CleverHans

By: Ian Goodfellow, Nicolas Papernot, Ryan Sheatsley

Presented by: Jennifer Fang [Week 04]

Department of Computer Science: University of Virginia

@ https://qdata.github.io/deep2Read/

CleverHans

Purpose: Benchmark machine learning systems’ vulnerabilities to adversarial attacks by providing adversarial algorithms to assess a machine learning system’s robustness

https://github.com/tensorflow/cleverhans

Previously seen in:

Adversarial Playground (Norton, Qi): Visualize the efficacy of current adversarial methods against convolutional NN systems through a web visualization tool.

- To visualize adversarial samples quickly, they introduced a new algorithm to quickly generate adversarial samples
- Used cleverhans’ JSMA algorithm as the baseline for their new, improved FJSMA algorithm

Background

Adversarial examples: Adversarial examples are inputs crafted by making slight perturbations to legitimate inputs with the intent of misleading machine learning models

How to combat adversarial examples: adversarial training aka training on adversarial samples; the first step to guarding against attacks

Interesting Related Work

Problem: Adversarial training is vulnerable to black-box attacks. With single-step methods, they overfit since regular adversarial training converges to a degenerate global minimum, where small curvature artifacts near the data points obfuscate a linear approximation of the loss.
Thus, the model learns to generate weak perturbations, instead of defending against strong ones.

New approach: Ensemble Adversarial Training is a technique that augments training data with perturbations transferred from other models.

https://openreview.net/forum?id=rkZvSe-RZ (Tramer, Kurakin, Papernot, Goodfellow, ...)

Algorithms in CleverHans

1. L-BFGS
2. FGSM fast gradient sign method
3. Carlini-Wagner Attack
4. Elastic Net Method
5. Basic Iterative Attack
6. Projected Gradient Descent
7. Momentum Iterative Method
8. JSMA *(used in Adversarial Playground)
9. Deep Fool
10. Feature Adversaries
11. SPSA

CleverHans’ purpose and advantages

CleverHans library provides reference implementations of the attacks, which are intended for use for two purposes.

1. Machine learning developers may construct robust models by using adversarial training
   a. This requires the construction of adversarial examples during the training procedure.
2. Provides researchers who report the accuracy of their models in the adversarial setting with a standardized reference implementation
   a. Without a standard reference, different benchmarks aren’t comparable
   b. A benchmark reporting high accuracy could indicate either a more robust model or the use of a weaker attack implementation.
   c. By using cleverhans, we are assured that accuracy on a benchmark corresponds to a robust model.

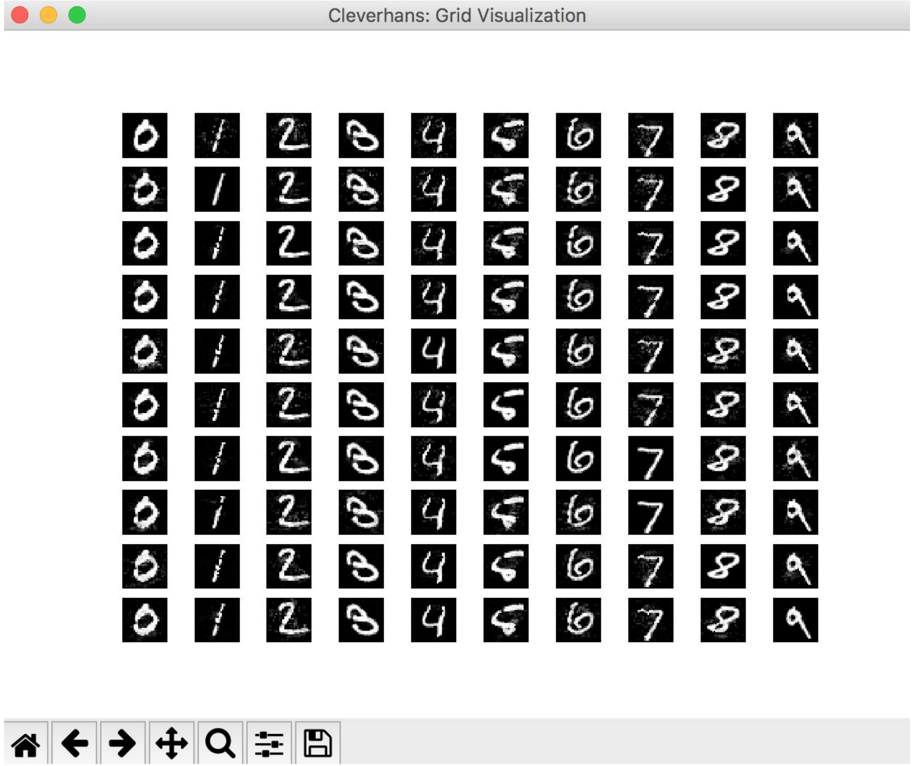
CleverHans approach

Reasoning behind adversarial training: inject adversarial examples during training to improve the generalization of the machine learning model.

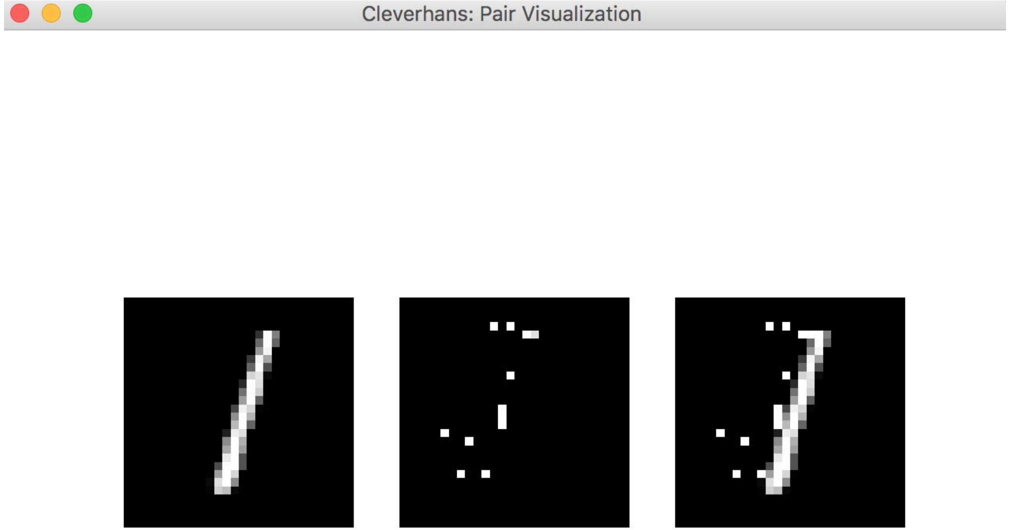
In cleverhans: use the training function tf model train() implemented in module utils tf

- Give it the tensor definition for an adversarial example
- When such a tensor is given, the training algorithm modifies the loss function used to optimize the model parameters:
- It is in that case defined as the average between the loss for predictions on legitimate inputs and the loss for predictions made on adversarial examples.
- The remainder of the training algorithm is left unchanged.

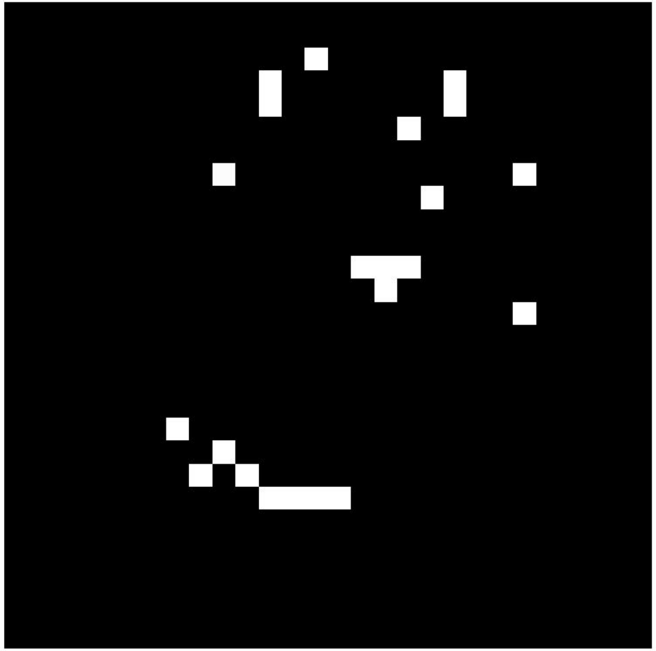
CleverHans with CW

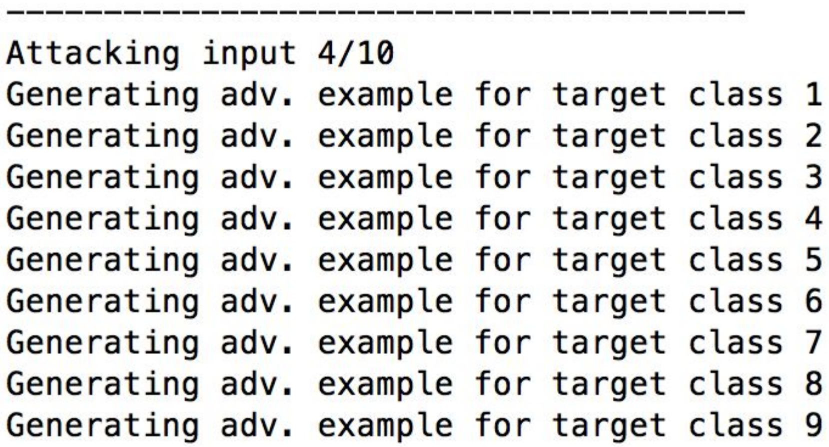


CleverHans with JSMA



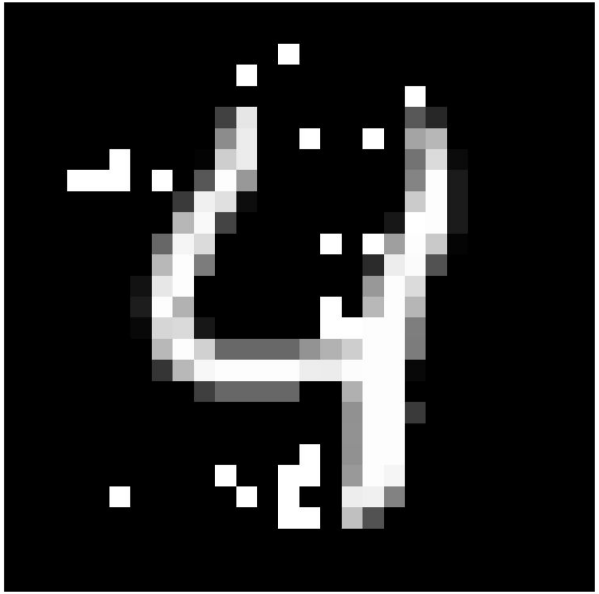
CleverHans with JSMA





```
--------------------------------------
Attacking input 4/10
Generating adv. example for target class 1
Generating adv. example for target class 2
Generating adv. example for target class 3
Generating adv. example for target class 4
Generating adv. example for target class 5
Generating adv. example for target class 6
Generating adv. example for target class 7
Generating adv. example for target class 8
Generating adv. example for target class 9
```

CleverHans with JSMA



Practicality

Pros:

1. Visualizations are easy to understand
2. If you already have necessary prerequisites, might be easy to install
3. Many supporting articles for the algorithms they support

Practicality

Cons:

1. Hard to download (problems with Tensorflow, pip)
2. Documentation is a bit sparse
3. Hard to get examples up and running; required additional setup
4. No sample output or expected output
5. Hard to understand tutorials
6. Difference between regular release and bleeding edge
7. Might have to download *additional* dependencies (Keras)

Links

1. Documentation: https://media.readthedocs.org/pdf/cleverhans/latest/cleverhans.pdf
2. Github: https://github.com/tensorflow/cleverhans
3. Docs: https://cleverhans.readthedocs.io/en/latest/source/model.html
4. Blog: http://www.cleverhans.io/
5. Technical report: https://arxiv.org/abs/1610.00768
6. Tutorial for JSMA: https://gist.github.com/miwong/936d8b12d565802358a924e1073cf6da