatter	74.36	88.18	↑ 13.82	41.22	62.98	↑ 21.76	40.99	55.31	↑ 14.32	46.07	57.04	↑ 10.97
Speckle Noise	55.86	85.26	↑ 29.40	27.81	54.82	↑ 27.01	18.31	47.48	↑ 29.17	20.28	54.39	↑ 34.11
Zoom Blur	70.63	86.02	↑ 15.39	43.60	61.81	↑ 18.21	16.20	41.13	↑ 24.93	26.45	54.34	↑ 27.89
Average	65.44	84.76	↑ 19.32	36.76	58.62	↑ 21.86	23.58	44.25	↑ 20.67	29.65	52.81	↑ 23.16

A.2. Compressibility: Full models versus RWSs under Quantization

As is shown in Table 5, full model weights are less compressible compared to RWSs, which suggests RWS-based methods to more easily achieve better storage efficiency. When applying the linear 16-bit quantization (Wu et al., 2020) to the full model weights, both the standard accuracy and natural corruption robustness have already degraded heavily. This is in stark contrast to RWSs which can retain most of their performance under the same quantization (16-bit) or even heavier (8-bit).

Note that we focus our study to provide the proof of concept that "RWSs are more easily amendable to quantization". We do not exclude the possibility that more sophisticated, robustness-aware quantization methods will sustain the robustness performance under heavy quantization, but further testing or developing such algorithms is out of this paper's scope.

Table 5. The standard and robust accuracy changes when applying different levels of quantization, showing the superior compressiblity of RWS-based methods.

Methods	CIFAR-10			CIFAR-100			Tiny-ImageNet					
	VGG-16			VGG-16			VGG-16			ResNet-50		
	N_{param} (MB)	TA (%)	RA (%)	N_{param} (MB)	TA (%)	RA (%)	N_{param} (MB)	TA (%)	RA (%)	N_{param} (MB)	TA (%)	RA (%)
Standard (32 bit)	58.8	92.59	65.44	58.8	71.44	36.76	59.2	61.28	23.58	95.7	65.72	29.65
Standard (16 bit)	29.4 (0.5×)	88.05	58.12	29.4 (0.5×)	53.60	21.17	29.6 (0.5×)	51.66	16.87	47.9 (0.5×)	40.27	20.26
Data Augmentation (32 bit)	58.8 (1×)	89.58	84.34	58.8 (1×)	67.34	56.95	59.2 (1×)	52.11	43.64	95.7 (1×)	59.17	47.96
Data Augmentation (16 bit)	29.4 (0.5×)	83.18	74.93	29.4 (0.5×)	60.58	51.19	29.6 (0.5×)	46.66	35.01	47.9 (0.5×)	39.19	26.03
All Models (32 bit)	1177.6 (20×)	92.59	88.97	1177.6 (20×)	71.44	64.72	1184.0 (20×)	61.28	51.55	1913.6 (20×)	65.72	55.97
All Models (16 bit)	588.8 (10×)	88.05	82.05	588.8 (10×)	53.60	52.96	592.0 (10×)	51.66	36.72	956.8 (10×)	40.27	23.00
Standard+**RWS**$_{\text{Full}}$	1177.6 (20×)	92.59	75.35	1177.6 (20×)	71.44	52.58	1184.0 (20×)	61.28	43.63	1913.6 (20×)	65.72	53.64
Standard+**RWS**$_{\text{Shallow}}$	101.0 (1.7×)	92.59	84.86	101.0 (1.7×)	71.44	58.78	101.4 (1.7×)	61.28	44.66	131.4 (1.4×)	65.72	52.84
Standard+**RWS**$_{\text{Shallow,16bits}}$	79.9 (1.4×)	92.59	84.76	79.9 (1.4×)	71.44	58.62	80.3 (1.4×)	61.28	44.25	113.6 (1.2×)	65.72	52.81
Standard+**RWS**$_{\text{Shallow,8bits}}$	69.4 (1.2×)	92.59	82.99	69.4 (1.2×)	71.44	53.52	69.7 (1.2×)	61.28	39.40	104.7 (1.1×)	65.72	52.79