

---

# The Pitfalls of Simplicity Bias in Neural Networks

---

**Harshay Shah**

Microsoft Research

harshay.rshah@gmail.com

**Kaustav Tamuly**

Microsoft Research

ktamuly2@gmail.com

**Aditi Raghunathan**

Stanford University

aditir@stanford.edu

**Prateek Jain**

Microsoft Research

prajain@microsoft.com

**Praneeth Netrapalli**

Microsoft Research

praneeth@microsoft.com

## Abstract

Several works have proposed Simplicity Bias (SB)—the tendency of standard training procedures such as Stochastic Gradient Descent (SGD) to find simple models—to justify why neural networks generalize well [1, 49, 74]. However, the precise notion of simplicity remains vague. Furthermore, previous settings [67, 24] that use SB to justify why neural networks generalize well do not simultaneously capture the non-robustness of neural networks—a widely observed phenomenon in practice [71, 36]. We attempt to reconcile SB and the superior standard generalization of neural networks with the non-robustness observed in practice by designing datasets that (a) incorporate a precise notion of simplicity, (b) comprise multiple predictive features with varying levels of simplicity, and (c) capture the non-robustness of neural networks trained on real data. Through theoretical analysis and targeted experiments on these datasets, we make four observations:

- (i) SB of SGD and variants can be extreme: neural networks can exclusively rely on the simplest feature and remain invariant to all predictive complex features.
- (ii) The extreme aspect of SB *could* explain why seemingly benign distribution shifts and small adversarial perturbations significantly degrade model performance.
- (iii) Contrary to conventional wisdom, SB can also hurt generalization on the same data distribution, as SB persists even when the simplest feature has less predictive power than the more complex features.
- (iv) Common approaches to improve generalization and robustness—ensembles and adversarial training—can fail in mitigating SB and its pitfalls.

Given the role of SB in training neural networks, we hope that the proposed datasets and methods serve as an effective testbed to evaluate novel algorithmic approaches aimed at avoiding the pitfalls of SB.

## 1 Introduction

Understanding the superior generalization ability of neural networks, despite their high capacity to fit randomly labeled data [84], has been a subject of intense study. One line of recent work [67, 24] proves that linear neural networks trained with Stochastic Gradient Descent (SGD) on linearly separable data converge to the maximum-margin linear classifier, thereby explaining the superior generalization performance. However, maximum-margin classifiers are inherently robust to perturbations of data at prediction time, and this implication is at odds with concrete evidence that neural networks, in practice, are brittle to adversarial examples [71] and distribution shifts [52, 58, 44, 65]. Hence, the linear setting, while convenient to analyze, is insufficient to capture the non-robustness of neural networks trained on real datasets. Going beyond the linear setting, several works [1, 49, 74] argue that neural networks generalize well because standard training procedures have a bias towards learning simple models. However, the exact notion of “simple” models remains vague and only intuitive. Moreover, the settings studied are insufficient to capture the brittleness of neural networks

Our goal is to formally understand and probe the *simplicity bias* (SB) of neural networks in a setting that is rich enough to capture known failure modes of neural networks and, at the same time, amenable to theoretical analysis and targeted experiments. Our starting point is the observation that on real-world datasets, there are several distinct ways to discriminate between labels (e.g., by inferring shape, color etc. in image classification) that are (a) predictive of the label to varying extents, and (b) define decision boundaries of varying complexity. For example, in the image classification task of white swans vs. bears, a linear-like “simple” classifier that only looks at color could predict correctly on most instances except white polar bears, while a nonlinear “complex” classifier that infers shape could have almost perfect predictive power. To systematically understand SB, we design modular synthetic and image-based datasets wherein different coordinates (or blocks) define decision boundaries of varying complexity. We refer to each coordinate / block as a *feature* and define a precise notion of feature *simplicity* based on the *simplicity of the corresponding decision boundary*.

**Proposed dataset.** Figure 1 illustrates a stylized version of the proposed synthetic dataset with two features,  $\phi_1$  and  $\phi_2$ , that can perfectly predict the label with 100% accuracy, but differ in simplicity. The simplicity of a feature is precisely determined by the *minimum* number of linear pieces in the decision boundary that achieves optimal classification accuracy using that feature. For example, in Figure 1, the simple feature  $\phi_1$  requires a linear decision boundary to perfectly predict the label, whereas complex feature  $\phi_2$  requires four linear pieces. Along similar lines, we also introduce a collection of image-based datasets in which each image concatenates MNIST images (simple feature) and CIFAR-10 images (complex feature). The proposed datasets, which incorporate features of varying predictive power and simplicity, allow us to systematically investigate and measure SB in SGD-trained neural networks.

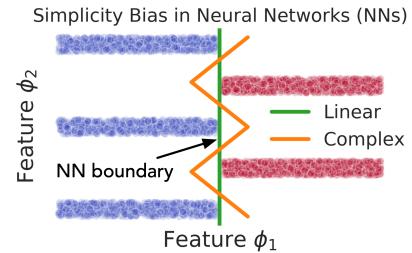


Figure 1: Simple vs. complex features

**Observations from new dataset.** The ideal decision boundary that achieves high accuracy *and* robustness relies on all features to obtain a large margin (minimum distance from any point to decision boundary). For example, the orange decision boundary in Figure 1 that learns  $\phi_1$  and  $\phi_2$  attains 100% accuracy and exhibits more robustness than the linear boundary because of larger margin. Given the expressive power of large neural networks, one might expect that a network trained on the dataset in Figure 1 would result in the larger-margin orange piecewise linear boundary. However, in practice, we find quite the opposite—trained neural networks have a linear boundary. Surprisingly, neural networks exclusively use feature  $\phi_1$  and remain *completely invariant* to  $\phi_2$ . More generally, we observe that SB is extreme: neural networks simply ignore several complex predictive features in the presence of few simple predictive features. We first theoretically show that one-hidden-layer neural networks trained on the piecewise linear dataset exhibit SB. Then, through controlled experiments, we validate the extreme nature of SB across model architectures and optimizers.

**Implications of extreme SB.** Theoretical analysis and controlled experiments reveal three major pitfalls of SB in the context of proposed synthetic and image-based datasets, which we *conjecture* to hold more widely across datasets and domains:

- (i) Lack of robustness: Neural networks exclusively latch on to the simplest feature (e.g., background) at the expense of very small margin and completely ignore complex predictive features (e.g., semantics of the object), *even when all features have equal predictive power*. This results in susceptibility to small adversarial perturbations (due to small margin) and spurious correlations (with simple features). Furthermore, in Section 4, we provide a concrete connection between SB and data-agnostic and model-agnostic universal adversarial perturbations [47] observed in practice.
- (ii) Lack of reliable confidence estimates: Ideally, a network should have high confidence only if all predictive features agree in their prediction. However due to extreme SB, the network has high confidence even if several complex predictive features contradict the simple feature, mirroring the widely reported inaccurate and substantially higher confidence estimates reported in practice [51, 25].
- (iii) Suboptimal generalization: Surprisingly, neural networks exclusively rely on the simplest feature *even if it less predictive of the label than all complex features* in the synthetic datasets. Consequently, contrary to conventional wisdom, extreme SB can hurt robustness as well as generalization.

In contrast, prior works [8, 67, 24] only extol SB by considering settings where all predictive features are simple and hence do not reveal the pitfalls observed in real-world settings. While our results on

the pitfalls of SB are established in the context of the proposed datasets, the two design principles underlying these datasets—combining multiple features of varying simplicity & predictive power and capturing multiple failure modes of neural networks in practice—suggest that our conclusions *could* be justifiable more broadly.

**Summary.** This work makes two key contributions. First, we design datasets that offer a precise stratification of features based on simplicity and predictive power. Second, using the proposed datasets, we provide theoretical and empirical evidence that neural networks exhibit extreme SB, which we postulate as a unifying contributing factor underlying key failure modes of deep learning: poor out-of-distribution performance, adversarial vulnerability and suboptimal generalization. To the best of our knowledge, prior works only focus on the positive aspect of SB: the lack of overfitting in practice. Additionally, we find that standard approaches to improve generalization and robustness—ensembles and adversarial training—do not mitigate simplicity bias and its shortcomings on the proposed datasets. Given the important implications of SB, we hope that the datasets we introduce serve (a) as a useful testbed for devising better training procedures and (b) as a starting point to design more realistic datasets that are amenable to theoretical analysis and controlled experiments.

**Organization.** We discuss related work in Section 2. Section 3 describes the proposed datasets and metrics. In Section 4, we concretely establish the extreme nature of Simplicity Bias (SB) and its shortcomings through theory and empirics. Section 5 shows that extreme SB can in fact hurt generalization as well. We conclude and discuss the way forward in Section 6.

## 2 Related Work

**Out-of-Distribution (OOD) performance:** Several works demonstrate that NNs tend to learn spurious features & low-level statistical patterns rather than semantic features & high-level abstractions, resulting in poor OOD performance [36, 21, 45, 52]. This phenomenon has been exploited to design backdoor attacks against NNs [6, 12] as well. Recent works [77, 76] that encourage models to learn higher-level features improve OOD performance, but require domain-specific knowledge to penalize reliance on spurious features such as image texture [21] and annotation artifacts [26] in vision & language tasks. Learning robust representations without domain knowledge, however, necessitates formalizing the notion of features and feature reliance; our work takes a step in this direction.

**Adversarial robustness:** Neural networks exhibit vulnerability to small adversarial perturbations [71]. Standard approaches to mitigate this issue—adversarial training [23, 42] and ensembles [69, 54, 37]—have had limited success on large-scale datasets. Consequently, several works have investigated reasons underlying the existence of adversarial examples: [23] suggests local linearity of trained NNs, [61] indicates insufficient data, [62] suggests inevitability in high dimensions, [9] suggests computational barriers, [16] proposes limitations of neural network architectures, and [33] proposes the presence of non-robust features. Additionally, Jacobsen et al. [34] show that NNs exhibit invariance to large label-relevant perturbations. Prior works have also demonstrated the existence of *universal adversarial perturbations* (UAPs) that are agnostic to model and data [55, 47, 73].

**Implicit bias of stochastic gradient descent :** Brutzkus et al. [8] show that neural networks trained with SGD provably generalize on linearly separable data. Recent works [67, 35] also analyze the limiting direction of gradient descent on logistic regression with linearly separable and non-separable data respectively. Empirical findings [49, 43] provide further evidence to suggest that SGD-trained NNs generalize well because SGD learns models of increasing complexity. Additional recent works investigate the implicit bias of SGD on non-linearly separable data for linear classifiers [35] and infinite-width two-layer NNs [13], showing convergence to maximum margin classifiers in appropriate spaces. In Section 4, we show that SGD’s implicit bias towards simplicity can result in small-margin and feature-impovery classifiers instead of large-margin and feature-dense classifiers.

**Feature reliance:** Two recent works study the relation between inductive biases of training procedures and the set of features that models learn. Hermann et al. [31] use color, shape and texture features in stylized settings to show that standard training procedures can (a) increase reliance on task-relevant features that are partially decodable using untrained networks and (b) suppress reliance on non-discriminative or correlated features. Ortiz et al. [53] show that neural networks learned using standard training (a) develop invariance to non-discriminative features and (b) adversarial training induces a sharp transition in the models’ decision boundaries. In contrast, we develop a *precise* notion of feature simplicity and subsequently show that SGD-trained models can exhibit invariance to multiple *discriminative-but-complex* features. We also identify three pitfalls of this phenomenon—poor OOD performance, adversarial vulnerability, suboptimal generalization—and show that adversarial training and standard ensembles do not mitigate the pitfalls of simplicity bias.

	Accuracy	AUC	Logits
S-Randomized	0.50	0.50	randomly shuffled
$S^c$ -Randomized	standard accuracy	standard AUC	essentially identical

Table 1: If the  $\{S, S^c\}$ -randomized metrics of a model behave as above, then that model relies *exclusively* on  $S$  and is *invariant* to  $S^c$ .

Multiple works mentioned above (a) differentially characterize *learned* features and *desired* features—statistical regularities vs. high-level concepts [36], syntactic cues vs. semantic meaning [45], robust vs. non-robust features [33]—and (b) posit that the mismatch between these features results in non-robustness. In contrast, our work probes *why* neural networks prefer one set of features over another and unifies the aforementioned feature characterizations through the lens of feature *simplicity*.

### 3 Preliminaries: Setup and Metrics

**Setting and metrics:** We focus on binary classification. Given samples  $\hat{\mathcal{D}} = \{(x_i, y_i)\}_{i=1}^n$  from distribution  $\mathcal{D}$  over  $\mathbb{R}^d \times \{-1, 1\}$ , the goal is to learn a scoring function  $s(x) : \mathbb{R}^d \rightarrow \mathbb{R}$  (such as logits), and an associated classifier  $f : \mathbb{R}^d \rightarrow \{-1, 1\}$  defined as  $f(x) = 2h(x)-1$  where  $h(x) = \mathbb{1}\{\text{softmax}(s(x)) < 0.5\}$ . We use two well-studied metrics for generalization and robustness:

**Standard accuracy.** The standard accuracy of a classifier  $f$  is:  $\mathbb{E}_{\mathcal{D}} [\mathbb{1}\{f(x) = y\}]$ .

**$\delta$ -Robust accuracy.** Given norm  $\|\cdot\|$  and perturbation budget  $\delta$ , the  $\delta$ -robust accuracy of a classifier  $f$  is:  $\mathbb{E}_{\mathcal{D}} [\min_{\|\hat{x}-x\| \leq \delta} \mathbb{1}\{f(\hat{x}) = y\}]$ .

Next, we introduce two metrics that quantitatively capture the extent to which a model relies on different input coordinates (or features). Let  $S$  denote some subset of coordinates  $[d]$  and  $\overline{\mathcal{D}}^S$  denote the  $S$ -randomized distribution, which is obtained as follows: given  $\mathcal{D}^S$ , the marginal distribution of  $S$ ,  $\overline{\mathcal{D}}^S$  independently samples  $((x^S, x^{S^c}), y) \sim \mathcal{D}$  and  $\bar{x}^S \sim \mathcal{D}^S$  and then outputs  $((\bar{x}^S, x^{S^c}), y)$ . In  $\overline{\mathcal{D}}^S$ , the coordinates in  $S$  are rendered independent of the label  $y$ . The two metrics are as follows.

**Definition 1** (Randomized accuracy). Given data distribution  $\mathcal{D}$ , and subset of coordinates  $S \subseteq [d]$ , the  $S$ -randomized accuracy of a classifier  $f$  is given by:  $\mathbb{E}_{\overline{\mathcal{D}}^S} [\mathbb{1}\{f(x) = y\}]$ .

**Definition 2** (Randomized AUC). Given data distribution  $\mathcal{D}$  and subset of coordinates  $S \subseteq [d]$ , the  $S$ -randomized AUC of classifier  $f$  equals the area under the precision-recall curve of distribution  $\overline{\mathcal{D}}^S$ .

Our experiments use  $\{S, S^c\}$ -randomized metrics—accuracy, AUC, logits—to establish that  $f$  depends *exclusively on some features*  $S$  and *remains invariant to the rest*  $S^c$ . First, if (a)  $S$ -randomized accuracy and AUC equal 0.5 and (b)  $S$ -randomized logit distribution is a random shuffling of the original distribution (i.e., logits in the original distribution are randomly shuffled across true positives and true negatives), then  $f$  depends *exclusively* on  $S$ . Conversely, if (a)  $S^c$ -randomized accuracy and AUC are equal to standard accuracy and AUC and (b)  $S^c$ -randomized logit distribution is essentially identical to the original distribution, then  $f$  is *invariant* to  $S^c$ ; Table 1 summarizes these observations.

#### 3.1 Datasets

**One-dimensional Building Blocks:** Our synthetic datasets use three one-dimensional data blocks—linear, noisy linear and  $k$ -slabs—shown in top row of Figure 2. In the **linear** block, positive and negative examples are uniformly distributed in  $[0.1, 1]$  and  $[-1, -0.1]$  respectively. In the **noisy linear** block, given a noise parameter  $p \in [0, 1]$ ,  $1-p$  fraction of points are distributed like the linear block described above and  $p$  fraction of the examples are uniformly distributed in  $[-0.1, 0.1]$ . In  **$k$ -slab** blocks, positive and negative examples are distributed in  $k$  well-separated, alternating regions.

**Simplicity of Building Blocks:** Linear classifiers can attain the optimal (Bayes) accuracy of 1 and  $1 - p/2$  on the linear and  $p$ -noisy linear blocks respectively. For  $k$ -slabs, however,  $(k-1)$ -piecewise linear classifiers are required to obtain the optimal accuracy of 1. Consequently, the building blocks have a natural notion of simplicity: *minimum number of pieces required by a piecewise linear classifier to attain optimal accuracy*. With this notion, the linear and noisy linear blocks are simpler than  $k$ -slab blocks when  $k > 2$ , and  $k$ -slab blocks are simpler than  $\ell$ -slab blocks when  $k < \ell$ .

**Multi-dimensional Synthetic Datasets:** We now outline four  $d$ -dimensional datasets wherein each coordinate corresponds to one of three building blocks described above. See Figure 2 for illustration.

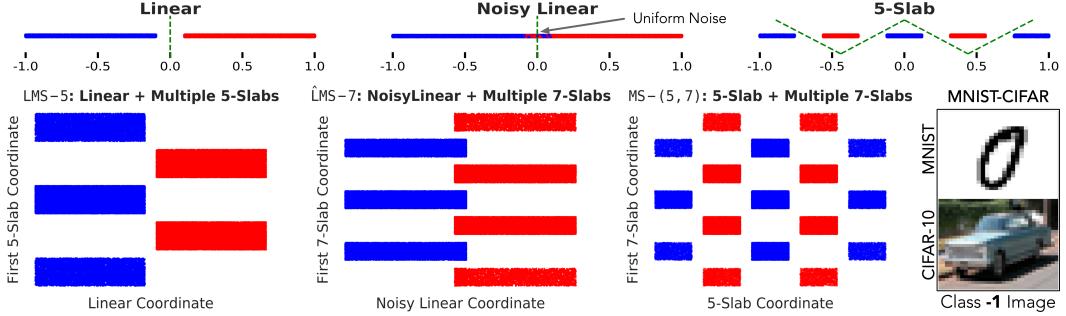


Figure 2: (Synthetic & Image-based Datasets) One-dimensional building blocks (top row)—linear, noisy linear,  $k$ -slab—are used to construct multi-dimensional datasets (bottom row): LMS-5 (linear & multiple 5-slabs),  $\hat{\text{LMS}}\text{-}5$  (noisy linear & multiple 5-slabs) and MS- $(5, 7)$  (5-slab & multiple 7-slabs). MNIST-CIFAR data vertically concatenates MNIST and CIFAR images (see Section 3.1).

- LMS-k: Linear and multiple  $k$ -slabs; the first coordinate is a linear block and the remaining  $d-1$  coordinates are independent  $k$ -slab blocks; we use LMS-5 & LMS-7 datasets in our analysis.
- $\hat{\text{LMS}}\text{-}k$ : Noisy linear and multiple  $k$ -slab blocks; the first coordinate is a *noisy* linear block and the remaining  $d-1$  coordinates are independent  $k$ -slab blocks. The noise parameter  $p$  is 0.1 by default.
- MS- $(5, 7)$ : 5-slab and multiple 7-slab blocks; the first coordinate is a 5-slab block and the remaining  $d-1$  coordinates are independent 7-slab blocks, as shown in Figure 2.
- MS-5: Multiple 5-slab blocks; all coordinates are independent 5-slab blocks.

We now describe the LSN (linear, 3-slab & noise) dataset, a stylized version of LMS- $k$  that is amenable to theoretical analysis. We note that recent works [53, 20] empirically analyze variants of the LSN dataset. In LSN, conditioned on the label  $y$ , the first and second coordinates of  $x$  are *singleton* linear and 3-slab blocks: linear and 3-slab blocks have support on  $\{-1, 1\}$  and  $\{-1, 0, 1\}$  respectively. The remaining coordinates are standard gaussians and not predictive of the label.

The synthetic datasets comprise features of varying simplicity; in LMS- $k$ ,  $\hat{\text{LMS}}\text{-}k$ , and MS- $(5, 7)$ , the first coordinate is the simplest feature and in MS-5, all features are equally simple. All datasets, even  $\hat{\text{LMS}}\text{-}k$ , can be *perfectly* classified via piecewise linear classifiers. Though the  $k$ -slab features are special cases of linear periodic functions on which gradient-based methods have been shown to fail for large  $k$  [64], we note that we use small values of  $k \in \{5, 7\}$  which are quickly learned by SGD in practice. Note that we (a) apply a random rotation matrix to the data and (b) use 50-dimensional synthetic data (i.e.,  $d = 50$ ) by default. Note that all code and datasets are available at the following repository: <https://github.com/harshays/simplicitybiaspitfalls>.

**MNIST-CIFAR Data:** The MNIST-CIFAR dataset consists of two classes: images in class -1 and class 1 are vertical concatenations of MNIST digit zero & CIFAR-10 automobile and MNIST digit one & CIFAR-10 truck images respectively, as shown in Figure 2. The training and test datasets comprise 50,000 and 10,000 images of size  $3 \times 64 \times 32$ . The MNIST-CIFAR dataset mirrors the structure in the synthetic LMS- $k$  dataset—both incorporate simple and complex features. The MNIST and CIFAR blocks correspond to the linear and  $k$ -slab blocks in LMS- $k$  respectively. Also note that MNIST images are zero-padded & replicated across three channels to match CIFAR dimensions before concatenation.

Appendix B provides details about the datasets, models, and optimizers used in our experiments. In Appendix C, we show that our results are robust to the exact choice of MNIST-CIFAR class pairs.

## 4 Simplicity Bias (SB) is Extreme and Leads to Non-Robustness

We first establish the *extreme* nature of SB in neural networks (NNs) on the proposed synthetic datasets using SGD and variants. In particular, we show that for the datasets considered, *if all features have full predictive power, NNs rely exclusively on the simplest feature S and remain invariant to all complex features S<sup>c</sup>*. Then, we explain why extreme SB on these datasets results in neural networks that are vulnerable to distribution shifts and data-agnostic & transferable adversarial perturbations.

### 4.1 Neural networks provably exhibit Simplicity Bias (SB)

We consider the LSN dataset (described in Section 3.1) that has one *linear* coordinate and one *3-slab* coordinate, both fully predictive of the label on their own; the remaining  $d-2$  noise coordinates do

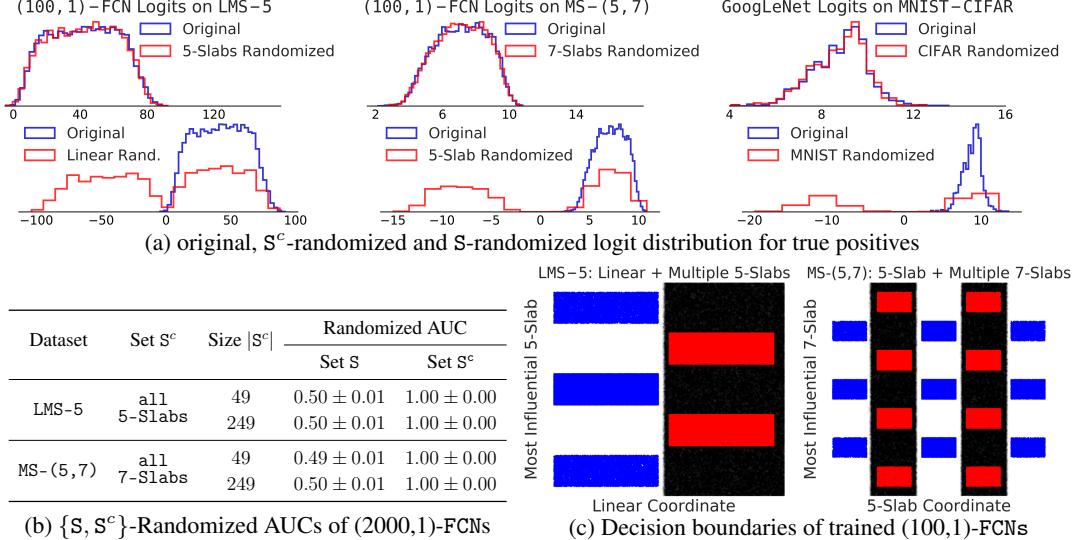


Figure 3: Extreme SB on LMS-5, MS-(5,7) and MNIST-CIFAR datasets (a) S-randomized logit distribution of true positives is essentially identical to the original logit distribution of true positives (before randomization). However,  $S^c$ -randomized logit distribution of true positives is a randomly shuffled version of original logit distribution;  $S^c$ -randomized logits are *shuffled across true positives and negatives*. (b)  $\{S, S^c\}$ -randomized AUCs (summarized in Table 1) are 0.5 and 1.0 respectively for varying number of complex features  $|S^c|$ . (c) FCN decision boundaries projected onto  $S$  & the most influential coordinate in  $S^c$  shows that the boundary depends only on  $S$  and is invariant to  $S^c$ .

not have any predictive power. Now, a "large-margin" one-hidden-layer NN with ReLU activation should give equal weight to the linear and 3-slab coordinates. However, we prove that NNs trained with standard mini-batch gradient descent (GD) on the LSN dataset (described in Section 3.1) provably learns a classifier that *exclusively* relies on the "simple" linear coordinate, thus exhibiting simplicity bias at the cost of margin. Further, our claim holds even when the margin in the linear coordinate (minimum distance between linear coordinate of positives and negatives) is significantly smaller than the margin in the slab coordinate. The proof of the following theorem is presented in Appendix F.

**Theorem 1.** Let  $f(x) = \sum_{j=1}^k v_j \cdot \text{ReLU}(\sum_{i=1}^d w_{i,j} x_i)$  denote a one-hidden-layer neural network with  $k$  hidden units and ReLU activations. Set  $v_j = \pm 1/\sqrt{k}$  w.p.  $1/2 \forall j \in [k]$ . Let  $\{(x^i, y^i)\}_{i=1}^m$  denote i.i.d. samples from LSN where  $m \in [cd^2, d^\alpha/c]$  for some  $\alpha > 2$ . Then, given  $d > \Omega(\sqrt{k} \log k)$  and initial  $w_{ij} \sim \mathcal{N}(0, \frac{1}{dk \log d})$ , after  $O(1)$  iterations, mini-batch gradient descent (over  $w$ ) with hinge loss, step size  $\eta = \Omega(\log d)^{-1/2}$ , mini-batch size  $\Theta(m)$ , satisfies:

- Test error is at most  $1/\text{poly}(d)$
- The learned weights of hidden units  $w_{ij}$  satisfy:

$$\underbrace{|w_{1j}| = \frac{2}{\sqrt{k}} \left(1 - \frac{c}{\sqrt{\log d}}\right) + O\left(\frac{1}{\sqrt{dk \log d}}\right)}_{\text{Linear Coordinate}}, \underbrace{|w_{2,j}| = O\left(\frac{1}{\sqrt{dk \log d}}\right)}_{\text{3-Slab Coordinate}}, \underbrace{\|w_{3:d,j}\| = O\left(\frac{1}{\sqrt{k \log d}}\right)}_{d-2 \text{ Noise Coordinates}}$$

with probability greater than  $1 - \frac{1}{\text{poly}(d)}$ . Note that  $c$  is a universal constant.

**Remarks:** First, we see that the trained model essentially relies only on the linear coordinate  $w_{1j}$ —SGD sets the value of  $w_{1j}$  roughly  $\tilde{\Omega}(\sqrt{d})$  larger than the slab coordinates  $w_{2j}$  that do not change much from their initial value. Second, the initialization we use is widely studied in the deep learning theory [46, 79] as it better reflects the practical performance of neural networks [14]. Third, given that LSN is linearly separable, Brutzkus et al. [8] also guarantee convergence of test error. However, our result additionally gives a precise description of the *final* classifier. Finally, we note that our result shows that extreme SB bias holds even for overparameterized networks ( $k = O(d^2 / \log d)$ ).

## 4.2 Simplicity Bias (SB) is Extreme in Practice

We now establish the extreme nature of SB on datasets with features of varying simplicity—LMS-5, MS-(5, 7), MNIST-CIFAR (described in Section 3)—across *multiple model architectures and optimizers*. Recall that (a) the simplicity of one-dimensional building blocks is defined as the number of pieces required by a piecewise linear classifier acting *only* on that block to get optimal accuracy and (b) LMS-5 has one linear block & multiple 5-slabs, MS-(5, 7) has one 5-slab and multiple 7-slabs and MNIST-CIFAR concatenates MNIST and CIFAR10 images. We now use  $S$  to denote the simplest feature in each dataset: linear in LMS-5, 5-slab in MS-(5, 7), and MNIST in MNIST-CIFAR.

We first consistently observe that SGD-trained models trained on LMS-5 and MS-(5, 7) datasets exhibit extreme SB: they *exclusively* rely on the simplest feature  $S$  and remain invariant to all complex features  $S^c$ . Using  $S$ -randomized &  $S^c$ -randomized metrics summarized in Table 1, we first establish extreme SB on fully-connected (FCN), convolutional (CNN) & sequential (GRU [15]) models. We observe that the  $S$ -randomized AUC is 0.5 across models. That is, unsurprisingly, all models are critically dependent on  $S$ . Surprisingly, however,  $S^c$ -randomized AUC of all models on both datasets equals 1.0. That is, arbitrarily perturbing  $S^c$  coordinates has *no impact* on the class predictions or the ranking of true positives’ logits against true negatives’ logits. One might expect that perturbing  $S^c$  would at least bring the *logits* of positives and negatives closer to each other. Figure 3(a) answers this in negative—the logit distributions over true positives of (100, 1)-FCNs (i.e., with width 100 & depth 1) remain unchanged even after randomizing *all* complex features  $S^c$ . Conversely, randomizing the simplest feature  $S$  randomly shuffles the original logits across true positives as well as true negatives. The two-dimensional projections of FCN decision boundaries in Figure 3(c) visually confirm that FCNs exclusively depend on the simpler coordinate  $S$  and are invariant to all complex features  $S^c$ .

Note that sample size and model architecture do not present any obstacles in learning complex features  $S^c$  to achieve 100% accuracy. In fact, if  $S$  is *removed* from the dataset, SGD-trained models with the same sample size indeed rely on  $S^c$  to attain 100% accuracy. Increasing the number of complex features does not mitigate extreme SB either. Figure 3(b) shows that even when there are 249 complex features and only one simple feature, (2000, 1)-FCNs exclusively rely on the simplest feature  $S$ ; randomizing  $S^c$  keeps AUC score of 1.0 intact but simply randomizing  $S$  drops the AUC score to 0.5. (2000, 1)-FCNs exhibit extreme SB despite their expressive power to learn large-margin classifiers that rely on all simple *and* complex features.

Similarly, on the MNIST-CIFAR dataset, MobileNetV2 [60], GoogLeNet [70], ResNet50 [27] and DenseNet121 [32] exhibit extreme SB. All models exclusively latch on to the simpler MNIST block to achieve 100% accuracy and remain invariant to the CIFAR block, even though the CIFAR block alone is almost fully predictive of its label—GoogLeNet attains 95.4% accuracy on the corresponding CIFAR binary classification task. Figure 3(a) shows that randomizing the simpler MNIST block randomly shuffles the logit distribution of true positives whereas randomizing the CIFAR block has no effect—the CIFAR-randomized and original logit distribution over true positives essentially overlap.

To summarize, we use  $S$ -randomized and  $S^c$ -randomized metrics to establish that models trained on synthetic and image-based datasets exhibit extreme SB: *If all features have full predictive power, NNs rely exclusively on the simplest feature  $S$  and remain invariant to all complex features  $S^c$ .* We further validate our results on extreme SB across model architectures, activation functions, optimizers and regularization methods such as  $\ell_2$  regularization and dropout in Appendix C.

## 4.3 Extreme Simplicity Bias (SB) leads to Non-Robustness

Now, we discuss how our findings about extreme SB in Section 4.2 can help reconcile poor OOD performance and adversarial vulnerability with superior generalization on the same data distribution.

**Poor OOD performance:** Given that neural networks tend to heavily rely on spurious features [45, 52], state-of-the-art accuracies on large and diverse validation sets provide a false sense of security; even benign distributional changes to the data (e.g., domain shifts) during prediction time can drastically degrade or even nullify model performance. This phenomenon, though counter-intuitive, can be easily explained through the lens of extreme SB. Specifically, we hypothesize that spurious features are *simple*. This hypothesis, when combined with extreme SB, explains the outsized impact of spurious features. For example, Figure 3(b) shows that simply perturbing the simplest (and potentially spurious in practice) feature  $S$  drops the AUC of trained neural networks to 0.5, thereby nullifying model performance. Randomizing *all* complex features  $S^c$ —5-slabs in LMS-5, 7-slabs in

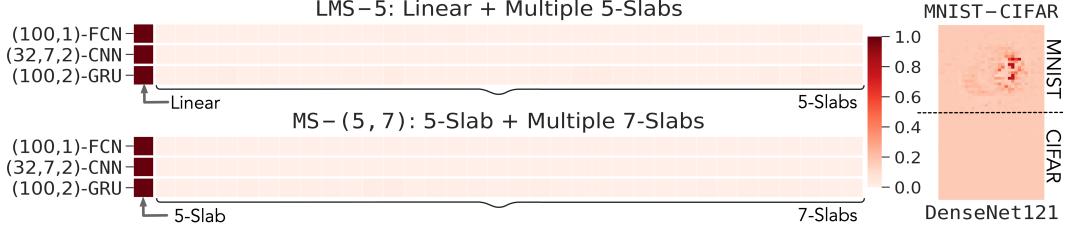


Figure 4: Extreme SB results in vulnerability to small-magnitude model-agnostic and data-agnostic Universal Adversarial Perturbations (UAPs) that nullify model performance by only perturbing the simplest feature  $S$ .  $\ell_2$  UAPs utilize most of the perturbation budget to attack  $S$  alone: 99.6% for linear in LMS-5, 99.9% for 5-slab in MS-(5,7) and 99.3% for MNIST pixels in MNIST-CIFAR. The UAPs in this figure are rescaled for visualization purposes.

MS-(5,7), CIFAR block in MNIST-CIFAR—has negligible effect on the trained neural networks— $S^c$ -randomized and original logits essentially overlap—even though  $S^c$  and  $S$  have equal predictive power. This further implies that approaches [29, 40] that aim to detect distribution shifts based on model outputs such as logits or softmax probabilities may themselves fail due to extreme SB.

**Adversarial Vulnerability:** Consider a classifier  $f^*$  that attains 100% accuracy on the LMS-5 dataset by taking an average of the linear classifier on the linear coordinate and  $d-1$  piecewise linear classifiers, one for every 5-slab coordinate. By relying on all  $d$  features,  $f^*$  has  $\mathcal{O}(\sqrt{d})$  margin. Now, given the large margin,  $f^*$  also attains high robust accuracy to  $\ell_2$  adversarial perturbations that have norm  $\mathcal{O}(\sqrt{d})$ —the perturbations need to attack at least  $\Omega(d)$  coordinates to flip model predictions. However, despite high model capacity, SGD-trained NNs do not learn robust and large-margin classifiers such as  $f^*$ . Instead, due to extreme SB, SGD-trained NNs exclusively rely on the simplest feature  $S$ . Consequently,  $\ell_2$  perturbations with norm  $\mathcal{O}(1)$  are enough to flip predictions and degrade model performance. We validate this hypothesis in Figure 4, where FCNs, CNNs and GRUs trained on LMS-5 and MS-(5,7) as well as DenseNet121 trained on MNIST-CIFAR are vulnerable to small universal (i.e., data-agnostic) adversarial perturbations (UAPs) of the simplest feature  $S$ . For example, Figure 4 shows that the  $\ell_2$  UAP of DenseNet121 on MNIST-CIFAR only attacks a few pixels in the simpler MNIST block and does not perturb the CIFAR block. Extreme SB also explains why data-agnostic UAPs of one model transfer well to another: the notion of simplicity is consistent across models; Figure 4 shows that FCNs, CNNs and GRUs trained on LMS-5 and MS-(5,7) essentially learn the same UAP. Furthermore, invariance to complex features  $S^c$  (e.g., CIFAR block in MNIST-CIFAR) due to extreme SB explains why “natural” [30] and semantic label-relevant perturbations [5] that modify the true image class do not alter model predictions.

To summarize, through theoretical analysis and extensive experiments on synthetic and image-based datasets, we (a) establish that SB is extreme in nature across model architectures and datasets and (b) show that extreme SB can result in poor OOD performance and adversarial vulnerability, *even when all simple and complex features have equal predictive power*.

## 5 Extreme Simplicity Bias (SB) can hurt Generalization

In this section, we show that, contrary to conventional wisdom, extreme SB can potentially result in suboptimal generalization of SGD-trained models on the same data distribution as well. This is because exclusive reliance on the simplest feature  $S$  can persist even when *every complex feature in  $S^c$  has significantly greater predictive power than  $S$* .

Accuracy	(100,1)-FCN	(200,1)-FCN	(300,1)-FCN	(100,2)-FCN	(200,2)-FCN	(300,2)-FCN
Training Data	$0.984 \pm 0.003$	$0.998 \pm 0.000$	$0.995 \pm 0.000$	$0.999 \pm 0.000$	$0.997 \pm 0.002$	$0.998 \pm 0.002$
Test Data	$0.940 \pm 0.002$	$0.949 \pm 0.003$	$0.948 \pm 0.002$	$0.945 \pm 0.003$	$0.946 \pm 0.003$	$0.947 \pm 0.002$
$S^c$ -Randomized	$0.941 \pm 0.001$	$0.946 \pm 0.001$	$0.946 \pm 0.001$	$0.944 \pm 0.001$	$0.945 \pm 0.001$	$0.946 \pm 0.001$
$S$ -Randomized	$0.498 \pm 0.001$	$0.498 \pm 0.000$	$0.497 \pm 0.001$	$0.498 \pm 0.001$	$0.497 \pm 0.000$	$0.498 \pm 0.001$

Table 2: Extreme SB can hurt generalization: FCNs of depth {1, 2} and width {100, 200, 300} trained on LMS-7 data with SGD attain 95% test accuracy. The randomized accuracies show that FCNs exclusively rely on the simpler noisy linear feature  $S$  and remain invariant to all 7-slab features that have 100% predictive power.

We verify this phenomenon on  $\hat{\text{LMS}}\text{-}7$  data defined in Section 3. Recall that  $\hat{\text{LMS}}\text{-}7$  has one noisy linear coordinate  $S$  with 95% predictive power (i.e., 10% noise in linear coordinate) and multiple 7-slab coordinates  $S^c$ , each with 100% predictive power. Note that our training sample size is large enough for FCNs of depth {1, 2} and width {100, 200, 300} trained on  $S^c$  *only* (i.e., after removing  $S$  from data) to attain 100% test accuracy. However, when trained on  $\hat{\text{LMS}}\text{-}7$  (i.e., including  $S$ ), SGD-trained FCNs exhibit extreme SB and *only* rely on  $S$ , the noisy linear coordinate. In Table 2, we report accuracies of SGD-trained FCNs that are selected based on validation accuracy after performing a grid search over four SGD hyperparameters: learning rate, batch size, momentum, and weight decay. The train, test and randomized accuracies in Table 2 collectively show that FCNs exclusively rely on the noisy linear feature and consequently attain 5% generalization error.

To summarize, the mere presence of a simple-but-noisy feature in  $\hat{\text{LMS}}\text{-}7$  data can significantly degrade the performance of SGD-trained FCNs due to extreme SB. Note that our results show that even an extensive grid search over SGD hyperparameters does not improve the performance of SGD-trained FCNs on  $\hat{\text{LMS}}\text{-}7$  data but does not necessarily imply that mitigating SB via SGD and its variants is impossible. We provide additional information about the experiment setup in Appendix D.

## 6 Conclusion and Discussion

We investigated Simplicity Bias (SB) in SGD-trained neural networks (NNs) using synthetic and image-based datasets that (a) incorporate a precise notion of feature simplicity, (b) are amenable to theoretical analysis and (c) capture the non-robustness of NNs observed in practice. We first showed that one-hidden-layer ReLU NNs provably exhibit SB on the LSN dataset. Then, we analyzed the proposed datasets to empirically demonstrate that SB can be extreme, and can help explain poor OOD performance and adversarial vulnerability of NNs. We also showed that, contrary to conventional wisdom, extreme SB can potentially hurt generalization.

**Can we mitigate SB?** It is natural to wonder if any modifications to the standard training procedure can help in mitigating extreme SB and its adverse consequences. In Appendix E, we show that well-studied approaches for improving generalization and adversarial robustness—ensemble methods and adversarial training—do not mitigate SB, at least on the proposed datasets. Specifically, “vanilla” ensembles of independent models trained on the proposed datasets mitigate SB to some extent by aggregating predictions, but continue to exclusively rely on the simplest feature. That is, the resulting ensemble remains *invariant to all* complex features (e.g., 5-Slabs in LMS-5). Our results suggest that in practice, the improvement in generalization due to vanilla ensembles stem from combining multiple simple-but-noisy features (such as color, texture) and not by learning diverse and complex features (such as shape). Similarly, adversarially training FCNs on the proposed datasets increases margin (and hence adversarial robustness) to some extent by combining multiple simple features but does not achieve the maximum possible adversarial robustness; the resulting adversarially trained models remain invariant to *all* complex features.

Our results collectively motivate the need for novel algorithmic approaches that avoid the pitfalls of extreme SB. Furthermore, the proposed datasets capture the key aspects of training neural networks on real world data, while being amenable to theoretical analysis and controlled experiments, and can serve as an effective testbed to understand deep learning phenomena. evaluating new algorithmic approaches aimed at avoiding the pitfalls of SB.

## Broader Impact

Our work is foundational in nature and seeks to improve our understanding of neural networks. We do not foresee any significant societal consequences in the short term. However, in the long term, we believe that a concrete understanding of deep learning phenomena is essential to develop reliable deep learning systems for practical applications that have societal impact.

## References

- [1] Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 233–242. JMLR.org, 2017.
- [2] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.
- [3] Peter L Bartlett. The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. *IEEE transactions on Information Theory*, 44(2):525–536, 1998.
- [4] Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems*, pages 6240–6249, 2017.
- [5] Anand Bhattacharjee, Min Jin Chong, Kaizhao Liang, Bo Li, and David Forsyth. Unrestricted adversarial examples via semantic manipulation. In *International Conference on Learning Representations*, 2020.
- [6] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012.
- [7] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017.
- [8] Alon Brutzkus, Amir Globerson, Eran Malach, and Shai Shalev-Shwartz. Sgd learns over-parameterized networks that provably generalize on linearly separable data. *arXiv preprint arXiv:1710.10174*, 2017.
- [9] Sébastien Bubeck, Eric Price, and Ilya Razenshteyn. Adversarial examples from computational constraints. *arXiv preprint arXiv:1805.10204*, 2018.
- [10] Jacob Buckman, Aurko Roy, Colin Raffel, and Ian Goodfellow. Thermometer encoding: One hot way to resist adversarial examples. 2018.
- [11] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. arxiv e-prints, page. *arXiv preprint arXiv:1608.04644*, 2, 2016.
- [12] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.
- [13] Lenaic Chizat and Francis Bach. Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. *arXiv preprint arXiv:2002.04486*, 2020.
- [14] Lenaic Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. In *Advances in Neural Information Processing Systems*, pages 2933–2943, 2019.
- [15] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.
- [16] Akshay Degwekar, Preetum Nakkiran, and Vinod Vaikuntanathan. Computational limitations in robust classification and win-win results. *arXiv preprint arXiv:1902.01086*, 2019.
- [17] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Xiaolin Hu, J Li, and J Zhu. Boosting adversarial attacks with momentum. arxiv preprint. *arXiv preprint arXiv: 1710.06081*, 2017.
- [18] Gintare Karolina Dziugaite and Daniel M Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*, 2017.
- [19] Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. Exploring the landscape of spatial robustness. *arXiv preprint arXiv:1712.02779*, 2017.
- [20] Matt Gardner, Yoav Artzi, Victoria Basmova, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, et al. Evaluating nlp models via contrast sets. *arXiv preprint arXiv:2004.02709*, 2020.
- [21] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.
- [22] Noah Golowich, Alexander Rakhlin, and Ohad Shamir. Size-independent sample complexity of neural networks. *arXiv preprint arXiv:1712.06541*, 2017.
- [23] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

- [24] Suriya Gunasekar, Jason D Lee, Daniel Soudry, and Nati Srebro. Implicit bias of gradient descent on linear convolutional networks. In *Advances in Neural Information Processing Systems*, pages 9461–9471, 2018.
- [25] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1321–1330. JMLR. org, 2017.
- [26] Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324*, 2018.
- [27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *corr abs/1512.03385* (2015), 2015.
- [28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.
- [29] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.
- [30] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *arXiv preprint arXiv:1907.07174*, 2019.
- [31] Katherine L Hermann and Andrew K Lampinen. What shapes feature representations? exploring datasets, architectures, and training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.
- [32] Gao Huang, Zhuang Liu, and Kilian Q Weinberger. Densely connected convolutional networks. *corr abs/1608.06993* (2016). *arXiv preprint arXiv:1608.06993*, 2016.
- [33] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, pages 125–136, 2019.
- [34] Joern-Henrik Jacobsen, Jens Behrmann, Richard Zemel, and Matthias Bethge. Excessive invariance causes adversarial vulnerability. In *International Conference on Learning Representations*, 2019.
- [35] Ziwei Ji and Matus Telgarsky. The implicit bias of gradient descent on nonseparable data. In *Conference on Learning Theory*, pages 1772–1798, 2019.
- [36] Jason Jo and Yoshua Bengio. Measuring the tendency of cnns to learn surface statistical regularities. *arXiv preprint arXiv:1711.11561*, 2017.
- [37] Sanjay Kariyappa and Moinuddin K Qureshi. Improving adversarial robustness of ensembles with diversity training. *arXiv preprint arXiv:1901.09981*, 2019.
- [38] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [39] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *arxiv e-prints*, page. *arXiv preprint arXiv:1612.01474*, 5, 2016.
- [40] Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*, 2017.
- [41] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3, 2013.
- [42] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.
- [43] Karttikeya Mangalam and Vinay Uday Prabhu. Do deep neural networks learn shallow learnable examples first? 2019.
- [44] Tom McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, 2019.
- [45] Tom McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy, July 2019. Association for Computational Linguistics.
- [46] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.
- [47] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1765–1773, 2017.

- [48] Vaishnavh Nagarajan and J Zico Kolter. Uniform convergence may be unable to explain generalization in deep learning. In *Advances in Neural Information Processing Systems*, pages 11611–11622, 2019.
- [49] Preetum Nakkiran, Gal Kaplun, Dimitris Kalimeris, Tristan Yang, Benjamin L Edelman, Fred Zhang, and Boaz Barak. Sgd on neural networks learns functions of increasing complexity. *arXiv preprint arXiv:1905.11604*, 2019.
- [50] Behnam Neyshabur, Srinadh Bhojanapalli, and Nathan Srebro. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. *arXiv preprint arXiv:1707.09564*, 2017.
- [51] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436, 2015.
- [52] Luke Oakden-Rayner, Jared Dunnmon, Gustavo Carneiro, and Christopher Ré. Hidden stratification causes clinically meaningful failures in machine learning for medical imaging. *arXiv preprint arXiv:1909.12475*, 2019.
- [53] Guillermo Ortiz-Jimenez, Apostolos Modas, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Hold me tight! influence of discriminative features on deep network boundaries. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.
- [54] Tianyu Pang, Kun Xu, Chao Du, Ning Chen, and Jun Zhu. Improving adversarial robustness via promoting ensemble diversity. *arXiv preprint arXiv:1901.08846*, 2019.
- [55] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.
- [56] Nicolas Papernot, Patrick D McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. corr abs/1511.04508 (2015). In *37th IEEE Symposium on Security and Privacy*, 2015.
- [57] Jie Ren, Peter J Liu, Emily Fertig, Jasper Snoek, Ryan Poplin, Mark Depristo, Joshua Dillon, and Balaji Lakshminarayanan. Likelihood ratios for out-of-distribution detection. In *Advances in Neural Information Processing Systems*, pages 14680–14691, 2019.
- [58] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.
- [59] Kevin Roth, Yannic Kilcher, and Thomas Hofmann. The odds are odd: A statistical test for detecting adversarial examples. *arXiv preprint arXiv:1902.04818*, 2019.
- [60] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.
- [61] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *Advances in Neural Information Processing Systems*, pages 5014–5026, 2018.
- [62] Ali Shafahi, W Ronny Huang, Christoph Studer, Soheil Feizi, and Tom Goldstein. Are adversarial examples inevitable? *arXiv preprint arXiv:1809.02104*, 2018.
- [63] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *Advances in Neural Information Processing Systems*, pages 3353–3364, 2019.
- [64] Shai Shalev-Shwartz, Ohad Shamir, and Shaked Shammah. Failures of gradient-based deep learning. *arXiv preprint arXiv:1703.07950*, 2017.
- [65] Becks Simpson, Francis Dutil, Yoshua Bengio, and Joseph Paul Cohen. Gradmask: reduce overfitting by regularizing saliency. *arXiv preprint arXiv:1904.07478*, 2019.
- [66] Chawin Sitawarin, Supriyo Chakraborty, and David Wagner. Improving adversarial robustness through progressive hardening. *arXiv preprint arXiv:2003.09347*, 2020.
- [67] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1):2822–2878, 2018.
- [68] Nitish Srivastava. Improving neural networks with dropout. *University of Toronto*, 182(566):7, 2013.
- [69] Thilo Strauss, Markus Hanselmann, Andrej Junginger, and Holger Ulmer. Ensemble methods as a defense to adversarial perturbations against deep neural networks. *arXiv preprint arXiv:1709.03423*, 2017.
- [70] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

- [71] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.
- [72] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.
- [73] Florian Tramèr, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. The space of transferable adversarial examples. *arXiv preprint arXiv:1704.03453*, 2017.
- [74] Guillermo Valle-Perez, Chico Q. Camargo, and Ard A. Louis. Deep learning generalizes because the parameter-function map is biased towards simple functions. In *International Conference on Learning Representations*, 2019.
- [75] Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge University Press, 2019.
- [76] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *Advances in Neural Information Processing Systems*, pages 10506–10518, 2019.
- [77] Haohan Wang, Zexue He, and Eric P. Xing. Learning robust representations by projecting superficial statistics out. In *International Conference on Learning Representations*, 2019.
- [78] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 2020.
- [79] Blake Woodworth, Suriya Gunasekar, Jason D Lee, Edward Moroshko, Pedro Savarese, Itay Golan, Daniel Soudry, and Nathan Srebro. Kernel and rich regimes in overparametrized models. *arXiv preprint arXiv:2002.09277*, 2020.
- [80] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 501–509, 2019.
- [81] Cihang Xie and Alan Yuille. Intriguing properties of adversarial training at scale. *arXiv preprint arXiv:1906.03787*, 2019.
- [82] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.
- [83] Valentina Zantedeschi, Maria-Irina Nicolae, and Ambrish Rawat. Efficient defenses against adversarial attacks. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 39–49, 2017.
- [84] C Zhang, S Bengio, M Hardt, B Recht, and O Vinyals. Understanding deep learning requires rethinking generalization. In *Int Conf. on Learning Representations*, 2017.
- [85] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P Xing, Laurent El Ghaoui, and Michael I Jordan. Theoretically principled trade-off between robustness and accuracy. *arXiv preprint arXiv:1901.08573*, 2019.

# Appendices

The supplementary material is organized as follows. We first discuss additional related work and provide experiment details in Section 2 and Appendix B respectively. In Appendix C, we provide additional experiments to further validate the extreme nature of Simplicity Bias (SB). Then, in Appendix D, we provide additional information about the experiment setup used to show that extreme SB can hurt generalization. We evaluate the extent to which ensemble methods and adversarial training mitigate Simplicity Bias (SB) in Appendix E. Finally, we provide the proof of Theorem 1 in Appendix F.

## A Additional Related Work

In this section, we provide a more thorough discussion of relevant work related to margin-based generalization bounds, adversarial attacks and robustness, and out-of-distribution (OOD) examples.

**Margin-based generalization bounds:** Building up on the classical work of [3], recent works try to obtain tighter generalization bounds for neural networks in terms of *normalized* margin [4, 50, 18, 22]. Here, margin is defined as the difference in the probability of the true label and the largest probability of the incorrect labels. While these bounds seem to capture generalization of neural networks at a coarse level, it has been argued [48] that these approaches may be incapable of fully explaining the generalization ability of neural networks. Furthermore, it is unclear if the notion of model complexity used in these works, based on Lipschitz constant, captures generalization ability accurately. In any case, our results suggest that due to extreme simplicity bias (SB), even if a formulation captures both margin and model complexity accurately, current optimization techniques may not be able to find the optimal solution in terms of generalization *and* robustness-, as they are strongly biased towards small-margin classifiers that exclusively rely on the simplest features.

**Adversarial Defenses:** Neural networks trained using standard procedures such as SGD are extremely vulnerable [23] to  $\epsilon$ -bound adversarial attacks such as FGSM [23], PGD [42], CW [11], and Momentum [17]; Unrestricted attacks [7, 19] can significantly degrade model performance as well. Defense strategies based on heuristics such as feature squeezing [82], denoising [80], encoding [10], specialized nonlinearities [83] and distillation [56] have had limited success against stronger attacks [2]. On the other hand, standard adversarial training [42] and its variants such as [85] are fairly effective on datasets such as MNIST, CIFAR-10 and CIFAR-100. However, on larger datasets such as ImageNet, these methods have limited success [63]; recent attempts [78, 63] that make adversarial training faster do not improve robustness either. In Appendix E, we show that  $\ell_2$  adversarial training on synthetic datasets can improve robustness by some extent but it is unable to learn optimal large-margin  $\ell_2$ -robust classifiers.

**Detecting OOD Examples:** Neural networks trained using standard training procedures tend to rely on low-level features and spurious correlations and hence exhibit brittleness to benign distributional changes to the data. Recent works thus aim to detect OOD examples using generative models [57], statistical tests [59], and model confidence scores [29, 40, 39]. Our experiments in Section 4 that validate extreme SB in practice also show that detectors that directly or indirectly rely on model scores to detect OOD examples may not work well as SGD-trained neural networks can exhibit complete invariance to predictive-but-complex features.

## B Experiment Details

In this section, we provide additional details on the datasets, models, optimization methods and training hyperparameters used in our experiments.

**One-dimensional Building Blocks:** We first describe the data generation process underlying each building block: linear, noisy linear, and  $k$ -slab. Then, we introduce a noisy version of the 5-slab block, which we later use in Appendix D.

- Linear( $\gamma, B$ ): The linear block is parameterized by the effective margin  $\gamma$  and width  $B$ . The distribution first samples a label  $y \in \{-1, 1\}$  uniformly at random, and then given  $y$ ,  $x$  is sampled as follows:  $x = y(B\gamma + (B - B\gamma) \cdot \text{U}(0, 1)$ , where  $\text{U}(0, 1)$  is the uniform distribution on  $[0, 1]$ .
- NoisyLinear( $\gamma, B, p$ ): The noisy linear block is parameterized by effective margin  $\gamma$ , width  $B$ , and noise parameter  $p$ . Linear classifiers can attain the optimal classification accuracy of  $1 - p/2$ . Given label  $y \in \{-1, 1\}$  sampled uniformly at random,  $x$  is sampled as follows:

$$x = \begin{cases} y(B\gamma + (B - B\gamma) \cdot \text{U}(0, 1)) & \text{w.p. } p \\ \text{U}(-\gamma, \gamma) & \text{w.p. } 1-p \end{cases}$$

- Slab( $\gamma, B, k$ ): The  $k$ -slab block is parameterized by effective margin  $\gamma$ , width  $B$ , and number of slabs  $k$ . We use  $k \in \{3, 5, 7\}$  in our paper. The width of each slab,  $w_k = 2B(1 - (k-1)\gamma)/k$ , in the  $k$ -slab block is chosen such that the farthest points are at  $-B$  and  $B$ . For example, given label  $y \in \{-1, 1\}$  and random sign  $z \in \{-1, 1\}$  sampled unif. at random, we can sample  $x$  from a 3-slab block as follows:

$$x = \begin{cases} z(\frac{1}{2}w_3 \cdot \text{U}(0, 1)) & \text{if } y = -1 \\ z(\frac{1}{2}w_3 + 2B\gamma + w_3 \cdot \text{U}(0, 1)) & \text{if } y = +1 \end{cases}$$

For  $k$ -slab blocks with  $k \in \{5, 7\}$ , the probability of sampling from the two slabs (one on each side) that are farthest away from the origin are  $1/4$  and  $1/8$  respectively to ensure that the variance of instances in positive and negative classes,  $x_+$  and  $x_-$ , are equal.

- NoisySlab( $\gamma, B, k, p$ ): Analogous to the noisy linear block, the noisy variant of the  $k$ -slab block is additionally parameterized by a noise parameter  $p$ . In this setting, a  $(k-1)$ -piecewise linear classifier can attain the optimal classification accuracy of  $1 - p/2$ . For example, For example, given label  $y \in \{-1, 1\}$  and random sign  $z \in \{-1, 1\}$  sampled uniformly at random, we can sample  $x$  from a  $p$ -noisy 3-slab block as follows:

$$x = \begin{cases} z(\frac{1}{2}w_3 \cdot \text{U}(0, 1)) & \text{if } y = -1 \\ z(\frac{1}{2}w_3 + 2B\gamma + w_3 \cdot \text{U}(0, 1)) & \text{if } y = +1 \\ z(\frac{1}{2}w_3 + (2B\gamma - \frac{1}{2}w_3) \cdot \text{U}(0, 1)) & \text{w.p. } p \end{cases}$$

**Datasets:** We now outline the default hyperparameters for generating the synthetic datasets used in the paper, provide additional details on the LSN dataset, and introduce two additional synthetic datasets as well as multiple versions of the MNIST-CIFAR dataset (i.e., with different class pairs).

- Synthetic Dataset Hyperparameters: Recall that we use four  $d$ -dimensional synthetic datasets—LMS- $k$ ,  $\hat{\text{LMS}}-k$ , MS-(5, 7), and MS-5—wherein each coordinate corresponds to one of the building blocks described above. Unless mentioned otherwise, for all four datasets, we set the effective margin parameter  $\gamma = 0.1$ , width parameter  $B = 1$ , and noise parameter  $p = 0.1$  in all blocks/coordinates. Also recall that each dataset comprises at most one “simple” feature  $S$  and multiple independent complex features  $S^c$ . In our experiments, all datasets have sample sizes that are large enough for all models considered in the paper to learn complex features  $S^c$  and attain optimal test accuracy, even in the absence of  $S$ ; we use sample sizes of 50000 for LMS-5 and MS-5 and 40000 for  $\hat{\text{LMS}}-7$ .
- LSN Dataset: Recall that the LSN dataset (described in Section 3) is a stylized version of the LMS- $k$  that is amenable to theoretical analysis. In LSN, conditioned on the label  $y$ , the first and second coordinates of  $x$  are *singleton* linear and 3-slab blocks: linear and 3-slab blocks have support on

$\{-1, 1\}$  and  $\{-1, 0, 1\}$  respectively. The remaining coordinates are standard gaussians and not predictive of the label. Each data point  $(x_i, y_i) \in \Re^d \times \{-1, 1\}$  can be sampled as follows:

$$\begin{aligned} y_i &= \pm 1, \text{ w.p. } 1/2, \quad \varepsilon_i = \pm 1, \text{ w.p. } 1/2, \\ x_{i1} &= y_i && \text{(Linear coordinate),} \\ x_{i2} &= \left(\frac{y_i + 1}{2}\right)\varepsilon_i && \text{(Slab coordinate),} \\ x_{i,3:d} &\sim \mathcal{N}(0, I_{d-2}) && \text{(d-2 Noise coordinates).} \end{aligned}$$

- Additional Datasets: We now introduce  $\hat{\text{MS}}-(5,7)$ , the noisy version of  $\text{MS}-(5,7)$ , and three MNIST-CIFAR datasets, each with different MNIST and CIFAR10 classes.
  - $\hat{\text{MS}}-(5,7)$ : Noisy 5-slab and multiple noiseless 7-slab blocks; the first coordinate is a noisy 5-slab block and the remaining  $d-1$  coordinates are independent 7-slab blocks. Note that this dataset comprises a noisy-but-simpler 5-slab block and multiple noiseless 7-slab blocks; a 6-piecewise linear classifier can attain 100% accuracy by learn any 7-slab block.
  - MNIST-CIFAR datasets: Recall that images in the MNIST-CIFAR datasets are concatenations of MNIST and CIFAR10 images. We introduce additional variants of the MNIST-CIFAR using different class pairs to show that our results in the paper are robust to the exact choice of pairs:

Datasets	Class -1		Class +1	
	MNIST	CIFAR10	MNIST	CIFAR10
MNIST-CIFAR:A	Digit 0	Automobile	Digit 1	Truck
MNIST-CIFAR:B	Digit 1	Automobile	Digit 4	Truck
MNIST-CIFAR:C	Digit 0	Airplane	Digit 1	Ship

Table 3: Three MNIST-CIFAR datasets. We use MNIST-CIFAR:A in the paper. In MNIST-CIFAR:B, we use different MNIST classes: digits 1 and 4. In MNIST-CIFAR:C, we use different CIFAR10 classes: airplane and ship. Our results in Section 4 hold on all three MNIST-CIFAR datasets.

**Models:** Here, we briefly describe the models (and its abbreviations) used in the paper. We use fully-connected (FCNs), convolutional (CNNs), and sequential neural networks (GRUs [15]) on synthetic datasets. Abbreviations  $(w, d)$ -FCN denotes FCN with width  $w$  and depth  $d$ ,  $(f, k, d)$ -CNN denotes  $d$ -layer CNNs with  $f$  filters of size  $k \times k$  in each layer with and  $(h, l, d)$ -GRU denotes  $d$ -layer  $d$ -layer GRU with input dimensionality  $l$  and hidden state dimensionality  $h$ . On MNIST-CIFAR, we train MobileNetV2 [60], GoogLeNet [70], ResNet50 [27] and DenseNet121 [32].

**Training Procedures:** Unless mentioned otherwise, we use the following hyperparameters for standard training and adversarial training on synthetic and MNIST-CIFAR data:

- Standard Training: On synthetic datasets, we use Stochastic Gradient Descent (SGD) with (fixed) learning rate 0.1 and batch size 256, and  $\ell_2$  regularization  $5 \cdot 10^{-7}$ . On MNIST-CIFAR datasets, we use SGD with initial learning rate 0.05 with decay factor of 0.2 every 30 epochs, momentum 0.9 and  $\ell_2$  regularization  $5 \cdot 10^{-5}$ . We do not use data augmentation. We run all models for at most 500 epochs and stop early if the training loss goes below  $10^{-2}$ .
- Adversarial Training: We use the same SGD hyperparameters (as described above) on synthetic and MNIST-CIFAR datasets. We use Projected Gradient Descent (PGD) Adversarial Training [42] to adversarially train models. We use learning rate 0.1 and 40 iterations to generate  $\ell_2$  &  $\ell_\infty$  perturbations

## C Additional Results on the Extreme Nature of Simplicity Bias (SB)

Recall that Section 4 of the paper establishes the extreme nature of SB: *If all features have full predictive power, NNs rely exclusively on the simplest feature S and remain invariant to all complex features  $S^c$* —in Section 4 of the paper. Now, we further validate the extreme nature of SB across model architectures, datasets, optimizers, activation functions and regularization. We also analyze the effect of input dimensionality, number of complex features, choice and scaling of random initialization and non-random initialization.

### C.1 Effect of Model Architecture

In this section, we supplement our results in Section 4 of the paper by showing that extreme simplicity bias (SB) persists across several model architectures and on synthetic as well as image-based datasets. In Table 4, we present  $\{S, S^c\}$ -Randomized AUCs for FCNs, CNNs and GRUs with depth  $\{1, 2\}$  trained on LMS and MS- $(5, 7)$  datasets and state-of-the-art CNNs trained on MNIST-CIFAR:A. While the  $S^c$ -randomized AUC equals 1.00 (perfect classification), we see that the  $S$ -randomized AUCs are approximately 0.5 for all models. This is because all models essentially only rely on the simplest feature  $S$  and remain invariant to all complex features  $S^c$ , even though all features have equal predictive power.

Dataset	Set S	Set $S^c$	Model	Randomized AUC	
				Set S	Set $S^c$
LMS-5	Linear	5-Slabs	(100, 1)-FCN	0.50	1.00
			(100, 2)-FCN	0.49	1.00
			(32, 7, 1)-CNN	0.50	1.00
			(32, 7, 2)-CNN	0.50	1.00
			(100, 10, 1)-GRU	0.51	1.00
			(100, 10, 2)-GRU	0.50	1.00
MS- $(5, 7)$	5-Slab	7-Slabs	(100, 1)-FCN	0.50	1.00
			(100, 1)-FCN	0.50	1.00
			(32, 7, 1)-CNN	0.50	1.00
			(32, 7, 2)-CNN	0.50	1.00
			(100, 10, 1)-GRU	0.50	1.00
			(100, 10, 2)-GRU	0.50	1.00
MNIST-CIFAR:A	MNIST block	CIFAR block	MobileNetV2	0.52	1.00
			GoogLeNet	0.51	1.00
			ResNet50	0.50	1.00
			DenseNet121	0.52	1.00

Table 4: Extreme SB across models trained on synthetic and image-based datasets show that all models exclusively rely on the simplest feature  $S$  and remain completely invariant to all complex features  $S^c$

### C.2 Effect of MNIST-CIFAR Class Pairs

In this section, we supplement our results on MNIST-CIFAR (in Section 4) in order to show that extreme SB observed in MobileNetV2 [60], GoogLeNet [70], ResNet50 [27] and DenseNet121 [32] does not depend on the exact choice of MNIST and CIFAR10 class pairs used to construct the MNIST-CIFAR datasets. To do so, we evaluate the MNIST-randomized and CIFAR10-randomized metrics of the aforementioned models on three datasets—MNIST-CIFAR:A, MNIST-CIFAR:B, MNIST-CIFAR:C—described in Appendix B.

Table 5 presents the standard, MNIST-randomized and CIFAR10-randomized AUC values of MobileNetV2, GoogLeNet, ResNet50 and DenseNet121 on three MNIST-CIFAR datasets. We observe that randomizing over the simpler MNIST block is sufficient to fully degrade the predictive power of all models; for instance, randomizing the MNIST block drops the AUC values of ResNet50 from 1.0 to 0.5 (i.e., equivalent to random classifier). However, randomizing the CIFAR10 block has no effect—standard AUC and CIFAR10-randomized AUCs equal 1.0. In contrast, an ideal classifier that relies on MNIST & CIFAR10 would attain non-trivial AUC even when the MNIST block is randomized.

Model	MNIST-CIFAR:A AUCs						MNIST-CIFAR:B AUCs						MNIST-CIFAR:C AUCs					
	CIFAR10		MNIST		CIFAR10		MNIST		CIFAR10		MNIST		CIFAR10		MNIST			
	Standard	Randomized	Standard	Randomized	Standard	Randomized	Standard	Randomized	Standard	Randomized	Standard	Randomized	Standard	Randomized	Standard	Randomized		
MobileNetV2	1.00 ± 0.00	1.00 ± 0.00	0.53 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	0.53 ± 0.02	1.00 ± 0.00	1.00 ± 0.00	0.50 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	0.53 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	0.50 ± 0.03	1.00 ± 0.00	1.00 ± 0.00	0.54 ± 0.01
GoogLeNet	1.00 ± 0.00	1.00 ± 0.00	0.52 ± 0.02	1.00 ± 0.00	1.00 ± 0.00	0.50 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	0.53 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	0.53 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	0.50 ± 0.03	1.00 ± 0.00	1.00 ± 0.00	0.50 ± 0.03
ResNet50	1.00 ± 0.00	1.00 ± 0.00	0.50 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	0.51 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	0.50 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	0.50 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	0.50 ± 0.03	1.00 ± 0.00	1.00 ± 0.00	0.50 ± 0.03
DenseNet121	1.00 ± 0.00	1.00 ± 0.00	0.53 ± 0.02	1.00 ± 0.00	1.00 ± 0.00	0.52 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	0.54 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	0.54 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	0.54 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	0.54 ± 0.01

Table 5: (Extreme SB in three MNIST-CIFAR datasets) Standard and randomized AUCs of four state-of-the-art CNNs trained on three MNIST-CIFAR datasets. The AUC values collectively indicate that all models exclusively rely on the MNIST block.

### C.3 Effect of Optimizers and Activation Functions

Now, we study the effect of activation function and optimizer on extreme SB. That is, can the usage of different activation functions and optimizer encourage trained neural networks to rely on complex features  $S^c$  in addition to the simplest feature  $S$ ?

Activation Function	LMS-7			MS-(5,7)		
	SGD	Adam	RMSProp	SGD	Adam	RMSProp
ReLU	0.499 ± 0.001	0.497 ± 0.003	0.502 ± 0.004	0.499 ± 0.003	0.499 ± 0.004	0.496 ± 0.004
Leaky ReLU	0.501 ± 0.001	0.497 ± 0.003	0.501 ± 0.005	0.499 ± 0.005	0.498 ± 0.002	0.498 ± 0.005
PReLU	0.500 ± 0.004	0.500 ± 0.003	0.501 ± 0.004	0.501 ± 0.004	0.496 ± 0.003	0.499 ± 0.002
Tanh	0.495 ± 0.001	0.502 ± 0.004	0.495 ± 0.004	0.498 ± 0.004	0.499 ± 0.004	0.498 ± 0.002

Table 6: (Effect of activation function and optimizers) (100, 2)-FCNs with multiple activation functions—ReLU, Leaky ReLU [41], PReLU [28], and Tanh—trained on LMS-7 data using common first-order optimization methods—SGD, Adam [38], and RMSProp [72]—exhibit extreme SB.

Table 6 presents the S-randomized AUCs of (100, 2)-FCNs with multiple activation functions—ReLU, Leaky ReLU [41], PReLU [28], and Tanh—trained on LMS-7 and MS-(5,7) datasets using multiple commonly-used optimizers: SGD, Adam [38], and RMSProp [72]. We observe that for all combinations of activations and optimizers, trained FCNs still only rely on simplest feature  $S$ ;  $S$ -randomized and  $S^c$ -randomized AUCs are approximately 0.50 and 1.0 respectively for all optimizers and activation functions. Therefore, in addition to SGD, commonly used first-order optimization methods such as Adam and RMSProp cannot jointly learn large-margin classifiers that rely on learn slab-structured features in the presence of a noisy linear structure. To summarize, the experiment in Appendix C.2 shows that simply altering the choice of optimizer and activation function does not have any effect on extreme SB. Similar to the experiments in Section 4 of the paper, all models exclusively rely on simplest feature  $S$  and remain invariant to complex features  $S^c$ .

### C.4 Effect of $\ell_2$ Regularization and Dropout

In this section, we use SGD-trained FCNs trained on LMS-7 data to examine the extent to which Dropout [68] and  $\ell_2$  regularization alters the extreme nature of SB. Specifically, we use Dropout probability parameter  $\{0.0, 0.05, 0.10\}$  and  $\ell_2$  regularization parameters  $\{0.01, 0.001\}$  when training FCNs with width 100 and depth  $\{1, 2\}$  on LMS-7 data using SGD. In Table 7, we show the standard and  $S^c$ -randomized AUCs equal 1.00 (perfect classification), whereas the S-randomized AUCs are approximately 0.5 for all models. Applying Dropout while reducing the amount of  $\ell_2$  regularization has negligible effect on the extreme nature of SB observed in the synthetic or image-based datasets.

### C.5 Effect of Input Dimension and Number of Complex Features

In this section, we evaluate the performance of FCNs trained on LMS-7 data using SGD to show the extreme SB persists in the low-dimensional setting ( $d < 10$ ) and also with varying number of 7-slab features  $1 \leq |S^c| \leq d$ . As shown in Table 8, decreasing the input dimension  $d$  or the number of complex features  $|S^c|$  has no effect on extreme SB of FCNs trained on the LMS-7 dataset. Similar to our results in Figure 3, the standard and randomized AUCs collectively show that the SGD-trained (100, 1)-FCNs exclusively rely on the linear component and do not rely on the 7-slab coordinates.

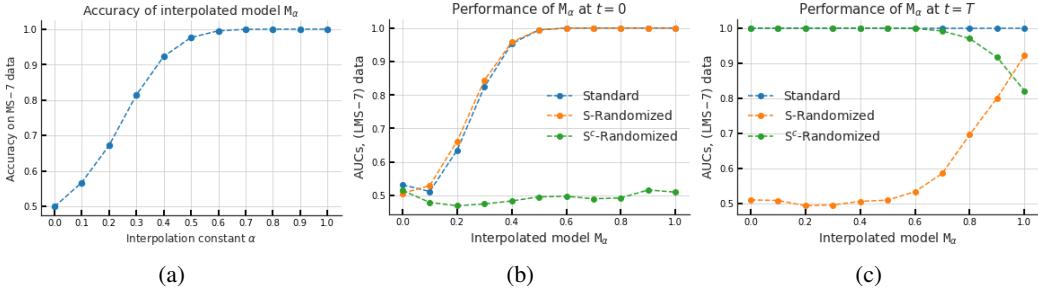


Figure 5: Effect of non-random initialization on simplicity bias. Subplot (a) standard test accuracy (on MS-7 data) of linearly-interpolated model  $M_\alpha \equiv \alpha \cdot M_{\text{slab}} + (1 - \alpha) \cdot M_{\text{rand}}$  increases monotonically with the interpolation constant  $\alpha$ . Subplots (b) and (c) show how standard, S-randomized and  $S^c$ -randomized AUCs vary with interpolated model  $M_\alpha$  before and after training on LMS-7 data.

### C.6 Effect of Random Initialization Scale

Now, we analyze the effect of the choice and scale (i.e., magnitude of the weights of randomly initialized FCNs) of random initialization on simplicity bias using FCNs trained on LMS-7 data.

As shown in Table 9, the choice (Kaiming and Xavier) and the scale of random initialization do not alter the extreme SB phenomenon on the LMS-7 dataset. That is, scaling the randomly initialized models by up to 0.1 and 10.0 has no effect on simplicity bias—SGD-trained (100, 1)-FCNs exclusively rely on the linear component and do not rely on the 7-slab coordinates.

### C.7 Effect of Non-random Initialization

In this section, we investigate the effect of non-random initialization on simplicity bias using FCNs trained on MS-7 and LMS-7 data. The goal of this experiment is to determine the extent to which simplicity bias persists when the untrained network (at timestep  $t = 0$ ) attains non-random standard accuracy by relying on one or more “complex” 7-slab features.

To vary the degree of non-random initialization  $\alpha$ , we obtain model  $M_\alpha$  by linearly interpolating the weights of a randomly initialized network  $M_{\text{rand}}$  and a network  $M_{\text{slab}}$  that exclusively relies on one or more “complex” 7-slab features to attain 100% accuracy on the LMS-7 dataset. That is,  $M_\alpha \equiv \alpha \cdot M_{\text{slab}} + (1 - \alpha) \cdot M_{\text{rand}}$ . Note that  $M_{\text{slab}}$  is trained on MS-7 data to attain 100% standard accuracy by relying on one or more 7-slab coordinates. As shown in Figure 5(a), increasing the interpolation constant  $\alpha$  monotonically increases the standard accuracy of  $M_\alpha$  on MS-7 data.

Now, we use the linearly interpolated model  $M_\alpha$  (for varying values of  $\alpha$ ) as initialization and train  $M_\alpha$  on LMS-7 data, which additionally consists of a “simple” linearly-separable coordinate. To maintain the input dimensionality, we obtain LMS-7 data by replacing a 7-slab coordinate by the simpler linear coordinate. In order to maintain the non-random accuracy of  $M_\alpha$ , we use coordinate-randomized AUCs to choose and replace a 7-slab coordinate that the model does not depend on.

Model	Dropout	Standard AUC		S-Randomized AUC		$S^c$ -Randomized AUC	
		$\lambda = 10^{-2}$	$\lambda = 10^{-4}$	$\lambda = 10^{-2}$	$\lambda = 10^{-4}$	$\lambda = 10^{-2}$	$\lambda = 10^{-4}$
(100, 1)-FCN	0.00	1.00 ± 0.00	1.00 ± 0.00	0.50 ± 0.00	0.50 ± 0.01	1.00 ± 0.00	1.00 ± 0.00
	0.05	1.00 ± 0.00	1.00 ± 0.00	0.50 ± 0.00	0.50 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	0.10	1.00 ± 0.00	1.00 ± 0.00	0.50 ± 0.00	0.50 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
(100, 2)-FCN	0.00	1.00 ± 0.00	1.00 ± 0.00	0.50 ± 0.00	0.50 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	0.05	1.00 ± 0.00	1.00 ± 0.00	0.50 ± 0.00	0.50 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	0.10	1.00 ± 0.00	1.00 ± 0.00	0.50 ± 0.00	0.50 ± 0.00	1.00 ± 0.00	1.00 ± 0.00

Table 7: Dropout and  $\ell_2$  regularization have no effect on extreme SB of FCNs trained on LMS-7 datasets. The standard and {S,S<sup>c</sup>} -randomized AUC values of (100, 1)-FCNs and (100, 2)-FCNs collectively indicate that the models still exclusively latch on to S (linear block) and remain invariant to S<sup>c</sup> (7-slab blocks).

Input Dimension $d$	Number of 7-Slabs $ \mathcal{S}^c $	Standard AUC	$\mathcal{S}$ -Randomized AUC	$\mathcal{S}^c$ -Randomized AUC
2	1	$1.00 \pm 0.00$	$0.51 \pm 0.02$	$1.00 \pm 0.00$
5	1	$1.00 \pm 0.00$	$0.50 \pm 0.00$	$1.00 \pm 0.00$
	3	$1.00 \pm 0.00$	$0.50 \pm 0.00$	$1.00 \pm 0.00$
10	1	$1.00 \pm 0.00$	$0.50 \pm 0.00$	$1.00 \pm 0.00$
	3	$1.00 \pm 0.00$	$0.50 \pm 0.01$	$1.00 \pm 0.00$
	6	$1.00 \pm 0.00$	$0.50 \pm 0.00$	$1.00 \pm 0.00$

Table 8: Effect of input dimension and number of complex features on extreme SB of FCNs trained on LMS-7 data. The standard and  $\{\mathcal{S}, \mathcal{S}^c\}$ -randomized AUCs of  $(100, 1)$ -FCNs collectively indicate that the models exclusively latch on to  $\mathcal{S}$  (linear) and remain invariant to  $\mathcal{S}^c$  (7-slabs).

Initialization	Scaling Factor	Standard AUC	$\mathcal{S}$ -Randomized AUC	$\mathcal{S}^c$ -Randomized AUC
Kaiming	0.1	$1.00 \pm 0.00$	$0.50 \pm 0.00$	$1.00 \pm 0.00$
	0.5	$1.00 \pm 0.00$	$0.49 \pm 0.01$	$1.00 \pm 0.00$
	2.0	$1.00 \pm 0.00$	$0.50 \pm 0.00$	$1.00 \pm 0.00$
	10.0	$1.00 \pm 0.00$	$0.50 \pm 0.00$	$1.00 \pm 0.00$
Xavier	0.1	$1.00 \pm 0.00$	$0.50 \pm 0.00$	$1.00 \pm 0.00$
	0.5	$1.00 \pm 0.00$	$0.50 \pm 0.00$	$1.00 \pm 0.00$
	2.0	$1.00 \pm 0.00$	$0.49 \pm 0.01$	$1.00 \pm 0.00$
	10.0	$1.00 \pm 0.00$	$0.51 \pm 0.01$	$1.00 \pm 0.00$

Table 9: Effect of choice (Kaiming and Xavier) and scale of random initialization on simplicity bias of FCNs trained on LMS-7 data. The AUCs of  $(100, 1)$ -FCNs collectively show increasing the scale of random initialization does not alleviate extreme simplicity bias in this setting.

As expected, Figure 5(b) shows that prior to training on LMS-7 data, the interpolated models  $M_\alpha$  attain  $\mathcal{S}^c$ -randomized AUC 0.50 and  $\mathcal{S}^c$ -randomized accuracy equals standard AUC because the models exclusively rely on one or more 7-slab coordinates. However, as shown in Figure 5(c), after training  $M_\alpha$  on LMS-7 data, the models no longer exhibit exclusive reliance on 7-slab features. In particular, when  $\alpha \leq 0.5$ , the models now exclusively rely on the linear coordinate only. When  $\alpha \geq 0.5$ , the dependence on the 7-slab coordinates is considerably reduced. Surprisingly, even when  $\alpha = 1.0$ , we observe that the model additionally relies on the linear coordinate, possibly due to non-zero training loss at initialization. These results collectively suggest that in this setting, non-random initialization by first training the model on complex features only can be effective in mitigating extreme simplicity bias to some extent.

## D Additional Results on the Effect of Extreme SB on Generalization

Recall that in Section 5 of the paper, we showed that extreme SB can result in suboptimal generalization of SGD-trained models on the same data distribution. In this section, we present additional information about the experiment setup used in Section 5 to show that SB can worsen standard generalization.

### D.1 Experiment Setup in Section 5

We now provide additional information about the experimental setup used in Section 5 of the paper, where we show that extreme simplicity bias can result in suboptimal generalization. We train fully-connected networks (FCNs) of width  $\{100, 200, 300\}$  and depth  $\{1, 2\}$  using SGD on 50-dimensional  $\hat{\text{LMS}}\text{-7}$  dataset of 40000 samples, which comprises of a noisy linear coordinate (10% noise) and 49 7-slab coordinates. For each model architecture, we perform a grid search over SGD hyperparameters—learning rate, batch size, momentum, and weight decay—and report standard and randomized test accuracies of the model (in Table 2) that perform best on a  $\hat{\text{LMS}}\text{-7}$  validation dataset. We perform a grid search over the following SGD hyperparameters:

- Learning rate:  $\{0.001, 0.01, 0.05, 0.1, 0.3\}$
- Batch size:  $\{4, 16, 64, 256\}$
- Weight decay:  $\{0, 0.00005, 0.0005\}$
- Momentum:  $\{0, 0.9, 0.95\}$

We train all models for at most 10 million updates with constant learning rate schedule and stop early if the training loss diverges or goes below 0.01. In this setting, 10 million updates is equivalent to 1000 epochs with batch size 4 and 64000 epochs with batch size 256. As mentioned in Section 5, the training sample size of 40000 data points is large enough for FCNs of depth  $\{1, 2\}$  and width  $\{100, 200, 300\}$  trained on MS-7 data (i.e.,  $S^c$  only, after removing  $S$  from data) to attain  $\approx 100\%$  test accuracy using SGD with *learning rate* 0.3, *batch size* 256, *weight decay* 0.0005, and *momentum* 0.9. The optimal hyperparameters for FCNs trained on  $\hat{\text{LMS}}\text{-7}$  data are provided in Table 10. Note that we do not consider  $\ell_2$  weight decay values larger than 0.0005 because it results in 95% test and train accuracy even after 10 million updates by preventing FCNs from overfitting to the noise in the linear component. Also note that  $(100, 1)$ -FCNs are not able to completely overfit due to insufficient representation capacity when trained with the chosen SGD hyperparameters (see Table 10).

Hyperparameter	$(100, 1)$ -FCN	$(200, 1)$ -FCN	$(300, 1)$ -FCN	$(100, 2)$ -FCN	$(200, 2)$ -FCN	$(300, 2)$ -FCN
Learning rate	0.30	0.10	0.01	0.30	0.10	1.00
Batch size	16	256	16	64	64	256
Weight decay	0.0	0.0	0.0	0.0	0.0005	0.0005
Momentum	0.90	0.0	0.0	0.0	0.0	0.0

Table 10: SGD hyperparameters of SGD-trained FCNs trained on  $\hat{\text{LMS}}\text{-7}$  data that result in the best validation accuracy out of all 180 hyperparameter combinations listed in Appendix D.1.

## E Can we mitigate Simplicity Bias?

In this section, we investigate whether standard approaches for improving generalization error and adversarial robustness—ensembles and adversarial training—help in mitigating SB.

### E.1 Ensemble Methods

We now study the extent to which ensembles mitigate SB and its adverse effect on generalization. Specifically, we evaluate the performance of ensembles of fully-connected networks (FCNs) that are trained on two datasets:  $\hat{\text{LMS}}\text{-}7$  and MS-5. Recall that the  $\hat{\text{LMS}}\text{-}7$  data comprises one simple-but-noisy linear coordinate and multiple relatively complex 7-slab coordinates that have no noise, whereas MS-5 data comprises multiple noiseless 5-slab coordinates only.

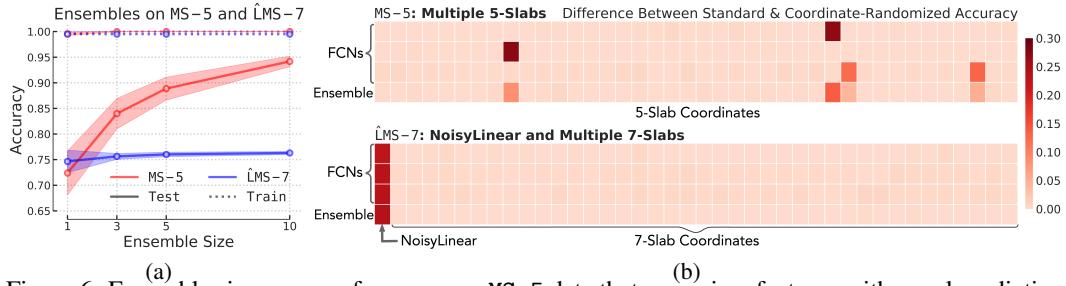


Figure 6: Ensembles improve performance on MS-5 data that comprises features with equal predictive power and simplicity. However, as shown in (a), ensembles do not improve performance on  $\hat{\text{LMS}}\text{-}7$  data that has a simple-but-noisy linear coordinate that has less predictive power than than the 7-slab coordinates; this is because individual FCNs trained on  $\hat{\text{LMS}}\text{-}7$  data exclusively rely on the noisy linear coordinate and consequently misclassify the same set of instances, as shown in subplot (b).

To better highlight the effect of ensembles on generalization, we choose a sample size (for both datasets) such that individual models (a) overfit to training data (i.e., non-zero generalization gap) but (b) still attain non-trivial test accuracy. We now discuss the performance of ensembles of independently trained models on MS-5 and  $\hat{\text{LMS}}\text{-}7$  datasets:

- **MS-5 data:** Recall that MS-5 data comprises multiple independent 5-slab blocks, one in each coordinate, that have equal simplicity and predictive power. Thus, since all features have equal simplicity, independent SGD-trained (100, 2)-FCN end up relying on different 5-slab coordinates due to random initialization, as shown in Figure 6(b). As the training sample size is small, FCNs overfit to the training data and attain approximately 75% test accuracy, as shown in Figure 6. Consequently, as shown in Figure 6, ensembles of these models rely on all 5-slab coordinates learned by the individual models and attain better test accuracy by aggregating model predictions and averaging out overfitting. For example, Figure 6 shows that ensembles of size 5 and 10 improves generalization by approximately 15% and 20% respectively.
- **$\hat{\text{LMS}}\text{-}7$  data:** Recall that  $\hat{\text{LMS}}\text{-}5$  data comprises one simple-but-noisy linear block (with 50% noise) and multiple independent 7-slab blocks that have no noise. Now, due to extreme SB, every independently trained FCN exclusively latches on (and overfits to) the simpler-but-noisy linear block, as shown in Figure 6(b). As a result, all models collectively lack diversity and essentially learn the same decision boundary because of extreme SB. Therefore, ensembles of these models do not improve generalization because the independent models make misclassifications on the same instances. As shown in Figure 6(a), ensembles of size 3, 5 and 10 do not improve generalization—the test accuracy remains 75%.

The ensemble performance on MS-5 data indicates that when datasets have *multiple* equally simple features, ensembles of independently trained models mitigate SB to some extent by aggregating predictions of models that rely on simple features. Conversely, the ensemble performance on  $\hat{\text{LMS}}\text{-}7$  data suggests that when datasets comprise few features that the more noisy and less predictive than the rest, ensembles may not improve generalization. Our results also suggest that the generalization improvements using ensemble methods in practice may stem from combining multiple simple-but-noisy features (such as color, texture) and not by learning complex features (such as shape).

## E.2 Adversarial Training

We now investigate the extent to which adversarial training [42] mitigates SB and its adverse effect on adversarial robustness using two datasets: MNIST-CIFAR and AdvMS-(5,7).

### E.2.1 Adversarially training FCNs on AdvMS-(5,7) data

Now, we first introduce AdvMS-(5,7), a variant of the MS-(5,7) dataset, and then investigate if adversarially trained FCNs that improve adversarial robustness by some extent also mitigate extreme simplicity bias (SB).

**AdvMS-(5,7) dataset:** Recall that  $d$ -dimensional MS-(5,7) data, introduced in Section 3, consists of  $d-1$  7-slab coordinates and a single relatively simpler 5-slab coordinate, all of which have perfect predictive power. Similar to MS-(5,7) data, the  $d$ -dimensional AdvMS-(5,7) data comprises 5-slab and 7-slab coordinates. Specifically, the first  $d/2$  coordinates correspond to independent 5-slabs, each with effective margin  $\gamma_5$  and the other  $d/2$  coordinates correspond to independent 7-slabs with effective margin  $\gamma_7$ . In contrast to MS-(5,7) data, the AdvMS-(5,7) dataset (a) comprises  $d/2$  5-slab coordinates and (b) the 5-slabs and 7-slabs do not necessarily share the same effective margin. In our experiments below, we set  $d = 20$ ,  $\gamma_5 = 0.05$  and  $\gamma_7 = 0.15$ . That is, we conduct our experiments on 20-dimensional data in which the simple features  $S$  and complex features  $S^c$  correspond to the 10 small-margin 5-slab and 10 large-margin 7-slab coordinates respectively.

**Experiment setup:** In Section 4, we observed that SGD-trained FCNs, due to extreme SB, exclusively rely on the simplest feature  $S$  and consequently learn small-margin classifiers that are highly vulnerable to adversarial attacks. To mitigate SB and attain optimal adversarial robustness, fully-connected networks with width  $w$  and depth  $d$ ,  $(w, d)$ -FCNs, must learn maximum-margin classifiers and consequently rely on *all* simple and complex features (i.e., features in  $S$  and  $S^c$ ). The margin of the 20-dimensional AdvMS-(5,7) dataset described above is approximately  $\gamma_{\text{data}} = 0.62$ , which implies that the maximum-margin classifier should exhibit robustness to  $\ell_2$  adversarial perturbations that have norm  $\epsilon < \gamma_{\text{data}}$ . Therefore, to check if adversarial training mitigates extreme SB, we evaluate the robustness of adversarially trained FCNs against  $\ell_2$  adversarial perturbations that have norm  $\epsilon < \gamma_{\text{data}}$ . In addition to  $\gamma_{\text{data}}$ , let  $\gamma_S = 0.30$  denote the maximum margin of the classifier that exclusively relies on all 5-slab features in  $S$ .

Also note that (a) adversarial perturbations are generated using PGD attacks [42], (b) (200,2)-FCNs and (1000,2)-FCNs are expressive enough to learn the maximum-margin classifier on the 20-dimensional AdvMS-(5,7) data, (c) FCNs are adversarially trained for 4000 epochs with initial learning rate 0.1 that decays by a multiplicative factor of 0.1 after every 1000 epochs, and (d) the training data comprises 6000 data points, which is enough for SGD-trained (1000,2)-FCNs to learn 7-slab coordinates and attain 100% generalization.

**Experiment results:** Recall that the feature sets  $S$  and  $S^c$  correspond to the set of ten 5-slab and 7-slab coordinates in the AdvMS-(5,7) dataset respectively. As shown in Figure 7, we evaluate the standard (blue),  $\epsilon$ -robust (orange),  $S$ -randomized (green) and  $S^c$ -randomized (red) accuracies, defined in Section 3, of FCNs with width  $\{200, 1000\}$  and depth 2 that are adversarially trained with  $\ell_2$  perturbation norm  $\epsilon \leq \gamma_{\text{data}}$ . The dashed and solid purple vertical bars denote  $\gamma_S$  and  $\gamma_{\text{data}}$  respectively. We make two key observations:

- Adversarial trained FCNs do not learn maximum-margin classifiers. When  $\epsilon \leq 0.1$ , (200,2)-FCNs learn classifiers that attain 100% standard and  $\epsilon$ -robust accuracies. However, when  $\epsilon \geq 0.2$ , due

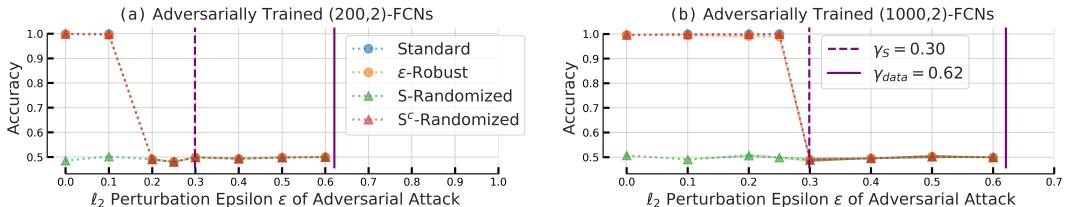


Figure 7: Adversarially trained FCNs on AdvMS-(5,7) data exhibit adversarial robustness to some extent, but are (a) unable to mitigate extreme simplicity bias, as shown by the  $\{S, S^c\}$ -randomized accuracies and (b) do not learn maximum-margin classifiers that attain optimal adversarial robustness (i.e., 100%  $\gamma_{\text{data}}$ -robust accuracy). See Appendix E.2 for more detail.

to optimization-related issues, adversarially trained (200, 2)-FCNs are unable to learn a non-trivial classifier that obtains more than 50% standard and  $\epsilon$ -robust accuracy. Increasing the model width from 200 to 1000 improves adversarial robustness to some extent—adversarially trained (1000, 2)-FCNs learn classifiers with 100% standard and robust accuracies when  $\epsilon \leq 0.25$ . However, when  $\epsilon \geq \gamma_S = 0.3$  (dashed purple line), adversarially trained (2000, 2)-FCNs are unable to learn  $\epsilon$ -robust classifiers as well. We note that further increasing the model width to 2000 does not improve robustness. Consequently, adversarial training does not result in maximum-margin classifiers that have optimal adversarial robustness (i.e., classifiers with 100%  $\gamma_{\text{data}}$ -robust accuracy) on AdvMS-(5, 7) data. These results reconcile two phenomena observed in practice: larger capacity models can improve adversarial robustness [81], but large-epsilon adversarial training can “fail” and result in trivial classifiers due to optimization-related issues [66].

- **Adversarial training does not mitigate extreme SB.** The  $\{\mathbf{S}, \mathbf{S}^c\}$ -randomized accuracies of adversarially trained FCNs collectively show that adversarial training does not mitigate extreme SB. When the perturbation budget  $\epsilon \leq \gamma_S$ , adversarially trained (2000, 2)-FCNs exhibit robustness by exclusively relying on *multiple* “simple” 5-slab coordinates. That is, randomizing  $\mathbf{S}$  drops the model accuracy to 50%, but randomizing the more complex 7-slab coordinates has no effect on model accuracy. Conversely, when  $\epsilon \geq \gamma_S$ , classifiers with 100%  $\epsilon$ -robust accuracy must rely on features in  $\mathbf{S}$  and  $\mathbf{S}^c$ . However, as shown in Figure 7, when  $\epsilon \geq \gamma_S$ , adversarial training fails and results in trivial classifiers that attain 50% standard and robust accuracy.

To summarize, the experiment above shows that adversarially trained FCNs do not mitigate simplicity bias or learn maximum-margin classifiers that are optimally robust to  $\ell_2$  adversarial attacks.

### E.2.2 Adversarially training CNNs on MNIST-CIFAR data

Recall that the MNIST-CIFAR images, described in Section 3, are vertically concatenations of “simple” MNIST images and the more complex CIFAR10 images. Next, we show that models that are  $\ell_\infty$ -adversarially trained on the MNIST-CIFAR data improve  $\epsilon$ -robust accuracy (defined in Section 3) but do not achieve the best possible  $\epsilon$ -robust accuracy due to extreme SB.

Table 11 evaluates the standard,  $\epsilon$ -robust and CIFAR10-randomized accuracies of SGD-trained and adversarially trained MobileNetV2, ResNet50 and DenseNet121 using the MNIST-CIFAR dataset. First, we observe that adversarial training with perturbation norm 0.3 significantly improves  $\epsilon$ -robust accuracies over those of SGD-trained models, without degrading the models’ standard test accuracies. However, the CIFAR10-randomized accuracies indicate that the adversarial training does not lead to reliance on the CIFAR10 block—adversarially trained CNNs continue to remain invariant to the CIFAR10 block even though it is almost fully predictive of the label. Consequently, these results suggest that adversarial training improves robustness but does not achieve the best  $\epsilon$ -robust accuracy on the MNIST-CIFAR dataset.

To summarize, our experiments on AdvMS-(5, 7) and MNIST-CIFAR datasets show that while adversarial training does improve the  $\epsilon$ -robust accuracy over that of SGD-trained model, adversarially trained models continue to remain susceptible to extreme SB and consequently do not achieve maximum possible adversarial robustness.

Model	$\ell_\infty$ budget $\epsilon$	Test Accuracy		$\epsilon$ -Robust Accuracy		CIFAR10-Randomized Accuracy	
		Standard SGD	$\ell_\infty$ Adv. Training	Standard SGD	$\ell_\infty$ Adv. Training	Standard SGD	$\ell_\infty$ Adv. Training
MobileNetV2	0.30	$0.999 \pm 0.001$	$0.999 \pm 0.000$	$0.000 \pm 0.000$	$0.991 \pm 0.000$	$0.493 \pm 0.005$	$0.493 \pm 0.001$
DenseNet121	0.30	$1.000 \pm 0.000$	$0.999 \pm 0.000$	$0.000 \pm 0.000$	$0.981 \pm 0.003$	$0.494 \pm 0.005$	$0.501 \pm 0.003$
ResNet50	0.30	$1.000 \pm 0.000$	$0.999 \pm 0.001$	$0.001 \pm 0.000$	$0.982 \pm 0.002$	$0.501 \pm 0.001$	$0.499 \pm 0.002$

Table 11: Adversarial training on MNIST-CIFAR: The table above presents standard,  $\epsilon$ -robust and CIFAR10-randomized accuracies of SGD-trained and adversarially trained MobileNetV2, DenseNet121 and ResNet50 models. While adversarial training significantly improves  $\epsilon$ -robust accuracy, it does not encourage models to learn complex features (CIFAR10 block in this case). The CIFAR10-randomized accuracies indicate that adversarially trained models do not mitigate extreme SB, as they exclusively rely on the MNIST block.

## F Proof of Theorem 1

In this section, we first re-introduce the data distribution and theorem. Then, we describe the proof sketch and notation, before moving on to the proof.

**Linear-Slab-Noise (LSN) data:** The LSN dataset is a stylized version of LMS-k that is amenable to theoretical analysis. In LSN, conditioned on the label  $y$ , the first and second coordinates of  $x$  are *singleton* linear and 3-slab blocks: linear and 3-slab blocks have support on  $\{-1, 1\}$  and  $\{-1, 0, 1\}$  respectively. The remaining coordinates are standard gaussians and not predictive of the label. Each data point  $(x_i, y_i) \in \mathbb{R}^d \times \{-1, 1\}$  from LSN can be sampled as follows:

$$\begin{aligned} y_i &= \pm 1, \text{ w.p. } 1/2, \quad \varepsilon_i = \pm 1, \text{ w.p. } 1/2, \\ x_{i1} &= y_i && \text{(Linear coordinate),} \\ x_{i2} &= \left(\frac{y_i + 1}{2}\right)\varepsilon_i && \text{(Slab coordinate),} \\ x_{i,3:d} &\sim \mathcal{N}(0, I_{d-2}) && \text{(d-2 Noise coordinates).} \end{aligned}$$

According to Theorem 1 (re-stated), one-hidden-layer ReLU neural networks trained with standard mini-batch gradient descent (GD) on the LSN dataset provably learns a classifier that *exclusively* relies on the “simple” linear coordinate, thus exhibiting simplicity bias at the cost of margin.

**Theorem 1.** Let  $f(x) = \sum_{j=1}^k v_j \cdot \text{ReLU}(\sum_{i=1}^d w_{ij}x_i)$  denote a one-hidden-layer neural network with  $k$  hidden units and ReLU activations. Set  $v_j = \pm 1/\sqrt{k}$  w.p.  $1/2 \forall j \in [k]$ . Let  $\{(x^i, y^i)\}_{i=1}^m$  denote i.i.d. samples from LSN where  $m \in [cd^2, d^\alpha/c]$  for some  $\alpha > 2$ . Then, given  $d > \Omega(\sqrt{k} \log k)$  and initial  $w_{ij} \sim \mathcal{N}(0, \frac{1}{dk \log^4 d})$ , after  $O(1)$  iterations, mini-batch gradient descent (over  $w$ ) with hinge loss, step size  $\eta = \Omega(\log d)^{-1/2}$ , mini-batch size  $\Theta(m)$ , satisfies:

- Test error is at most  $1/\text{poly}(d)$
- The learned weights of hidden units  $w_{ij}$  satisfy:

$$\underbrace{|w_{1j}| = \frac{2}{\sqrt{k}} \left(1 - \frac{c}{\sqrt{\log d}}\right) + O\left(\frac{1}{\sqrt{dk} \log d}\right)}_{\text{Linear Coordinate}}, \underbrace{|w_{2,j}| = O\left(\frac{1}{\sqrt{dk} \log d}\right)}_{\text{3-Slab Coordinate}}, \underbrace{\|w_{3:d,j}\| = O\left(\frac{1}{\sqrt{k} \log d}\right)}_{\text{d-2 Noise Coordinates}}$$

with probability greater than  $1 - \frac{1}{\text{poly}(d)}$ . Note that  $c$  is a universal constant.

**Proof Sketch** Since the number of iterations  $t = O(1)$ , we partition the dataset into  $t$  minibatches each of size  $n := m/t$  samples. This means that each iteration uses a fresh batch of  $n$  samples and the  $t$  iterations together form a single pass over the data. The overall outline of the proof is as follows. If the step size is  $\eta$ , then for  $t \lesssim \frac{4}{\eta}$  iterations, with probability  $\geq 1 - \frac{1}{\text{poly}(d)}$ ,

- Lemma 2 shows that the hinge loss is “active” (i.e.,  $yf(x) < 1$ ) for all data points in a given batch.
- Under this condition, we derive closed-form expressions for *population* gradients in Lemmas 4, 5 and 6.
- Lemma 1 uses the above lemmas to establish precise estimates of the linear, slab and noise coordinates for all iterations until  $t$ .

The proof is organized as follows. Appendix F.1 presents the main lemmas that will directly lead to Theorem 1. Appendix F.2 derives closed form expressions for population gradients and Appendix F.3 presents auxiliary lemmas that are useful in the main proofs.

**Notation** Recall that  $f(x) = \sum_{j=1}^k v_j \cdot \text{ReLU}(\sum_{i=1}^d w_{ij}x_i) = v^T \text{ReLU}(W^T x)$  where  $W \in \mathbb{R}^{d \times k}$  and  $v \in \mathbb{R}^k$ . Note that  $w_i = [w_{1i} w_{2i}, \dots w_{di}]^T$  is the  $i^{\text{th}}$  column in  $W$ . Let  $\bar{w}_i$  and  $\bar{x}_j$  denote the  $w_{3:d,i}$  and  $x_{3:d,j}$  respectively. Also, let  $\mathcal{S}_n = \{(x_i, y_i)\}_{i=1}^n$  denote a set of  $n$  i.i.d. points randomly sampled from LSN. For simplicity, we also assume  $|\{i : v_i = 1/\sqrt{k}\}| = |\{i : v_i = -1/\sqrt{k}\}| = k/2$ . We can now define the loss function as  $\mathcal{L}_f(\mathcal{S}_n) = 1/n \sum_i^n \ell(x_i, y_i)$ , where  $\ell(x, y) = \max(0, 1 - yf(x))$  denotes the hinge loss. For notational simplicity, we use  $X = \mu \pm \delta$  and  $|X - \mu| \leq \delta$  interchangeably. Also let  $\varphi$  and  $\phi$  denote the probability density function and cumulative distribution function of standard normal distribution.

*Proof of Theorem 1.* The proof directly follows from Lemma 1 and Lemma 2. In Lemma 1, we show that the weights in the linear coordinate are  $\Omega(\sqrt{d})$  larger than the weights in the slab and noise coordinates. Applying Lemma 1 at  $\hat{t} = \lfloor \frac{4}{\eta}(1 - \frac{c_n}{\sqrt{\log d}}) \rfloor$  gives the following result:

$$w_{1i}^{(\hat{t})} \stackrel{(a)}{=} \frac{2}{\sqrt{k}}(1 - \frac{c_n}{\sqrt{\log d}}) + O(\frac{1}{\sqrt{dk} \log d}) \text{ and } |w_{2i}^{(\hat{t})}| \stackrel{(a)}{=} O(\frac{1}{\sqrt{dk} \log d}) \text{ and } \|\bar{w}_i^{(\hat{t})}\| \stackrel{(a)}{=} O(\frac{1}{\sqrt{k} \log d})$$

where (a) is due to  $c_0(1 + \hat{c})^t \leq c_0 e^{\hat{c}t} \leq c_0 e^1 = O(1)$ .

The  $0 - 1$  error of the function  $f$  at timestep  $\hat{t}$  is small as well, because we can directly use Lemma 2 to get  $\Pr(yf(x) < 0) = 2/c^3 d^6$ . Therefore, the  $0 - 1$  error is at most  $\frac{2}{c^3 d^6} = O(\frac{1}{d^6})$ . ■

## F.1 Proof by Induction

In this section, we use proof by induction to show that for the first  $t = O(1/\eta)$  steps, (1) the hinge loss is “active” for all data points (Lemma 2) and (2) hidden layer weights in the linear coordinate are  $\Omega(\sqrt{d})$  larger than the hidden layer weights in the slab and noise coordinates (Lemma 1).

**Lemma 1.** Let  $|\mathcal{S}_n| \in [cd^2, d^\alpha / c]$  and initialization  $w_{ij} \sim \mathcal{N}(0, 1/dk \log^2 d)$ . Also let  $\hat{c} = \eta/4$ ,  $c_0 = 2$  and  $c_n = 5\sqrt{c}c_0(1 + \hat{c})^t$ . Then, for all  $t \leq \frac{4}{\eta}(1 - c_n/\sqrt{\log d})$ ,  $d \geq \exp((8c_n/\eta)^2)$ ,  $\sqrt{d/\log^3(d)} > 24\sqrt{k}/c_0 c$  and  $i \in [k]$ , w.p. greater than  $1 - O(\frac{1}{d^2})$ , we have:

$$y_i f(x_i) \leq 1 \quad \forall (x_i, y_i) \in \mathcal{S}_n \quad (1)$$

$$w_{1i}^{(t)} = \frac{t\eta v_i}{2} \pm \frac{c_0(1 + \hat{c})^t}{\sqrt{dk} \log d} \quad (2)$$

$$|w_{2i}^{(t)}| \leq \frac{c_0(1 + \hat{c})^t}{\sqrt{dk} \log d} \quad (3)$$

$$\|\bar{w}^{(t)}\|_2 \leq \frac{c_0(1 + \hat{c})^t}{\sqrt{k} \log d} \quad (4)$$

*Proof.* First, we prove that equations (2), (3) & (4) hold at initialization (i.e.,  $t = 0$ ) with high probability. Using 7 and 1:

$$\max_{i \in \{1, 2\}} \max_{j \leq k} |w_{ij}| \leq \frac{2}{\sqrt{dk} \log d} \text{ and } \max_{i \leq k} \|\bar{w}_i\| \leq \frac{2}{\sqrt{k} \log d} \text{ w.p. } 1 - \frac{2}{d^4}$$

Therefore,  $w_{1i} = \frac{(0)\eta v_i}{2} \pm \frac{c_0(1 + \hat{c})^0}{\sqrt{dk} \log d}$  and  $|w_{2i}| \leq \frac{c_0(1 + \hat{c})^0}{\sqrt{dk} \log d}$  and  $\|\bar{w}_i\| \leq \frac{c_0(1 + \hat{c})^0}{\sqrt{k} \log d}$ . Since equations (2), (3) & (4) hold at  $t = 0$ , we can use Lemma 2 to show that the hinge loss is “active” with high probability:

$$y_i f(x_i) = \pm \frac{c_n}{\sqrt{\log d}} < 1 \quad \text{when } d \geq \exp(c_n^2)$$

Now, we assume that the inductive hypothesis—equations (1), (2), (3) and (4)—is true after every timestep  $\tau$  where  $\tau \in \{0, \dots, t\}$ .

We now prove that the inductive hypothesis is true at timestep  $t + 1$ , after applying gradient descent using the  $(t + 1)^{th}$  batch. Since  $z(1)$  holds at timestep  $t$ , we can use the closed-form expression of the gradient along the linear coordinate (lemma 4) to prove that equation (2) holds at timestep  $t + 1$

as well:

$$\begin{aligned}
w_{1i}^{(t+1)} &= w_{1i}^{(t)} + \frac{\eta v_i}{4} \left[ 2 + \phi\left(\frac{w_{1i} + w_{2i}}{\|\bar{w}_i\|}\right) + \phi\left(\frac{w_{1i} - w_{2i}}{\|\bar{w}_i\|}\right) - 2\phi\left(\frac{w_{1i}}{\|\bar{w}_i\|}\right) \right] \pm \frac{5\eta v_i}{d} \sqrt{\frac{\log(cd^2)}{c}} \\
&= w_{1i}^{(t)} + \frac{\eta v_i}{2} + \frac{\eta v_i}{2} |w_{2i}^{(t)}| \cdot \max_{|\delta| \leq |w_{2i}^{(t)}|} \frac{1}{\|\bar{w}^t\|} \varphi\left(\frac{w_{1i}^{(t)} + \delta}{\|\bar{w}_i^{(t)}\|}\right) \pm \frac{5\eta v_i}{d} \sqrt{\frac{\log(cd^2)}{c}} \\
&\stackrel{(a)}{=} \frac{t\eta v_i}{2} \pm \frac{c_0(1 + \hat{c})^t}{\sqrt{dk} \log d} + \frac{\eta v_i}{2} + \frac{\eta v_i}{2} \frac{c_0(1 + \hat{c})^t}{\sqrt{dk} \log d} \pm \frac{5\eta v_i}{d} \sqrt{\frac{\log(cd^2)}{c}} \\
&\stackrel{(b)}{=} \frac{(t+1)\eta v_i}{2} \pm \frac{c_0(1 + \hat{c})^t}{\sqrt{dk} \log d} \pm \eta v_i \frac{c_0(1 + \hat{c})^t}{\sqrt{dk} \log d} = \frac{(t+1)\eta v_i}{2} \pm \frac{c_0(1 + \hat{c})^t(1 + \eta v_i)}{\sqrt{dk} \log d} \\
&\stackrel{(c)}{=} \frac{(t+1)\eta v_i}{2} \pm \frac{c_0(1 + \hat{c})^{t+1}}{\sqrt{dk} \log d}
\end{aligned}$$

where (a) is via equation (12) in Lemma 3, (b) is because  $d/\log^3(d) \geq 20/c_0 e^1 \sqrt{c}$  and (c) is due to  $\eta v_i \leq \hat{c}$ .

Similarly, since equation (1) holds at timestep  $t$  (via the inductive hypothesis), we can use the closed-form expression of the gradient along the slab coordinate (lemma 5) to show that the weights in the slab (i.e., second) coordinate are small (equation (3)) at timestep  $t+1$  as well:

$$\begin{aligned}
w_{2i}^{(t+1)} &= w_{2i}^{(t)} + \frac{\eta v_i}{4} \left[ \phi\left(\frac{w_{1i} + w_{2i}}{\|\bar{w}_i\|}\right) - \phi\left(\frac{w_{1i} - w_{2i}}{\|\bar{w}_i\|}\right) \right] \pm \frac{5\eta v_i}{d} \sqrt{\frac{\log(cd^2)}{c}} \\
&= w_{2i}^{(t)} + \frac{\eta v_i}{2} |w_{2i}^{(t)}| \cdot \max_{|\delta| \leq |w_{2i}^{(t)}|} \frac{1}{\|\bar{w}^t\|} \varphi\left(\frac{w_{1i}^{(t)} + \delta}{\|\bar{w}_i^{(t)}\|}\right) \pm \frac{5\eta v_i}{d} \sqrt{\frac{\log(cd^2)}{c}} \\
&\stackrel{(a)}{=} \pm \frac{c_0(1 + \hat{c})^t}{\sqrt{dk} \log d} \pm \frac{\eta v_i}{2} \frac{c_0(1 + \hat{c})^t}{\sqrt{dk} \log d} \pm \frac{\eta v_i}{2} \frac{c_0(1 + \hat{c})^t}{\sqrt{dk} \log d} \leq \pm \frac{c_0(1 + \hat{c})^{t+1}}{\sqrt{dk} \log d}
\end{aligned}$$

where (a) is due to equations (11) in Lemma 3, (3) and  $d/\log^3(d) \geq 20/c_0 e^1 \sqrt{c}$ .

Finally, we can use the closed-form expression of the gradient along the noise coordinate (lemma 6) to prove that the norm of the gradient along the noise coordinates (i.e., coordinates 3 to  $d$ ) is small (equation (4)) at timestep  $t+1$ :

$$\bar{w}_i^{(t+1)} = \bar{w}_i^{(t+1)} + \underbrace{\frac{\eta v_i}{4} \left[ \varphi\left(\frac{w_{1i} + w_{2i}}{\|\bar{w}_i\|}\right) + \varphi\left(\frac{w_{1i} - w_{2i}}{\|\bar{w}_i\|}\right) - 2\varphi\left(\frac{w_{1i}}{\|\bar{w}_i\|}\right) \right]}_{\bar{\mathcal{G}}_1} \frac{\bar{w}_i^{(t)}}{\|\bar{w}_i\|} \pm \underbrace{\frac{3\eta|v_i| \log(\sqrt{cd})}{\sqrt{cd}} \frac{\bar{w}_i}{\|\bar{w}_i\|} \pm \frac{6\eta|v_i|}{\sqrt{cd}} u_i^\perp}_{\bar{\mathcal{G}}_2}$$

We first show that the the first part of the noise gradient,  $\bar{\mathcal{G}}_1$ , is at most  $\eta v_i/2$ :

$$\begin{aligned}
&\frac{\eta v_i}{4} \left[ \varphi\left(\frac{w_{1i} + w_{2i}}{\|\bar{w}_i\|}\right) + \varphi\left(\frac{w_{1i} - w_{2i}}{\|\bar{w}_i\|}\right) - 2\varphi\left(\frac{w_{1i}}{\|\bar{w}_i\|}\right) \right] \\
&= \frac{\eta v_i}{4} \underbrace{\frac{1}{\|\bar{w}_i^{(t)}\|} \varphi\left(\frac{w_{1i}^{(t)}}{\|\bar{w}_i^{(t)}\|}\right)}_{\leq 1 \text{ (see eq. 13 in Lemma 3)}} \underbrace{\left[ 2 - \varphi\left(\frac{w_{2i}^{(t)}}{\|\bar{w}_i^{(t)}\|}\right) \left( \exp\left(\frac{w_{1i}^{(t)} w_{2i}^{(t)}}{\|\bar{w}_i^{(t)}\|^2}\right) + \exp\left(\frac{-w_{1i}^{(t)} w_{2i}^{(t)}}{\|\bar{w}_i^{(t)}\|^2}\right) \right) \right]}_{\leq 2} \leq \frac{\eta v_i}{2}
\end{aligned}$$

Next, we show that the  $\ell_2$  norm of the second part of the noise gradient,  $\|\bar{\mathcal{G}}_2\|$ , is  $O(1/\sqrt{d})$ :

$$\|B\| \leq 3\eta|v_i| \frac{\log(\sqrt{cd})}{\sqrt{cd}} + \frac{6\eta|v_i|}{\sqrt{cd}} \leq \frac{12\eta|v_i|}{\sqrt{cd}}$$

Now, we can use the upper bounds on  $\mathcal{G}_1$  and  $\mathcal{G}_2$  to show that the  $\ell_2$  norm of the gradient along the noise gradients is small as well:

$$\begin{aligned}
\|\bar{w}_i^{(t+1)}\| &\leq \|\bar{w}_i^{(t)}\| + \frac{\eta v_i}{2} \|\bar{w}_i^{(t)}\| + \frac{12\eta|v_i|}{\sqrt{cd}} \stackrel{(a)}{\leq} \|\bar{w}_i^{(t)}\| + \frac{\eta v_i}{2} \|\bar{w}_i^{(t)}\| + \frac{\eta v_i}{2} \|\bar{w}_i^{(t)}\| \\
&\stackrel{(b)}{\leq} \frac{c_0(1 + \hat{c})^t(1 + \eta v_i)}{\sqrt{k} \log d} \stackrel{(c)}{\leq} \frac{c_0(1 + \hat{c})^{t+1}}{\sqrt{k} \log d}
\end{aligned}$$

where (a) is because  $d/\log d \geq (24\sqrt{k}/c_0 c)^2$ , (b) is due to equation (4) and (c) is because  $\eta v_i \leq \hat{c}$ . ■

Since equations (2), (3) & (4) hold at timestep  $t$  (from Lemma 1), we can show that the hinge loss is positive (i.e.,  $y_i f(x_i) < 1$ ) for all data points with high probability as well.

**Lemma 2.** Let  $\mathcal{S}_n$  denote a set of  $n \in [cd^2, d^\alpha/c]$  i.i.d. samples from LSN, where  $\alpha > 2$  and  $c > 1$ . Suppose equations (2), (3) & (4) hold at timestep  $t$ . Also let  $d \geq \exp((\frac{8c_n}{\eta})^2)$  where  $c_n = 5\sqrt{\alpha}c_0(1 + \hat{c})^t$ . Then, w.p. greater than  $1 - \frac{2}{c^3d^6}$ , we have:

$$y_i f(x_i) = \frac{t\eta}{4} \pm \frac{c_n}{\sqrt{\log d}} = (t \pm 1/2) \frac{\eta}{4} \quad \forall (x_i, y_i) \in \mathcal{S}_n \quad (5)$$

*Proof.* We use equations (2), (3) & (4) to obtain simplify the dot product between  $w_i^{(t)}$  &  $x_j$  and the indicator  $\mathbb{1}\{w_i^{(t)} \cdot x_j \geq 0\}$ . First, we show that the dot product between  $w_i^{(t)}$  and  $x_j$  is in the band  $\frac{t\eta v_i y_j}{2} \pm \frac{c_n}{\sqrt{k \log d}}$  with high probability:

$$\begin{aligned} w_i^{(t)} \cdot x_j &= w_{1i}^{(t)} y_j + w_{2i}^{(t)} \frac{y_j + 1}{2} \varepsilon_j + \bar{w}_i^{(t)} \cdot \bar{x}_j \stackrel{(a)}{=} w_{1i}^{(t)} y_j + w_{2i}^{(t)} \frac{y_j + 1}{2} \varepsilon_j + \|\bar{w}_i^{(t)}\| Z_j \\ &\stackrel{(b)}{=} w_{1i}^{(t)} y_j + w_{2i}^{(t)} \frac{y_j + 1}{2} \varepsilon_j \pm \|\bar{w}_i^{(t)}\| \sqrt{8\alpha \log d} \quad \text{w.p. } 1 - \frac{2}{d^6} \\ &= \frac{t\eta v_i y_j}{2} \pm \frac{c_0(1 + \hat{c})^t}{\sqrt{dk \log d}} (y_j + \frac{y_j + 1}{2} \varepsilon_j) \pm \frac{c_0(1 + \hat{c})^t}{\sqrt{k \log d}} \sqrt{8\alpha \log d} \quad \text{via eq. 2, 3, 4} \\ &\stackrel{(c)}{=} \frac{t\eta v_i y_j}{2} \pm \frac{2c_0(1 + \hat{c})^t}{\sqrt{dk \log d}} + \frac{3\sqrt{\alpha}c_0(1 + \hat{c})^t}{\sqrt{k \log d}} = \frac{t\eta v_i y_j}{2} \pm \frac{c_n}{\sqrt{k \log d}} \quad (6) \\ &= (ty_j \pm 1/4) \frac{\eta v_i}{2} \quad \text{when } d \geq \exp((\frac{8c_n}{\eta})^2) \quad (7) \end{aligned}$$

where (a) is because  $\bar{w}_i^{(t)} \cdot \bar{x}_j = \|\bar{w}_i^{(t)}\| \mathcal{N}(0, 1)$ , (b) is via lemma 7 &  $c > 1$ , and (c) is because  $(y_j + \frac{y_j + 1}{2} \varepsilon_j) < 2$ . Next, when  $d \geq \exp((\frac{8c_n}{\eta})^2)$ , we can simplify  $\mathbb{1}\{w_i^{(t)} \cdot x_j \geq 0\}$  as follows:

$$\mathbb{1}\{w_i^{(t)} \cdot x_j \geq 0\} \stackrel{\text{eq. 7}}{\leq} \mathbb{1}\{(ty_j \pm 1/4) \frac{\eta v_i}{2} \geq 0\} \leq \begin{cases} 1, & \text{if } t = 0 \\ 1, & \text{if } t > 0 \text{ and } y_j v_i \geq 0 \leq \mathbb{1}\{t = 0 \vee yv_i \geq 0\} \\ 0, & \text{if } t > 0 \text{ and } y_j v_i < 0 \end{cases} \quad (8)$$

We can now use equations (6) & (8) to show that  $y_j f^{(t)}(x_j)$  is in the band  $t\eta/4 \pm O(1/\sqrt{\log d})$  with high probability:

$$\begin{aligned} y_j f^{(t)}(x_j) &= \sum_{i=1}^k y_j v_i \cdot \text{ReLU}(w_i \cdot x_j) = \sum_{i=1}^k v_i \mathbb{1}\{t = 0 \vee yv_i \geq 0\} \left( \frac{t\eta v_i}{2} \pm \frac{c_n}{\sqrt{k \log d}} \right) \\ &= \sum_{i=1}^k \mathbb{1}\{t = 0 \vee yv_i \geq 0\} \left( \frac{t\eta}{2k} \pm \frac{c_n}{k \sqrt{\log d}} \right) \stackrel{(a)}{=} \begin{cases} \pm k \frac{c_n}{k \sqrt{\log d}}, & \text{if } t = 0 \\ \frac{k}{2} \left( \frac{t\eta}{2k} \pm \frac{c_n}{k \sqrt{\log d}} \right), & \text{if } t > 0 \end{cases} \quad (9) \\ &= \frac{t\eta}{4} \pm \frac{c_n}{\sqrt{\log d}} \stackrel{(b)}{=} (t \pm 1/2) \frac{\eta}{4} \end{aligned}$$

where (a) is due to  $|\{v_i \mid v_i > 0\}| = |\{v_i \mid v_i < 0\}| = k/2$  and (b) follows from  $c_n/\sqrt{\log d} \leq \eta/8$  when  $d \geq \exp((8c_n/\eta)^2)$  ■

**Lemma 3.** If equations (2), (3) & (4) hold at timestep  $t$ ,  $d > \exp((\frac{4c_0 e^1}{\eta})^2)$  and  $d/\log d > \sqrt{k}$ , we have:

$$\max_{|\delta| \leq |w_{2i}^{(t)}|} \frac{1}{\|\bar{w}^{(t)}\|} \varphi\left(\frac{w_{1i}^{(t)} + \delta}{\|\bar{w}_i^{(t)}\|}\right) \leq 1 \quad (10)$$

$$\left| \phi\left(\frac{w_{1i}^{(t)} + w_{2i}^{(t)}}{\|\bar{w}^{(t)}\|}\right) - \phi\left(\frac{w_{1i}^{(t)} - w_{2i}^{(t)}}{\|\bar{w}^{(t)}\|}\right) \right| \leq \frac{2c_0(1 + \hat{c})^t}{\sqrt{dk \log d}} \quad (11)$$

$$\left| \phi\left(\frac{w_{1i}^{(t)} + w_{2i}^{(t)}}{\|\bar{w}^{(t)}\|}\right) - \phi\left(\frac{w_{1i}^{(t)}}{\|\bar{w}^{(t)}\|}\right) \right| \leq \frac{c_0(1 + \hat{c})^t}{\sqrt{dk \log d}} \quad (12)$$

$$\frac{1}{\|\bar{w}^{(t)}\|} \varphi\left(\frac{w_{1i}^{(t)}}{\|\bar{w}_i^{(t)}\|}\right) \leq \frac{c_0(1 + \hat{c})^t}{\sqrt{dk \log d}} \quad (13)$$

*Proof.* Let  $g_z(x) = \frac{1}{x}\varphi(\frac{z}{x})$  and  $h(x) = \max_{|\delta| \leq |w_{2i}^{(t)}|} \frac{1}{x}\varphi(\frac{w_{1i}^{(t)} + \delta}{x}) = \max_{|\delta| \leq |w_{2i}^{(t)}|} g_{w_{1i}^{(t)} + \delta}(x)$ .

To prove Equation (10), we show that an upper bound on  $\|w^{(t)}\|$  is less than a lower bound on  $\arg \max_x h(x)$ , which subsequently implies that  $h(\|w^{(t)}\|) < \max_x h(x)$  because  $h$  is an increasing function for all  $|x| \leq \arg \max_x h(x)$ .

First, we find the maximizer  $x^*$  of  $h(x)$  as follows:

$$\max_x h(x) = \max_x \max_{|\delta| \leq |w_{2i}^{(t)}|} g_{w_{1i}^{(t)} + \delta}(x) \stackrel{(a)}{=} \max_{|\delta| \leq |w_{2i}^{(t)}|} \frac{e^{-1}}{|w_{1i}^{(t)} + \delta|} \quad \text{when } x^* = |w_{1i}^{(t)} + \delta|$$

where (a) follows from lemma 11. Next, we lower bound the maximizer  $x^*$  of  $h(x)$ :

$$\begin{aligned} x^* = |w_{1i}^{(t)} + \delta| &\geq |w_{1i}^{(t)} + w_{2i}^{(t)}| \geq \left| |w_{1i}^{(t)}| - |w_{2i}^{(t)}| \right| \stackrel{(a)}{\geq} \left| \left| \frac{t\eta v_i}{2} \pm \frac{c_0(1+\hat{c})^t}{\sqrt{dk} \log d} \right| - \left| \frac{c_0(1+\hat{c})^t}{\sqrt{dk} \log d} \right| \right| \\ &\stackrel{(b)}{\geq} \left| \left| \frac{t\eta v_i}{2} \pm \frac{\eta v_i}{8} \right| - \left| \frac{\eta v_i}{8} \right| \right| \geq \left| t - \frac{1}{2} \right| \frac{\eta v_i}{2} \geq \frac{\eta v_i}{4} \end{aligned}$$

where (a) follows from the weights in the linear and slab coordinate at timestep  $t$  (equations (2) & (3)) and (b) is because  $\frac{c_0(1+\hat{c})^t}{\sqrt{dk} \log d} \leq \frac{\eta v_i}{8}$  when  $\sqrt{d} \geq \frac{8c_0 e^1}{\eta}$ . Therefore,  $\arg \max_x h(x) \geq \eta v_i / 4$ . We can use the upper bound on the  $\ell_2$  norm of the gradient along the noise coordinates (equation (4)) and  $d \geq \exp(\frac{4c_0 e^1}{\eta})$  to show that  $\|\bar{w}_i^{(t)}\|$  is less than  $x^*$ :

$$\|\bar{w}_i^{(t)}\| \leq \frac{c_0(1+\hat{c})^t}{\sqrt{dk} \log d} \leq \frac{\eta v_i}{4} \leq \arg \max_x h(x)$$

From lemma 11, we know that  $h(x)$  is an increasing function for all  $|x| < x^*$ . This implies that  $h(\|w_i^{(t)}\|) \leq h(\frac{c_0(1+\hat{c})^t}{\sqrt{dk} \log d}) \leq h(\frac{\eta v_i}{4}) \leq h(x^*)$ . Therefore, when  $d \geq \exp((\frac{4c_0 e^1}{\eta})^2)$  and  $d/\log d \geq \sqrt{k}$ , we obtain the desired result as follows:

$$\max_{|\delta| \leq |w_{2i}^{(t)}|} \frac{1}{\|\bar{w}^{(t)}\|} \varphi\left(\frac{w_{1i}^{(t)} + \delta}{\|\bar{w}_i^{(t)}\|}\right) = h(\|w_i^{(t)}\|) \leq h\left(\frac{c_0 e^1}{\sqrt{dk} \log d}\right) \leq \frac{\sqrt{k} \log d}{c_0 e^1} \frac{1}{d^{(\frac{\eta}{4c_0 e^1})^2 \log d}} \leq 1$$

Now, we can prove equations (11), (12) and (13) using equation (10) as follows:

$$\left| \phi\left(\frac{w_{1i}^{(t)} + w_{2i}^{(t)}}{\|\bar{w}^{(t)}\|}\right) - \phi\left(\frac{w_{1i}^{(t)} - w_{2i}^{(t)}}{\|\bar{w}^{(t)}\|}\right) \right| \leq 2|w_{2i}^{(t)}| \cdot \max_{|\delta| \leq |w_{2i}^{(t)}|} \frac{1}{\|\bar{w}^{(t)}\|} \varphi\left(\frac{w_{1i}^{(t)} + \delta}{\|\bar{w}_i^{(t)}\|}\right) \stackrel{(a)}{\leq} \frac{2c_0(1+\hat{c})^t}{\sqrt{dk} \log d} \quad (14)$$

$$\left| \phi\left(\frac{w_{1i}^{(t)} + w_{2i}^{(t)}}{\|\bar{w}^{(t)}\|}\right) - \phi\left(\frac{w_{1i}^{(t)}}{\|\bar{w}^{(t)}\|}\right) \right| \leq |w_{2i}^{(t)}| \cdot \max_{|\delta| \leq |w_{2i}^{(t)}|} \frac{1}{\|\bar{w}^{(t)}\|} \varphi\left(\frac{w_{1i}^{(t)} + \delta}{\|\bar{w}_i^{(t)}\|}\right) \stackrel{(a)}{\leq} \frac{c_0(1+\hat{c})^t}{\sqrt{dk} \log d} \quad (15)$$

$$\frac{1}{\|\bar{w}^{(t)}\|} \varphi\left(\frac{w_{1i}^{(t)}}{\|\bar{w}_i^{(t)}\|}\right) \leq \max_{|\delta| \leq |w_{2i}^{(t)}|} \frac{1}{\|\bar{w}^{(t)}\|} \varphi\left(\frac{w_{1i}^{(t)} + \delta}{\|\bar{w}_i^{(t)}\|}\right) \stackrel{(a)}{\leq} \frac{c_0(1+\hat{c})^t}{\sqrt{dk} \log d} \quad (16)$$

where (a) is due to equation (4). ■

## F.2 Closed-form Gradient Expressions

In this section, we provide closed-form expressions for gradients along the linear, slab and noise coordinates:  $\nabla_{w_{1i}} \mathcal{L}_f(\mathcal{S}_n)$ ,  $\nabla_{w_{2i}} \mathcal{L}_f(\mathcal{S}_n)$  and  $\nabla_{\bar{w}_i} \mathcal{L}_f(\mathcal{S}_n)$ . First, we provide a closed-form expression for the gradient along the linear coordinate:

**Lemma 4.** If  $n > cd^2$  and  $y_i f(x_i) < 1 \forall (x_i, y_i) \in \mathcal{S}_n$ , then w.p. greater than  $1 - \frac{3}{n}$ :

$$\nabla_{w_{1i}} \mathcal{L}_f(\mathcal{S}_n) = -\frac{v_i}{4} \left[ 2 + \phi\left(\frac{w_{1i} + w_{2i}}{\|\bar{w}_i\|}\right) + \phi\left(\frac{w_{1i} - w_{2i}}{\|\bar{w}_i\|}\right) - 2\phi\left(\frac{w_{1i}}{\|\bar{w}_i\|}\right) \right] \pm \frac{5v_i}{d} \sqrt{\frac{\log(cd^2)}{c}}$$

*Proof.*

$$\begin{aligned}
\nabla_{w_{1i}} \mathcal{L}_f(\mathcal{S}_n) &= -\frac{v_i}{n} \sum_{j=1}^n \mathbb{1}\{y_j f(x_j) \leq 1\} \mathbb{1}\{w_i^T x_j \geq 0\} y_j x_{1j} \\
&\stackrel{(a)}{=} -\frac{v_i}{n} \sum_{j=1}^n \mathbb{1}\{\bar{w}_i^T \bar{x}_j \geq -w_{i1} y_j - w_{i2} \mathbb{1}\{y_j = 1\} \varepsilon_j\} \\
&\stackrel{(b)}{=} -\frac{v_i}{n} \sum_{j=1}^n \mathbb{1}\left\{Z_j \geq \frac{-w_{1i} y_j - w_{2i} \mathbb{1}\{y_j = 1\} \varepsilon_j}{\|\bar{w}_i\|}\right\} \quad \text{where } Z_j \sim \mathcal{N}(0, 1) \\
&= -v_i \sum_l^{\{0,1,-1\}} \frac{1}{n} \sum_{i=1}^n \mathbb{1}\{x_{2j} = l\} \mathbb{1}\left\{Z_j \geq \frac{-w_{1i}(2l^2-1)-w_{2i}l}{\|\bar{w}_i\|}\right\} \\
&= -v_i \sum_l^{\{0,1,-1\}} \left( \mathbb{P}(x_{2j} = l) \phi\left(\frac{w_{1i}(2l^2-1)+w_{2i}l}{\|\bar{w}_i\|}\right) \pm \sqrt{\frac{\log n}{n}} \right) \quad \text{via lemma 9} \\
&= -\frac{v_i}{4} [\phi\left(\frac{w_{1i}+w_{2i}}{\|\bar{w}_i\|}\right) + \phi\left(\frac{w_{1i}-w_{2i}}{\|\bar{w}_i\|}\right) + 2\phi\left(\frac{-w_{1i}}{\|\bar{w}_i\|}\right)] \pm \frac{5v_i}{d} \sqrt{\frac{\log(cd^2)}{c}} \\
&= -\frac{v_i}{4} \left[ 2 + \phi\left(\frac{w_{1i}+w_{2i}}{\|\bar{w}_i\|}\right) + \phi\left(\frac{w_{1i}-w_{2i}}{\|\bar{w}_i\|}\right) - 2\phi\left(\frac{w_{1i}}{\|\bar{w}_i\|}\right) \right] \pm \frac{5v_i}{d} \sqrt{\frac{\log(cd^2)}{c}} \quad \text{w.p. } 1 - \frac{3}{n}
\end{aligned}$$

where (a) is due to  $y_i x_{1i} = y_i^2 = 1$  &  $\mathbb{1}\{y_j f(x_j) \leq 1\} = 1$  and (b) is due to  $\mathbb{1}\{\bar{w}_i^T \bar{x}_j \geq k\} = \mathbb{1}\{\|\bar{w}_i\| Z_j \geq k\}$ .  $\blacksquare$

Similarly, we provide a closed-form expression for the gradient along the slab coordinate:

**Lemma 5.** If  $n > cd^2$  and  $y_i f(x_i) < 1 \forall (x_i, y_i) \in \mathcal{S}_n$ , then w.p. greater than  $1 - \frac{3}{n}$ :

$$\nabla_{w_{2i}} \mathcal{L}_f(\mathcal{S}_n) = -\frac{v_i}{4} \left[ \phi\left(\frac{w_{1i}+w_{2i}}{\|\bar{w}_i\|}\right) - \phi\left(\frac{w_{1i}-w_{2i}}{\|\bar{w}_i\|}\right) \right] \pm \frac{5v_i}{d} \sqrt{\frac{\log(cd^2)}{c}}$$

*Proof.*

$$\begin{aligned}
\nabla_{w_{2i}} \mathcal{L}_f(\mathcal{S}_n) &= -\frac{v_i}{n} \sum_{j=1}^n \mathbb{1}\{y_j f(x_j) \leq 1\} \mathbb{1}\{w_i^T x_j \geq 0\} y_j x_{2j} \\
&\stackrel{(a)}{=} -\frac{v_i}{n} \sum_{j=1}^n \mathbb{1}\{\bar{w}_i^T \bar{x}_j \geq -w_{1i} y_j - w_{2i} \mathbb{1}\{y_j = 1\} \varepsilon_j\} \mathbb{1}\{y_j = 1\} \varepsilon_j \\
&\stackrel{(b)}{=} v_i \sum_l^{\{-1,1\}} \frac{1}{n} \sum_{j=1}^n (-1)^{\mathbb{1}\{\varepsilon_j=l\}} \mathbb{1}\left\{Z_j \geq \frac{-w_{1i} - w_{2i}l}{\|\bar{w}_i\|}\right\} \\
&= -v_i \sum_l^{\{1,-1\}} \left( \mathbb{P}(x_{2j} = l) \phi\left(\frac{w_{1i} + w_{2i}l}{\|\bar{w}_i\|}\right) \pm \sqrt{\frac{\log n}{n}} \right) \quad \text{via lemma 9} \\
&= -\frac{v_i}{4} [\phi\left(\frac{w_{1i}+w_{2i}}{\|\bar{w}_i\|}\right) - \phi\left(\frac{w_{1i}-w_{2i}}{\|\bar{w}_i\|}\right)] \pm \frac{5v_i}{d} \sqrt{\frac{\log(cd^2)}{c}} \quad n > cd^2 \\
&= -\frac{v_i}{4} \left[ \phi\left(\frac{w_{1i}+w_{2i}}{\|\bar{w}_i\|}\right) - \phi\left(\frac{w_{1i}-w_{2i}}{\|\bar{w}_i\|}\right) \right] \pm \frac{5v_i}{d} \sqrt{\frac{\log(cd^2)}{c}} \quad \text{w.p. } 1 - \frac{3}{n}
\end{aligned}$$

where (a) is due to  $y_i x_{i2} = \mathbb{1}\{y_i = 1\} \varepsilon_i$  &  $\mathbb{1}\{y_j f(x_j) \leq 1\} = 1$  and (b) is due to  $\mathbb{1}\{\bar{w}_i^T \bar{x}_j \geq k\} = \mathbb{1}\{\|\bar{w}_i\| Z_j \geq k\}$ .  $\blacksquare$

Next, we provide a closed-form expression for the gradient along the noise coordinates:

**Lemma 6.** If  $n > cd^2$  and  $y_i f(x_i) < 1 \forall (x_i, y_i) \in \mathcal{S}_n$ , then w.p. greater than  $1 - \frac{1}{3n}$ :

$$\begin{aligned}\nabla_{\bar{w}_i} \mathcal{L}_f(\mathcal{S}_n) &= \bar{\mathcal{G}} \bar{w}_i \pm \frac{3|v_i| \log(\sqrt{cd})}{\sqrt{cd}} \frac{\bar{w}_i}{\|\bar{w}_i\|} \pm \frac{6|v_i|}{\sqrt{cd}} u_i^\perp \\ \bar{\mathcal{G}} &= -\frac{v_i}{4\|\bar{w}_i\|} \left[ \varphi\left(\frac{w_{1i} + w_{2i}}{\|\bar{w}_i\|}\right) + \varphi\left(\frac{w_{1i} - w_{2i}}{\|\bar{w}_i\|}\right) - 2\varphi\left(\frac{w_{1i}}{\|\bar{w}_i\|}\right) \right]\end{aligned}$$

where  $u_i^\perp$  is some unit vector orthogonal to  $\bar{w}_i$ .

*Proof.* Let  $S \subset \mathbb{R}^{d-2}$  denote the subspace spanned by  $\bar{w}_i$ . Then, for any  $x \in \mathbb{R}^d$ ,  $x = x^S + x^{S^\perp}$  where  $x^S$  &  $x^{S^\perp}$  are the orthogonal projections of  $x$  onto  $S$  and its orthogonal complement  $S^\perp$ . We show the  $\ell_2$  norm of the orthogonal projections of  $\nabla_{\bar{w}_i} \mathcal{L}_f(\mathcal{S}_n)$  onto  $S$  and  $S^\perp$  are  $O(\frac{1}{\sqrt{d}})$ :

$$\begin{aligned}\nabla_{\bar{w}_i} \mathcal{L}_f(\mathcal{S}_n) &= -\frac{v_i}{n} \sum_{j=1}^n \mathbb{1}\{y_j f(x_j) \leq 1\} \mathbb{1}\{w_i^T x_j \geq 0\} y_j \bar{x}_j \\ &= \underbrace{-\frac{v_i}{n} \sum_{j=1}^n \mathbb{1}\{w_i^T x_j \geq 0\} y_j (\bar{x}_j^S)}_{\text{case 1}} - \underbrace{\frac{v_i}{n} \sum_{j=1}^n \mathbb{1}\{w_i^T x_j \geq 0\} y_j (\bar{x}_j^{S^\perp})}_{\text{case 2}}\end{aligned}$$

Next, we show that the projection of  $\nabla_{\bar{w}_i} \mathcal{L}_f(\mathcal{S}_n)$  onto  $S^\perp$  (i.e., case 2) has small norm w.p. greater than  $1 - \frac{1}{d}$ :

$$\begin{aligned}\|\nabla_{\bar{w}_i} \mathcal{L}_f(\mathcal{S}_n)\|^{S^\perp} &= \left\| \frac{v_i}{n} \sum_{j=1}^n \mathbb{1}\{w_i^T x_j \geq 0\} y_j \bar{x}_j^{S^\perp} \right\| = \left\| \frac{v_i}{n} \sum_{j=1}^n \mathbb{1}\{w_i^T x_j^S \geq 0\} y_j \bar{x}_j^{S^\perp} \right\| \\ &\stackrel{(a)}{\leq} |v_i| \cdot \left\| \sum_{j=1}^n \mathcal{N}(0, \frac{1}{n^2} I_{d-2}) \right\| = |v_i| \cdot \|\mathcal{N}(0, \frac{1}{n} I_{d-2})\| \\ &\stackrel{(b)}{\leq} 4|v_i| \sqrt{\frac{d}{n}} \pm 2|v_i| \sqrt{\frac{\log n}{n}} \stackrel{(c)}{\leq} \frac{6|v_i|}{\sqrt{cd}} \quad \text{w.p. } 1 - \frac{1}{n}\end{aligned}$$

where (a) is because  $x_j^S \perp \bar{x}_j^{S^\perp}$ , (b) is via fact 1 and (c) is due to  $n \geq cd^2$ . Next, we show that the norm of the gradient in the direction of  $\bar{w}_i$  (i.e., case 1) is close to  $\bar{\mathcal{G}}$  w.h.p.:

$$\begin{aligned}\nabla_{\bar{w}_i} \mathcal{L}_f(\mathcal{S}_n)|^S &= -\frac{v_i}{n} \sum_{j=1}^n \mathbb{1}\{w_i^T x_j \geq 0\} y_j \bar{x}_j^S \\ &\stackrel{(a)}{=} -\left( \frac{1}{n} \sum_{j=1}^n \mathbb{1}\{w_i^T x_j^S \geq 0\} y_j \bar{w}_i^T \bar{x}_j \right) \frac{v_i \bar{w}_i}{\|\bar{w}_i\|^2} \\ &\stackrel{(b)}{=} -\left( \frac{1}{n} \sum_{j=1}^n \mathbb{1}\left\{ Z_j \geq \frac{-w_{1i}y - w_{2i}\mathbb{1}\{y_j = 1\}\varepsilon_j}{\|\bar{w}_i\|} \right\} y_j Z_j \right) \frac{v_i \bar{w}_i}{\|\bar{w}_i\|} \\ &= \left( \sum_l^{\{0,\pm 1\}} \frac{(-1)^{\mathbb{1}\{l \neq 0\}}}{n} \sum_{i=1}^n \mathbb{1}\left\{ x_{2j} = l \wedge Z_j \geq \frac{-w_{1i}(2l^2-1)-w_{2i}l}{\|\bar{w}_i\|} \right\} Z_j \right) \frac{v_i \bar{w}_i}{\|\bar{w}_i\|} \\ &= \left[ 2\varphi\left(\frac{w_{1i}}{\|\bar{w}_i\|}\right) - \varphi\left(\frac{w_{1i} + w_{2i}}{\|\bar{w}_i\|}\right) - \varphi\left(\frac{w_{1i} - w_{2i}}{\|\bar{w}_i\|}\right) \pm \frac{5 \log n}{\sqrt{n}} \right] \frac{v_i \bar{w}_i}{4\|\bar{w}_i\|} \quad \text{via lemma 10} \\ &= \bar{\mathcal{G}} \bar{w}_i \pm \frac{3|v_i| \log(\sqrt{cd})}{\sqrt{cd}} \frac{\bar{w}_i}{\|\bar{w}_i\|} \quad \text{w.p. } 1 - \frac{12}{n}\end{aligned}$$

where (a) is because  $\bar{x}_j^S = \frac{\bar{w}_i^T x_j}{\|\bar{w}_i\|^2} \bar{w}_i$  and (b) is because (b) is due to  $\mathbb{1}\{\bar{w}_i^T \bar{x}_j \geq k\} = \mathbb{1}\{\|\bar{w}_i\| Z_j \geq k\}$ . Therefore, by combining the results in case 1 and 2, the following holds w.p. greater than  $1 - \frac{13}{n}$ :

$$\nabla_{\bar{w}_i} \mathcal{L}_f(\mathcal{S}_n) = \bar{\mathcal{G}} \bar{w}_i \pm \frac{3|v_i| \log(\sqrt{cd})}{\sqrt{cd}} \frac{\bar{w}_i}{\|\bar{w}_i\|} \pm \frac{6|v_i|}{\sqrt{cd}} u_i^\perp$$

■

### F.3 Miscellaneous Lemmas

**Lemma 7.** Let  $X_i \sim \mathcal{N}(0, \sigma^2)$  and  $\delta \in (0, 1)$ . Then,  $\max_{i \in [k]} |X_i| \leq \sigma \sqrt{2 \log(\frac{2k}{\delta})}$  with probability greater than  $1 - \delta$ ,

*Proof.* Let  $\varphi$  denote the probability density function of the standard normal. Also let  $Z \sim \mathcal{N}(0, 1)$ . Then, for  $t \geq 1$ , we have:

$$\mathbb{P}(|X| \geq \sigma t) = \mathbb{P}(|Z| \geq t) = 2 \int_t^\infty x \varphi(x) dx \leq \frac{2}{t} \int_t^\infty x \varphi(x) dx \stackrel{(a)}{\leq} \frac{2}{t} \int_\infty^t \varphi'(x) dx \leq \frac{2}{t} \varphi(t) \leq 2\varphi(t)$$

where (a) is because  $\varphi'(x) = -x\varphi(x)$ . Using union bound with  $t = \sqrt{2 \log(\frac{2k}{\delta})} \geq 1 \forall \delta \in (0, 1)$  gives the desired result.  $\blacksquare$

**Lemma 8.** Let  $\phi$  and  $\varphi$  denote the cumulative distribution function and the probability density function of the standard gaussian. Then, for any  $Z \sim \mathcal{N}(0, 1)$  and  $k \in \mathbb{R}$ :

$$\mathbb{E}[\mathbb{1}\{Z \geq k\}Z] = \varphi(k) = \exp(-k^2/2)$$

*Proof.* The expectation  $\mathbb{E}[\mathbb{1}\{Z \geq k\}Z]$  can be simplified as follows:

$$\mathbb{E}[\mathbb{1}\{Z \geq c\}Z] = \Pr[Z \geq c] \mathbb{E}[Z|Z \geq c] = \phi^c(k) \int_k^\infty x \frac{\varphi(x)}{\phi^c(k)} dx \stackrel{(a)}{=} - \int_k^\infty \varphi'(x) dx = \varphi(k)$$

where (a) is due to  $\varphi'(x) = -x\varphi(x)$ .  $\blacksquare$

**Lemma 9.** Let  $b_i \sim \text{bernoulli}(p)$  and  $Z_i \sim \mathcal{N}(0, 1)$ . Let  $X_i = b_i \mathbb{1}\{Z_i \geq k\}$  and  $\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i$ . Then:

$$\Pr\left(|\bar{X} - p\phi(-k)| \geq \sqrt{\frac{\log n}{n}}\right) \leq \frac{1}{n}$$

*Proof.* Note that  $\mathbb{E}[\bar{X}] = \mathbb{E}[X_i] = \mathbb{E}[b_i] \mathbb{E}[\mathbb{1}\{Z_i \geq k\}] = p\phi(-k)$  and  $|X_i| \leq 1$ . Therefore, using Hoeffding's inequality with  $t = \sqrt{\frac{\log n}{n}}$  directly gives the result.  $\blacksquare$

**Lemma 10.** Let  $b_i \sim \text{bern}(p)$  and  $Z_i \sim \mathcal{N}(0, 1)$ . Let  $X_i = b_i \mathbb{1}\{Z_i \geq k\} Z_i$  and  $\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i$ . Then:

$$\Pr\left(|\bar{X} - p\varphi(k)| \leq \sqrt{\frac{2}{n} \log n}\right) \geq 1 - \frac{4}{n}$$

*Proof.* Since  $|X_i| = |b_i \mathbb{1}\{Z_i \geq k\} Z_i| \leq |Z_i|$ , we have  $\max_{i \in [n]} |X_i| \leq \sqrt{4 \log(n)}$  w.p. at least  $1 - \frac{2}{n}$  via lemma 7. From lemma 8, we get  $\mathbb{E}[X_i] = \mathbb{E}[b_i] \mathbb{E}[\mathbb{1}\{Z_i \geq k\} Z_i] = p\varphi(k)$ . Let  $A = \mathbb{1}\{|X_i| \leq \sqrt{4 \log(n)} \forall i \in [n]\}$ . Given  $A$ , we can use Hoeffding's inequality with  $t^* = \sqrt{\frac{2}{n} \log n}$  (and  $\delta = 2/n$ ) to get the desired result, as follows:

$$\mathbb{P}(|\bar{X} - p\varphi(k)| \leq t^*) \geq \mathbb{P}(|\bar{X} - p\varphi(k)| \leq t^* | A) \mathbb{P}(A) \geq (1 - \frac{2}{n})^2 \geq 1 - \frac{4}{n}$$

Therefore,  $\bar{X} = p\varphi(k) \pm \sqrt{\frac{2}{n} \log n}$  w.p. at least  $1 - \frac{4}{n}$ .  $\blacksquare$

**Lemma 11.** Let  $g : \mathbb{R} \setminus \{0\} \rightarrow \mathbb{R}$  be defined as  $g_z(x) = \frac{1}{x} \exp(-\frac{z^2}{2x^2})$ . Then, (1)  $|z|$  and  $-|z|$  are the global maximizer and minimizer respectively, and (2)  $g$  monotonically increases from  $-|z|$  to  $|z|$ .

*Proof.* Note that  $g'_z(x) = \frac{1}{x^2} \exp(-\frac{z^2}{2x^2})(\frac{z^2}{x^2} - 1)$ . Therefore, the critical points of  $g$  are  $|z|$  and  $-|z|$ . Let  $S = \{t : |t| \geq |z|, t \in \mathbb{R} \setminus \{0\}\}$ . Note that  $g'_z(x) < 0$  for all  $x \in S$  and  $g'_z(x) > 0$  for all  $x \in S^c$ . Therefore, (1) and (2) hold.  $\blacksquare$

**Fact 1.** Let  $X \sim \mathcal{N}(0, \sigma^2 I_d)$  denote a  $d$ -dimensional gaussian vector. Then, from [75], w.p. greater than  $1 - \delta$ :

$$\|X\|_2 \leq 4\sigma\sqrt{d} + 2\sigma$$