Bad Global Minima Exist and SGD Can Reach Them

Shengchao Liu, Dimitris Papailiopoulos
University of Wisconsin–Madison

Dimitris Achlioptas
University of California, Santa Cruz

Abstract

Several recent works have aimed to explain why severely overparameterized models, generalize well when trained by Stochastic Gradient Descent (SGD). The emergent consensus explanation has two parts: the first is that there are “no bad local minima”, while the second is that SGD performs implicit regularization by having a bias towards low complexity models. We revisit both of these ideas in the context of image classification with common deep neural network architectures. Our first finding is that there exist bad *global* minima, *i.e.*, models that fit the training set perfectly, yet have poor generalization. Our second finding is that given only *unlabeled* training data, we can easily construct initializations that will cause SGD to quickly converge to such bad global minima. For example, on CIFAR, CINIC10, and (Restricted) ImageNet, this can be achieved by starting SGD at a model derived by fitting random labels on the training data: while subsequent SGD training (with the correct labels) will reach zero training error, the resulting model will exhibit a test accuracy degradation of up to 40% compared to training from a random initialization. Finally, we show that regularization seems to provide SGD with an escape route: once heuristics such as data augmentation are used, starting from a complex model (adversarial initialization) has no effect on the test accuracy.

1 Introduction

In [1], Zhang et al. demonstrated that several popular deep neural network architectures for image classification have enough capacity to perfectly memorize the CIFAR10 training set. That is, they can achieve zero training error, even after the training examples are relabeled with uniformly random labels. Moreover, such memorizing models are not even hard to find; they are reached by standard training methods such as stochastic gradient descent (SGD) in about as much time as it takes to train with the correct labels. It would stand to reason that since these architectures have enough capacity to “fit anything,” models derived by fitting the correctly labeled data, *i.e.*, “yet another anything,” would fail to generalize. Yet, miraculously, they do not: models trained by SGD on even severly overparameterized architectures generalize spectacularly. Following recent work [2, 3, 4, 5, 6, 7], our study is motivated by the desire to shed some light onto this miracle which stands at the center of the recent machine learning revolution.

When the training set is labeled randomly, all models that minimize the corresponding loss function are equivalent in terms of generalization, in the sense that, we expect none of them to generalize. The first question we ask is: when the true labels are used, are *all* models that minimize the loss function equivalent in terms of generalization, or are some better than others? We show that not all global minima are created equal: there exist bad *global* minima, *i.e.*, global minima that generalize poorly.

The existence of bad global minima implies that the optimization method used for training, *i.e.*, to select among the (near-)global minima, has germane effect on generalization. In practice, SGD appears to avoid bad global minima, as different models produced by SGD from independent random initializations tend to all generalize equally well, a phenomenon attributed to an inherent bias of the algorithm to converge to models of low complexity [8, 9, 10, 11, 12, 13]. This brings about our second question: does SGD deserve all the credit for avoiding bad global minima, or are there

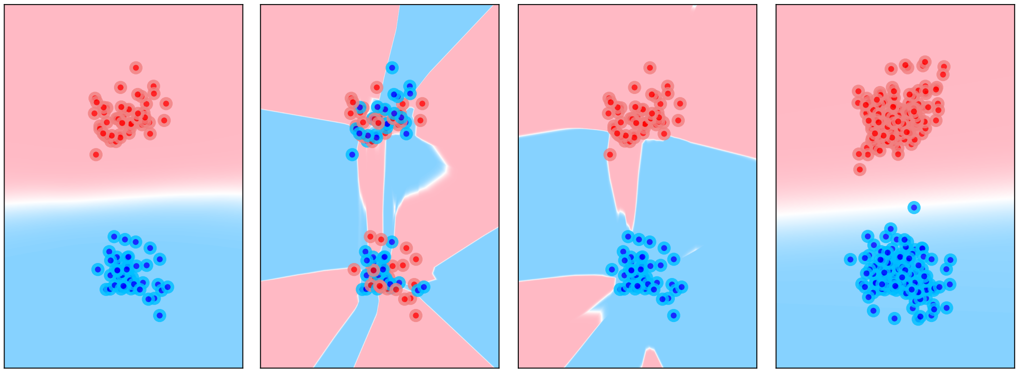
Preprint. Under review.

also other factors at play? More concretely, can we initialize SGD so that it ends up at a bad global minimum? Of course, since we can always start SGD *at* a bad global minimum, our question has a trivial positive answer as stated. We show that initializations that cause SGD to converge to bad global minima can be constructed given only *unlabeled* training data, *i.e.*, without any idea of the true loss landscape.

The fact that we can construct adversarial initializations without knowledge of the loss landscape suggests that these initializations correspond to models whose *inherently undesirable* characteristics persist, at least partially, in the trained models that perfectly fit the training examples with correct labels. Such a priori undesirability justifies a priori preference of some models over others, *i.e.*, regularization. In particular, if a regularization term makes an adversarial initialization appear a far worse model than before, this correspondingly incentivizes SGD to move away from it, a tendency amplified by the use of momentum during optimization. This is precisely what we find in our experiments: adding regularization and momentum allows SGD to overcome our adversarial initializations and end up at good global minima. In other words, in penalizing inherent characteristics of models, it appears that regularization plays a role beyond that of distinguishing between different models that fit the data equally well: it affects training dynamics, making good models easier to find, perhaps by making bad models more evidently bad.

A Sketch of the Phenomenon Consider training a two-layer, fully-connected, neural network for a binary classification task, where the training data are sampled from two identical, well-separated 2-dimensional Gaussians. Each class comprises 50 samples, while the network has 100 hidden units in each layer and uses ReLU activations. In Figure 1, we show the decision boundary of the model reached by training with SGD with a batch size of 10 until 100% accuracy is reached under the following four settings:

1. True labels, random initialization.
2. Random labels, random initialization.
3. True labels, initialization at the model in Fig. 1(b) (reached after training under setting 2).
4. Same as setting 3, but with data augmentation, l_2 regularization, and momentum[1].



(a) Setting 1 (b) Setting 2 (c) Setting 3 (d) Setting 4

Figure 1: The decision boundary of the model reached by SGD in Settings 1–4, respectively.

Figure 1(a) shows that from a random initialization, SGD converges to a model with near max margin, which may attributed to its implicit bias. Figure 1(b) shows that when fitting random labels, the decision margin becomes extremely complex and has miniscule margin. Figure 1(c) shows that when SGD is initialized at such an extremely complex model, it converges to a "nearby" model whose decision boundary is unnatural and has small margin. Finally, in Figure 1(d), we see that when data augmentation, regularization and momentum are added, SGD escapes the bad initialization and again reaches a model with a max margin decision boundary.

In the next section, we show that the phenomenon sketched above in a toy setting persists in state-of-the-art-neural network architectures over real data sets. We specifically examine VGG16, ResNet18, ResNet50, and DenseNet40, when trained on CIFAR, CINIC, and a restricted version of ImageNet.

[1] Data augmentation was performed by replicating each training point twice and adding Gaussian noise.

We consistently observe the following: 1) bad global minima exist; 2) initializations that cause SGD to converge to them can be easily derived given only unlabeled training data; 3) each of data augmentation, regularization, and momentum help SGD avoid reaching bad global minima by allowing the model to escape far away from such adversarial initializations.

2 Experimental Setup

Datasets and Architectures We ran experiments on the CIFAR [14] data set (including both CIFAR10 and CIFAR100), CINIC10 [15] and a resized Restricted ImageNet [16].[2] We train on them four models: VGG16 [17], ResNet18 and ResNet50 [18] and DenseNet40 [19].

Implementation and Reproducibility We run our experiments on PyTorch 3.0. Our figures, models, and all results can be reproduced using the code available at a GitHub repository: `https://github.com/chao1224/BadGlobalMinima`.

Training methods We consider state-of-the-art (SOTA) SGD training and vanilla SGD training. The former corresponds to SGD with ℓ_2-regularization, data augmentation (random crops and flips), and momentum. The latter to SGD without any of these features.

Initialization We consider two kinds of initialization: random and adversarial. For random initializations we use the PyTorch default. To create an adversarial initialization, we train the model on an augmented version of the original training dataset in which we have labeled every example uniformly at random.

Algorithm 1 Adversarial initialization

```
Input: Original training dataset $S$; Replication factor $R$; Noise factor $N$
$C = \emptyset$
for every image $x \in S$ do
    for $i$ from 1 to $R$ do
        $x_i \leftarrow$ zero-out a random subset comprising $N\%$ of the pixels in $x$
        $y_i \leftarrow$ Uniformly random label
        Add $(x_i, y_i)$ to $C$
Train the architecture to $100\%$ accuracy on $C$ from a random initialization using vanilla SGD [3]
Output: The weight vector of the architecture when training ends
```

Hyperparameters For CIFAR, CINIC10, and Restricted ImageNet, we use batch size 128, while the momentum term is set to 0.9 when it is used. When we use ℓ_2 regularization, the regularization parameter is $5 \cdot 10^{-4}$ for CIFAR and Restricted ImageNet and 10^{-4} for CINIC10. We use the following learning rate schedule for CIFAR: 0.1 for epochs 1 to 150, 0.01 for epoch 151 to 250, and 0.001 for epochs 251 to 350. We use the following learning rate schedules for CINIC and Restricted ImageNet: 0.1 epochs 1 to 150, 0.01 for epoch 151 to 225, and 0.001 for epochs 226 to 300. The CIFAR training set consists of 50k data points and the test set consists of 10k data points. The CINIC10 training set consists of 90k data points and the test set consists of 90k data points. The Restricted ImageNet training set consists of approximately 123k data points and the test set consists of 4.8k data poitns.

3 Experimental Findings and Observed Phenomena

The motivation behind our adversarial initialization comes from our expectation that memorizing random labels will consume some of the learning capacity of the network, and potentially reduce the positive effects of overparametrization. Furthermore, as seen in our toy example in Section 1,

[2] We resize the images to 32×32 for faster computation, which leads to a test accuracy drop from 96% to 87%.

[3] Some architectures cannot reach 100% accuracy on random labels. In this case, we train the model until it reaches its highest possible accuracy.

adversarial initialization tends to encourage SGD towards extremely complex decision boundaries, *i.e.*, decision regions that look surprising given the expectation of an implicit SGD bias toward simplicity.

In the following we present several metrics related to the trained models. To generate the figures below, we ran each setup 5 times with different random seeds and reported the min, max and average test accuracy for each model and data set.

We first report the test and train error curves for our 16 setups (4 data sets and 4 models), followed by the impact of the replication parameter R on the test error. We then examine how different combinations of training heuristics (*e.g.*, momentum, regularization, and data augmentation) impact the overall test error with and without adversarial initialization. After this, we report the distance that a model travels, first from a random initialization to the adversarial initializer and then from the adversarial initializer to the final model trained on the correct labels. Finally, we report several norms that are known proxies for model complexity, and also the robustness of all models trained against adversarial perturbations, *i.e.*, yet another model complexity proxy.

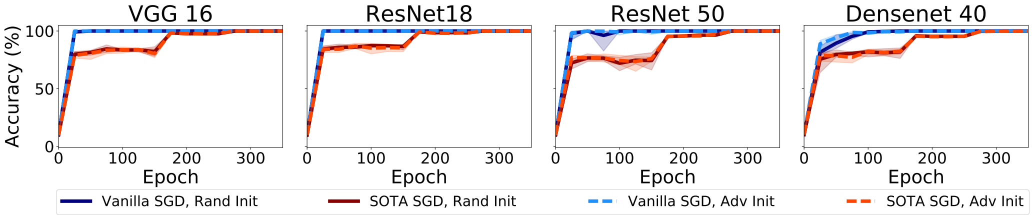
Our main observations as taken from the figures below and our experimental data are as follows:

1. Vanilla SGD with random initialization reaches 100% training accuracy for all models and data sets tested, which is consistent with [1].
2. Vanilla SGD with adversarial initialization suffers up to a 40% test accuracy degradation compared to random initialization. That is, SGD models that are near global optima can have a difference of up to 40% in test accuracy: not all training global optima are equally good.
3. SOTA SGD with explicit regularization, converges to the same test error from random vs. adversarial initialization.
4. Data augmentation, momentum, and ℓ_2 regularizatrion allow SGD to move far away (in euclidean distance) from adversarial initializations; in sharp contrtast, vanilla SGD finds a minimizer close to the adversarial initialization.

3.1 Training Accuracy

In Figure 2 we report the training accuracy convergence for all models setups on CIFAR10. For brevity, we omit reporting the remaining data sets (where nearly identical behavior is observed); all figures can be found in the supplemental material. We consistently observe two phenomena.

First, we see that after sufficient number of epochs for most setups we can reach 100% training accuracy irrespective of the initialization, or the use of momentum, regularization or data augmentation. The only outlier here would be DenseNet since it seems to only achieve a little less than 100% training accuracy in some data sets. However, in the vast majority of settings the models can perfectly fit the training data.



(a) Training Accuracy on CIFAR10

Figure 2: Training accuracy vs number of epochs on CIFAR10 on all four neural network models.

The second phenomenon we observe is that the speed of convergence to 100% training accuracy is irrespective of whether we start with adversarial or random initialization. This may hint at the possibility that models that fit the train labels exist close to most initializations, including one that is adversarial.

3.2 Test accuracy

Our most important findings are the test accuracy curves shown in Figure 3, showing the test accuracy convergence during training. We see that the test accuracy of adversarially initialized vanilla SGD flattens out significantly below the corresponding accuracy under a random initialization, even though both methods achieve 100% training accuracy. The test error degradation can be up to 40% on CIFAR100, while for DenseNet the test degradation is comparatively smaller.

At the same time, we see that the detrimental effect of adversarial initialization vanishes once we use data augmentation, momentum, and ℓ_2 regularization. This demonstrates that by also changing the optimization landscape *far* from good models, regularization plays a role that has not received much attention, namely effecting the *dynamics* of the search for good models.

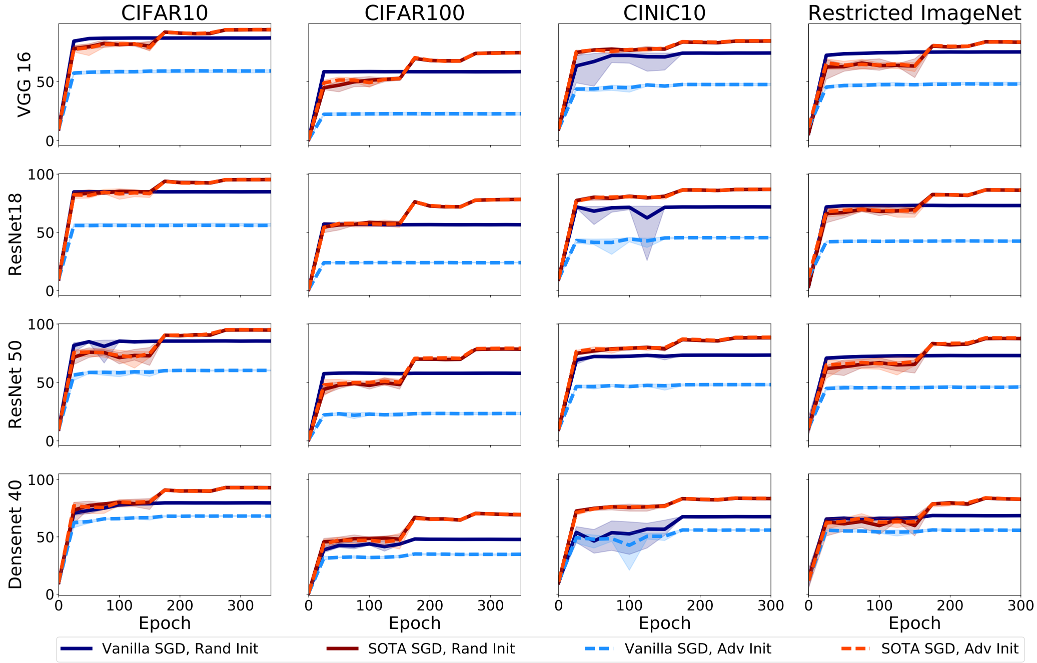


Figure 3: Test accuracy (%) vs number of epochs on CIFAR, CINIC10 and Restricted ImageNet on all four neural network models.

It is natural to ask if the bad global models to which SGD converges when adversarially initialized have some shared properties. In the following, we argue that one property that stands out is that such bad global minima are in a small neighborhood within the adversarial initializers; and, it appears, that as long as it can find a perfect fit, SGD prefers these models, as it take less “effort" to converge to. In contrast, as we see later on, SOTA SGD forces travel far from the bad initialization.

3.3 Distance Travelled during Training

Here we report on the distance travelled from an initializer till the end of training in our different setups. First we define the distance between the parameters of two models W_1 and W_2 models as $d(W_1, W_2) = \frac{\|W_1 - W_2\|_F}{\|W_2\|_F}$, where $\|\cdot\|_F$ denotes the Frobenius norm.

In Table 1, we report the distance travelled from a random vs. an adversarial initialization to the final model (achieving 100% training accuracy). A subscript 0 denotes an initializer; an S or V subscript indicates SOTA, or vanilla SGD training. The R or A superscripts indicate whether the model has been initialized at random or adversarially.

We observe that from a random initialization, the distance that vanilla SGD travels to a model with 100% training accuracy is independent of whether we use the correct labels or random labels. Specifically, whether we want to train a genuinely good model, or to find an adversarial initialization the distance traveled is about 0.9. Intriguingly, when vanilla SGD is initialized adversarially, the distance to a model with 100% training accuracy on the correct labels is far less than 0.9, being approximately 0.2. This can be interpreted as SGD only spending a modicum of effort to “fix up” a bad model, just enough to fit the training labels.

In contrast, when training with the correct labels, the distance travelled by SOTA SGD to a model with 100% training accuracy is significantly larger if we initialize adversarially vs. if we initialize randomly, in most cases by nearly an order of magnitude. This hints at the possibility that regularization enables SGD to escape the bad initialization by dramatically altering the landscape in its vicinity (and beyond).

Finally, we observe that bad models seem to always be in close proximity to random initializers, *i.e.*, bad models are easy to find from almost every point of the parameter space.

Table 1: The model distance (mean of 5 random runs) for the different data sets and models.

Dataset	Model	$d(W_0^R, W_V^R)$	$d(W_0^R, W_S^R)$	$d(W_0^R, W_0^A)$	$d(W_0^R, W_V^A)$	$d(W_0^R, W_S^A)$	$d(W_0^A, W_V^A)$	$d(W_0^A, W_S^A)$
CIFAR10	DenseNet40	0.810	3.715	0.946	0.950	3.715	0.347	28.669
	ResNet18	0.953	3.917	0.873	0.879	3.902	0.207	12.421
	ResNet50	0.894	8.552	0.919	0.923	8.608	0.194	36.311
	VGG16	0.907	3.464	0.950	0.953	3.433	0.264	26.838
CIFAR100	DenseNet40	0.917	2.008	0.946	0.968	1.980	0.538	13.290
	ResNet18	0.915	2.609	0.852	0.862	2.597	0.210	7.243
	ResNet50	0.917	4.941	0.925	0.927	4.882	0.172	24.598
	VGG16	0.919	2.029	0.960	0.962	2.035	0.253	19.388
CINIC10	DenseNet40	0.917	1.415	0.955	0.976	1.440	0.491	12.102
	ResNet18	0.836	1.565	0.942	0.951	1.560	0.212	10.961
	ResNet50	0.837	3.260	0.952	0.955	3.274	0.157	24.940
	VGG16	0.844	1.406	0.974	0.977	1.397	0.241	17.916
Restricted ImageNet	DenseNet 40	0.940	3.378	0.966	0.981	3.383	0.518	39.364
	ResNet 18	0.849	3.154	0.969	0.973	3.164	0.163	47.184
	ResNet 50	0.883	7.123	0.913	0.916	7.139	0.164	30.841
	VGG 16	0.836	2.885	0.985	0.986	2.871	0.172	83.035

3.4 The Effect of the Replication Factor R on Test Error

Here, we report the test accuracy effect of the replication factor R, *i.e.*, the number of randomly labeled augmentations that are applied to each point during adversarial initialization. In Figure 4, we plot the test accuracy performance for all networks we tested as a function of the number of the randomly labeled augmentations R. When we vary R we observe that SOTA SGD essentially achieves the same test error, while the test performance of vanilla SGD degrades, initially fast, and then slower for larger R.

We would like to note that although it would be interesting to make R even bigger, the time needed to generate the adversarial initializer grows proportional to R, as it requires training a data set (of size proportional to R) to full accuracy.

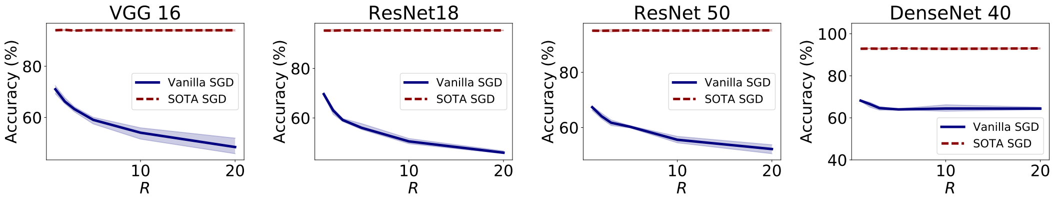


Figure 4: The effect of R on CIFAR10, the zero-out ratio is fixed to 10%. Clearly, increasing R causes vanilla SGD to suffer more. In contrast, SOTA SGD is always unaffected.

3.5 The Effect of Different Training Heuristics

In our implementation, SOTA SGD involves the simultaneous use of ℓ_2 regularization, data augmentation, and momentum. Here we tease out the different effects, by exploring all 8 combinations of the 3 techniques as shown in Table 2, in order to examine if a specific technique is particularly effective at repairing the test error damage done by an adversarial initialization. What we find is that each technique by itself is enough to allow SGD to largely escape a bad initialization, but not to reach the same level of test accuracy as from a random initialization. When a second technique is added, though, the test error becomes independent of the initialization, in all three combinations.

Table 2: Multiple SGD results with random init and adversarial init respectively using model ResNet-18 on CIFAR10.

	Random Init		Adversarial Init	
Mode	Train Acc	Test Acc	Train Acc	Test Acc
Vanilla SGD	100.000 ± 0.000	84.838 ± 0.193	100.000 ± 0.000	56.024 ± 0.883
DA	100.000 ± 0.000	93.370 ± 0.115	99.995 ± 0.003	89.402 ± 0.175
ℓ_2	100.000 ± 0.000	87.352 ± 0.055	100.000 ± 0.000	83.172 ± 3.732
Momentum	100.000 ± 0.000	89.200 ± 0.176	100.000 ± 0.000	89.016 ± 0.142
DA+ℓ_2	100.000 ± 0.000	94.680 ± 0.071	100.000 ± 0.000	94.192 ± 0.173
DA+Momentum	100.000 ± 0.000	93.448 ± 0.242	100.000 ± 0.001	92.756 ± 0.347
ℓ_2+Momentum	100.000 ± 0.000	89.050 ± 0.110	100.000 ± 0.000	88.932 ± 0.373
DA+ℓ_2+Momentum (SOTA)	100.000 ± 0.000	95.324 ± 0.086	100.000 ± 0.001	95.346 ± 0.098

3.6 Proxies for Model Complexity

Here we report on some popular proxies for model complexity, *e.g.*, the Frobenius norm of the weights of the network, and recently studied path norms [2]. For brevity, we only report the results for ResNet50, while all remaining figures can be found in the supplemental material.

We observe that across all three norms, the network tends to have smaller norm for SOTA SGD irrespective of the initialization. At the same time, we observe that the adversarially initialized model trained with vanilla SGD has larger norms compared to random initialization.

If these norms are a proxy for generalization, then they indeed do track our observations that adversarial initialization leads vanilla SGD to worse generalization, and SOTA SGD explicitly biases towards good global models, while being unaffected by adversarial initialization.

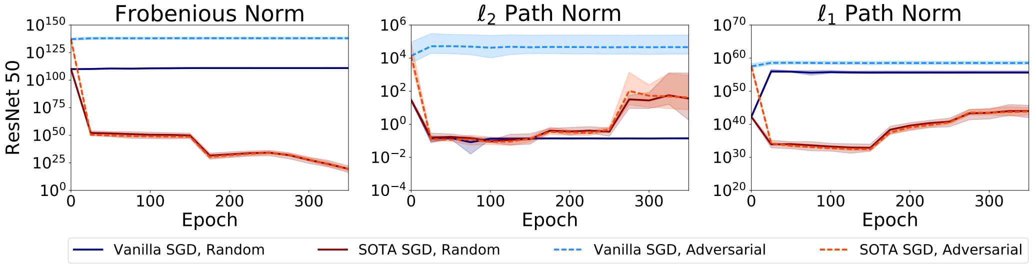


Figure 5: Norm measurement on CIFAR10 and ResNet 50.

3.7 Robustness to Adversarial Examples

Robustness to adversarial examples is another metric we examine. In this case, model robustness is a direct proxy for margin, *i.e.*, the proximity of the decision boundary to the train, or test data. In Figure 6, we report the test accuracy degradation once adversarial perturbations are crafted for Resnet50 on all four data sets. To compute the adversarial perturbation we use the Fast Gradient Sign Attack (FGSM) on the final model.

Consistent with the norm measures in the previous subsection, we observe that when adversarially initialized, vanilla SGD finds models that are more prone to small perturbations that lead to misclassification. This is to be expected, since (as also observed in our toy example), the decision boundaries for adversarially initialized vanilla SGD tend to be complex, and also are very close to several training points (*i.e.*, their margin is small).

As observed in the previous figures, the decision boundaries of models derived by SOTA SGD are less prone to adversarial attacks, potentially due to the fact that they achieve better margin.

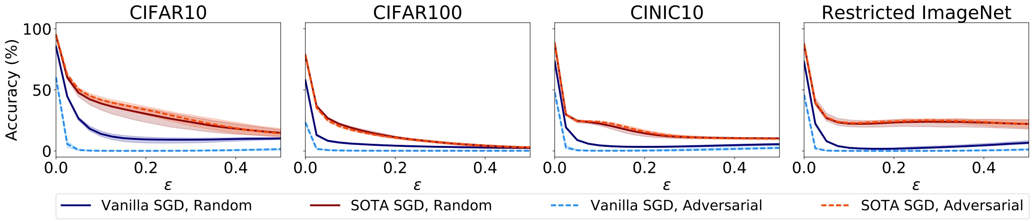


Figure 6: FGSM on CIFAR, CINIC10 and Restricted ImageNet using ResNet-50.

4 Conclusion

Understanding empirical generalization in the face of severe overparameterization has emerged as a central and fascinating challenge in machine learning, primarily due to the dramatic success of deep neural networks. Several studies aim to explain this phenomenon. Besides "no bad local minima," the emergent consensus explanation is that SGD has "a taste for" minima of low complexity.

In this work, we show that not only bad *global* minima exist, *i.e.*, models that fit the training set perfectly, yet have poor generalization but, moreover, that these bad global minima are attractive to SGD even from initializations constructed from *unlabeled* data. We also demonstrate empirically that regularization rescues SGD from these adversarial initializations.

We believe that the main value of our work is in pointing out the role played by regularization in enabling SGD to escape from our adversarial initializations. Not because we consider such initializations particularly important in and of themselves, but because the phenomenon observed highlights the role of regularization in altering the dynamics of the search for good models. In other words, while the typical view of regularization is as a way to distinguish between good models (by minimizing risk), our work shows that the alteration of the optimization landscape induced by regularization is highly relevant even *very far* from good models. In that sense, we view the value of our work as an invitation for further work on this seemingly germane but largely unexplored point.

References

[1] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. 2017.

[2] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems*, pages 5947–5956, 2017.

[3] Tomaso Poggio, Kenji Kawaguchi, Qianli Liao, Brando Miranda, Lorenzo Rosasco, Xavier Boix, Jack Hidary, and Hrushikesh Mhaskar. Theory of deep learning iii: explaining the non-overfitting puzzle. *arXiv preprint arXiv:1801.00173*, 2017.

[4] Roman Novak, Yasaman Bahri, Daniel A Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. Sensitivity and generalization in neural networks: an empirical study. *arXiv preprint arXiv:1802.08760*, 2018.

[5] Alon Brutzkus, Amir Globerson, Eran Malach, and Shai Shalev-Shwartz. Sgd learns over-parameterized networks that provably generalize on linearly separable data. *arXiv preprint arXiv:1710.10174*, 2017.

[6] Mikhail Belkin, Siyuan Ma, and Soumik Mandal. To understand deep learning we need to understand kernel learning. *arXiv preprint arXiv:1802.01396*, 2018.

[7] Siyuan Ma, Raef Bassily, and Mikhail Belkin. The power of interpolation: Understanding the effectiveness of sgd in modern over-parametrized learning. *arXiv preprint arXiv:1712.06559*, 2017.

[8] Suriya Gunasekar, Blake E Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro. Implicit Regularization in Matrix Factorization. page 9, 2017.

[9] Behnam Neyshabur, Ryota Tomioka, Ruslan Salakhutdinov, and Nathan Srebro. Geometry of Optimization and Implicit Regularization in Deep Learning. *arXiv:1705.03071 [cs]*, May 2017. arXiv: 1705.03071.

[10] Junhong Lin, Raffaello Camoriano, and Lorenzo Rosasco. Generalization Properties and Implicit Regularization for Multiple Passes SGM. page 9, 2016.

[11] Behnam Neyshabur. Implicit Regularization in Deep Learning. *arXiv:1709.01953 [cs]*, September 2017. arXiv: 1709.01953.

[12] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In Search of the Real Inductive Bias: On the Role of Implicit Regularization in Deep Learning. *arXiv:1412.6614 [cs, stat]*, December 2014. arXiv: 1412.6614.

[13] 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.

[14] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[15] Amos Storkey, Antreas Antoniou, Luke N Darlow, Elliot J Crowley, et al. Cinic-10 is not imagenet or cifar-10. 2018.

[16] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *stat*, 1050:11, 2018.

[17] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016.

[19] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely Connected Convolutional Networks. pages 2261–2269. IEEE, July 2017.

A CIFAR10 Complete Results

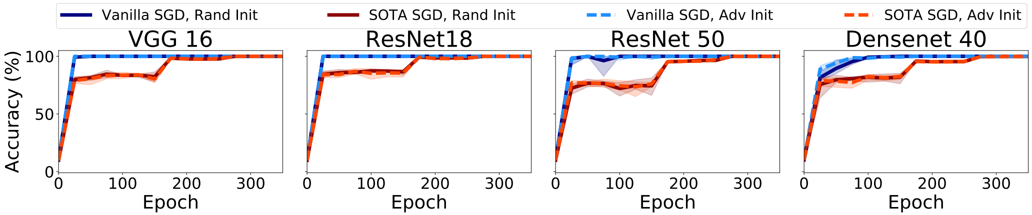


Figure 7: The training accuracy convergence on CIFAR10.

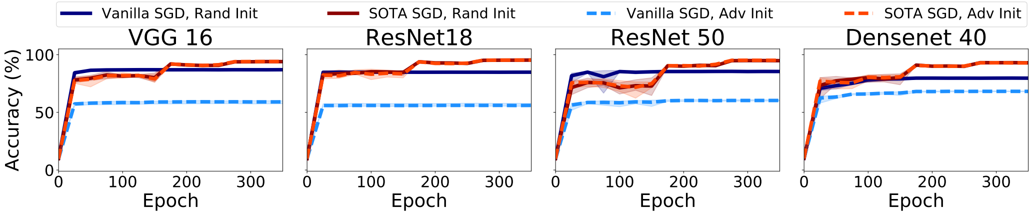


Figure 8: The test accuracy convergence on CIFAR10.

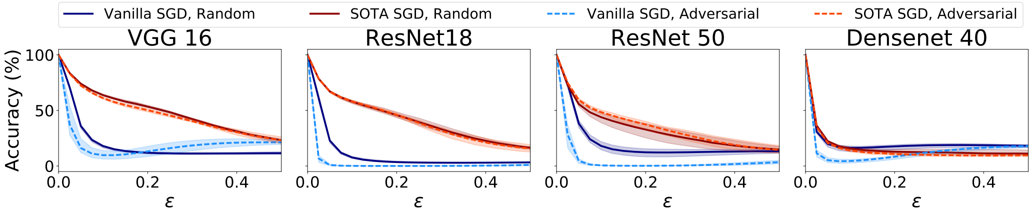


Figure 9: Fast Gradient Sign Attack (FGSM) on CIFAR10.

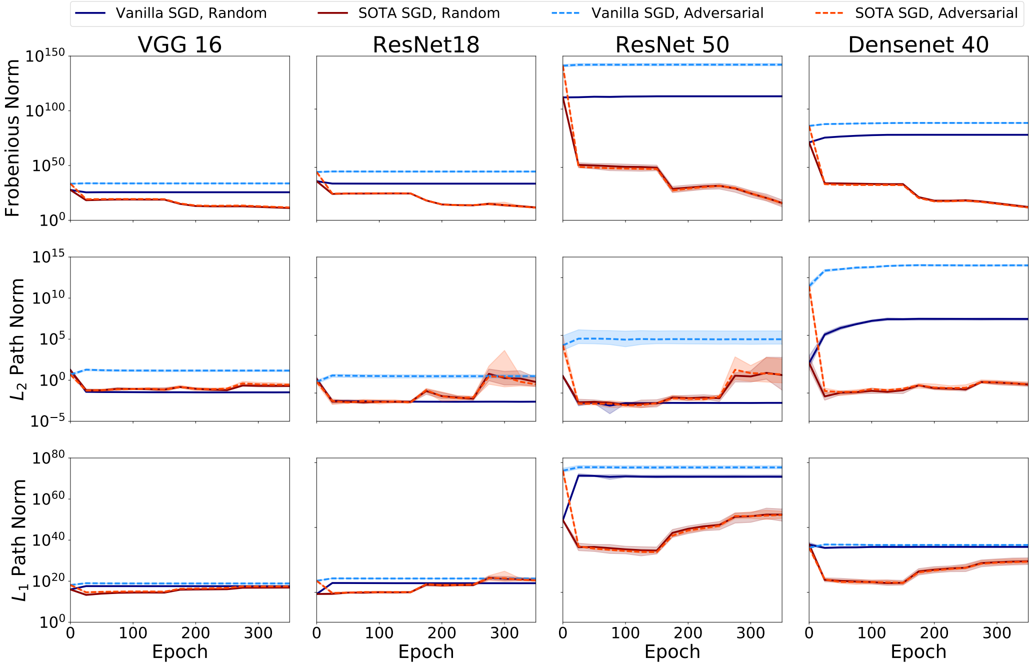


Figure 10: Model Complexity on CIFAR10.

B CIFAR100 Complete Results

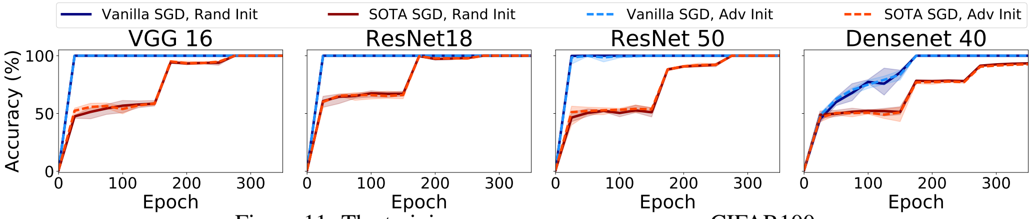


Figure 11: The training accuracy convergence on CIFAR100.

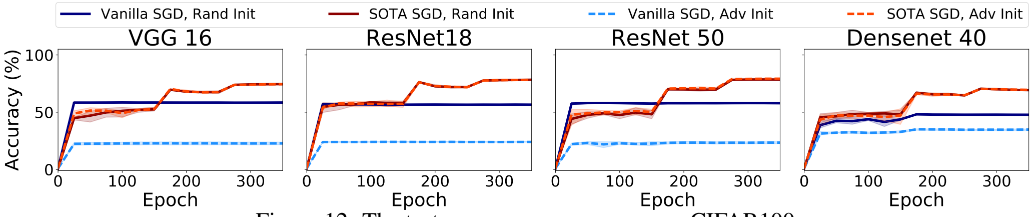


Figure 12: The test accuracy convergence on CIFAR100.

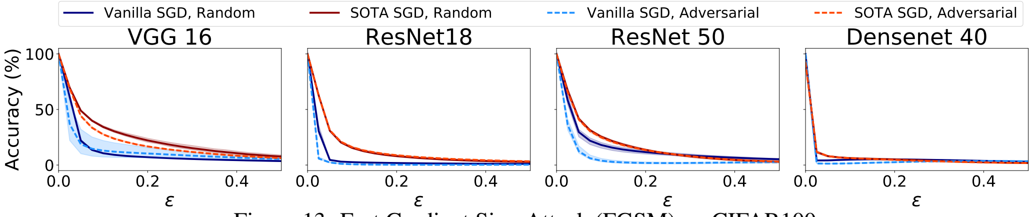


Figure 13: Fast Gradient Sign Attack (FGSM) on CIFAR100.

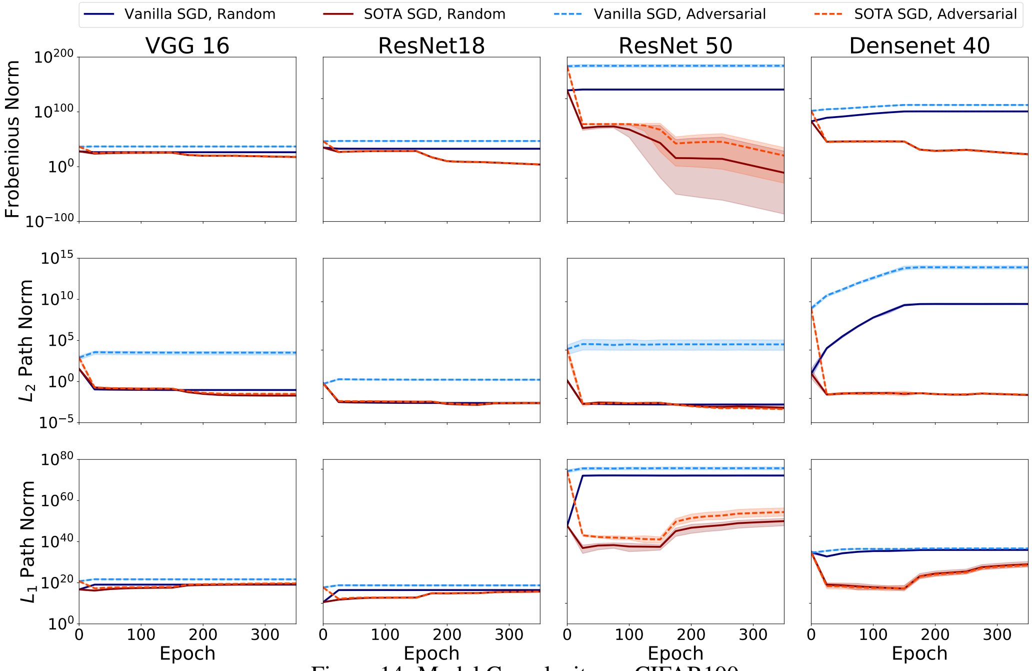


Figure 14: Model Complexity on CIFAR100.

C CINIC10 Complete Results

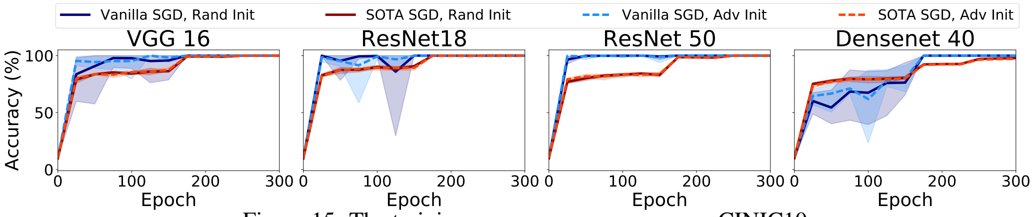


Figure 15: The training accuracy convergence on CINIC10.

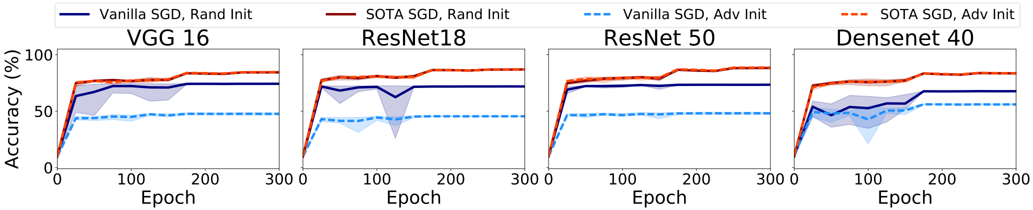


Figure 16: The test accuracy convergence on CINIC10.

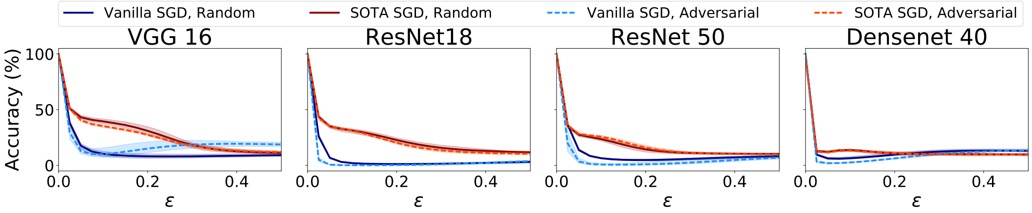


Figure 17: Fast Gradient Sign Attack (FGSM) on CINIC10.

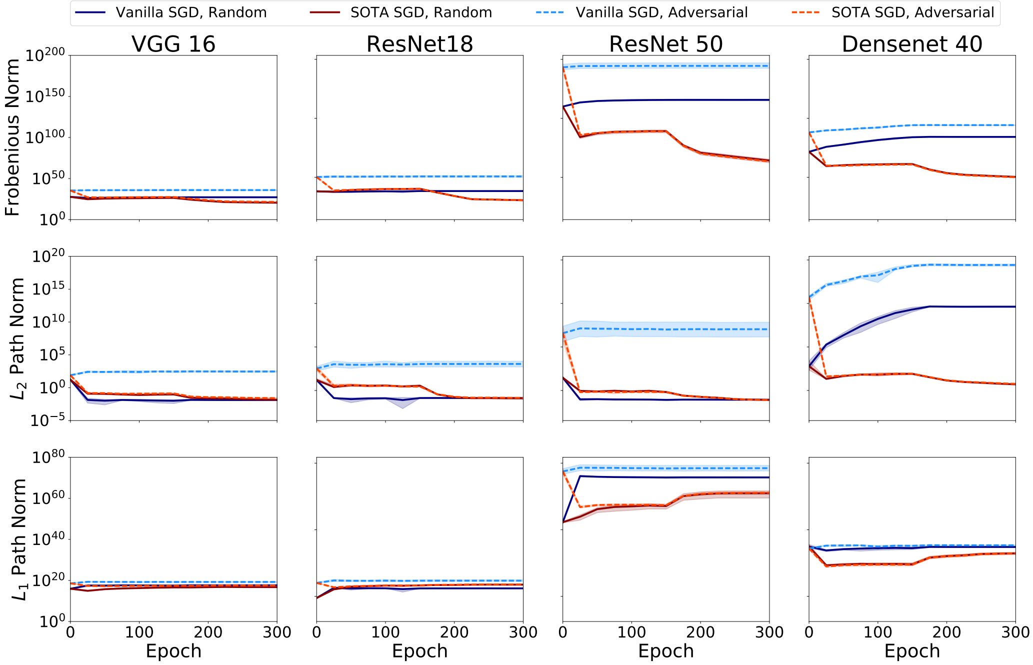


Figure 18: Model Complexity on CINIC10.

D Restricted ImageNet Complete Results

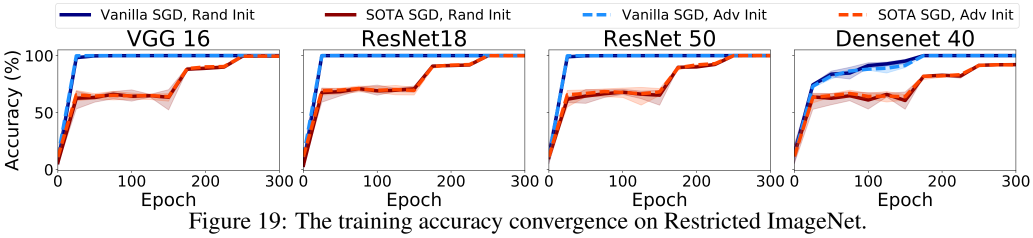


Figure 19: The training accuracy convergence on Restricted ImageNet.

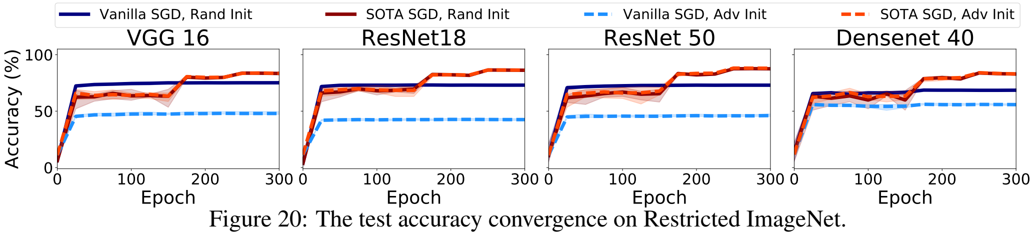


Figure 20: The test accuracy convergence on Restricted ImageNet.

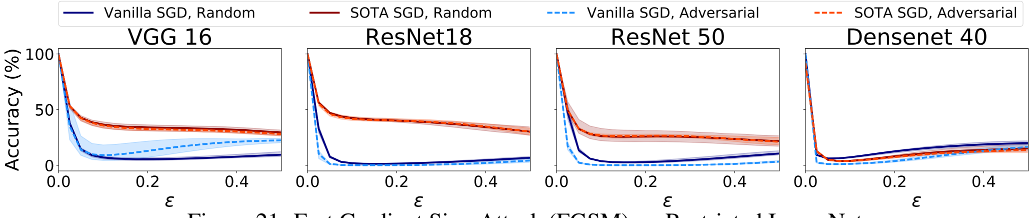


Figure 21: Fast Gradient Sign Attack (FGSM) on Restricted ImageNet.

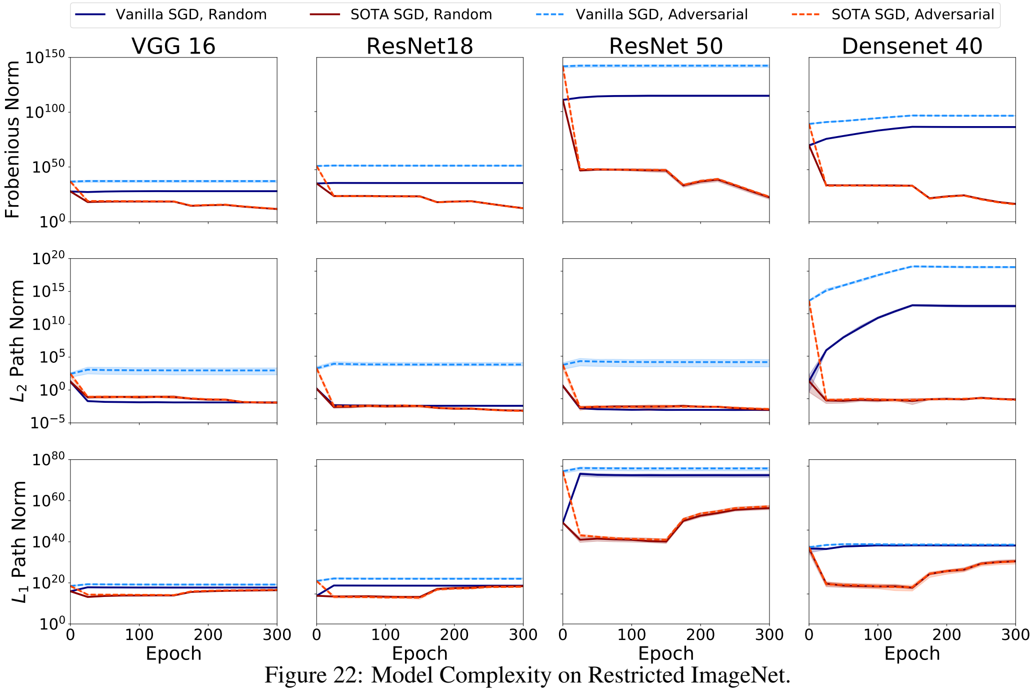


Figure 22: Model Complexity on Restricted ImageNet.