Fooling neural networks and adversarial examples

Nils Wireklint

April 22, 2015
Fooling neural networks and adversarial examples

Nils Wireklint

April 22, 2015
Introduction
Articles

- Intriguing properties of neural networks  
  *Szgedy et al.*

- Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images  
  *Nguyen et al.*

- Explaining and Harnessing Adversarial Examples  
  *Goodfellow, et al.*
Article 1: Intriguing properties of neural networks
*Szegedy et al.*

- Smoothness assumption does not hold.
- Images imperceptibly close can have different classifications.
- Generated by optimizing the classification error for a trained network.
1: Method

Minimize $\|r\|_2$ subject to:
1. $f(x + r) = l$
2. $x + r \in [0, 1]^m$

The author simplified this by approximating $D$ and using linesearch according to

Min $c|r| + \text{loss}_f(x + r, l)$ subject to $x + r \in [0, 1]^m$. 
1: Conclusions

- Easy to generate adversarial examples
- These generalize to other networks with similar training (set)
- Some robustness was achieved by including adversarial examples in training
2: Fooling Networks

Article 2: Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images
*Nguyen et al.*

- Evolves fooling examples (that maximize classification error) from random noise.
- Images looks very weird when using geometric patterns in evolution
2: Method

Two approaches, 1 random data for each pixel or 2 random rules for a compositional pattern-producing network, CPPN. Tested on networks trained for digit recognition on the Lenet data set and on regular images in the ImageNet data set.
2: Results

example results

<table>
<thead>
<tr>
<th>robin</th>
<th>cheetah</th>
<th>armadillo</th>
<th>lesser panda</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Image" /></td>
<td><img src="image2" alt="Image" /></td>
<td><img src="image3" alt="Image" /></td>
<td><img src="image4" alt="Image" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>king penguin</th>
<th>starfish</th>
<th>baseball</th>
<th>electric guitar</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image5" alt="Image" /></td>
<td><img src="image6" alt="Image" /></td>
<td><img src="image7" alt="Image" /></td>
<td><img src="image8" alt="Image" /></td>
</tr>
</tbody>
</table>
2: Results

- Directly encoded (pixel) images got low confidence on the regular image set (21.59%)
- Indirectly encoded (geometric) images got high confidence on the regular image set (88.11%)
- Geometric images share some superficial features with the training data images
2: New Conclusions

- in independent runs similar and dissimilar geometric patterns were obtained, indicating that the discerning 'features' between classes.
- Author notes the difference between the fooling images for different classes is large, even though we know that even a small perturbation is enough to shift class.
- some runs similar classes got similar pictures, other times very different
- It’s hard to fool images of cats, due to a large sample size and many different classes of cats, so hard to isolate only one subclass
2: Remedies

- introduce a class 'fooling images' and generate new during training and dump them in this class
- no effect on digits but dropped confidence to 11% for regular images.
- sanity check: manually created geometric CPPN images that do look like a class still got high confidence
- sanity check: no decrease in verification on the original verification set.
Article 3: Explaining and Harnessing Adversarial Examples
*Goodfellow, et al.*

- Continues article 1 with some mutual authors.
- Introduces a cheap algorithm for generating adversarial examples.
- Relates the adversarial examples to properties of linear operations in high dimensional space.
3: Method

proof of adversarial examples in one-layer networks: Assume perturbed input $x' = x + \nu$, $w^T x' = w^T x + w^T \nu$ activation is maximized by $\nu = \text{sign}(w)$, with $n$ dimensions and average weight $m$ the activation grows with $\varepsilon nm$ which is linear in $n$ even though $\|\nu\|_\infty < \varepsilon$. Thus adversarial examples will always exist for large $n$. 
3: Method

Fast gradient sign method, for deep networks $J(\theta, x, y)$, cost function of training the network w.r.t. parameters, input image, image’s target class.
Linearizing $J$ around $\theta$: $\eta = \varepsilon \text{sign} (\nabla_x J(\theta, x, y))$.
$\varepsilon$ is a step length parameter, they just picked something that worked. Adversarial example is then $a = x + \eta$.
generates the closes adversarial example, can be generalized to finding a specific class.
3: Method

Feasible to use adversarial examples in training. Updated stopping criterion for training.
3: Results

example results

\[ x + .007 \times \text{sign}(\nabla_x J(\theta, x, y)) = x + \epsilon \text{sign}(\nabla_x J(\theta, x, y)) \]

- “panda” 57.7% confidence
- “nematode” 8.2% confidence
- “gibbon” 99.3% confidence
This led to improvements in the verification on real images (slight, but significant). Especially with more nodes in the hidden layer.

The robustness of the fully trained network was much greater, from an 90% misclassification rate on adversarial examples to 18%.
3: New Conclusions

- adversarial examples are caused by the linearity of the models
- linear models have the strength of fast training and generalization.
- adversarial examples are aligned with weight vectors, explaining their applicability across similar networks.
- adversarial examples can be found along many lines in image-space more common than previously thought
- Article 2 was overkill, cheap to start with random and taking a few fast gradient steps.
3: Remedies

- Radial Basis Functions are found to be much more robust w.r.t. adversarial examples.
- Adversarial examples can be generated in the same way, but yield much lower confidence due to the necessity of moving away from the images.
End