ADVERSARIAL SPHERES

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce a simple synthetic task in order to study the phenomenon of adversarial examples. For this task, the data manifold is mathematically defined and there is an analytic characterization of the model's decision boundary. We show that when the data is high dimensional, generalization error may be so low that model errors may only be found *adversarially*. Despite these errors being extremely unlikely they appear to always be close to randomly sampled "clean" data. These adversarial examples exist on the data manifold and even for a model class which can provably obtain perfect accuracy. We conclude by drawing connections to adversarial examples for other machine learning models.

1 INTRODUCTION

There has been substantial work demonstrating that standard machine learning models are susceptible to *adversarial examples*: small, imperceptible perturbations added to the input data causing misclassifications (Goodfellow et al., 2014; Szegedy et al., 2014). These examples can be constructed to be surprisingly robust, invariant to viewpoint, orientation and scale (Athalye & Sutskever, 2017) and cause potential security risks to deploying deep neural network based models in a variety of settings. Despite some theoretical work and proposed defense strategies (Cisse et al., 2017; Madry et al., 2017; Papernot et al., 2016) the cause of this vulnerability is still poorly understood.

There have been several hypotheses proposed regarding the cause of adversarial examples. We briefly survey some of them here. One common hypothesis is that neural network classifiers are too linear in various regions of the input space, (Goodfellow et al., 2014; Luo et al., 2015). Another hypothesis is that adversarial examples are off the data manifold (Goodfellow et al., 2016). Fawzi et al. (2015) define a mathematical notion of flexibility of a hypothesis class and argue that the vulnerability of models to adversarial perturbations are due to the low flexibility of classifiers compared to the difficulty of the classification task. Cisse et al. (2017) argue that large singular values of internal weight matrices may cause the classifier to be vulnerable to small perturbations of the input.

Alongside works endeavoring to explain adversarial examples, others have proposed defenses in order to increase robustness. Some works increase robustness to small perturbations by changing the non-linearities used (Krotov & Hopfield, 2017), distilling a large network into a small network (Papernot et al., 2016), or using regularization (Cisse et al., 2017). Other works explore detecting adversarial examples using a second statistical model (Feinman et al., 2017; Abbasi & Gagné, 2017; Grosse et al., 2017; Metzen et al., 2017). However, many of these methods have been shown to fail against a strong enough attacker (Carlini & Wagner, 2017a;b). Finally, adversarial training has been shown in many instances to increase robustness (Madry et al., 2017; Kurakin et al., 2016; Szegedy et al., 2014; Goodfellow et al., 2014). Despite the progress on increasing robustness of classifiers, these defenses still do not scale well.

To better understand this phenomenon and test existing hypothesis, we define a simple synthetic task. We study the task of classifying two concentric high dimensional spheres and the worst case behavior of neural networks on this task. This allows us to study adversarial examples in a setting where the data manifold is well defined mathematically, where we have an analytic characterization of the decision boundary learned by the model, and when the hypothesis class can in theory achieve perfect accuracy. Our experiments and theoretical analysis on this dataset demonstrate the following:

- When the input is high dimensional, the probability of observing a model error under the data distribution may be so low that model errors may only be found *adversarially*. In

particular, a large random sample from the data distribution will with high probability contain no errors.

- Models may become nearly perfect even if they ignore most relevant dimensions in high dimensional space.
- For this dataset, the probability of an adversarial example under the data distribution is identical to that of random "clean" data.
- For this dataset, as the dimension of the spheres tends to infinity, the distance to the nearest error will tend to zero *for any model which has a fixed non-zero error rate*.

In Section 6 we provide a detailed discussion about the connection between adversarial examples for the sphere and those for image models. Our results highlight the fact that the epsilon norm ball adversarial examples often studied in defence papers are not the real problem but are rather a tractable research problem. We demonstrate this further by showing how a recent SOTA defense technique (Madry et al., 2017) is still vulnerable to adversarial perturbations which are different than those used in training.

2 THE CONCENTRIC SPHERES DATASET

The data distribution is mathematically described as two concentric spheres in d dimensions: we generate a random $x \in \mathbb{R}^d$ where $||x||_2$ is either 1.0 or R, with equal probability assigned to each norm (for this work we choose $R = 1.3$). We associate with each x a target y such that $y = 0$ if $||x||_2 = 1$ and $y = 1$ if $||x||_2 = R$.

Studying a synthetic high dimensional dataset has many advantages:

- The probability density of the data $p(x)$ is well defined and is uniform over all x in the support. We can also sample uniformly from $p(x)$ by sampling $z \sim N(\vec{0}, I)$ and then setting $x = z/||z||_2$ or $x = Rz/||z||_2$.
- There is a theoretical max margin boundary which perfectly separates the two classes (the sphere with radius $(R + 1)/2$).
- We can design machine learning models which provably can learn a decision boundary which perfectly separate the two spheres.
- We can control the difficulty of the problem by varying d, and R.

It should be noted that some of the dataset choices were somewhat arbitrary. Our choice of $R = 1.3$ was a bit arbitrary and we did not explore in detail the relationship between adversarial examples and the distance between to two spheres. Additionally, our choice to restrict the data distribution to be the shells of the two spheres was made to simplify the problem further.

In our experiments we investigate training on this dataset in two regimes: the online setting where each minibatch is a uniform sample from $p(x)$, and where there is a fixed training set of size N and the network is trained for many epochs on this finite sample.

3 ADVERSARIAL EXAMPLES FOR A DEEP RELU NETWORK

Our first experiment used an input dimensionality of $d = 500$. We then train a 2 hidden layer ReLU network with 1000 hidden units on this dataset. We applied batch normalization (Ioffe & Szegedy, 2015) to the two hidden layers, but not to the readout layer. We train with minibatch SGD, minimizing the sigmoid cross entropy loss. We use Adam optimizer (Kingma & Ba, 2014) for 1 million training steps with mini batch size 50 and learning rate 0.0001. Because this is training in the online setting with batch size 50 and 1 million training points, the model saw 50 million data points during training.

We evaluated the final model on 10 million uniform samples from each sphere - 20 million points in total and observed no errors on these finite samples. Thus the error rate of this model is unknown, we only have a statistical upper bound on the error rate. Despite this, we are able to adversarially find errors *on the data manifold* by performing a manifold attack (see Section 3.1). There are two types of adversarial examples we generate using this attack, the first are *worst-case examples*, where we

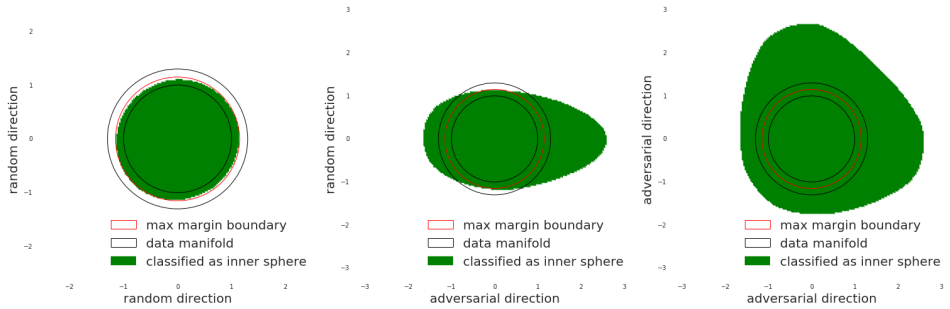

Figure 1: Visualizing a 2d slice of the input space where the subspace is spanned by: 2 randomly chosen directions **(left)**, 1 random and 1 "adversarial direction" **(center)**, and 2 orthogonal "adversarial directions" **(right)**. The data manifold is indicated in black and the max margin boundary in red. The green area indicates points which are classified by the ReLU network as being on the inner sphere. In the last plot, the projection of the entire outer sphere is misclassified despite the fact that the error rate of the model is less than 1 in 10 million.

iterate the attack until the attack objective converges and do not restrict to a local region around the starting point. The second type are *nearest neighbor examples*, where we terminate the attack on the first misclassification found.

In Figure 1 we visualize the decision boundary by taking different 2d projections of the 500 dimensional space. When we take a random projection, the model has closely approximated the max margin boundary on this projection. Note the model naturally interpolates between the two spheres despite only being trained on samples from the surfaces of the spheres. By contrast, when we take a 2d projection where one basis vector is a worst-case adversarial example, the model's decision boundary is highly warped along this "adversarial direction". There are points of norm > 2 for which the model is confident is on the inner sphere. We can also take a slice where the x and y axis are an orthogonal basis for the subspace spanned to two different worst-case examples. Although the last plot shows that the entire projection of the outer sphere is misclassified, the volume of this adversarial region is exceedingly small due to the high dimensional space.

Despite being extremely rare, these misclassifications appear close to randomly sampled points on the sphere. The mean L2 distance to the nearest error on the data manifold is 0.18, by comparison two randomly sampled points on the inner sphere are typically around $\sqrt{2} \approx 1.41$ distance from each other. If we look for the nearest point in between the two spheres which is classified as being on the outer sphere, then we get an average L2 distance of 0.0796, and an average norm of 1.07. Thus the nearest example of the other class is typically about half the distance to the theoretical margin.

This phenomenon appears only when the spheres are high dimensional. In Figure 2 (right), we visualize the same model trained on 100 samples in the case where $d = 2$. The model makes no errors on the data manifold. In our experiments the highest dimension we were able to train the ReLU net without adversarial examples seems to be around $d = 60$. We did not investigate if larger networks will work for larger d.

.

3.1 Finding Adversarial Examples with a Manifold Attack

We wanted to test if adversarial examples were off the data manifold. To that end we designed an attack which specifically produces adversarial examples on the data manifold which we call a *manifold attack*. Traditional attack methods for image models start with an input x and target class $\hat{y}$ and finds an input $\hat{x}$ that maximizes $P(\hat{y} \mid \hat{x})$ subject to the constraint $||x - \hat{x}|| < \epsilon$, where $|| \cdot ||$ is often chosen to be the L_∞ norm.

The manifold attack maximizes $P(\hat{y} \mid \hat{x})$ subject to the constraint $||\hat{x}||_2 = ||x||_2$. This ensures that the produced adversarial example is of the same class as the starting point and lies in the support of

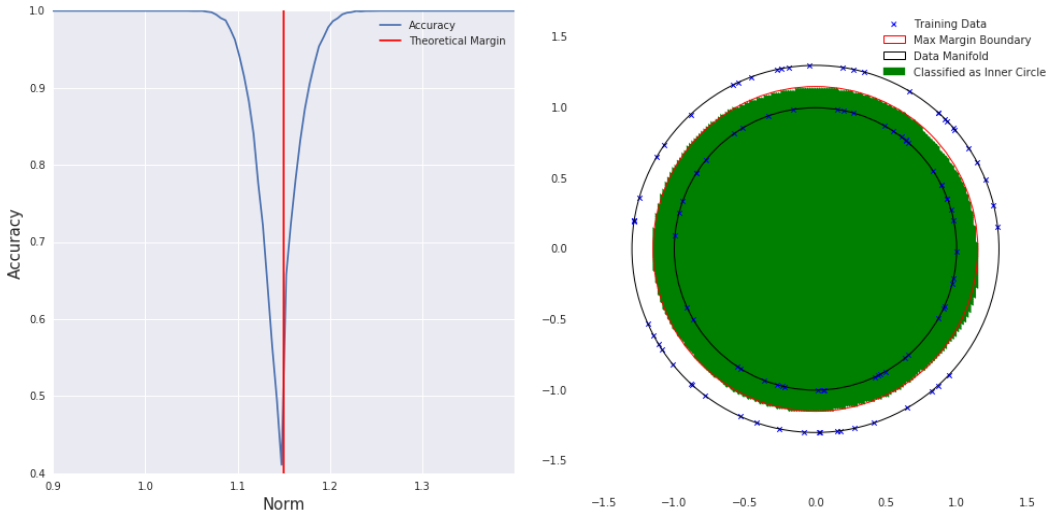

Figure 2: **Left:** We consider the ReLU net trained on 50 million samples from two 500 dimensional spheres of radius 1.0 and 1.3. We evaluate the accuracy of this network on the entire space using a theoretical decision boundary of 1.15. For each norm considered we plot the accuracy among 10000 random samples. We see the accuracy rapidly increases as we move away from the margin. As we move far enough away we no longer observe errors on the random samples. However, we are able to adversarially find errors as far as norms .6 and 2.4. **Right:** We trained the same ReLU net on 100 samples from the data distribution when $d = 2$. By visualizing predictions on a dense subset of the entire space it appears that the model makes no errors on either circle.

the data distribution. We solve this constraint problem using projected gradient descent (PGD), only for the projection step we project back on the sphere by normalizing $||\hat{x}||_2$.

Traditional attacks which create a small adversarial perturbation may be thought of as a special case of a manifold attack in that they exploit the property that nearby images to the starting image should be of the same class[1].

4 Analytic Forms for a Simpler Network

It is difficult to reason about the learned decision boundary of the ReLU network. To obtain a more complete understanding of the decision boundary, we next study a simpler model. The network, dubbed "the quadratic network", is a single hidden layer network where the pointwise non-linearity is a quadratic function, $\sigma(x) = x^2$. There is no bias in the hidden layer, and the output simply sums the hidden activations, multiplies by a scalar and adds a bias. With hidden dimension h the network has $d \times h + 2$ learn-able parameters. The logit is written as

$$\hat{y}(x) = w\vec{1}^T(W_1 x)^2 + b \tag{1}$$

where $W_1 \in \mathbb{R}^{hxd}$, $\vec{1}$ is a column vector of h 1's. Finally, w and b are learned scalars. In the Appendix, we show that the output of this network can be rewritten in the form

$$\hat{y}(x) = \sum_{i=1}^{d} \alpha_i z_i^2 - 1 \tag{2}$$

where α_i are scalars which depend on the model's parameters and the vector $\vec{z}$ is a rotation of the input vector $\vec{x}$. The decision boundary of the quadratic network is all inputs where $\sum_{i=1}^{d} \alpha_i z_i^2 = 1$. It is an ellipsoid in d dimensions centered at the origin. This allows us to analytically determine when

[1]See Section 6 for a discussion regarding whether or not adversarial examples for image models are off the data manifold.

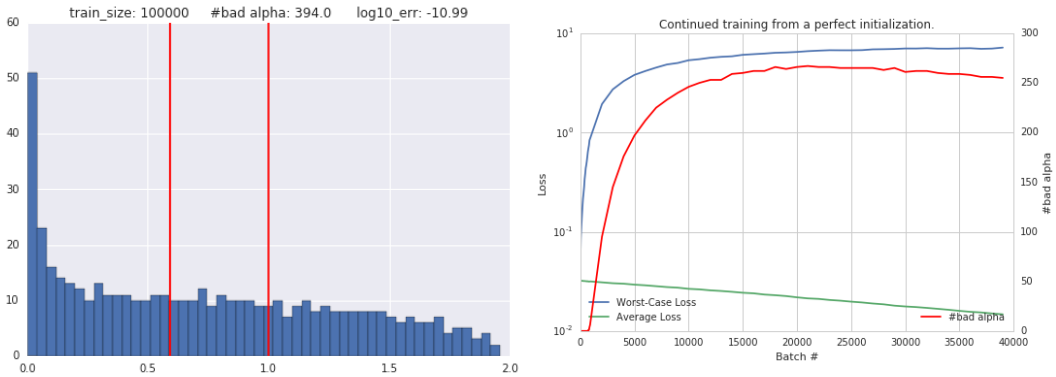

Figure 3: **Left:** The final distribution of α_i when the quadratic network is trained on 100k examples. Red lines indicate the range of needed for perfect classification. Despite a theoretical error rate of 1e-11 most of the α are incorrect. **Right:** Training curves of the quadratic network initialized at a perfect initialization with no classification errors. As training progresses average case loss gets minimized at the cost of a dramatically worse worst case loss. The number of incorrect α_i increases at a similar rate.

the model has adversarial examples. In particular, if there is any $\alpha_i > 1$, then there are errors on the inner sphere. If there are any $\alpha_i < 1/R^2$ then there are errors on the outer sphere. Therefore, the model has perfect accuracy if and only if all $\alpha_i \in [1/R^2, 1]$.

When we train the quadratic network with $h = 1000$ using the same setup as in Section 3 we arrive at the perfect solution: all of the $\alpha_i \in [1/R^2, 1]$ and there are no adversarial examples. This again was in the online learning setup where each minibatch was an iid sample from the data distribution. The story is different, however, if we train on a finite sample from the data distribution. In particular if we sample $N = 10^6$ data points from $p(x)$ as a fixed finite training set and train using the same setup we arrive at a model which empirically has a very low error rate - randomly sampling 10 million datapoints from each sphere results in no errors, but for which there are adversarial examples. In fact, 394 out of 500 of the learned α_i are incorrect in that $\alpha_i \notin [1/R^2, 1]$ (for a complete histogram see Figure 3).

We can use the Central Limit Theorem (CLT) to estimate the error rate of the quadratic network from the α_i (Section 4.1). The estimated error rate of this particular model to be $\approx 10^{-11}$. Note, we are applying the CLT at the tails of the distribution, so it is unclear how accurate this estimate is. However, we found the CLT closely approximates the error rate in the regime where it is large enough to estimate numerically.

Next we augmented the above setup with a "perfect" initialization; we initialize the quadratic network at a point for which all of the α_i are "correct" but there are non-zero gradients due to the sigmoid cross-entropy loss. The network is initialized at a point where the sigmoid probability of $y = 1$ for the inner sphere and outer spheres is $.0016$ and 0.9994 respectively. As shown in Figure 3 continued training from this initialization results in a rapid divergence of the worst and average case loss. Although the average loss on the test set decreases with further training, the worst case rapidly increases and adversarial examples can once again be found after 1000 training steps. This behavior results from the fact that the training objective (average sigmoid cross entropy loss) does not directly track the accuracy of the models. It also demonstrates how the worst and average case losses may diverge when the input is high dimensional.

4.1 Analytic Estimates of the Error Rate

We can use the CLT[2] to analytically estimate the accuracy for the quadratic network in terms of the α_i. The following proposition estimates the error rate on the inner sphere:

[2]We explored using concentration inequalities to get a true upper bound on the error rate, but found the resulting bounds to be too weak to be of interest. We found the CLT estimate to accurately track the empirical accuracy in the settings where the error rate was large enough to be measured 10 million random samples.

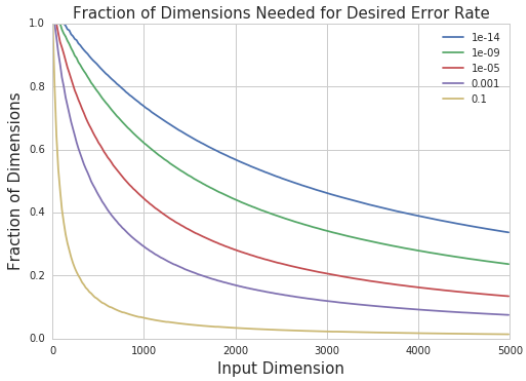

Figure 4: We consider a classification model which only sees a projection of the input, size d, onto a k dimensional subspace. We then plot what k/d needs to be in order for the model to obtain a certain error rate. We find that as the input dimension grows, the ratio k/d needed quickly decreases.

Proposition 4.1 *Consider the decision boundary of the quadratic network of the form*

$$\sum_{i=1}^{d} \alpha_i z_i^2 = 1$$

.

Let $Z = N(0,1)$. We let $\vec{z} \sim S_0$ to denote that the vector $\vec{z}$ is uniformly distributed on the inner sphere. Finally let $\mu = \sum_{i=1}^{d}(\alpha_i - 1)$ and $\sigma = 2\sum_{i=1}^{d}(\alpha_i - 1)^2$. Then the error rate on the inner sphere can be estimated as

$$\mathbb{P}_{\vec{z}\sim S_0}\left[\sum_{i=1}^{d} \alpha_i z_i^2 > 1\right] \approx \mathbb{P}_{Z\sim\mathcal{N}(0,1)}\left[Z > -\frac{\mu}{\sigma}\right] = 1 - \Phi\left(\frac{\mu}{\sigma}\right)$$

Proposition 4.1 implies that there are many settings of α_i which obtain very low error rates. As long as $\mathbb{E}[\alpha_i] \approx (1 + R^{-2})/2)$ and their variance is not too high, the model will be extremely accurate. The histogram in Figure 3 illustrates this; i.e. the learned model has an error rate of 10^{-11} but 80% of the α_i are incorrect. For a typical sample, the model sums incorrect numbers together and obtains the correct answer. Flexibility in choosing α_i while maintaining good accuracy increases dramatically with the input dimension.

To illustrate this further consider a special case of the quadratic network where the decision boundary is of the form

$$\sum_{i=1}^{k} x_i^2 = b.$$

This simplified model has two parameters, k the number of dimensions the model looks at and b a threshold separating the two classes. How large does k need to be in order for the model to obtain a desired error rate? (Assuming b is chosen optimally based on k). We answer this question using the CLT approximation in Proposition 4.1. In Figure 4 we plot the fraction of input dimensions needed to obtain a desired accuracy using this simplified model. For example, if $d = 3000$ then the model can obtain an estimated accuracy of 10^{-14} while only looking at 50% of the input.

5 Small Fooling Perturbations

One counter-intuitive property of adversarial examples is it that nearly every natural image which is correctly classified can be turned into an adversarial image by a small perturbation. This happens for SOTA models which correctly classify most natural images.

Why then does it often appear that most natural images are correctly classified, yet is extremely close to a misclassified image?

We call this the "very unlikely errors appear to be everywhere property" of adversarial examples. We hypothesize that this behavior may be a natural result of the high dimensional geometry of the data manifold. In this work, we do not attempt to compare the geometry of the natural image manifold to that of the sphere, but we can explain why this property occurs on the sphere dataset. Let S_0 be in the sphere of radius 1 in d dimensions and fix $E \subseteq S_0$ (we interpret E to be the set of points on the inner sphere which are misclassified by some model). For $x \in S_0$ let $d(x, E)$ denote the L2 distance between x and the nearest point in the set E. Finally let $d(E) = \mathbb{E}_{x \sim S_0} d(x, E)$ denote the average distance from a uniformly sampled point on the sphere to the set E. We make the following conjecture:

Conjecture 5.1 *For any $E \subset S_0$ of constant measure, $d(E) = O(1/\sqrt{d})$.*

This conjecture directly links the probability of an error on the test set to the average distance to the nearest error *independently of the model*. Any model which misclassifies a small constant fraction of the sphere will appear to make errors close to most data points, no matter how the model errors are distributed on the sphere. In the Appendix we prove Conjecture 5.1 for a special case where all of the errors on the inner sphere are concentrated in a local region near a pole.

Not all high dimensional manifolds will satisfy Conjecture 5.1. For example, if the data manifold is the hypercube $H = [0, 1]^d$ and $E = \{x \in H : x_1 > .9\}$. Then $|E| = 1$ and $d(E) = .405$ is a constant independent of the input dimension. Whether or not a similar conjecture holds for image manifolds is unclear and should be investigated in future work.

6 Connection to Image Models and Why They are Vulnerable to Adversarial Examples

We studied this dataset in order to obtain a better understanding of why adversarial examples occur for image models. What then are the similarities and differences with image models? There are of course many differences, and we do not claim that we have explained all of the properties of adversarial examples that appear for image models with the sphere dataset. One particular aspect of adversarial examples for image models is that the perturbations are often imperceptible to a human, and it is not clear whether or not the adversarial examples for the sphere would satisfy that property. It also appears that transferability of adversarial examples is a bit more limited for the sphere dataset (see Section D in the Appendix).

What is common between the two settings, is we trained a statistical model on a high dimensional dataset and the learned model has very low probability errors which we never see naturally but instead only see when we look for them with gradient descent. The fact that we are using gradient descent to find an error has completely removed the meaning of probability from the picture and the notion of typical example no longer applies.

In the sphere setting, $p(x) = p(x_{adv})$, and there is nothing special about x_{adv} in terms of the underlying data distribution. Thus the only difference between a test error and an adversarial example is the method by which they are found. It is perhaps not true that $p(x) = p(x_{adv})$ in the image setting. However, if the image model was trained with noise added to the input, then any adversarial perturbation which is in the support of the noise distribution used in training will satisfy $p(x_{adv}) > 0$. In other words, the adversarial example will be on the data manifold used in training.

In the introduction we discussed previous hypotheses on why models are vulnerable to adversarial examples. Our explanation of this vulnerability is quite simply that the generalization error of most ML models is non-zero, and attackers can reliably find generalization error adversarially. If the question is then "why is generalization error non-zero?" then there are many potential reasons: not enough data, noisy data, issues with training the model, the model is over-fitting/under-fitting, the model class has low inductive bias or that we are evaluating the model out of distribution. In this sense many of the previous hypothesis in the literature are valid reasons as to why models may make mistakes. If the question is "why are most data points close to an error?", then the answer for the sphere dataset is that this a fundamental property of the geometry of the data manifold itself - if Conjecture 5.1 is true then *any* model which misclassifies a constant fraction of the sphere will appear to have errors close to most points. The answer to the last question in the context of image models is still unclear.

It may be possible to construct an image model which still makes errors on the test set, but for which most correctly classified images are "far" from an incorrectly classified image. Indeed recent work has made some progress towards this by applying adversarial training on both the MNIST and CIFAR10 datasets (Madry et al., 2017). However, in Section A in the Appendix we show how this recent progress, although produced an MNIST model which appears robust with $\epsilon < 0.3$, is still vulnerable to larger adversarial perturbations to the background. This attack is motivated as an extension of the manifold attack on the sphere - by perturbing only the background the attacker can, in a sense, walk farther along the manifold of digits which are recognizable to a human. This manifold is different than the data manifold that MNIST was sampled from, and thus these adversarial errors are out of the distribution the model was trained on. Because these *particular* adversarial examples are off the data manifold, this attack (and most other attack methods) could easily be detected statistically by say looking at a histogram of pixel activations. However, once the defender has fixed some way to detect for adversarial examples, the combination of the detection method and the underlying image model *is in itself another statistical model and thus is likely to have non-zero generalization error which an attacker can find adversarially given white box access*. This sort of back and forth between the attacker and defender has already occurred in the literature when Carlini & Wagner (2017b) broke many existing detection methods.

It is important to note that given enough data the quadratic network does eventually become perfect. Perhaps a properly trained image model on a very large image dataset will naturally become robust to adversarial perturbations. An important lesson from the sphere dataset is that the architecture size required to achieve low generalization error may be much smaller than the size required to achieve 0 generalization error. Indeed the quadratic network with 1000 hidden nodes trained on the sphere dataset with $d = 2000$ obtains an estimated generalization error of 10^{-21} and 10^{-9} on the inner and outer spheres respectively despite being only 1/2 the size required to achieve 0 error. For image datasets, Madry et al. (2017) also reported requiring larger models in order to obtain robustness to small adversarial perturbations.

Most of this discussion assumes the attacker has white box access to the model. Defending against black box attacks may be an easier task where the defender needs to "hide" the generalization error of their model from the attacker, or at least detect an attack in some way. Of relevance to the black box setting is that it is quite common that independently trained models on the same data set have correlated error sets. Indeed ensembling many independently trained models together often results in diminishing reduction to test error as the number of models in the ensemble increases. Perhaps this is related to the fact that adversarial examples often transfer between models, and why the strongest black box attacks use an ensemble of networks (Tramèr et al., 2017).

7 Conclusions

In this work we studied the worst case behavior of different neural networks trained to classify between two high dimensional concentric spheres. For this dataset, we observed that when the dimension is large, the generalization error of these networks is sometimes so low that it can only be found adversarially. We showed that these adversarial examples exist even when they are generated on the data manifold, when the model class can provably achieve 0 generalization error, and even when we used a "perfect" initialization of the model. We then showed under some assumptions that for a fixed error rate, the average distance to nearest error may decrease on the order of $O(1/\sqrt{d})$ as the input dimension grows large.

We concluded by discussing similarities and differences for adversarial examples on the sphere dataset and image datasets. We also argued that the reason most ML models have adversarial examples is simply because the generalization error is non-zero, and that there are many potential reasons why generalization error may be non-zero.

Our results highlight the fact that epsilon norm ball adversarial examples are not the real problem but rather a tractable research problem. We have shown that recent progress on increasing robustness to small perturbations on MNIST is still vulnerable to large perturbations in the background.

REFERENCES

Mahdieh Abbasi and Christian Gagné. Robustness to adversarial examples through an ensemble of specialists. *arXiv preprint arXiv:1702.06856*, 2017.

Anish Athalye and Ilya Sutskever. Synthesizing robust adversarial examples. *CoRR*, abs/1707.07397, 2017. URL http://arxiv.org/abs/1707.07397.

Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. *arXiv preprint arXiv:1705.07263*, 2017a.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pp. 39–57. IEEE, 2017b.

Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 854–863, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL http://proceedings.mlr.press/v70/cisse17a.html.

Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Analysis of classifiers' robustness to adversarial perturbations. *arXiv preprint arXiv:1502.02590*, 2015.

Reuben Feinman, Ryan R Curtin, Saurabh Shintre, and Andrew B Gardner. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*, 2017.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

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

Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick McDaniel. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280*, 2017.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In David Blei and Francis Bach (eds.), *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pp. 448–456. JMLR Workshop and Conference Proceedings, 2015. URL http://jmlr.org/proceedings/papers/v37/ioffe15.pdf.

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

Dmitry Krotov and John J Hopfield. Dense associative memory is robust to adversarial inputs. *arXiv preprint arXiv:1701.00939*, 2017.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

Yan Luo, Xavier Boix, Gemma Roig, Tomaso Poggio, and Qi Zhao. Foveation-based mechanisms alleviate adversarial examples. *arXiv preprint arXiv:1511.06292*, 2015.

Aleksander Madry, Aleksander Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial examples. *arXiv preprint arXiv:1706.06083*, 2017.

Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017.

Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *Security and Privacy (SP), 2016 IEEE Symposium on*, pp. 582–597. IEEE, 2016.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. URL http://arxiv.org/abs/1312.6199.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

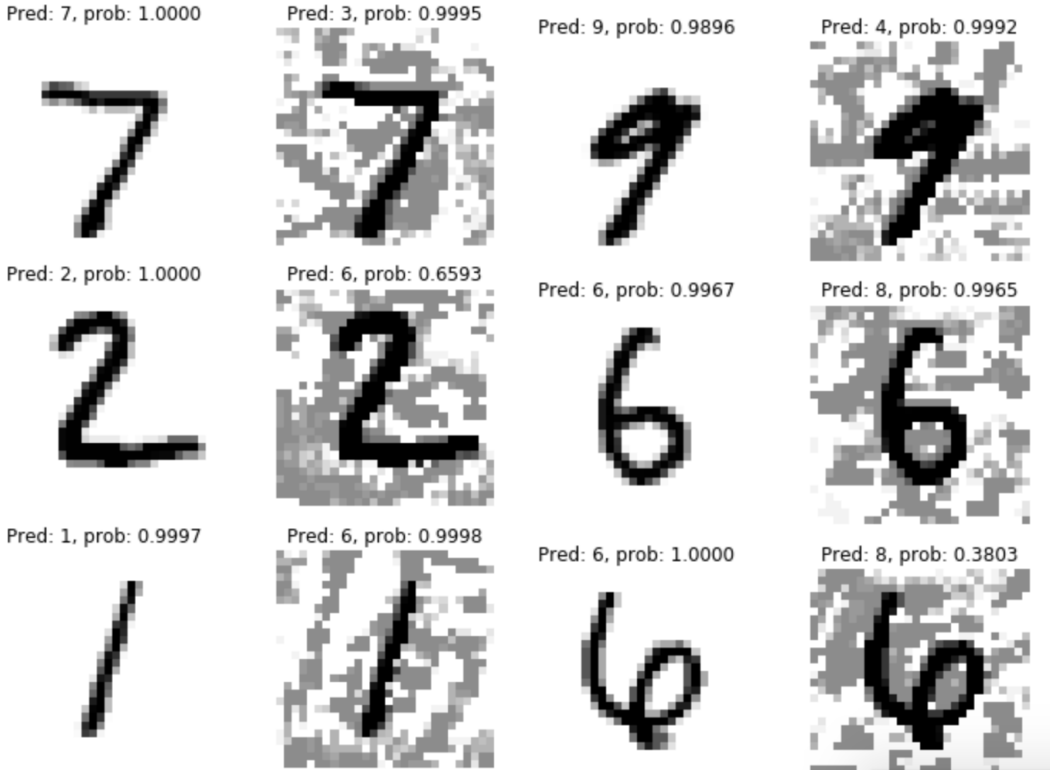

Figure A.1: Adversarial examples for the adversarially trained model described in Madry et al. (2017). The attack has a success rate of 93% and works by perturbing the background of the image only by a larger ϵ than was adversarially trained for.

Appendix

A Adversarial Examples for the Madry et al. (2017) MNIST Adversarial Challenge

Recently, Madry et al. (2017) showed how to use adversarial training to produce a model which is robust on MNIST when $||x - x_{adv}||_\infty < \epsilon$ for $\epsilon = 0.3$. They then uploaded the model to github and issued a challenge for anyone to generate adversarial examples for their model [3]. To date, the strongest attack against their model has a success rate of 8%. Our attempts to increase the success rate of this attack for $\epsilon < .3$ were unsuccessful. However, we were still able to generate adversarial examples by perturbing the background with an $\epsilon = .45$. By perturbing the background only we can use larger perturbations while still keeping the human classification of the image the same. In fact, to make the digit further stand out from the background we automatically bolden the digit by first setting any pixel with value $> .7$ to 1.0, and any neighboring pixel also to 1.0. The resulting digit is still of the same class semantically, but the attack has a success rate of 93%. Using a larger epsilon of .6 gives a success rate of 100%, but the resulting digits are a little less clear semantically. We suspect using attacks which are designed to perturb a few image pixels by a large amount, such as those described in Carlini & Wagner (2017b), may be even better suited here. In Figure A.1 we show the results of successful attacks with a background $\epsilon = .45$.

[3] `https://github.com/MadryLab/mnist_challenge`

B THE DECISION BOUNDARY OF THE QUADRATIC NETWORK

Here we show the decision boundary of the quadratic network is a d-dimensional ellipsoid. We'll use $\boldsymbol{x}$ to denote the column vector representation of the input. The logit of the quadratic network is of the form

$$l(x) = (W_1\boldsymbol{x})^T(W_1\boldsymbol{x})w + b \tag{3}$$

W_1 is the input to hidden matrix, and w and b are scalars. We can greatly simplify this by taking SVD of $W_1 = U\Sigma V^T$. So we have the following:

$$\begin{aligned} l(x) &= (W_1\boldsymbol{x})^T(W_1\boldsymbol{x})w + b \\ &= (U\Sigma V^T\boldsymbol{x})^T(U\Sigma V^T\boldsymbol{x})w + b \\ &= (\boldsymbol{x}^T V\Sigma U^T U\Sigma V^T\boldsymbol{x})w + b \end{aligned}$$

Let $\boldsymbol{z} = V^T\boldsymbol{x}$ which is a rotation of the input. Then the above becomes

$$l(x) = \boldsymbol{z}^T\Sigma^2\boldsymbol{z}w + b \tag{4}$$

Letting the singular values of W_1 be the sequence (s_i) we have

$$l(x) = w\sum_{i=1}^{d} s_i^2 z_i^2 + b \tag{5}$$

The decision boundary is of the form $l(x) = 0$, thus we have

$$\begin{aligned} w\sum_{i=1}^{d} s_i^2 z_i^2 + b &= 0 \\ \sum_{i=1}^{d} \alpha_i z_i^2 - 1 &= 0 \end{aligned}$$

where $\alpha_i = w * s_i^2/(-b)$.

Note the distribution of $\boldsymbol{z}$ is the same as $\boldsymbol{x}$ (they are rotations of each other) so replacing z_i with x_i above yields a rotation of the decision boundary.

C ESTIMATING THE ACCURACY WITH THE CENTRAL LIMIT THEOREM

Here we prove thm. 4.1. As in the statement let $\{\alpha_i\}_{i=1}^d$ be nonnegative real numbers and $b > 0$. Let S_0 be the unit sphere in d-dimensions. Suppose z is chosen uniformly on S_0 then we wish to compute the probability that

$$\sum_{i=1}^{d} \alpha_i z_i^2 > 1. \tag{6}$$

One way to generate z uniformly on S_0 is to pick $u_i \sim N(0,1)$ for $1 \leq i \leq d$ and let $z_i = u_i/||u||$. It follows that we may rewrite eq. (6) as,

$$\begin{aligned} \frac{1}{||u||^2}\sum_{i=1}^{d} \alpha_i u_i^2 &> 1 \\ \sum_{i=1}^{d} \alpha_i u_i^2 &> \sum_{i=1}^{d} u_i^2 \\ \sum_{i=1}^{d} (\alpha_i - 1)u_i^2 &> 0. \end{aligned} \tag{7}$$

Thus we have converted the condition in terms of points drawn uniformly on S to a condition on points that are i.i.d. Gaussian distributed.

Let $X = \sum_{i=1}^{d}(\alpha_i - 1)u_i^2$. In the case that d is sufficiently large we may use the central limit theorem conclude that $X \sim N(\mu, \sigma^2)$. In this regime X will be determined exclusively by its first two moments. We can work out each of these separately,

$$\mu = \mathbb{E}[X] = \sum_{i=1}^{d}(\alpha_i - 1) \tag{8}$$

$$\sigma^2 = \text{Var}[X] = 2\sum_{i=1}^{d}(\alpha_i - 1)^2. \tag{9}$$

It follows that,

$$P(X > 0) = P\left(\sigma Z + \mu > 0\right) = P\left(Z > -\frac{\mu}{\sigma}\right) = 1 - \Phi\left(-\frac{\mu}{\sigma}\right) \tag{10}$$

which proves the result.

D TRANSFERABILITY

Adversarial examples have been shown to often generalize accross independently trained image models and different architectures. This allows for "black box attacks" where the attacker does not have access to the weights of the model they are attacking, but instead trains a private model, generates adversarial examples the private model and uses these to attack the black box model. Performing a successful black box attack requires finding a common error between two models given access to only one of the models. This dataset has some unique properties that affect transferability of adversarial examples. In particular, the data distribution $p(x)$ is rotationally symmetric, as a result when we train two different models on two different finite samples from the data distribution, we find that their error sets are uncorrelated. In particular, generating an adversarial example for model 1 is very unlikely to transfer to model 2. We are still able to find common errors between the two models by performing a joint white box attack against the two models. This story changes a bit once we start training models on the same finite sample from $p(x)$. To test transferability we trained different models on the same 100,000 samples from the sphere dataset. We then generated 100 "worst case" examples for the attacking model and measured the probability that they transfer to the defending model. In this experiment none of the adversarial examples transferred between two independently trained ReLU networks. But they all transferred between two independently trained quadratic network.

E PROOF OF A SPECIAL CASE OF CONJECTURE 5.1

In this section we sketch a proof a special case of Conjecture 5.1, when the set E is concentrated near a pole. Let $x \sim S_0$ be a uniform sample on the inner sphere. The proof follows from the fact that as d becomes large, the distribution of a single coordinate x_i on the sphere approaches $N(0, \frac{1}{d})$. Thus if $E_c = \{x \in S_0 : x_1 > c/\sqrt{d}\}$, then

$$\mathbf{P}[x \in E_c] \approx \mathbf{P}[N(0, \frac{1}{d}) > c/\sqrt{d}] = \mathbf{P}[N(0,1) > c].$$

Thus for constant c the measure of E_c will, in the limit, be constant. Note the distance from E_c to the equator is $\sqrt{2}c/\sqrt{d}$ which tends to 0 as d becomes large.

For $x \in S_0$ let $d(x, E_c)$ denote the distance from x to the set E_c. This distance is equal to $max(\sqrt{2}(c/\sqrt{d} - x_1), 0)$. Thus we have

$$d(E_c) = \mathbf{E}_{x\sim S_0} d(x, E_c) \approx \mathbf{E}\left[max\left(\sqrt{2}(c/\sqrt{d} - N(0, \frac{1}{d})), 0\right)\right] = O(1/\sqrt{d}).$$

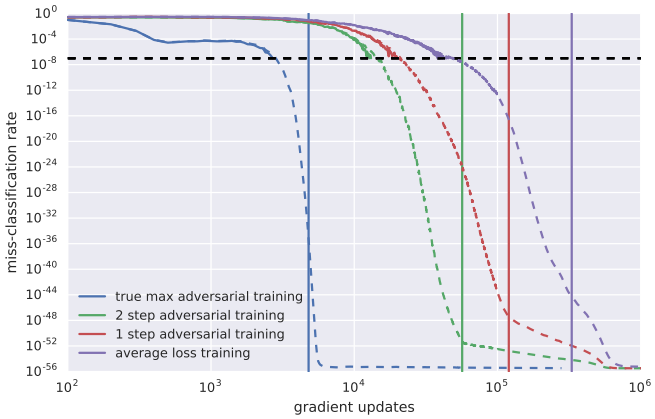

Figure F.2: Training curves training with adversarial training. We use 1, or 2 steps of SGD to approximate the inner max of the min max objective as well as the true max computed analytically for our simple model. We also show a training curve without adversarial training as comparison. The vertical lines indicates the first instance of the model containing no adversarial examples, all α_i are in the correct range. The solid lines indicate accuracies computed via brute force monte carlo sampling. The dashed black line indicates the number of samples drawn when numerically evaluating error. The dashed line is the analytic estimate of error rate computed in 4.1. Adversarial training of any form decreases the amount of samples needed to find a perfect model. Using the exact solution for the inner max speeds up training speed by an order of magnitude.

F ADVERSARIAL TRAINING AS DEFENSE

In this section we explore adversarial training as an attempt to remove adversarial examples. Adversarial training can be formulated as minimizing the worst case error around a particular input example (Madry et al., 2017). This inner max in the loss is normally approximated via a few steps of SGD. We perform experiments on the the quadratic network, equation 1. Doing this allows enables analytic estimation of miss-classification rate without the need to perform any kind of attack. Additionally, this allows efficient computation of a true max in min max optimization enabling a more accurate adversarial training. In this experiment, we train 4 models, no adversarial training, estimation of the inner max with 1 and 2 steps of SGD, and adversarial training using the true worst case loss. We train minimizing a sigmoid cross entropy loss. Training curves can be found in figure F.2. We find adversarial training does help learn better models. When using worst case adversarial training we get an order of magnitude improvement in training speed over standard training.