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, 47, 69]. However, the precise notion of simplicity remains vague. Furthermore, previous settings [62, 23] that use SB to justify why neural networks generalize well do not simultaneously capture the brittleness of neural networks—a widely observed phenomenon in practice [66, 34]. To this end, we introduce a collection of piecewise-linear and image-based datasets that (a) naturally incorporate a precise notion of simplicity and (b) capture the subtleties of neural networks trained on real datasets. Through theory and experiments on these datasets, we show that SB of SGD and variants is extreme: neural networks rely exclusively on the simplest feature and remain invariant to all predictive complex features. Consequently, the extreme nature of SB explains why seemingly benign distribution shifts and small adversarial perturbations significantly degrade model performance. Moreover, 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. We also demonstrate that common approaches for improving generalization and robustness—ensembles and adversarial training—do not mitigate SB and its shortcomings. Given the central role played by SB in generalization and robustness, we hope that the datasets and methods in this paper serve as an effective testbed to evaluate novel algorithmic approaches aimed at avoiding the pitfalls of extreme SB.

1 Introduction

The surprisingly good generalization ability of neural networks, despite their high capacity to fit even randomly labeled data [78], has been a subject of intense study. One line of recent work [62, 23] 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 good 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 [66] and distribution shifts [50, 55, 42, 61]. Hence, the linear setting, while convenient to analyze, is insufficient to capture the subtleties of neural networks trained on real datasets. Going beyond the linear setting, several works [1, 47, 69] 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.

Preprint. Under review.
Our goal is to formally understand and probe the **simplicity bias** (SB) of neural networks in a setting that is rich enough to capture real-world problems and, at the same time, amenable to theoretical analysis & 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 would almost perfect predictive power. To systematically understand SB, we design 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 trained neural networks.

**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 on the proposed synthetic and image-based datasets reveal three major pitfalls of SB:

(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 [45] 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 [49, 24].

(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 stark contrast, prior works [8, 62, 23] only extol SB by considering settings where all useful predictive features are simple and hence do not reveal the pitfalls observed in real-world settings.
Summary. This work makes two key contributions. First, we provide theoretical and empirical evidence that neural networks exhibit extreme SB, which we identify as a unifying explanation for various failure modes of deep learning in practice. To the best of our knowledge, prior works only focus on the positive aspect of SB: the lack of overfitting in practice. Second, we design datasets that offer a precise stratification of features based on simplicity and predictive power. Additionally, we find that standard approaches to improve generalization and robustness—ensembles and adversarial training—do not mitigate simplicity bias and its shortcomings. Given the important implications of SB, we hope these datasets serve as a useful testbed for devising better training procedures.

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

Given space constraints, we only discuss directly related work and defer the rest to Appendix A.

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 [34, 20, 43, 50]. This phenomenon has been exploited to design backdoor attacks against NNs [6, 12] as well. Recent works [22, 71] 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 [20] and annotation artifacts [25] 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 [66]. Standard approaches to mitigate this issue—adversarial training [22, 40] and ensembles [64, 51, 35]—have had limited success on large-scale datasets. Consequently, several works have investigated reasons underlying the existence of adversarial examples: [22] suggests local linearity of trained NNs, [58] indicates insufficient data, [59] suggests inevitability in high dimensions, [9] suggests computational barriers, [16] proposes limitations of neural network architectures, and [31] proposes the presence of non-robust features. Additionally, Jacobsen et al. [32] 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 [52, 45, 68].

Multiple works mentioned above (a) differently characterize learned features and desired features—statistical regularities vs. high-level concepts [24], syntactic cues vs. semantic meaning [43], robust vs. non-robust features [31]—and (b) posit that the mismatch between these features results in non-robustness. Our work instead probes why neural networks prefer one set of features over another and unifies the aforementioned disparate feature characterizations through the lens of feature simplicity.

3 Preliminaries: Setup and Metrics

Setting and metrics: We focus on binary classification. Given samples \( \tilde{D} = \{(x_i, y_i)\}_{i=1}^n \) from distribution \( D \) over \( \mathbb{R}^d \times \{-1, 1\} \), the goal is to learn a scoring function \( s(x) : \mathbb{R}^d \to \mathbb{R} \) (such as logits), and an associated classifier \( f : \mathbb{R}^d \to \{-1, 1\} \) 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: \( \mathbb{E}_D[\mathbb{1}\{f(x) = y\}] \).

**\( \delta \)-Robust accuracy.** Given norm \( \|\cdot\| \) and perturbation budget \( \delta \), the \( \delta \)-robust accuracy of a classifier \( f \) is: \( \mathbb{E}_D[\min_{\|\hat{x} - x\|_2 \leq \delta} \mathbb{1}\{f(\hat{x}) = y\}] \).

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 \( D^S \) denote the \( S \)-randomized distribution, which is obtained as follows: given \( D^S \), the marginal distribution of \( S, D^S \) independently samples \( (x^S, x^S), y) \sim D \) and \( \pi^S \sim D^S \) and then outputs \( (\pi^S, x^S), y) \). In \( D^S \), the coordinates in \( S \) are rendered independent of the label \( y \). The two metrics are as follows.

**Definition 1 (Randomized accuracy).** Given data distribution \( D \), and subset of coordinates \( S \subseteq [d] \), the \( S \)-randomized accuracy of a classifier \( f \) is given by: \( \mathbb{E}_{D^S}[\mathbb{1}\{f(x) = y\}] \).
We now describe the S-AUC are equal to standard accuracy and AUC and (b) Definition 2 (Randomized AUC). Given data distribution $D$ and subset of coordinates $S \subseteq [d]$, the $S$-randomized AUC of classifier $f$ equals the area under the precision-recall curve of distribution $\mathcal{D}^S$.

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 $k$-slab—shown in top row of Figure 2. In the linear block, positive and negative examples are uniformly distributed in $[0,1]$ 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.

- $\text{LMS-}k$: Linear and multiple $k$-slab; the first coordinate is a linear block and the remaining $d-1$ coordinates are independent $k$-slab blocks; we use $\text{LMS-}5$ & $\text{LMS-}7$ datasets in our analysis.
- $\text{LMS-}k$: 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 0.2 by default.
- $\text{MS-}5(5,7)$: 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.
- $\text{MS-}5$: Multiple 5-slab blocks; all coordinates are independent 5-slab blocks.

We now describe the LSIN (linear, 3-slab & noise) dataset, a stylized version of $\text{LMS-}k$ that is amenable to theoretical analysis. In LSIN, conditioned on the label $y$, the first and second coordinates of $x$ are

|       | Accuracy | AUC | Logits       |
|-------|----------|-----|--------------|
| $S$-Randomized | 0.5     | 0.5 | 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 $S^c$. 

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): $\text{LMS-}5$ (linear & multiple 5-slabs), $\text{LMS-}5$ (noisy linear & multiple 5-slabs) and $\text{MS-}(5,7)$ (5-slab & multiple 7-slabs). MNIST-CIFAR data vertically concatenates MNIST and CIFAR images (see Section 3.1).
We now establish the extreme nature of SB on datasets with features of varying simplicity—LSN. We consider the extreme nature of SB in neural networks (NNs) trained via stochastic gradient descent (SGD) and variants. If all features have full predictive power, NNs rely exclusively on the simplest feature & remain invariant to all complex features S. Then, we explain why extreme SB results in neural networks that are vulnerable to distribution shift and 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 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) = ∑j=1k vj · ReLU(∑i=1d wijx_i) denote a one-hidden-layer neural network with k hidden units and ReLU activations. Set v_j = ±1/√d w.p. 1/2 ∀j ∈ [k]. Let \{(x^i, y^i)\}_{i=1}^m denote i.i.d. samples from LSN where m ∈ [cd^2, d^3/c] for some c > 2. Then, given d > Ω(√k log k) and initial w_1,j ∼ N(0, 1/(dk log d)), after O(1) iterations, mini-batch gradient descent (over w) with hinge loss, constant step size, mini-batch size Ω(m), satisfies:

- Test error is at most 1/poly(d).
- The learned weights of hidden units w_1,j satisfy:
  \[ |w_{1,j}| = \frac{2}{\sqrt{k}} \left(1 - \frac{c}{\sqrt{\log d}}\right) + O\left(\frac{1}{\sqrt{dk \log d}}\right), \quad |w_{2,j}| = O\left(\frac{1}{\sqrt{dk \log d}}\right), \quad \|w_{3,d,j}\| = O\left(\frac{1}{\sqrt{k \log d}}\right) \]

**Linear Coordinate**

**3-Slab Coordinate**

**d−2 Noise Coordinates**

with probability greater than 1 − 1/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_1,j—SGD sets the value of w_1,j roughly Ω(√d) larger than the slab coordinates w_2,j that do not change much from their initial value. Second, the initialization we use is widely studied in the deep learning theory as it better reflects the practical performance of neural networks. Third, given that LSN is linearly separable, Brutzkus et al. guarantee convergence of test error; in contrast, however, our result additionally gives a precise description of the final classifier.

4.2 Simplicity Bias (SB) is Extreme in Practice

We now establish the extreme nature of SB on datasets with features of varying simplicity—LSN. 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
Decision boundaries of trained (100,1)-FCNs remain unchanged even after randomizing all LMS-5. We first consistently observe that SGD-trained models trained on S with the same sample size indeed rely on the simplest feature S to achieve 100% accuracy and remain invariant to the ranking of true positives’ logits against true negatives. One might expect that perturbing S or removing S from the dataset, SGD-trained models exclusively latch on to the simpler coordinate S and are invariant to all complex features S'.

We first consistently observe that SGD-trained models trained on LMS-5 and MS-5 datasets exhibit extreme SB: they exclusively rely on the simplest feature S and remain invariant to all complex features S'. Using S-randomized & S'-randomized metrics summarized in Table I, we first establish extreme SB on fully-connected (FCN), convolutional (CNN) & sequential (GRU) 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'-randomized AUC of all models on both datasets equals 1.0. That is, arbitrarily perturbing S' 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' 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'. Conversely, randomizing the simplest feature S randomly shuffles the original logit distributions over true positives as well as true negatives.

We now use S to denote the simplest feature in each dataset: linear in LMS-5, 5-slab in MS-5, and MNIST in MNIST-CIFAR. Figure 3: Extreme SB on LMS-5, MS-5, 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’-randomized logit distribution of true positives is a randomly shuffled version of original logit distribution; S’-randomized logits are shuffled across true positives and negatives. (b) {S, S’}-randomized AUCs (summarized in Table I) are 0.5 and 1.0 respectively for varying number of complex features |S'|. (c) FCN decision boundaries projected onto S & the most influential coordinate in S' shows that the boundary depends only on S and is invariant to S'.

Note that sample size and model architecture do not present any obstacles in learning complex features S' 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' 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' 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 [57], GoogLeNet [63], ResNet50 [26] and DenseNet121 [30] 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(c) visually confirm that FCNs exclusively depend on the simpler coordinate S and are invariant to all complex features S'.

| Dataset        | Set S | Size $|S'|$ | Randomized AUC |
|----------------|-------|-------|----------------|
| LMS-5          | all   | 49    | 0.50 ± 0.01    |
|                | 5-slabs |      | 1.00 ± 0.00    |
|                | 249   |       | 0.50 ± 0.01    |
| MS-5           | all   | 49    | 0.49 ± 0.01    |
|                | 7-slabs |      | 1.00 ± 0.00    |
|                | 249   |       | 0.50 ± 0.01    |

Figure 3: Extreme SB on LMS-5, MS-5, 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’-randomized logit distribution of true positives is a randomly shuffled version of original logit distribution; S’-randomized logits are shuffled across true positives and negatives. (b) {S, S’}-randomized AUCs (summarized in Table I) are 0.5 and 1.0 respectively for varying number of complex features |S'|. (c) FCN decision boundaries projected onto S & the most influential coordinate in S' shows that the boundary depends only on S and is invariant to S'.
To summarize, we use $S$ to validate our results on extreme SB across model architectures, activation functions, optimizers and modify the true image class do not alter model predictions. Due to extreme SB explains why “natural” [29] and semantic label-relevant perturbations [5] that the same UAP. Furthermore, invariance to complex features models; Figure 4 shows that FCN data-agnostic UAPs of one model transfer well to another: the notion of simplicity is consistent across MNIST in the simpler example, Figure 4 shows that the trained on LMS-5 degrade model performance. We validate this hypothesis in Figure 4, where simplest feature $f$ classifies such as $\text{FCN}$ given the large margin, classifiers, one for every $5$—slab coordinate. By relying on all $d$ features, $f^*$ has $O(\sqrt{d})$ margin. Now, given the large margin, $f^*$ also attains high robust accuracy to $\ell_2$ adversarial perturbations that have norm $O(\sqrt{d})$—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 $O(1)$ 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” [29] and semantic label-relevant perturbations [5] that modify the true image class do not alter model predictions.

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

Now, we discuss how extreme SB helps reconcile the fact that NNs have poor OOD performance and adversarial vulnerability despite generalizing well on the same data distribution.

**Poor OOD performance:** Given that neural networks tend to heavily rely on spurious features [43, 50], 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 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 [28, 38] 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 $O(\sqrt{d})$ margin. Now, given the large margin, $f^*$ also attains high robust accuracy to $\ell_2$ adversarial perturbations that have norm $O(\sqrt{d})$—the perturbations need to attack at least $\Omega(d)$ coordinates to flip model predictions.
We investigated Simplicity Bias (SB) in SGD-trained neural networks (NNs) using 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.

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.

### Table 2: Extreme SB can hurt generalization: FCNs of varying depth and width trained on $\hat{\text{LMS-7}}$ data

| Accuracy | SGD            | Adam            |
|----------|----------------|-----------------|
|          | (100,1)-FCN    | (200,1)-FCN     | (200,2)-FCN     |
| Training Data | 0.969 ± 0.004 | 0.970 ± 0.006   | 1.000 ± 0.000   |
| Test Data  | 0.898 ± 0.002 | 0.892 ± 0.002   | 0.892 ± 0.002   |
| $S^c$-Randomized | 0.888 ± 0.002 | 0.892 ± 0.002   | 0.892 ± 0.002   |
| S-Randomized | 0.498 ± 0.002 | 0.505 ± 0.005   | 0.502 ± 0.002   |

Table 2: Extreme SB can hurt generalization: FCNs of varying depth and width trained on $\hat{\text{LMS-7}}$ data with SGD and Adam [16] obtain approximately 100% train accuracy but at most 90% test accuracy. This is because, due to extreme SB, FCNs exclusively rely on the simpler noisy linear feature $\mathcal{S}$ and remain invariant to all 7-slab features that have 100% predictive power.

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

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

We verify this phenomenon on $\hat{\text{LMS-7}}$ data defined in Section 3. Recall that $\hat{\text{LMS-7}}$ has one noisy linear coordinate $\mathcal{S}$ with 90% predictive power and multiple 7-slab coordinates $\mathcal{S}^c$, each with 100% predictive power. Note that our training sample size is large enough for FCNs of varying depth and width that are trained on $\mathcal{S}^c$ only (i.e., after removing $\mathcal{S}$ from data) to attain 100% test accuracy. However, when trained with SGD or Adam [16] on all features in $\hat{\text{LMS-7}}$, (i.e., including $\mathcal{S}$), FCNs exhibit extreme SB and only rely on $\mathcal{S}$, the noisy linear coordinate. Specifically, as shown in Table 2, the train, test, and $\{\mathcal{S}, \mathcal{S}^c\}$-randomized accuracies collectively show that FCNs (a) exclusively rely on (and overfit to) the noisy linear feature and (b) consequently suffer 10% generalization error.

To summarize, the mere presence of a simple-but-noisy feature significantly degrades generalization because of extreme SB. In Appendix E we additionally show that (a) this phenomenon holds on other datasets, (b) increasing learning rate does not improve generalization on $\hat{\text{LMS-7}}$ data, and (c) deeper models exhibit stronger bias towards noisy-but-simple features.

### 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 subtleties of trained NNs in practice. We first showed that one-hidden-layer ReLU NNs provably exhibit SB. Then, we empirically demonstrated that SB is extreme in practice and leads to poor OOD performance and adversarial vulnerability. We also showed that, contrary to conventional wisdom, extreme SB can in fact 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. Specifically, when datasets have *multiple* simple features (e.g., multiple linear coordinates in LMS-5 or multiple 5-Slab coordinates in MS- (5, 7)), ensembles of independently trained models mitigate SB to some extent by aggregating predictions based on multiple simple features. However, the resulting ensemble remains *invariant to all* complex features: 5-Slabs in LMS-5 and 7-Slabs in MS- (5, 7). Our results suggest that the generalization improvements using ensemble models in practice stem from combining multiple simple-but-noisy features (such as color, texture) and not by learning complex features (such as shape). Similarly, adversarial training increases margin (and hence adversarial robustness) to some extent by combining multiple simplest features but does not achieve maximum adversarial robustness since the resulting models are still invariant to *all* complex features.

Our results collectively motivate the need for new algorithms that avoid the pitfalls of extreme SB. Additionally, our 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 act as an effective testbed for 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 Bhattad, 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 overparameterized 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] 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.

[21] Noah Golowich, Alexander Rakhlin, and Ohad Shamir. Size-independent sample complexity of neural networks. *arXiv preprint arXiv:1712.06541*, 2017.

[22] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[23] 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.

[24] 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.

[25] 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.

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. corr abs/1512.03385 (2015), 2015.

[27] 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.

[28] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.

[29] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *arXiv preprint arXiv:1907.07174*, 2019.

[30] Gao Huang, Zhuang Liu, and Kilian Q Weinberger. Densely connected convolutional networks. corr abs/1608.06993 (2016). *arXiv preprint arXiv:1608.06993*, 2016.

[31] 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.

[32] Joern-Henrik Jacobsen, Jens Behrmann, Richard Zemel, and Matthias Bethge. Excessive invariance causes adversarial vulnerability. In *International Conference on Learning Representations*, 2019.

[33] Ziwei Ji and Matus Telgarsky. The implicit bias of gradient descent on nonseparable data. In *Conference on Learning Theory*, pages 1772–1798, 2019.

[34] Jason Jo and Yoshua Bengio. Measuring the tendency of cnns to learn surface statistical regularities. *arXiv preprint arXiv:1711.11561*, 2017.

[35] Sanjay Kariyappa and Moiuddin K Qureshi. Improving adversarial robustness of ensembles with diversity training. *arXiv preprint arXiv:1901.09981*, 2019.

[36] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[37] 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.

[38] 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.
[39] 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.

[40] 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.

[41] Karttikeya Mangalam and Vinay Uday Prabhu. Do deep neural networks learn shallow learnable examples first? 2019.

[42] 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.

[43] 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.

[44] 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.

[45] 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.

[46] 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.

[47] 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.

[48] 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.

[49] 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.

[50] 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.

[51] Tianyu Pang, Kun Xu, Chao Du, Ning Chen, and Jun Zhu. Improving adversarial robustness via promoting ensemble diversity. arXiv preprint arXiv:1901.08846, 2019.

[52] 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.

[53] 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.

[54] 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.

[55] 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.

[56] Kevin Roth, Yannic Kilcher, and Thomas Hofmann. The odds are odd: A statistical test for detecting adversarial examples. arXiv preprint arXiv:1902.04818, 2019.

[57] 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.
[58] 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.

[59] Ali Shafahi, W Ronny Huang, Christoph Studer, Soheil Feizi, and Tom Goldstein. Are adversarial examples inevitable? arXiv preprint arXiv:1809.02104, 2018.

[60] 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.

[61] Becks Simpson, Francis Dutil, Yoshua Bengio, and Joseph Paul Cohen. Gradmask: reduce overfitting by regularizing saliency. arXiv preprint arXiv:1904.07478, 2019.

[62] 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.

[63] Nitish Srivastava. Improving neural networks with dropout. University of Toronto, 182(566):7, 2013.

[64] 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.

[65] 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.

[66] 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.

[67] 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.

[68] Florian Tramèr, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. The space of transferable adversarial examples. arXiv preprint arXiv:1704.03453, 2017.

[69] Guillermo Valle-Perez, Chico Q. Camargo, and Arda Louis. Deep learning generalizes because the parameter-function map is biased towards simple functions. In International Conference on Learning Representations, 2019.

[70] Martin J Wainwright. High-dimensional statistics: A non-asymptotic viewpoint, volume 48. Cambridge University Press, 2019.

[71] 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.

[72] Haohan Wang, Zexue He, and Eric P. Xing. Learning robust representations by projecting superficial statistics out. In International Conference on Learning Representations, 2019.

[73] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. arXiv preprint arXiv:2001.03994, 2020.

[74] 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.

[75] 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.

[76] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. arXiv preprint arXiv:1704.01155, 2017.

[77] 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.
[78] C Zhang, S Bengio, M Hardt, B Recht, and O Vinyals. Understanding deep learning requires rethinking generalization. In Int Conf. on Learning Representations, 2017.

[79] 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 present additional experiments to show that extreme SB can hurt generalization. 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 the implicit bias of SGD, margin-based generalization bounds, adversarial robustness, and out-of-distribution (OOD) examples.

Implicit bias of stochastic gradient descent: Brutzkus et al. [8] shows that neural networks trained with SGD provably generalize on linearly separable data. Recent works [62, 33] also analyze the limiting direction of gradient descent on logistic regression with linearly separable and non-separable data respectively; Gunasekar et al. [23] proves similar results for linear convolutional networks. Empirical findings [47, 41] provide further evidence to suggest that neural networks trained using SGD generalize well because SGD learns models of increasing complexity over time. A few recent works have investigated the implicit bias of SGD on non-linearly separable data for linear classifiers [33] and infinite width two layer NNs [13], in both cases showing convergence to maximum margin classifiers in appropriate spaces. As discussed previously, we believe that this implicit bias of SGD (towards simplicity) can in fact be a challenge for learning robust large-margin classifiers as it is naturally biased towards simple, small-margin and feature-impoverished classifiers at the cost of feature-dense, large-margin classifiers. Our result in Theorem 1 exhibits this issue in a stylized setting.

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, 48, 18, 21]. 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 [46] 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 [22] to ϵ-bound adversarial attacks such as FGSM [22], PGD [40], 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 [76], denoising [75], encoding [10], specialized nonlinearities [77] and distillation [53] have had limited success against stronger attacks [2]. On the other hand, standard adversarial training [40] and its variants such as [79] 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 [60]; recent attempts [73, 60] that make adversarial training faster do not improve robustness either. In Appendix E, we show that ℓ2 adversarial training on synthetic datasets can improve robustness by some extent but it is unable to learn optimal large-margin ℓ2-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 [54], statistical tests [56], and model confidence scores [28, 58, 57]. 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 the paper.

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 $3$-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: $x = y(B\gamma + (B - B\gamma) \cdot U(0, 1))$, where $U(0, 1)$ 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 $1 - \frac{p}{2}$. Given label $y \in \{-1, 1\}$ sampled uniformly at random, $x$ is sampled as follows:
  
  $$x = \begin{cases} 
  y(B\gamma + (B - B\gamma) \cdot U(0, 1)) & \text{w.p. } p \\
  U(-\gamma, \gamma) & \text{w.p. } 1-p 
  \end{cases}$$

- **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 uniformly at random, we can sample $x$ from a 3-slab block as follows:
  
  $$x = \begin{cases} 
  z(\frac{1}{2}w_3 \cdot U(0, 1)) & \text{if } y = -1 \\
  z(\frac{1}{2}w_3 + 2B\gamma + w_3 \cdot U(0, 1)) & \text{if } y = +1 
  \end{cases}$$

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 $1 - \frac{p}{2}$. For example, for 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:

  $$x = \begin{cases} 
  z(\frac{1}{2}w_3 \cdot U(0, 1)) & \text{if } y = -1 \text{ w.p. } 1-p \\
  z(\frac{1}{2}w_3 + 2B\gamma + w_3 \cdot U(0, 1)) & \text{if } y = +1 \text{ w.p. } p \\
  z(\frac{1}{2}w_3 + (2B\gamma - \frac{3}{2}w_3) \cdot U(0, 1)) & \text{w.p. } p 
  \end{cases}$$

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—$\text{LMS}-k$, $\text{LMS}-k$, $\text{MS}-5(5, 7)$, and $\text{MS}-5$—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 $p = 0.2$ in all blocks/coordinates. Also recall that each dataset comprises at most one “simple” feature $S$ and multiple independent complex features $S'$. In our experiments, all datasets have sample sizes that are large enough for all models considered in the paper to learn complex features $S'$ and attain optimal test accuracy, even in the absence of $S$; we use sample sizes of 500000 for $\text{LMS}-5$ and $\text{MS}-5$ and 100000 for $\text{LMS}-k$ and $\text{MS}-5(5, 7)$ datasets.

- **LSN Dataset:** Recall that the LSN dataset (described in Section 3) is a stylized version of the $\text{LMS}-k$ 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 \((x_i, y_i) \in \mathbb{R}^d \times \{-1,1\}\) can be sampled as follows:

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

- Additional Datasets: We now introduce \(\hat{\mathcal{S}}-(5,7)\), the noisy version of \(\mathcal{S}-(5,7)\), and three \textsc{MNIST-CIFAR} datasets, each with different \textsc{MNIST} and \textsc{CIFAR10} classes.

  - \(\hat{\mathcal{S}}-(5,7)\): 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.

  - \textsc{MNIST-CIFAR} datasets: Recall that images in the \textsc{MNIST-CIFAR} datasets are concatenations of \textsc{MNIST} and \textsc{CIFAR10} images. We introduce additional variants of the \textsc{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 |
|--------------|----------|----------|
| \textsc{MNIST-CIFAR}:A | Digit 0 | Automobile | Digit 1 | Truck |
| \textsc{MNIST-CIFAR}:B | Digit 1 | Automobile | Digit 4 | Truck |
| \textsc{MNIST-CIFAR}:C | Digit 0 | Airplane  | Digit 1 | Ship  |

Table 3: Three \textsc{MNIST-CIFAR} datasets. We use \textsc{MNIST-CIFAR}:A in the paper. In \textsc{MNIST-CIFAR}:B, we use different \textsc{MNIST} classes: digits 1 and 4. In \textsc{MNIST-CIFAR}:C, we use different \textsc{CIFAR10} classes: airplane and ship. Our results in Section 4 hold on all three \textsc{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 and \((h, l, d)\)-GRU denotes \(d\)-layer \(d\)-layer GRU with input dimensionality \(l\) and hidden state dimensionality \(h\). On \textsc{MNIST-CIFAR}, we train MobileNetV2 [57], GoogLeNet [65], ResNet50 [26] and DenseNet121 [30].

Training Procedures: Unless mentioned otherwise, we use the following hyperparameters for standard training and adversarial training on synthetic and \textsc{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 \textsc{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 \(10^{-3}\).

- Adversarial Training: We use the same SGD hyperparameters (as described above) on synthetic and \textsc{MNIST-CIFAR} datasets. We use Projected Gradient Descent (PGD) Adversarial Training [40] 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* —in Section 4 of the paper. Now, we further validate the extreme nature of SB across model architectures, datasets, optimizers, activation functions and regularization.

### 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 5, we present \{S, S*\}-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. While the S*-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*, even though all features have equal predictive power.

| Dataset       | Set $S$ | Set $S^c$ | Model          | Randomized AUC |
|---------------|---------|-----------|----------------|----------------|
|               |         |           | (100,1) -FCN   | Set $S$ | Set $S^c$ | 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 |
|               | Linear  | 5-Slabs   | (100,10,1) -GRU|          |          | 0.51 | 1.00 |
|               |         |           | (100,10,2) -GRU|          |          | 0.50 | 1.00 |
| MNIST-CIFAR:A | MNIST   | CIFAR     | MobileNetV2    | 0.52 | 1.00 |
|               | block   | block     | 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.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 [57], GoogLeNet [65], ResNet50 [26] and DenseNet121 [30] 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.

| Model          | MNIST-CIFAR:A AUCs | MNIST-CIFAR:B AUCs | MNIST-CIFAR:C AUCs |
|----------------|--------------------|--------------------|--------------------|
|                | Standard | Randomized | Standard | Randomized | Standard | Randomized | Standard | Randomized |
| MobileNetV2    | 1.00 ± 0.00       | 1.00 ± 0.00       | 0.51 ± 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       |
| 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       |
| 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 ± 0.03       |
| DenseNet121    | 1.00 ± 0.00       | 1.00 ± 0.00       | 0.52 ± 0.01       | 1.00 ± 0.00       | 1.00 ± 0.00       | 0.52 ± 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.

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.

### 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 $S^\circ$ in addition to the simplest feature $S$?

| Activation Function | LMS-7 SGD | Adam | RMSProp | NS-$(5, 7)$ 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 [39], PReLU [27], and Tanh—trained on LMS-7 data using common first-order optimization methods—SGD, Adam [36], and RMSProp [67]—exhibit extreme SB.

Table 6 presents the $S$-randomized AUCs of $(100, 2)$-FCNs with multiple activation functions—ReLU, Leaky ReLU [39], PReLU [27], and Tanh—trained on LMS-7 and $S^\circ$-randomized AUCs of $(5, 7)$ datasets using multiple commonly-used optimizers: SGD, Adam [36] and RMSProp [67]. We observe that for all combinations of activations and optimizers, trained FCNs still only rely on simplest feature $S$; $S$-randomized and $S^\circ$-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 $S^\circ$.

### 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 [63] 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 $S^\circ$-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.

| Model     | Dropout | Standard AUC $\lambda = 10^{-2}$ | $\lambda = 10^{-4}$ | S-Randomized AUC $\lambda = 10^{-2}$ | $\lambda = 10^{-4}$ | $S^\circ$-Randomized AUC $\lambda = 10^{-2}$ | $\lambda = 10^{-4}$ |
|-----------|---------|----------------------------------|-------------------|--------------------------------------|-------------------|--------------------------------------|-------------------|
| (100, 1) -FCN | 0.00 | 1.00 ± 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 | 0.10 | 1.00 ± 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 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 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 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 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 $\{S, S^\circ\}$-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 $S^\circ$ (7-slab blocks).
D  Additional Results on the Effect of Extreme SB on Generalization

Recall that in Section 5 of the paper, we showed that, contrary to conventional wisdom, extreme SB can result in suboptimal generalization of SGD-trained models on the same data distribution as well. In this section, we present additional experiments to concretely validate the aforementioned phenomenon. First, we analyze the effect of $\ell_2$ regularization and learning rate on models trained on $\hat{\text{LMS}}$-k (noisy linear & multiple noiseless $k$-slab blocks) data. Second, we investigate the effect of model depth and the amount of noise in “simple” features on generalization in the context of extreme SB. Third, we show that SGD-trained fully-connected networks trained on $\hat{\text{MS}}$- (5,7) data (noisy 5-slab & multiple 7-slabs) exhibit suboptimal generalization performance as well.

D.1 Effect of $\ell_2$ Regularization

Now, we show that increasing the $\ell_2$ regularization hyper-parameter $\lambda$ has no effect on extreme SB or its effect on generalization. SGD-trained FCNs continue to exclusively latch on to the noisy linear block and consequently attain suboptimal generalization.

| $\ell_2$ regularization | LMS-7 | MS-7 |
|-------------------------|-------|------|
| $\lambda = 10^{-6}$     | 0.896 ± 0.003 | 0.995 ± 0.004 |
| $\lambda = 10^{-5}$     | 0.896 ± 0.001 | 0.991 ± 0.003 |
| $\lambda = 10^{-4}$     | 0.891 ± 0.001 | 0.990 ± 0.002 |
| $\lambda = 10^{-3}$     | 0.897 ± 0.002 | 0.991 ± 0.003 |
| $\lambda = 10^{-2}$     | 0.899 ± 0.001 | 0.999 ± 0.000 |

Table 8: (Effect of $\ell_2$ regularization) Increasing the $\ell_2$ regularization parameter $\lambda$ from $10^{-6}$ to $10^{-2}$ reduces the extent to which FCNs overfit to the noisy linear component. However, increasing $\lambda$ does not make FCNs learn the 7-slab components and obtain 100% test accuracy—all models continue to overfit to the noisy linear component and obtain 90% test accuracy.

As shown in Table 8, increasing $\lambda$ from $10^{-6}$ to $10^{-2}$ does not enable FCNs to learn the 7-slab blocks and obtain 100% test accuracy—all models continue to overfit to the noisy linear component and obtain 90% test accuracy. On the contrary, we observe that increasing $\lambda$ to $10^{-3}$ or $10^{-2}$ leads to severe underfitting, which prevents models from learning the 7-slab block in $\text{MS}$-7 data and thus attaining 100% test accuracy.

D.2 Effect of Learning Rate

We now evaluate the effect of learning rate on the generalization of (100,2)-FCNs trained with SGD on $\hat{\text{LMS}}$-5 (noisy linear + multiple 5-slab coordinates) and $\hat{\text{LMS}}$-7 (noisy linear + multiple 7-slab coordinates). Recall that in this setting, both datasets, $\text{LMS}$-5 and $\text{LMS}$-7, have a single noisy linear coordinate that has 10% noise whereas all $k$-slab coordinates have no noise. Therefore, the optimal accuracy attainable by any model that relies only on the noisy linear coordinate is 98%. On the other hand, models can attain 100% accuracy by learning one or more $k$-slab coordinates.

![Figure 5: The effect of learning rate on the generalization of (100,2)-FCNs trained with SGD on $\hat{\text{LMS}}$-5 (noisy linear + multiple 5-slab coordinates) and $\hat{\text{LMS}}$-7 (noisy linear + multiple 7-slab coordinates) datasets. Increasing the learning rate results in models that generalize well on the $\text{LMS}$-5 dataset. However, models trained on the $\text{LMS}$-7 with large learning rates attain suboptimal generalization.](image)
\textbf{LMS-5 data:} We first consider the effect of learning rate using the LMS-5 dataset with 50000 samples. In this setting, while the noisy linear coordinate does not have perfect predictive power, the remaining 5-slab coordinates are capable of perfect prediction. (100, 2)-FCNs trained on the dataset comprising 5-slab coordinates only (i.e., without noisy linear coordinate) achieves perfect accuracy with the same sample size. When we train the same model on the LMS-5 dataset (i.e., with the noisy linear coordinate), we observe that SGD-trained models learned with small learning rate end up having only 94\% test accuracy. However, the test accuracy of models trained with larger learning rates approaches 100\%, as shown by the blue line in Figure 5. The two-dimensional projections of the decision boundaries in Figure 5 show that models learned with a small learning rate (LR) rely almost exclusively on the noisy linear coordinate, whereas those learned with large LR rely on the noisy linear and 5-slab coordinates.

\textbf{LMS-7 data:} However, the effect of LR disappears when we consider the LMS-7 dataset. The orange line in Figure 5 shows that increasing the learning rate from 0.01 to 2.0 has no effect on the generalization of (100, 2)-FCNs trained on the LMS-7 dataset. All models attain suboptimal generalization by latching on to the simple-but-noisy linear coordinate instead of learning the more complex 7-slab coordinates.

To summarize, these results demonstrate that when the simplest feature has less predictive power than the more complex features, larger LR may help offset extreme SB and lead to better generalization, as observed with the LMS-5 dataset. However, more generally, as observed with the LMS-7 dataset, simply increasing the learning rate may be insufficient to mitigate SB and its adverse effect on generalization.

\textbf{D.3 Effect of Model Depth and Noise in LMS-7}

In this section, we evaluate the effect of model depth on extreme SB and generalization using SGD-trained FCNs on LMS-7 data. In particular, we evaluate the generalization of FCNs with width 100 and depth \{1, 2, 3\} by varying the noise parameter $p$ to control the noise in the noisy linear coordinate.

We consider the LMS-7 dataset with 70000 samples. Recall that when noise parameter $p > 0$, the noisy linear coordinate does not have perfect predictive power; the optimal accuracy attainable is $1 - p/2$. On the other hand, all 7-slab coordinates are capable of perfect prediction. Now, we vary the noise parameter $p$ from 0.0 (linearly separable) to 1.0 (zero predictive power) and train FCNs of depth \{1, 2, 3\} on the resulting LMS-7 dataset. The standard accuracy and linear-randomized AUCs in Figure 6 show that when $p \leq 0.2$, all models exclusively latch on to the noise linear due to extreme SB and hence suffer 10\% generalization error. More surprisingly, FCNs with depth 3 continue to exclusively rely on the noisy linear coordinate even when $p = 0.8$ and rely on noiseless 7-slab coordinates only when $p > 0.8$. Consequently, (100, 3)-FCNs can exhibit generalization error of up to 40\%.

To summarize, even though the data has complex features that can be quickly learned with the same sample complexity, model and algorithm, the presence of a noisier-but-simpler feature leads to trained models that overfit onto this feature, which subsequently leads to suboptimal generalization. We also observe that the deeper FCNs are more susceptible to simple-but-noisy features.

\begin{figure}[h]
    \centering
    \begin{subfigure}{0.45\textwidth}
        \includegraphics[width=\textwidth]{figure6a.png}
        \caption{$p$-NoisyLinear vs. Standard Accuracy in LMS-7}
    \end{subfigure}\hspace*{0.05\textwidth}
    \begin{subfigure}{0.45\textwidth}
        \includegraphics[width=\textwidth]{figure6b.png}
        \caption{$p$-NoisyLinear vs. Randomized Accuracy in LMS-7}
    \end{subfigure}
    \caption{Effect of noise parameter $p$ and FCN depth on generalization. Subplots (a) & (b) depict standard accuracy & S-randomized AUCs of FCNs with varying depth trained on LMS-7 data.}
\end{figure}
D.4 Generalization in Noisy $\mathbb{MS}-(5,7)$

Recall that in Section 5 of the paper, we showed that FCNs trained on $\hat{\mathbb{MS}}-7$ data (with 20% noise only in the linear block) obtain 90% test accuracy, even though the optimal FCN classifier can obtain 100% accuracy by relying on one or more (noiseless) 7-slab blocks. Now, we show that extreme SB can hurt generalization even in a non-linear setting. We consider the $\hat{\mathbb{MS}}-(5,7)$ data that has a relatively simple 5-slab block (with 20% noise) and multiple 7-slab blocks that have no noise.

| FCN Width | FCN Depth | Standard Test Accuracy $\mathbb{MS}-(5,7)$ | $\hat{\mathbb{MS}}-(5,7)$ |
|-----------|-----------|---------------------------------|--------------------------|
| 100       | 1         | 0.999 ± 0.000 0.907 ± 0.026  | 0.907 ± 0.037           |
|           | 2         | 0.999 ± 0.001 0.902 ± 0.037          |                          |
|           | 3         | 1.000 ± 0.003 0.888 ± 0.002       |                          |
| 300       | 1         | 0.988 ± 0.008 0.885 ± 0.003  |                          |
|           | 2         | 0.974 ± 0.010 0.915 ± 0.020          |                          |
|           | 3         | 1.000 ± 0.000 0.892 ± 0.003       |                          |

Table 9: (Generalization on $\hat{\mathbb{MS}}-(5,7)$ data) FCNs have test accuracy ≈ 90% on $\hat{\mathbb{MS}}-(5,7)$ data. In contrast, FCNs obtain 100% test accuracy on $\mathbb{MS}-(5,7)$ data (i.e. without noisy 5-slab block is removed) with the same sample size.

As shown in Table 9, FCNs with depth \{1, 2, 3\} and width \{100, 300\} trained using SGD on 50-dimensional $\hat{\mathbb{MS}}-(5,7)$ data (with 20% noise) only obtain ≈ 90% accuracy, even though the optimal FCN classifier can obtain 100% accuracy by learning one or more 7-slab structures. Due to extreme SB, The mere presence of a noisy 5-slab block that is “easier” to learn than 7-slab blocks brings down the accuracy by 10%. Therefore, the tendency of SGD to favor simple-but-noisy feature $S$ even if they are less predictive than every complex feature in $S^c$ can result in suboptimal generalization. More broadly, this experiment also indicates that (1) noisy 5-slab blocks are easier to learn than noiseless 7-slab block and (2) the proposed notion of feature simplicity extends beyond linearity.
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: LMS-7 and MS-5. Recall that the LMS-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 7: 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 LMS-7 data that has a simple-but-noisy linear coordinate that has less predictive power than the 7-slab coordinates; this is because individual FCNs trained on LMS-7 data exclusively rely on the noisy linear coordinate and consequently misclassify the same set of instances, as shown in subplot (b).](image)

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 LMS-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 7(b). As the training sample size is small, FCNs overfit to the training data and attain approximately 75% test accuracy, as shown in Figure 7. Consequently, as shown in Figure 7, 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 7 shows that ensembles of size 5 and 10 improve generalization by approximately 15% and 20% respectively.

- **LMS-7 data**: Recall that LMS-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 7(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 7(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 LMS-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 mitigates SB and its adverse effect on adversarial robustness through (100, 2)-FCNs (fully-connected network with 2 hidden layers and 100 hidden units in each layer) that are trained on two datasets: LMS-5 and MNIST-CIFAR. Recall
that the LMS-5 data comprises one simple linear coordinate and multiple relatively complex 7-slab coordinates, all of which have perfect predictive power whereas the MNIST-CIFAR data vertically concatenates MNIST and CIFAR10 images; the simplest feature S is \{linear\} and \{MNIST\} in the LMS-5 and MNIST-CIFAR datasets respectively.

**LMS-5 data:** Figure 8(a) shows the standard (in blue), adversarial (in orange), and S-randomized (in green) accuracies of $\ell_2$ adversarially trained (100, 2)-FCNs. Since the margin in the linear coordinate S is 0.2, any model that relies exclusively on S cannot achieve adversarial accuracy of 1.0 beyond an $\ell_2$ perturbation budget of $0.2/2 = 0.1$. In fact, as shown in Figure 8(b), the $\epsilon$-robust accuracy (in red) of FCN trained using SGD (i.e., not adversarial training) starts to dip sharply once the perturbation budget $\epsilon$ is larger than 0.1. However, as shown by S-randomized accuracy (in green) in Figure 8(a), the adversarially trained FCN continues to maintain $\epsilon$-robust accuracy of 1.0 until a perturbation budget of 0.5 by partially relying on the slab coordinates when $\epsilon > 0.1$. Nevertheless, both, standard and $\epsilon$-robust accuracies, drop to 0.50 (i.e., random chance) once the perturbations are larger than 0.6. This is in stark contrast to the fact that (100, 2)-FCNs are capable of fully relying on at least 25 of the 49 5-Slab coordinates (using 4 ReLUs for each 5-Slab coordinate) and hence are capable of achieving $\epsilon$-robust accuracy of 1.0 all the way up to perturbations of size 2.15.

![Figure 8: Adversarially training (100,2)-FCNs on LMS-5](image)

To summarize, our experiments on LMS-5 and MNIST-CIFAR datasets show that while adversarial training does improve the $\epsilon$-robust accuracy over that of SGD-trained model, it (a) does not fully mitigate SB and consequently (b) does not achieve maximum possible adversarial robustness.

| Model            | $\ell_\infty$ budget | $\epsilon$-Robust Accuracy | CIFAR10-Randomized Accuracy |
|------------------|-----------------------|----------------------------|----------------------------|
|                  | $\epsilon$           | Standard SGD               | $\epsilon$-Robust Training | Standard SGD               | $\epsilon$-Robust Training | Standard SGD               | $\epsilon$-Robust Training |
| MobileNetV2      | 0.30                  | 0.999 ± 0.001              | 0.999 ± 0.000              | 0.900 ± 0.000              | 0.991 ± 0.003              | 0.438 ± 0.005              | 0.451 ± 0.001              |
| DenseNet121      | 0.30                  | 1.000 ± 0.000              | 0.990 ± 0.000              | 0.961 ± 0.003              | 0.494 ± 0.005              | 0.501 ± 0.003              | 0.499 ± 0.002              |
| ResNet50         | 0.30                  | 1.000 ± 0.000              | 0.999 ± 0.001              | 0.992 ± 0.002              | 0.501 ± 0.001              | 0.499 ± 0.002              | 0.501 ± 0.003              |

Table 10: Adversarial training on MNIST-CIFAR: The table above presents standard, $\epsilon$-robust and CIFAR10-randomized accuracies for MobileNetV2, DenseNet121 and ResNet50 that are training using standard SGD and adversarial training. 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:

\[
\begin{align*}
    y_i &= \pm 1, \text{ w.p. } 1/2, \quad \varepsilon_i = \pm 1, \text{ w.p. } 1/2, \\
    x_{i1} &= y_i \\
    x_{i2} &= \left(\frac{y_i + 1}{2}\right) \varepsilon_i \\
    x_{i,3:d} &\sim \mathcal{N}(0, I_{d-2})
\end{align*}
\]

(Linear coordinate), (Slab coordinate),

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 \(m \in [cd^2, d^3/c]\) 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, constant step size, mini-batch size \(\Theta(m)\), satisfies:

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

\[
\begin{align*}
|w_{1j}| &= \frac{2}{\sqrt{k}} \left(1 - \frac{c}{\sqrt{\log d}}\right) + O\left(\frac{1}{\sqrt{dk \log d}}\right), \\
|w_{2j}| &= O\left(\frac{1}{\sqrt{dk \log d}}\right), \\
\|w_{3:d,j}\| &= O\left(\frac{1}{\sqrt{k \log d}}\right)
\end{align*}
\]

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 \(t \lesssim \frac{1}{\eta}\) iterations, with probability \(\geq 1 - \frac{1}{\text{poly}(d)}\),

- Lemma 3 shows that the hinge loss is “active” (i.e., \(y f(x) < 1\)) for all data points in a given batch.
- Under this condition, we derive closed-form expressions for population gradients in Lemmas 3 and 6.
- Lemma 4 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 \(w_{ij} = [w_{ij1}, w_{ij2}, \ldots, w_{ijd}]^T\) is the \(j^{th}\) column in \(W\). Let \(\bar{w}\) and \(\bar{x}\) denote the \(w_{3:d,i}\) and \(x_{3:d,i}\) respectively. Also, let \(S_n = \{(x_i, y_i)\}_{i=1}^n\) denote a set of \(n\) i.i.d. points randomly sampled from LSN. For simplicity, we also assume \(\{|i : v_i = 1/\sqrt{k}\} = \{|i : v_i = -1/\sqrt{k}\}\). We can now define the loss function as \(L_f(S_n) = 1/n \sum_{i=1}^n \ell(x_i, y_i)\), where \(\ell(x, y) = \max(0, 1 - y f(x))\) 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 7. 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 $t = \lfloor \frac{1}{\eta}(1 - \frac{c_0}{\sqrt{d} \log d}) \rfloor$ gives the following result:

$$w_{1i}^{(t)} = \frac{2}{\sqrt{k}} (1 - \frac{c_n}{\log d}) + O(\frac{1}{\sqrt{d} \log d})$$ and $$|w_{2i}^{(t)}| = O(\frac{1}{\sqrt{d} \log d})$$ and $$||\bar{w}_{t}^{(t)}|| = O(\frac{1}{\sqrt{k} \log d})$$

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

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 $\Pr(yf(x) < 0) = 2^{-\sigma d_n^2}$. Therefore, the 0 - 1 error is at most $\frac{2}{\sqrt{d} \log d^4} = O(\frac{1}{\sqrt{d}})$.

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

$$y_i f(x_i) \leq 1 \forall (x_i, y_i) \in S_n$$

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

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

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

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

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

Therefore, $w_{1i} = \frac{(0)v_i}{2} \pm \frac{c_0(1 + \hat{c})^t}{\sqrt{d} \log d}$ and $|w_{2i}| \leq \frac{c_0(1 + \hat{c})^t}{\sqrt{d} \log d}$ and $||\bar{w}_i|| \leq \frac{c_0(1 + \hat{c})^t}{\sqrt{k} \log d}$. 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:

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

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

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:

\[
\omega_i^{(t+1)} = \omega_i^{(t)} + \eta v_i \left[ 2 + \phi \left( \frac{w_i^{(t)} + \omega_i^{(t)}}{||w_i^{(t)}||} \right) - 2 \phi \left( \frac{w_i^{(t)} - \omega_i^{(t)}}{||w_i^{(t)}||} \right) \right] \pm 5\eta v_i \sqrt{\log(cd^2)}
\]

\[
= \omega_i^{(t)} + \frac{\eta v_i}{2} \left[ \frac{w_i^{(t)} + \omega_i^{(t)}}{||w_i^{(t)}||} \right] \cdot \max_{\|a\| \leq \|w_i^{(t)}\|} \frac{1}{\|w_i^{(t)}\|} \left[ \phi \left( \frac{w_i^{(t)} + \delta}{\|w_i^{(t)}\|} \right) + \frac{5\eta v_i}{d} \sqrt{\log(cd^2)} \right]
\]

where \((a)\) is via equation (12) in Lemma [3] \((b)\) is because \(d/\log^2(d) \geq 20/\kappa v \sqrt{c}\) 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:

\[
\omega_i^{(t+1)} = \omega_i^{(t)} + \eta v_i \left[ \frac{w_i^{(t)} - \omega_i^{(t)}}{||w_i^{(t)}||} \right] \cdot \max_{\|a\| \leq \|w_i^{(t)}\|} \frac{1}{\|w_i^{(t)}\|} \left[ \phi \left( \frac{w_i^{(t)} - \delta}{\|w_i^{(t)}\|} \right) + \frac{5\eta v_i}{d} \sqrt{\log(cd^2)} \right]
\]

where \((a)\) is due to equations (11) in Lemma [3] (3) and \(d/\log^2(d) \geq 20/\kappa v \sqrt{c}\).

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 \(t + 1\):

\[
\omega_i^{(t+1)} = \omega_i^{(t+1)} + \eta v_i \left[ \frac{w_i^{(t)} - \omega_i^{(t)}}{||w_i^{(t)}||} \right] \cdot \max_{\|a\| \leq \|w_i^{(t)}\|} \frac{1}{\|w_i^{(t)}\|} \left[ \phi \left( \frac{w_i^{(t)} - \delta}{\|w_i^{(t)}\|} \right) + \frac{5\eta v_i}{d} \sqrt{\log(cd^2)} \right]
\]

\[
= \frac{\eta v_i}{4} \left[ \frac{w_i^{(t)} + \omega_i^{(t)}}{||w_i^{(t)}||} - \frac{w_i^{(t)} - \omega_i^{(t)}}{||w_i^{(t)}||} \right] \cdot \max_{\|a\| \leq \|w_i^{(t)}\|} \frac{1}{\|w_i^{(t)}\|} \left[ \phi \left( \frac{w_i^{(t)} + \delta}{\|w_i^{(t)}\|} \right) + \frac{5\eta v_i}{d} \sqrt{\log(cd^2)} \right]
\]

\[
\leq \frac{\eta v_i}{2}
\]

Next, we show that the \(\ell_2\) norm of the second part of the gradient, \(\|\tilde{G}_2\|\), is \(O(1/\sqrt{c})\):

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

Now, we can use the upper bounds on \(G_1\) and \(G_2\) to show that the \(\ell_2\) norm of the gradient along the noise gradients is small as well:

\[
\|\omega_i^{(t+1)}\| \leq \|\omega_i^{(t)}\| + \frac{\eta v_i}{2} \|\tilde{w}_i^{(t)}\| \leq \|\omega_i^{(t)}\| + \frac{\eta v_i}{2} \|\tilde{w}_i^{(t)}\| \leq \frac{c_0(1 + \hat{c})(1 + \eta v_i)}{\sqrt{k \log d}} \leq \frac{c_0(1 + \hat{c})^{t+1}}{\sqrt{k \log d}}
\]

where \((a)\) is because \(d/\log d \geq (24\sqrt{\pi}/v^2)\), \((b)\) 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., \( y_f(x) < 1 \)) for all data points with high probability as well.

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

\[
y_t f(x_t) = \frac{t_n}{4} + \frac{c_n}{\log d} = (t + \frac{1}{2})\frac{\eta}{4} \quad \forall (x_t, y_t) \in S_n
\]

**(Proof.** We use equations (3), (4) & (5) to obtain simplify the dot product between \( w_i(t) \) & \( x_j \) and the indicator \( 1 \{ w_i(t) \cdot x_j \geq 0 \} \). First, we show that the dot product between \( w_i(t) \) and \( x_j \) is in the band \( \frac{t_n y_j}{2} \pm \frac{c_n}{\sqrt{k \log d}} \) with high probability:

\[
\begin{align*}
\frac{t_n y_j}{2} &\pm \frac{c_n}{\sqrt{k \log d}} = \frac{t n y_j}{2} \pm \frac{c n}{\sqrt{k \log d}} \\
\text{w.p.} &\quad 1 - \frac{2}{d^6}
\end{align*}
\]

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 \( 1 \{ w_i(t) \cdot x_j \geq 0 \} \) as follows:

\[
\begin{align*}
1 \{ w_i(t) \cdot x_j \geq 0 \} &\quad \text{if } t = 0, \\
1 \{ t > 0 \text{ and } y_i, y_j \geq 0 \} &\quad \text{if } t > 0 \text{ and } y_i, y_j \geq 0
\end{align*}
\]

We can now use equations (6) & (5) to show that \( y_j f(t)(x) \) is in the band \( \frac{t n}{4} \pm O(1/\sqrt{k \log d}) \) with high probability:

\[
\begin{align*}
y_j f(t)(x) = \sum_{i=1}^k y_i v_i \cdot \text{ReLU}(w_i \cdot x) = \sum_{i=1}^k v_i \begin{cases} 
1 &\quad \text{if } t = 0 \lor y_i, y_j \geq 0 \\
\frac{t n}{2k} &\quad \text{if } t > 0
\end{cases} \\
&= \sum_{i=1}^k 1 \{ t = 0 \lor y_i, y_j \geq 0 \} \frac{t n}{2k} \pm \frac{c_n}{k \sqrt{k \log d}}
\end{align*}
\]

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

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

\[
\begin{align*}
\max_{\delta \leq \|w_i(t)\|} \frac{1}{\|w_i(t)\|} \varphi \left( \frac{w_i(t) + \delta}{\|w_i(t)\|} \right) &\leq 1 \\
\left| \varphi \left( \frac{w_i(t) + w_2(t)}{\|w_i(t)\|} \right) - \varphi \left( \frac{w_1(t) - w_i(t)}{\|w_i(t)\|} \right) \right| &\leq \frac{2c_0(1+\hat{c})^t}{\sqrt{k \log d}} \\
\left| \varphi \left( \frac{w_i(t) + w_2(t)}{\|w_i(t)\|} \right) - \varphi \left( \frac{w_1(t) + w_i(t)}{\|w_i(t)\|} \right) \right| &\leq \frac{c_0(1+\hat{c})^t}{\sqrt{k \log d}} \\
\frac{c_0(1+\hat{c})^t}{\sqrt{k \log d}} &\leq \frac{c_0(1+\hat{c})^t}{\sqrt{k \log d}}
\end{align*}
\]
Proof. Let $g_x(x) = \frac{1}{2} \varphi(\frac{x}{a})$ and $h(x) = \max_{\|\delta\| \leq \|w_{t_1}\|} \frac{1}{2} \varphi(\frac{w_{t_1} + \delta}{x}) = \max_{\|\delta\| \leq \|w_{t_1}\|} g_{w_{t_1} + \delta}(x)$.

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:

$$
\max h(x) = \max_x \max_{\|\delta\| \leq \|w_{t_1}\|} g_{w_{t_1} + \delta}(x) \triangleq \max_{\|\delta\| \leq \|w_{t_1}\|} \frac{e^{-\|\delta\|}}{|w_{t_1} + \delta|} \quad \text{when } x^* = |w_{t_1}| + \delta
$$

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

$$
x^* = |w_{t_1}| + \delta \geq |w_{t_1} + w_{t_2}| \geq |w_{t_1}| - |w_{t_2}| \geq \frac{t \eta v}{2} + \frac{c_0(1 + \hat{c}) t}{\sqrt{d} \log d} \geq \frac{c_0(1 + \hat{c}) t}{\sqrt{d} \log d} \geq \frac{\eta v}{4}
$$

where (a) follows from the weights in the linear and slab coordinate at timestep $t$ (equations (2) & (3)) and (b) is because $\frac{c_0(1 + \hat{c}) t}{\sqrt{d} \log d} \leq \frac{\eta v}{8}$ when $\sqrt{d} \geq 8c_0e^t \eta$. Therefore, $\arg \max_x h(x) \geq \eta v/4$. We can use the upper bound on the $\ell_2$ norm of the gradient along the noise coordinates (equation [4]) and $d \geq \exp(\frac{4c_0e^t}{\eta})$ to show that $\|w_{t_1}^{(t)}\|$ is less than $x^*$:

$$
\|\bar{w}_{t_1}^{(t)}\| \leq \frac{c_0(1 + \hat{c}) t}{\sqrt{d} \log d} \leq \frac{\eta v}{4} \leq \arg \max_x h(x)
$$

From lemma [11] we know that $h(x)$ is an increasing function for all $|x| < x^*$. This implies that $h(\|w(t)\|) \leq h(\frac{c_0(1 + \hat{c}) t}{\sqrt{d} \log d}) \leq h(\frac{\eta v}{4}) \leq h(x^*)$. Therefore, when $d \geq \exp(\frac{4c_0e^t}{\eta})^2$ and $d^{1/\log d} \geq \sqrt{k}$, we obtain the desired result as follows:

$$
\max_{\|\delta\| \leq \|w_{t_1}\|} \frac{1}{\|w_{t_1}\|} \varphi\left(\frac{w_{t_1}^{(t)} + \delta}{\|w_{t_1}\|}\right) = h(\|w_{t_1}\|) \leq h(\frac{c_0e^t}{\sqrt{d} \log d}) \leq \frac{\sqrt{k} \log d}{c_0e^t} \leq \frac{1}{d^{1/\log d} \log d} \leq 1
$$

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

$$
\left| \phi\left(\frac{w_{t_1}^{(t)} + w_{t_2}^{(t)}}{\|w_{t_1}\|}\right) - \phi\left(\frac{w_{t_1} + w_{t_2}}{\|w_{t_1}\|}\right) \right| \leq 2\|w_{t_2}\| \cdot \max_{\|\delta\| \leq \|w_{t_1}\|} \frac{1}{\|w_{t_1}\|} \varphi\left(\frac{w_{t_1}^{(t)} + \delta}{\|w_{t_1}\|}\right) \triangleq \frac{2c_0(1 + \hat{c}) t}{\sqrt{d} \log d} \quad (14)
$$

$$
\left| \phi\left(\frac{w_{t_1}^{(t)} + w_{t_2}^{(t)}}{\|w_{t_1}\|}\right) - \phi\left(\frac{w_{t_1} + w_{t_2}}{\|w_{t_1}\|}\right) \right| \leq \|w_{t_2}\| \cdot \max_{\|\delta\| \leq \|w_{t_1}\|} \frac{1}{\|w_{t_1}\|} \varphi\left(\frac{w_{t_1}^{(t)} + \delta}{\|w_{t_1}\|}\right) \triangleq \frac{c_0(1 + \hat{c}) t}{\sqrt{d} \log d} \quad (15)
$$

$$
1 \varphi\left(\frac{w_{t_1}^{(t)} + \delta}{\|w_{t_1}\|}\right) \leq \max_{\|\delta\| \leq \|w_{t_1}\|} 1 \varphi\left(\frac{w_{t_1}^{(t)} + \delta}{\|w_{t_1}\|}\right) \triangleq \frac{c_0(1 + \hat{c}) t}{\sqrt{d} \log d} \quad (16)
$$

where (a) is due to equation (4). \hfill \blacksquare

F.2 Closed-form Gradient Expressions

In this section, we provide closed-form expressions for gradients along the linear, slab and noise coordinates: $\nabla_{w_{t_1}} L_f(S_n)$, $\nabla_{w_{t_2}} L_f(S_n)$ and $\nabla_{\bar{w}_i} L_f(S_n)$. First, we provide a closed-form expression for the gradient along the linear coordinate:

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

$$
\nabla_{w_{t_1}} L_f(S_n) = -\frac{\eta v}{4} \left[ 2 + \phi\left(\frac{w_{t_1} + w_{t_2}}{\|w_{t_1}\|}\right) + \phi\left(\frac{w_{t_1} - w_{t_2}}{\|w_{t_1}\|}\right) - 2\phi\left(\frac{w_{t_1}}{\|w_{t_1}\|}\right) \right] \pm \frac{5\eta v}{d} \frac{\log(\text{cd}^2)}{\sqrt{c}}
$$

28
Proof.

\[
\nabla_{w_1} \mathcal{L}_f(S_n) = -\frac{v_i}{n} \sum_{j=1}^{n} \mathbb{1}\{y_j f(x_j) \leq 1\} \mathbb{1}\{w_1^T x_j \geq 0\} y_j x_{i,j}
\]

\[
= -\frac{v_i}{n} \sum_{j=1}^{n} \mathbb{1}\{w_i^T \tilde{x}_j \geq -w_{1i} y_j - w_{2i} \mathbb{1}\{y_j = 1\} \varepsilon_j\}
\]

\[
= -\frac{v_i}{n} \sum_{j=1}^{n} \mathbb{1}\left\{Z_j \geq \frac{-w_{1i} y_j - w_{2i} \mathbb{1}\{y_j = 1\} \varepsilon_j}{||w_i||}\right\}
\]

where \(Z_j \sim \mathcal{N}(0, 1)\)

\[
= -v_i \sum_{l} \frac{1}{n} \sum_{j=1}^{n} \mathbb{1}\{x_{i,j} = l\} \mathbb{1}\left\{Z_j \geq \frac{-w_{1i}(2d^2 - 1) - w_{2i} l}{||w_i||}\right\}
\]

\[
= -v_i \sum_{l} \mathbb{P}(x_{i,j} = l) \phi\left(\frac{w_{1i}(2d^2 - 1) + w_{2i} l}{||w_i||}\right) \pm \frac{\sqrt{\log n}}{n}
\]

via lemma \(\circledast\)

\[
= \frac{v_i}{4} \left[2 + \phi\left(\frac{w_{1i} + w_{2i}}{||w_i||}\right) + \phi\left(\frac{w_{1i} - w_{2i}}{||w_i||}\right) - 2\phi\left(\frac{w_{1i} - w_{2i}}{||w_i||}\right)\right] \pm \frac{5v_i}{d} \sqrt{\frac{\log(c^2d^2)}{c}}
\]

w.p. \(1 - \frac{3}{n}\)

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

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

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

\[
\nabla_{w_2} \mathcal{L}_f(S_n) = -\frac{v_i}{n} \sum_{j=1}^{n} \mathbb{1}\{y_j f(x_j) \leq 1\} \mathbb{1}\{w_1^T x_j \geq 0\} y_j x_{i,j}
\]

Proof.

\[
\nabla_{w_2} \mathcal{L}_f(S_n) = -\frac{v_i}{n} \sum_{j=1}^{n} \mathbb{1}\{y_j f(x_j) \leq 1\} \mathbb{1}\{w_1^T x_j \geq 0\} y_j x_{i,j}
\]

\[
= -\frac{v_i}{n} \sum_{j=1}^{n} \mathbb{1}\{w_i^T \tilde{x}_j \geq -w_{1i} y_j - w_{2i} \mathbb{1}\{y_j = 1\} \varepsilon_j\} \mathbb{1}\{y_j = 1\} \varepsilon_j
\]

\[
= v_i \sum_{j=1}^{n} \frac{1}{n} \sum_{j=1}^{n} (-1) \varepsilon_j \mathbb{1}\{Z_j \geq \frac{-w_{1i} - w_{2i} l}{||w_i||}\}
\]

via lemma \(\circledast\)

\[
= -v_i \sum_{l} \mathbb{P}(x_{i,j} = l) \phi\left(\frac{w_{1i} + w_{2i} l}{||w_i||}\right) \pm \frac{\sqrt{\log n}}{n}
\]

\[
= -\frac{v_i}{4} \left[\phi\left(\frac{w_{1i} + w_{2i}}{||w_i||}\right) - \phi\left(\frac{w_{1i} - w_{2i}}{||w_i||}\right)\right] \pm \frac{5v_i}{d} \sqrt{\frac{\log(c^2d^2)}{c}}
\]

n > cd^2

w.p. \(1 - \frac{3}{n}\)

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

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

\[\]
Lemma 6. If \( n > cd^2 \) and \( y_i, f(x_i) < 1 \) for all \( (x_i, y_i) \in S_n \), then w.p. greater than \( 1 - \frac{1}{5n} \):

\[
\nabla_{\bar{w}_i} L_f(S_n) = \bar{G} \bar{w}_i + 3|v_i| \log(\sqrt{cd}) \frac{\bar{w}_i}{||\bar{w}_i||} \pm \frac{6|v_i|}{\sqrt{cd}} u_i^\perp
\]

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 \( x^S \) and \( x^{S^\perp} \) 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 \( \nabla \bar{w}_i L_f(S_n) \) onto \( S \) and \( S^\perp \) are \( O(\frac{1}{\sqrt{d}}) \):

\[
\nabla_{\bar{w}_i} L_f(S_n)_S = -\frac{v_i}{n} \sum_{j=1}^{n} \{ y_j f(x_j) \leq 1 \} \{ w_i^T x_j \geq 0 \} y_j \bar{x}_j = -\frac{v_i}{n} \sum_{j=1}^{n} \{ v_i^T x_j \geq 0 \} y_j \bar{x}_j^S
\]

where (a) is due to \( \bar{x}_j \perp x_j^{S^\perp} \), (b) is due to \( n \geq cd^2 \). Next, we show that the projection of \( \nabla \bar{w}_i L_f(S_n) \) onto \( S^\perp \) (i.e., case 2) has small norm w.p. greater than \( 1 - \frac{1}{d} \):

\[
||\nabla_{\bar{w}_i} L_f(S_n)_{S^\perp}||_2 = \left\| \frac{v_i}{n} \sum_{j=1}^{n} \{ v_i^T x_j \geq 0 \} y_j \bar{x}_j^{S^\perp} \right\|_2 = \left\| \frac{v_i}{n} \sum_{j=1}^{n} \{ w_i^T x_j \geq 0 \} y_j \bar{x}_j^{S^\perp} \right\|_2
\]

\[
\leq ||v_i|| \cdot \left\| \sum_{j=1}^{n} \mathcal{N}(0, \frac{1}{n^2} I_{d-2}) \right\| = ||v_i|| \cdot ||\mathcal{N}(0, \frac{1}{n} I_{d-2})||
\]

\[
\leq 4|v_i| \sqrt{\frac{d}{n}} + 2|v_i| \sqrt{\frac{\log n}{n}} \leq 6|v_i| \sqrt{\frac{1}{cd}} \text{ w.p. } 1 - \frac{1}{n}
\]

where (a) is due to \( x_j \perp x_j^{S^\perp} \), (b) 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 \( \bar{G} \) w.h.p.:

\[
\nabla_{\bar{w}_i} L_f(S_n)_S = -\frac{v_i}{n} \sum_{j=1}^{n} \{ w_i^T x_j \geq 0 \} y_j \bar{x}_j^S
\]

\[
\leq \left( \frac{1}{n} \sum_{j=1}^{n} \{ w_i^T x_j \geq 0 \} y_j \bar{x}_j^S \right) \frac{v_i \bar{w}_i}{||\bar{w}_i||^2}
\]

\[
= \left( \frac{1}{n} \sum_{j=1}^{n} \{ Z_j \geq -w_{11} v_i - w_{24} \mathbb{1} \{ y_j = 1 \} \frac{v_i}{||\bar{w}_i||} \} \frac{v_i \bar{w}_i}{||\bar{w}_i||} \right)
\]

\[
= \sum_{l}^{(0,1)} \left\{ l \right\} \sum_{i=1}^{n} \{ x_{2j} = l \wedge Z_j \geq \frac{-w_{11}(2l^2-1)-w_{24} l}{||\bar{w}_i||} \} \left\{ \frac{v_i \bar{w}_i}{||\bar{w}_i||} \right\}
\]

\[
\leq 2\varphi(\frac{w_{11}}{||\bar{w}_i||}) - \varphi(\frac{w_{11} + w_{24}}{||\bar{w}_i||}) - \varphi(\frac{w_{11} - w_{24}}{||\bar{w}_i||}) + \frac{5 \log n}{\sqrt{n}} \frac{v_i \bar{w}_i}{||\bar{w}_i||} \text{ w.p. } 1 - \frac{12}{n}
\]

where (a) is due to \( \bar{x}_j \perp x_j^S \), (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 \( 1 - \frac{13}{n} \):

\[
\nabla_{\bar{w}_i} L_f(S_n) = \bar{G} \bar{w}_i + 3|v_i| \log(\sqrt{cd}) \frac{\bar{w}_i}{||\bar{w}_i||} \pm \frac{6|v_i|}{\sqrt{cd}} u_i^\perp
\]

\[\square\]
F.3 Miscellaneous Lemmas

Lemma 7. Let $X_i \sim \mathcal{N}(0, \sigma^2)$ and $\delta \in (0, 1)$. Then, $\max_{i \in [k]} |X_i| \leq \sigma \sqrt{2 \log \left( \frac{2k}{\delta} \right)}$ 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:
\[
P(|X| \geq \sigma t) = P(|Z| \geq t) = 2 \int_t^{\infty} x \varphi(x) \, dx \leq \frac{2}{t} \int_t^{\infty} x \varphi(x) \, dx \leq \frac{2}{t} \int_0^{\infty} \varphi'(x) \, dx \leq 2 \varphi(t)
\]
where $(a)$ is because $\varphi'(x) = -x \varphi(x)$. Using union bound with $t = \sqrt{2 \log \left( \frac{2k}{\delta} \right)} \geq 1 \forall \delta \in (0, 1)$ gives the desired result. ■

Lemma 8. Let $\phi$ and $\varphi$ denote the cumulative distribution function and the probability density function of the standard normal. Then, for any $Z \sim \mathcal{N}(0, 1)$ and $k \in \mathbb{R}$:
\[
E[1 \{ Z \geq k \} Z] = \varphi(k) = \exp(-k^2/2)
\]

Proof. The expectation $E[1 \{ Z \geq k \} Z]$ can be simplified as follows:
\[
E[1 \{ Z \geq c \} Z] = \Pr[Z \geq c]E[Z | Z \geq c] = \phi'(c) \int_c^{\infty} x \varphi(x) \, dx = \varphi(c)
\]
where $(a)$ is due to $\varphi'(x) = -x \varphi(x)$. ■

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

Then:
\[
\Pr \left( |\bar{X} - p \varphi(-k)| \geq \sqrt{\frac{\log n}{n}} \right) \leq \frac{1}{n}
\]

Proof. Note that $E[1] = E[|X_i|] = E[b_i]E[1 \{ Z_i \geq k \}] = p \varphi(k)$ and $|X_i| \leq 1$. Therefore, using Hoeffding’s inequality with $t = \sqrt{\frac{\log n}{n}}$ directly gives the result. ■

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

Then:
\[
P(|\bar{X} - p \varphi(k)| \leq t^*) \geq P(|\bar{X} - p \varphi(k)| \leq t^* | A) P(A) \geq \left(1 - \frac{2}{n}\right)^2 \geq 1 - \frac{4}{n}
\]

Proof. Since $|X_i| = |b_i 1 \{ Z_i \geq k \} Z_i| \leq |Z_i|$, 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 $E[X_i] = E[b_i]E[1 \{ Z_i \geq k \} Z_i] = p \varphi(k)$. Let $A = \{ |X_i| \leq \sqrt{4 \log(n)} \forall i \in [n] \}$. 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:
\[
P(|\bar{X} - p \varphi(k)| \leq t^*) \geq P(|\bar{X} - p \varphi(k)| \leq t^* | A) P(A) \geq \left(1 - \frac{2}{n}\right)^2 \geq 1 - \frac{4}{n}
\]

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

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

Proof. Note that $g_z'(x) = \frac{1}{x^2} \exp(-\frac{x^2}{2z^2}) \left( \frac{x^2}{z^2} - 1 \right)$. 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. ■

Fact 1. Let $X \sim \mathcal{N}(0, \sigma^2 I_d)$ denote a $d$-dimensional gaussian vector. Then, from [70], w.p. greater than $1 - \delta$:
\[
||X||_2 \leq 4\sigma \sqrt{d} + 2\sigma
\]