Adversarial Transform Networks
Learning to Generate Adversarial Examples

S. Baluja, I. Fischer

Google Research

arXiv:1703.0938
Reviewed by: Bill Zhang
University of Virginia
https://qdata.github.io/deep2Read/
Outline

Introduction

Adversarial Transformation Networks

MNIST Experiments

ATN Extensions

ImageNet Experiments

Summary
Introduction
Basic Premise and Motivation

>Create a network which learns how to generate adversarial networks given a model
>Generate either untargeted or targeted examples
>Current approaches include using optimizers, fast single-step gradients, and iterative variants of gradient-based techniques
Adversarial Transformation Networks
Network, Optimization, and Inference

- Focus on targeted, white-box ATNs
- ATNs transforms an input into an adversarial example against a target network or set of networks. $\theta$ is the parameter vector of $g$, $f$ is the target network

$$g_{f,\theta}(x) : x \in X \rightarrow x'$$

- To find $g_{f,\theta}$, solve following optimization where $L_X$ is a loss function on the input space and $L_Y$ is a loss function on the output space to avoid learning identity function

$$\arg\min_{\theta} \sum_{x_i \in X} \beta L_X(g_{f,\theta}(x_i), x_i) + L_Y(f(g_{f,\theta}(x_i)), f(x_i))$$

- Inference does not require any further gradient calculations or access to $f$, so generation is very quick
Adversarial Transformation Networks
Loss Function and Reranking

- $L_X$ was picked to be $L_2$ loss
- $L_Y$ determines whether ATN is targeted; define $L_Y$ to be the following equation, where $t$ is the target class, $y = f(x)$, $y' = f(g_f(x))$, and $r$ is a reranking function

$$L_{Y,t}(y', y) = L_2(y', r(y, t))$$

- $r$ reranks $y$ such that $y_k < y_t, \forall k \neq t$ and attempts to keep most of structure from $y$ to minimize distortions; defined where $\alpha > 1$ and $\text{norm}$ rescales output to be valid probability distribution

$$r_\alpha(y, t) = \text{norm}(\{\alpha \max(y) \text{ if } k = t, y_k \text{ otherwise}\})$$
Two approaches of generating examples

- Perturbation ATN (P-ATN): Similar to He et al. 2015, set 
  \( g_f(x) = \tanh(x + G(x)) \), where \( G(x) \) is the core function of \( g_f \); easy to generate small, but effective, perturbations

- Adversarial Autoencoding (AAE): Similar to standard autoencoders, attempt to reconstruct input subject to regularization (\( L_Y \)) and noise

For both approaches, enforce that \( x' \) is in \( X \) by restricting \( x' \) to valid input range of \( f \); adding a \( \tanh \) function in the last layer sufficiently restricts the range to \([-1, 1]\)
MNIST Experiments

Procedure

- Train 5 separate models with varying architectures (combinations of fully connected and convolution layers) and weight initializations on MNIST
  - Baseline accuracy around 98.5-99.1 percent
- Attempt to create an autoencoding ATN using previously detailed process
- During training of ATN, weights of classifier model are frozen
- Empirically set $\alpha = 1.5$ for reranking
MNIST Experiments
Optimizing Beta

- Vary $\beta$ on one various ATN architectures, find average accuracy on all 10 targets (all trained on original model of 5x5 conv, 5x5 conv, FC, FC); 1 separate network for each target
  - Row 1: classifier labeled $x'$ as $t$
  - Row 2: classifier labeled $x'$ as $t$ that kept previous argmax($y$) in second
  - Row 3: argmax($y$) in second place

<table>
<thead>
<tr>
<th>ATN$_a$</th>
<th>$\beta$: 0.010</th>
<th>0.005</th>
<th>0.001</th>
</tr>
</thead>
<tbody>
<tr>
<td>FC $\rightarrow$ FC $\rightarrow$ 28x28 Image</td>
<td>69.1%</td>
<td>84.1%</td>
<td>95.9%</td>
</tr>
<tr>
<td>ATN$_b$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3x3 Conv) $\rightarrow$ (3x3 Conv) $\rightarrow$</td>
<td>61.8%</td>
<td>77.7%</td>
<td>89.2%</td>
</tr>
<tr>
<td>(3x3 Conv) $\rightarrow$ FC $\rightarrow$ 28x28 Image</td>
<td>93.8%</td>
<td>95.8%</td>
<td>97.4%</td>
</tr>
<tr>
<td>ATN$_c$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3x3 Conv) $\rightarrow$ (3x3 Conv) $\rightarrow$ (3x3 Conv) $\rightarrow$ Deconv: 7x7 $\rightarrow$ Deconv: 14x14 $\rightarrow$ 28x28 Image</td>
<td>66.6%</td>
<td>82.5%</td>
<td>91.4%</td>
</tr>
</tbody>
</table>
MNIST Experiments

Optimizing Beta

- Smaller beta values more easily fool networks, but at a cost of losing similarity to original input image
MNIST Experiments

Key Observations

- Transformation maintains empty space; no salt-and-pepper type noise
- In majority of generated examples, shape does not drastically change; by training output to maintain rank except for the top-output, this should be true
- Vertical components seem to be emphasized in some digits (especially if $t = 1$)
- Novel aspect of ATN is the rank preservation
ATN Extensions

Multiple Networks

- ATN trained on one classifier generated examples which did not generalize well to other classifiers, even those with similar architectures.
- ATN trained while minimizing $L_Y$ for three classifiers performed better on all three classifiers and also two outside classifiers.
ATN Extensions

Inside Information

- Since treating classifier as white-box, may be helpful to look at more inside information than just outputs and error derivatives
- Look at hidden unit activations (for practicality, only for penultimate FC layer)
- Increased conditional success of second position
ATN Extensions
Parallel and Serial ATNs

- First, take 1000 random images from MNIST test set, then apply each ATN for each digit separately; track how many ATNs can successfully transform each image
- Next, sequentially apply all 10 ATNs on images
- In parallel application, 283/1000 were transformed successfully; in serial application, 741/1000 were transformed successfully
  - Likely because each transformation diminished underlying original image, so only a few pixels needed to be added to change top class to target
- Order preservation was not maintained in serial application
ImageNet Experiments

Procedure

- Use state-of-the-art Inception ResNet (IR2)
- Trained both AAE and P-ATNs against IR2
- 5 separate architectures for IR2
  - IR2-Base-Deconv (AAE, P-ATN)
  - IR2-Resize-Conv (AAE)
  - IR2-Conv-Deconv (AAE)
  - IR2-Conv-FC (P-ATN)

The use of FC layer makes AAE too slow, so only P-ATN used FC layer

- All 5 architectures were trained with same hyperparameters for 4 separate targets (binoculars, soccer ball, volcano, zebra) for a total of 20 ATNs
- Hyperparameters found using grid search using only volcano target
ImageNet Experiments

Results

- AAE more successful than P-ATN
- P-ATN tends to preserve most of image at cost of small area with high perturbation, while AAE distributes changes across image
- AAEs, however, can produce checkerboard patterns, a common problem in image generation
- For AAEs, many high frequency patterns are replaced by high frequencies which encode adversarial signal

**Table 8. IL2 ATN Performance**

<table>
<thead>
<tr>
<th></th>
<th>Binoculars</th>
<th>Soccer Ball</th>
<th>Volcano</th>
<th>Zebra</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>P-ATN</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IR2-Base-Decov</td>
<td>66.0%</td>
<td>56.5%</td>
<td>0.2%</td>
<td>43.2%</td>
</tr>
<tr>
<td>IR2-Conv-FC</td>
<td>79.9%</td>
<td>78.8%</td>
<td>0.0%</td>
<td>85.6%</td>
</tr>
<tr>
<td><strong>AAE</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IR2-Base-Decov</td>
<td>83.0%</td>
<td>92.1%</td>
<td>88.1%</td>
<td>88.2%</td>
</tr>
<tr>
<td>IR2-Resize-Conv</td>
<td>69.8%</td>
<td>61.4%</td>
<td>91.1%</td>
<td>80.2%</td>
</tr>
<tr>
<td>IR2-Conv-Decov</td>
<td>56.6%</td>
<td>75.0%</td>
<td>87.3%</td>
<td>79.1%</td>
</tr>
</tbody>
</table>
ImageNet Experiments

Results

- AAE produces more diverse adversarial examples, which may be useful for adversarial training
- P-ATNs sometimes produces the same perturbation in the same place for all input examples (similar to DeepDream)
Summary

- ATNs are a fundamentally different approach than previous gradient descent based approaches for generating adversarial examples.
- ATNs are efficient to train, fast to execute, and produce diverse examples; may allow for more robust models in future by improving adversarial training procedures.
References