Brittle ML: Playing Satan 40- MLG 40

Aditya Vikram(adityavk@iitk.ac.in) **Amur Ghose(amur@iitk.ac.in)**

Ankit Kumar(ankitkur@iitk.ac.in) **Hemant Kumar(hemantk@iitk.ac.in)**

Tushant Mittal(tushant@iitk.ac.in)

Mentor - **Purushottam Kar**

Abstract

Machine Learning models, especially deep neural nets are vulnerable to attack by adversaries because of their linear behaviour in high dimensional spaces. We aim at crafting adversarial inputs for Inception-v3, one of the most prominent image classification convolution neural nets. We study both whitebox and blackbox attacks in our project. Additionally, we also discuss the use of earthmover distances instead of the originally proposed $l_{\inf}$ norm. Crafting adversarial inputs for ranking using decision trees is also discussed briefly towards the end.

1 Problem Statement

Given a model and a certain input, craft an adversarial input. Intentionally designed to make the model err thus revealing its brittleness. Adversarial image should be "close" to the original.

Input (Whitebox) : Complete access to a trained model, set of inputs for the model, constraints(optional).
Input (Blackbox) : Limited Query access to a trained model, set of inputs for the model, constraints(optional).
Output: Adversarial data point(s), which will break the trained model.

2 Problem motivation

Most of our image classification algorithms used in day-to-day life are CNNs. These models are highly vulnerable to attack by adversaries, e.g. we could change a stop sign on a road to fool a self driving car leading to road accidents. This motivates us to inspect adversarial attacks on CNNs so that in the future, we can strengthen these models by counter-acting these attacks (*Adversarial training*). Most of the time, the adversary doesn't know the underlying model type and parameters. Thus, we primarily focus on the blackbox attacks wherein we can only query the model for the output for a given input.

3 Existing work/literature survey

As the field is very recent (2014), not a lot of work has been done. However, many models have been studied.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

3.1 Simple Black-Box Adversarial Perturbations for Deep Networks

https://arxiv.org/pdf/1612.06299.pdf
This paper from Samsung Research America, focus on deep convolutional neural networks and demonstrate that adversaries can easily craft adversarial examples even without any internal knowledge of the target network. The attacks treat the network as an oracle (black-box) and only assume that the output of the network can be observed on the probed inputs.

3.2 Adversarial Examples in the physical World

A research paper discussing few methods of generating adversarial images. This discusses here fast method, basic iterative method, iterative least-likely class method, and also provides the experimental results and relative comparision. This also provides a black box attack demonstration where a clean image from the ImageNet dataset is used to generate adversarial images with various sizes of adversarial perturbation. Experimentation results show the adversarial images are mis-classified.

3.3 Ensemble Adversarial Training: Attacks and Defenses

https://arxiv.org/pdf/1705.07204.pdf
Goodfellow et al. reported that an adversarially trained maxout network on MNIST has slightly higher error rate on transferred examples than on examples crafted from the model itself. Papernot et al. found that a model trained on small perturbations can be evaded by transferring larger perturbations from another model. This paper shows that adversarially trained models on MNIST and ImageNet are significantly more robust to adversarial examples computed on the model itself (i.e., a white-box attack), than to transferred examples computed using the same method on another model (i.e., a black-box attack).

3.4 Cleverhans

A very popular work done by Ian Goodfellow and Nicolas Papernot about security and privacy in machine learning provides the open-source cleverhans library for benchmarking the vulnerability of machine learning models to adversarial examples.

The cleverhans library provides reference implementations of the attacks.

Library is completely Implemented in TensorFlow. Cleverhans is designed as a tool to help developers add defenses against adversarial examples to their models and benchmark the robustness of their models to adversarial examples. The interface for cleverhans is designed to accept models implemented using any model framework (e.g. Keras) or implemented without any specific model abstraction

3.5 TensorFlow

TensorFlow is an open-source software library for dataflow programming across a range of tasks. It is a symbolic math library, and also used for machine learning applications such as neural networks.

4 The Project

4.1 Inception v3

Inception v3 is the state-of-the-art CNN for image classification by created by Google trained using Imagenet. Imagenet is a large dataset of labeled images belonging to commonly seen real-world objects like dogs, cars, aeroplanes etc. The idea of Inception was not simply going deep, but using different sized filters on the same image and then concatenating the feature to generate a more robust representation.

4.2 The Whitebox Attack

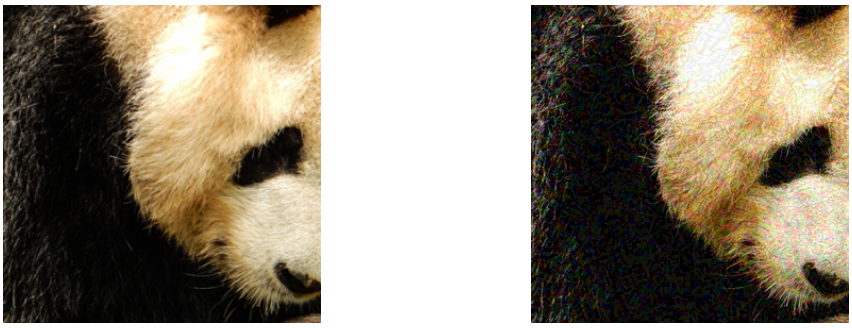
The whitebox attack has been studied in some detail for CNNs but no attack on Inceptionv3 has been carried out yet.

The attack exploits the vulnerability of CNNs which arises due to their piecewise linearity in high-dimensional spaces. The idea is then to move along the direction of the gradient to maximize loss

$$\bar{x} = x + \epsilon sign\left((\nabla_x J(\theta, x, y)\right)$$

4.2.1 Result

As is expected, the whitebox attack performs exceedingly well and creates almost indistinguishable adversarial images



(a) Original Image
92.47% Panda

(b) Adversarial Image
10.02% Samoyed

Figure 1: The Whitebox Attack

4.3 The Blackbox Attack

4.3.1 Challenges

A blackbox attack is much harder to mount as we no longer have access to the gradients which is crucial for the FGSM attack. Moreover since we have no knowledge of the underlying model size, choosing the right

4.3.2 Solution

4.4 Methodology

We perform the blackbox attack on Inception-v3 model following [4], wherein we first create a substitute model to mimic Inception. Our substitute model consisits of 2 hidden layers with 200 nodes each with ReLU activations. The output is a softmax layer. The model was chosen keeping in mind the simplicity of the model and computational ease. [6] allows us to keep the model simple, yet mimic the larger CNN. Next task is to train the substitute model.

We choose a small dataset ($\sim$150 images) from [3] for training the substitute. The images are first pre-processed (resized and mean-centered) to fit to the tensorflow's input standards. This dataset is then augmented using Jacobian augmentation of cleverhans library, using which we generate new images by taking each image, and adding to it the Jacobian of the substitute model w.r.t that image. This way, we increase the variance in our dataset, which improves the model training. The above procedure is repeated multiple times (5-10) to fully train the model.

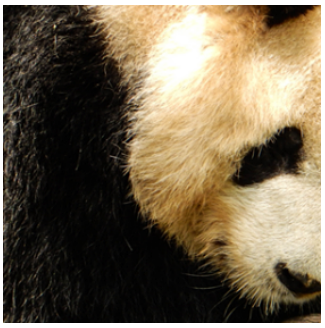
Now that we have simulated Inception-v3, we use FGSM on the substitute model to craft adversarial images like the whitebox.

Algorithm 1 - Substitute DNN Training: for oracle $\tilde{O}$, a maximum number max_ρ of substitute training epochs, a substitute architecture F, and an initial training set S_0.

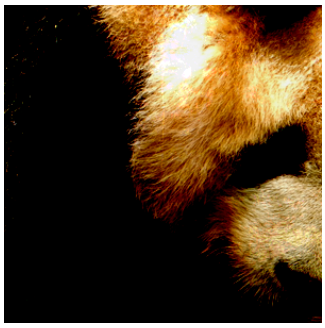
Input: $\tilde{O}$, max_ρ, S_0, λ

1: Define architecture F
2: **for** $\rho \in 0$ .. $max_\rho - 1$ **do**
3: *// Label the substitute training set*
4: $D \leftarrow \left\{(\vec{x}, \tilde{O}(\vec{x})) : \vec{x} \in S_\rho\right\}$
5: *// Train F on D to evaluate parameters θ_F*
6: $\theta_F \leftarrow \text{train}(F, D)$
7: *// Perform Jacobian-based dataset augmentation*
8: $S_{\rho+1} \leftarrow \{\vec{x} + \lambda \cdot \text{sgn}(J_F[\tilde{O}(\vec{x})]) : \vec{x} \in S_\rho\} \cup S_\rho$
9: **end for**
10: **return** θ_F

Figure 2: The Blackbox attack algorithm [6]



(a) Original Image
92.47% Panda



(b) Adversarial Image
25.26% Panda

Figure 3: The Blackbox Attack

5 Novel Ideas

5.1 Restricted Query Model

- Usually, a blackbox attack places no restriction on the number of queries and this is exploited by the algorithm to create a nice substitute model.
- But in many settings it might not be possible to actually make a lot of queries.
- We thus look at how the performance changes by such a restriction

5.2 Tables

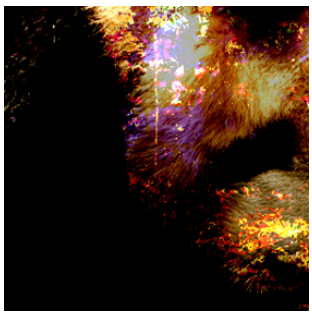
5.3 Distorted Images - large epsilon

As we can see the misclassification rate increases despite a decrease in the number of query images. This can be explained when we look at the images. Basically, the l_2 norm though restricted can give

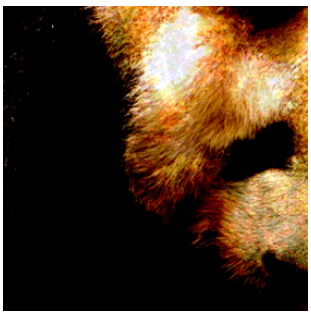
Table 1: Effect of query restrictions

Number of Images (n)	Epochs	Epsilon (ϵ)	Misclassification Rate
150	4	0.3	58.00%
150	3	0.3	33.53%
150	1	0.3	52.118%
125	4	0.3	54.52%
100	03	0.3	62.12%

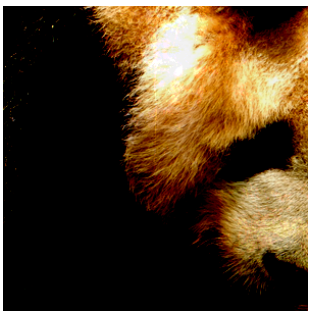
rise to visually dissimilar images and this causes misclassification. As we observe, decreasing epsilon even further shows that the misclassification rates go back up.



(a) $n = 150, \epsilon = 0.3$
55.39% polecat



(b) $n = 150, \epsilon = 0.1$
68.94% Panda



(c) $n = 150, \epsilon = 0.05$
25.26% Panda

5.4 A probabilistic approach

Under the FGSM approach, every input X can be changed into the adversarial input X' that swaps the label in a binary classification system whilst being closest in terms of the $L2$ norm. Now, let us treat X' as the MAP solution given a particular input X instead of a deterministic solution.

We assume that :

- The solved-for X' is an acceptable substitute for the mean of the posterior distribution on the set of adversarial images, i.e. there exists a random variable I distributed as $\mathcal{N}(\mu, \Sigma)$ where μ is X'
- To approximate Σ, we further assume that FGSM, being a gradient approach, yields a first order optimum. Thus, the normal distribution around the optimum satisfies $\Sigma = H^{-1}$ where H is the Hessian matrix at X'.
- Since we already assume the existence of gradient oracles, computing the Hessian is not difficult.

Sampling from the corresponding normal instead of playing the mean everytime is in some ways an UCB-like approach to the adversarial problem, and analogous to corresponding bandit strategies such as Thompson sampling.

We test this approach on KDE and iGMM (Infinite GMM) density estimators, feeding the blackbox gradient of the Log-likelihood to construct the Hessian. Note that the expected $L2$ norm is slightly increased and should be rescaled by the Monte-Carlo mean of the minibatch for a particular round. We present our results below - note that ϵ was learnt as twice the standard deviation along an axis and not used as static - using the setosa and versicolour classes for Iris, and the Male and Infant categories for Abalone :

Table 2: Results of the Hessian attack

Dataset	Hessian flip chance (iGMM)	FGSM flip chance (iGMM)	Hessian flip chance (KDE)	FGSM flip chance (KDE)
Iris	60.3%	57.2%	30.2%	12.7%
Abalone	32 to 58%	47.6%	22.4%	5.6%

6 Future Directions

6.1 Generalizing the norm

Current methods [5] of crafting adversarial examples use l_p norms to constrain the added noise in order to prevent a visible change in the input. The idea is to try to construct an FGSM-like attack using Earthmover Distance (EMD) to encapsulate the perceptual similarity of images better than l_{inf} norm, or any l_p norm in general.

6.2 Adversarial crafting for Ranking with Decision Trees

Papernot et. al. proposed an attack [6] for decision trees as classifiers. Future work could focus at breaking LambdaMart [7], a boosted regression trees model to rank search queries. However, in order to rank using decision trees, finding the nearest leaf that output a different regression value doesn't help because the aim is to change the sorting order of the inputs, and not just the regression values. We infer from this that ranking using DTs is much more robust than CNNs, but we aim to break the former in future.

References

[1]Goodfellow *et.al.* cleverhans v2.0.0: an adversarial machine learning library

[2]Tramèr *et.al.* Ensemble Adversarial Training: Attacks and Defenses https://arxiv.org/pdf/1705.07204.pdf

[3]NIPS 2017: Adversarial Learning Development Set. https://www.kaggle.com/google-brain/nips-2017-adversarial-learning-development-set

[4]Papernot *et.al.* Practical Black-Box Attacks against Machine Learning. https://arxiv.org/pdf/1602.02697.pdf

[5]Tramer *et.al.* Space of Transferable Adversarial Examples. https://arxiv.org/abs/1704.03453

[6]Papernot *et.al.* Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples. https://arxiv.org/pdf/1605.07277.pdf

[7]Chris J.C. Burges. From RankNet to LambdaRank to LambdaMART: An Overview https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/MSR-TR-2010-82.pdf

[8]Inception classification using pretrained model tutorial. http://cv-tricks.com/tensorflow-tutorial/understanding-alexnet-resnet-squeezenetand-running-on-tensorflow/

[9]https://github.com/sankit1/cv-tricks.com/blob/master/Tensorflow-tutorials/Tensorflow-slim-run-prediction/run_inference_on_v3.py

[10]https://www.kaggle.com/benhamner/fgsm-attack-example---fgsmattackonpretrainedinception

[11]Nina Narodytska & Shiva Prasad Kasiviswanathan, Samsung Research America, Simple Black-Box Adversarial Perturbations for Deep Networks