EXPLAINING AND HARNESSING ADVERSARIAL EXAMPLES

Ian Goodfellow, Jonathon Shlens and Christian Szegedy
ICLR 2015

Table of Contents

Table of Contents

Introduction

Szegedy et al. [1] made an intriguing discovery: several machine learning models, including state-of-the-art neural networks, are vulnerable to adversarial examples.

The cause of these adversarial examples was a mystery, and speculative explanations have suggested it is due to *extreme nonlinearity of deep neural networks*.

They show that these speculative hypotheses are **unnecessary**. Linear behavior in high-dimensional spaces is sufficient to cause adversarial examples.

Introduction

Their explanation suggests a fundamental tension between designing models that are easy to train due to their linearity and designing models that use nonlinear effects to resist adversarial perturbation.

Previous works suggest that classifiers based on modern machine learning techniques, even those that obtain *excellent* performance on the test set, are not learning the true underlying concepts that determine the correct output label.

The Linear Explanation Of Adversarial Examples

In many problems, the precision of an individual input feature is limited. For example, digital images often use only 8 bits per pixel so they discard all information below 1/255 of the dynamic range.

Formally, for problems with well-separated classes, we expect the classifier to assign the same class to $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$ so long as $||\boldsymbol{\eta}||_\infty < \epsilon$, where ϵ is small enough to be discarded.

Consider the dot product between a weight vector w and an adversarial example $\tilde{\boldsymbol{x}}$:

$$w^T \tilde{\boldsymbol{x}} = w^T \boldsymbol{x} + w^T \boldsymbol{\eta}.$$

The Linear Explanation Of Adversarial Examples

The adversarial perturbation causes the activation to grow by $w^T \eta$.

If w has n dimensions and the average magnitude of an element of the weight vector is m, then the activation will grow by ϵmn.

We can make many infinitesimal changes to the input that add up to one large change to the output.

This explanation shows that a simple linear model can have adversarial examples if its input has sufficient dimensionality.

Previous explanations for adversarial examples invoked hypothesized properties of neural networks, such as their supposed highly non-linear nature.

Linear Perturbation of Non-Linear Examples

They hypothesize that neural networks are *too linear* to resist linear adversarial perturbation.

LSTMs, ReLUs, and maxout networks are all intentionally designed to behave in very linear ways, so that they are easier to optimize.

This linear behavior suggests that cheap, analytical perturbations of a linear model should also damage neural networks.

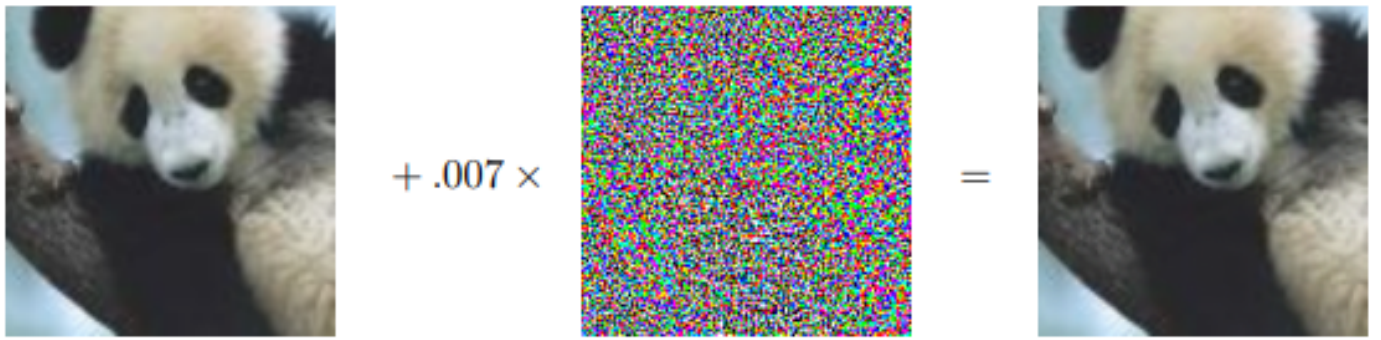
Let $\boldsymbol{\theta}$ be the parameters of a model, $\boldsymbol{x}$ the input to the model, y the targets associated with $\boldsymbol{x}$ and $J(\boldsymbol{\theta}, \boldsymbol{x}, y)$ be the cost used to train the neural network.

Linear Perturbation of Non-Linear Examples

We can linearize the cost function around the current value of $\boldsymbol{\theta}$, obtaining an optimal max-norm constrained pertubation of

$$\boldsymbol{\eta} = \epsilon sign(\nabla_x J(\boldsymbol{\theta}, \boldsymbol{x}, y)).$$

They refer to this as the **fast gradient sign method** of generating adversarial examples.



Linear Perturbation of Non-Linear Examples

They find that using $\epsilon = .25$, we cause a shallow softmax classifier to have an error rate of 99.9% with an average confidence of 79.3% on the MNIST test set.

In the same setting, a maxout network misclassifies 89.4% of our adversarial examples with an average confidence of 97.6%.

The fact that these simple, cheap algorithms are able to generate misclassified examples serves as evidence in favor of our interpretation of adversarial examples as a result of linearity.

Adversarial Training of Deep Neural Networks

The criticism of deep networks as vulnerable to adversarial examples is somewhat misguided, because unlike shallow linear models, deep networks are at least able to represent functions that resist adversarial perturbation.

Obviously, standard supervised training does not specify that the chosen function be resistant to adversarial examples. This must be encoded in the training procedure somehow.

We found that training with an adversarial objective function based on the fast gradient sign method was an effective regularizer:

$$\tilde{J}(\boldsymbol{\theta}, \boldsymbol{x}, y) = \alpha J(\boldsymbol{\theta}, \boldsymbol{x}, y) + (1 - \alpha) J(\boldsymbol{\theta}, \boldsymbol{x} + \epsilon \mathit{sign}(\nabla_x J(\boldsymbol{\theta}, \boldsymbol{x}, y)))$$

Adversarial Training of Deep Neural Networks

In all experiments, they use $\alpha = 0.5$.

Using this approach to train a maxout network that was also regularized with dropout, they were able to reduce the error rate from 0.94% without adversarial training to 0.84% with adversarial training.

They also used early stopping on the adversarial validation set error. Using this criterion to choose the number of epochs to train for, they then retrained on all 60,000 examples.

Five different training runs result in four trials that each had an error rate of 0.77% on the test set and one trial that had an error rate of 0.83%.

The average of 0.782% is the best result reported on the permutation invariant version of MNIST.

Adversarial Training of Deep Neural Networks

The model also became somewhat resistant to adversarial examples.

Recall that without adversarial training, this same kind of model had an error rate of 89.4% on adversarial examples based on the fast gradient sign method. With adversarial training, the error rate fell to 17.9%.

Adversarial examples are transferable between the two models but with the adversarially trained model showing greater robustness.

Adversarial examples generated via the original model yield an error rate of 19.6% on the adversarially trained model, while adversarial examples generated via the new model yield an error rate of 40.9% on the original model.

Adversarial Training of Deep Neural Networks

When the adversarially trained model does misclassify anadversarial example, its predictions are unfortunately still highly confident. The average confidence on a misclassified example was 81.4%.

We could also regularize the model to be insensitive to changes in its features that are smaller than the ϵ precision simply by training on all points within the ϵ max norm box.

However, noise with zero mean and zero covariance is very inefficient at preventing adversarial examples.

Adversarial Training of Deep Neural Networks

As control experiments, they trained training a maxout network with noise based on randomly adding $- + \epsilon$ to each pixel, or adding noise in $U(-\epsilon, \epsilon)$ to each pixel. These obtained an error rate of 86.2% with confidence 97.3% and an error rate of 90.4% with a confidence of 97.8% respectively on fast gradient sign adversarial examples.

Why Do Adversarial Examples Generalize?

An intriguing aspect of adversarial examples is that an example generated for one model is often misclassified by other models, even when they have different architecures or were trained on disjoint training sets.

Moreover, when these different models misclassify an adversarial example, they often agree with each other on its class.

Explanations based on extreme non-linearity and overfitting cannot readily account for this behavior.

To explain why mutiple classifiers assign the same class to adversarial examples, they hypothesize that neural networks trained with current methodologies all resemble the linear classifier learned on the same training set.

Why Do Adversarial Examples Generalize?

To test this hypothesis, they generated adversarial examples on a deep maxout network and classified these examples using a shallow softmax network and a shallow RBF network.

Their hypothesis does not explain all of the maxout network's mistakes or all of the mistakes that generalize across models.

Bibliography I


C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus.
Intriguing properties of neural networks.
arXiv preprint arXiv:1312.6199, 2013.