Exploring Adversarial Input Spaces for Convolutional Neural Network Defense

Austin Wang austinw2@stanford.edu
Daniel Kunin kunin@stanford.edu
Justin Pyron pyron@stanford.edu

1. Introduction

Despite the remarkable success of deep learning in recent years, the susceptibility of deep networks to being fooled by adversaries remains an ever present problem. Szegedy et al. demonstrated that images can be strategically perturbed so that the resulting images are indistinguishable to humans, but the network confidently misclassifies them [8]. See Figure 1 for a grid of adversarial examples. Adversarial images fall into two categories: targeted and untargeted. In a targeted attack, an original image is perturbed to cause the model to misclassify to a specific target class, while in an untargeted attack, an original image is perturbed with the goal of causing misclassification to *any* class. A clearer understanding of the space and structure of these adversarial inputs would help shed light on ways to defend against adversarial attacks and increase the robustness of current deep networks.

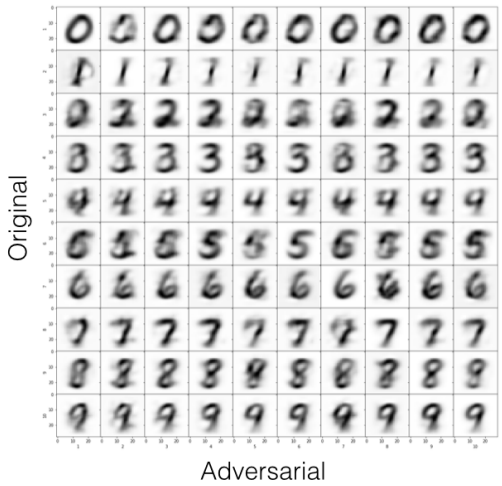

Figure 1. A grid showing all the average adversarial images we generated for MNIST. Image i, j is an image originally labeled as class i, but perturbed such that it would be classified as class j. The diagonal images are original MNIST images.

1.1. Generation of Adversarial Images

There exist several methods for generating adversarial images, each of which utilizes the gradient of the outputs of a network with respect to input pixels. Let F denote a model that maps input image x to a vector of predicted class scores $F(x)$. Letting F_j denote the function that maps input image x to the score for class j, the gradient $\nabla_x F_j(x)$ tells how impactful each input pixel is on the score for class j. Methods that generate adversarial images utilize this information about the responsiveness of class scores with respect to pixels to strategically modify pixel values in order to push outputted class scores in the desired direction.

The Fast Gradient Sign Method (FGSM) is one approach that generates targeted adversarial examples. For target class t, gradients of the score of class t with respect to input pixels are computed. These signs of the gradients are scaled by a small value ϵ and then added to the original image x to produce an adversarial image x' [1]:

$$x' = x + \epsilon \text{ sign}\Big(\nabla_x F_t(x)\Big)$$

The Jacobian Based Saliency Map Approach (JSMA) follows a similar approach by constructing a saliency map, which is the Jacobian of class output scores with respect to input pixels. But rather than update all pixels, JSMA is more selective and only updates the pixels that have the greatest impact on fooling the model. The impact is measured by the pixel's ability to both increase the score to the target class *and* decrease the score to all other classes. For a single pixel $x_{i,j}$, one way to measure impactfulness is a sum of derivatives:

$$\frac{\partial F_t(x)}{\partial x_{i,j}} + \sum_{k \neq t} -\frac{\partial F_k(x)}{\partial x_{i,j}}$$

Only the fraction of pixels with the highest impactfulness are updated.

Note that although a base network is required to generate adversarial images, they are often transferable to distinct models, i.e. images calibrated to fool one specific model often are able to fool different ones [8].

1.2. Traditional Approaches to Adversarial Defense

The vulnerability of deep networks to adversarial attack has spurred attempts to increase robustness of networks. The most popular method to defend against adversarial images so far has involved generating a set of adversarial images, assigning to them the proper label, and retraining the model with these adversarial images in the training set [10]. Such techniques lead to increased robustness, but make no attempt to understand the properties of the network responsible for the vulnerabilities, nor attempt to modify the model to remedy such vulnerabilities.

Another popular approach involves building a classifier that discriminates between natural images and adversarial images, with the ultimate goal of using it to filter out adversarial inputs before feeding them to the original model. Classifiers that successfully discriminate between original images and adversarial images generated from them have been successfully built. However, models augmented with a classifier still remain vulnerable to attack, since new adversarial images can be generated to trick the new model with an attached classifier filter.

1.3. Problem Statement

Our goal is to conduct an exploration of the space of adversarial inputs for a convolutional neural network to help motivate strategies for adversarial defense. To this end, we use the MNIST and CIFAR-10 datasets and generate thousands of adversarial images. We also compute the differences between these adversarial images and the images used to generate them, which we call perturbations.

By looking at statistics of these adversarial images and perturbations such as mean image, standard deviation, histograms of pixel values, and norm size, we hope to uncover some type of discernible structure, especially if we find the transferability of adversarial inputs across networks to be true. Based on the initial findings of Szegedy et al., we expect the perturbations to exhibit patterns resembling structured noise, rather than a visually coherent structure [8].

We attempt to use the analysis of the perturbations and adversarial images to find new approaches to adversarial defense. One idea is to create a training set of adversarial images and learn a network to filter out the perturbations, similar to an autoencoder. A metric for evaluating the robustness of our new model is the proportion of adversarial image labels we were able to fix. We can also generate adversarial images for the new, hopefully more robust model, and see if those images fool the original model in order to determine if we have or have not made improvements.

2. Motivation and Methodology

2.1. An Analogy to Numerical Linear Algebra

In numerical linear algebra we are very concerned with how error propagates through an algorithm. Because numbers cannot be represented to infinite precision, roundoff errors are introduced into any calculation. The goal is to bound this error and design algorithms that minimize its propagation. To this end we generally talk about three forms of error: Forward Error, Backward Error and Sensitivity.

We assume our algorithm can be represented as a surjective function $f : X \to Y$ (a map from the input space X to the output space Y), however $\tilde{f} : X \to Y$ is the actual result of running our algorithm when accounting for roundoff error. The goal of error analysis in numerical linear algebra is to understand the relationship between f and $\tilde{f}$. These three forms of error can formally and pictorially be defined as shown in Figure 2.

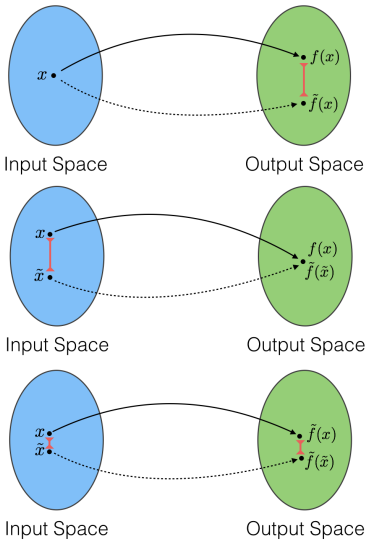

Figure 2. Forward Error (top) is the distance in output space for a given input: $||f(x) - \tilde{f}(x)||$. Backward Error (middle) is the distance in input space such that the outputs are equal: $||x - \tilde{x}||$. Sensitivity (bottom) is the ratio of forward error with backward error. It can be understood as a measure of how much our output, $\tilde{f}(x)$, changes when our input, x, is perturbed. [3]

Linear algebra is the study of linear transformations, however, the error analysis paradigm described above is not unique to linear transformations. In fact this framework lends itself very nicely to deep learning. A deep learning task can be understood as learning a non-linear transformation $\tilde{f}$ by minimizing some loss function evaluated on a set of training samples $(x, f(x))$. This loss function is generally defined as a distance or distortion measure between $f(x)$ and $\tilde{f}(x)$. Notice that this scheme is analogous to minimizing forward error in the numerical linear algebra

setting. Similarly, we can understand adversarial attacks as the sensitivity of our neural network. An adversarial attack is a small perturbation in the input space that propagate to large errors in the output space. Using this framework for understanding a deep learning task, a natural strategy to increasing adversarial robustness would be minimizing forward error and maximizing backward error.

2.2. Technical Approach

We begin our investigation of the robustness (or lack thereof) of deep networks by analyzing the MNIST dataset of handwritten digits. We generate a dataset of adversarial examples following an approach similar to FGSM in section 1.1, but slightly modified. Instead of perturbing images by the sign of the gradient, we update by the gradient. As a result, the perturbations applied to a pixel are not uniform. See Figure 1 for a grid of generated adversarial images.

With our adversarial images generated, we create a Perturbation Stats Python class equipped with functionality to read in and store both adversarial images and the associated perturbations, along with methods to produce summary statistics and plots. These include histograms of the L_1 norm, L_2 norm, and the raw pixel values, tables of the mean, standard deviation, maximum, and minimum for arbitrary adversarial images or perturbations, and visualization capabilities for images with a given original label and adversarial label. Using this class, we are able to quickly analyze key features of adversarial inputs to help us learn more about this space and guide our defense approaches.

3. Preliminary Results

3.1. Initial Statistics

Our first observation from the analysis of our data set of adversarial MNIST images is that while the adversarial images look very similar to their original images, the distribution of their pixel values is actually quite different. As seen in Figure 3, the distribution of pixels for both the original and adversarial images are bimodal, however the distribution of the adversarial images has a much larger spread. This observation has been used in literature to motivate defense approaches that detect adversarial examples through classic statistical methods such as mean discrepancy and kernel density estimation [2].

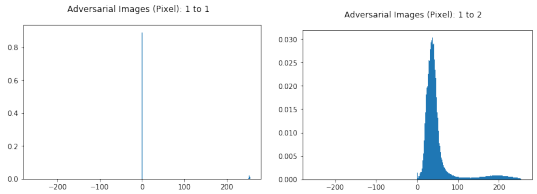

Figure 3. Histogram of pixels from original MNIST images with label 1 (left) and histogram of pixels from adversarial MNIST images with original label 1 and adversarial label 2 (right).

The next thing we notice is that our perturbations appear to be highly correlated to the original and target class. As seen in Figure 4 the perturbations resemble two superimposed images, one subtracting the original image class and the second adding the target class. This is slightly unexpected as most of the adversarial perturbations reported in literature resemble something closer to random noise [5]. We believe this is probably a condition of the simplicity of MNIST and that the method we used to generate the perturbations is targeted to a specific class.

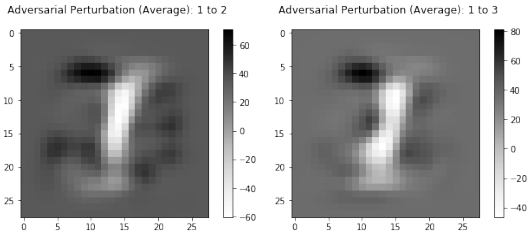

Figure 4. Average perturbation between original MNIST images with label 1 and their adversarial examples with label 2 (left) and label 3 (right).

3.2. Dropout Defense

The first defense strategy we test is motivated by a hypothesis that the process which generates adversarial examples is highly tailored to specific components of a model's architecture. Although the transferability of adversarial examples to distinct models is well-documented, we hypothesize that there exists common regularity in the internal architecture of models trained to classify specific types of data (e.g. CIFAR-10), and that adversarial examples exploit this common regularity. If our hypothesis is correct, then disrupting the ordinary architecture of the network should attenuate the effectiveness of adversarial images.

To embed disruptive behavior into a model, we propose an approach that injects randomness in predictions through dropout. More specifically, we test an approach where a collection of predictions is computed for each input, where each prediction is the result of a forward pass through the network with dropout enabled between layers. The final

output classification is the class which receives a plurality of votes from the collection of predictions for that example. When an adversarial example is fed forward through the network, the components of the architecture that it depends on exploiting to successfully fool the model will be jumbled, decreasing its adversarial strength.

Even if the dropout ensemble method is not strong enough to cause the correct classification of adversarial inputs, it still provides information potentially useful toward detecting an adversarial image. In particular, since we hypothesize that the injection of randomness in the prediction stage disproportionately affects adversarial images, it would follow that the distribution of class predictions for adversarial examples will have much greater variability than that of original images. In preliminary investigations, this phenomenon was confirmed. See Figure 5.

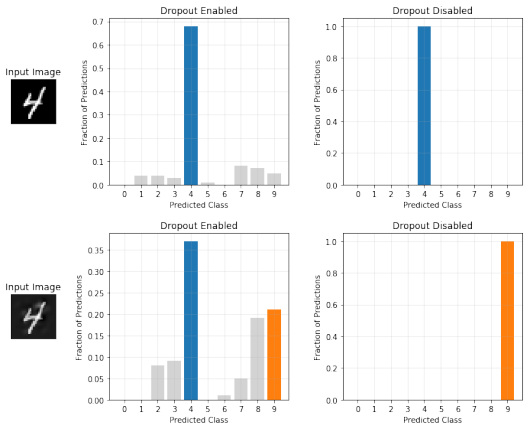

Figure 5. Histogram of predictions for a single original image (top). Histogram of predictions for a single adversarial image (bottom).

References

[1] N. Carlini and D. A. Wagner. Towards evaluating the robustness of neural networks. *CoRR*, abs/1608.04644, 2016.

[2] N. Carlini and D. A. Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. *CoRR*, abs/1705.07263, 2017.

[3] E. Darve and M. Wootters. Numerical linear algebra with julia.

[4] Y. Dong, F. Liao, T. Pang, X. Hu, and J. Zhu. Boosting adversarial examples with momentum. *CoRR*, abs/1710.06081, 2017.

[5] I. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[6] A. Kurakin, I. J. Goodfellow, and S. Bengio. Adversarial machine learning at scale. *CoRR*, abs/1611.01236, 2016.

[7] N. Papernot, P. D. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. The limitations of deep learning in adversarial settings. *CoRR*, abs/1511.07528, 2015.

[8] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013.

[9] F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel. The Space of Transferable Adversarial Examples. *ArXiv e-prints*, Apr. 2017.

[10] X. Yuan, P. He, Q. Zhu, R. R. Bhat, and X. Li. Adversarial examples: Attacks and defenses for deep learning. *CoRR*, abs/1712.07107, 2017.