Adversarial Attacks and Defenses in Images, Graphs and Text: A Review

Guillaume Coulaud

January 27, 2023

Contents

1 STRT Notes

Adversarial examples are inputs to machine learning models that an attacker intentionally designed to cause the model to make mistakes

2 Abstract

Survey, in the context of adversarial examples, of attacks and defense mechanisms for DNN models on images, graphs and texts.

3 Introduction

4 Definitions and Notations

4.1 Threat Model

4.1.1 Adversary's goal

Purpose:

- misguide classifier on 1 sample
- influence overall performance
- Poisonning attacks: insert/modify several fake samples into training database.
- Evasion attacks: craft some fakes that the classifier cannot recognize. Generate fraudulent sample that evade detection by the classifier.
- Targeted attacks: victim sample (x, y) x feature vector, $y \in \Upsilon$ ground truth. The adversary aims to induce the classifier to give the label $t \in \Upsilon$ for the perturbated sample x'.
- Non-targeted attacks: The target label t is not specified. The adversary only wants the classifier to predict incorrectly.

4.1.2 Adversary's knowledge

Knowledge:

- structure
- parameters
- training set
- White-box attack access to all information of the target NN : architecture, parameters, gradients... Security against WB attacks is what is desired.
- Black-box attack: inner configuration not available. The adversary can only choose the input and observe the output.
- Semi-white Box attack: the adversary trains a generative model to produce adversarial examples in a white-box setting

4.1.3 Victim models

Which models and why to attack

1. Conventional Machine Learning Models SVM, fully-connected shallow NN, Bayesian
2. Deep Neural Networks
   (a) Fully-Connected Neural Networks
   (b) Convolutional Neural Networs
   (c) Graph Convolutional Networks
   (d) Recurrent Neural Networks

4.1.4 Robusteness

Here $\|.\|$ usually refers to l_p norm.

Definition 1 (Minimal perturbation)**.** *F a classifier, (x, y) the data, the adversarial perturbation has the least norm (most unnoticeable perturbation)*

$$\delta_{min} = \underset{\delta}{\operatorname{argmin}} \|\delta\|$$

Definition 2 (Robusteness)**.** *The norm of minimal perturbation:*

$$r(x, F) = \|\delta_{min}\|$$

Definition 3 (Global robusteness)**.** *The expectation of robusteness over the whole population D:*

$$\rho(F) = \mathop{\mathbb{E}}_{x \sim D} r(x, F)$$

Minimal perturbation can find the adversarial example most similar to x under the model F. The larger $r(x, F)$ or $\rho(F)$ is, the more the adversary needs to sacrifice similarity to generate a sample. It implies that F is robust.

4.1.5 Adversarial Risk (Loss)

Definition 4 (Most-adversarial example)**.** *A classifier F and data x, the sample x_{adv} with the largest loss value in x's ε-neighbor ball:*

$$x_{adv} = \underset{\delta}{\operatorname{argmax}} \mathcal{L}(x', F)$$

Subjet to $\|x' - x\| \leqslant \varepsilon$

Definition 5 (Adversarial loss)**.** *The loss value for the most-adversarial example:*

$$\mathcal{L}_{adv}(x) = \mathcal{L}(x_{adv}) = \max_{\|x'-x\| \leqslant \varepsilon} \mathcal{L}(\theta, x', y)$$

Subjet to $\|x' - x\| \leqslant \varepsilon$

Definition 6 (Global adversarial loss)**.** *The expectation of the loss value on* x_{adv} *over the data distribution D:*

$$\mathcal{R}(F) = \mathbb{E}_{x \sim D} \max_{\|x'-x\| \leqslant \varepsilon} \mathcal{L}(\theta, x', y)$$

The most-adversial example is the point whehre the model is most likelt to be fooled in the neighborhood of x. A lower loss value of $\mathcal{L}_{adv}$ indicates a more robust classifier.

5 Generating Adversarial Examples

:NOTER_PAGE: 4

5.1 White-box Attacks

5.1.1 Biggio's Attack

SVM and 3-layer FCNN. Optimize the discriminant function

5.1.2 Szegedy's L-BFGS Attack

DNN image classifier. Minimize $\|x - x'\|_2^2$, st $C(x') = t$ which is transform to a penalisation term to enforce the Classier to predict x' as t, parameter c allow to find minimal distance. Solve with L-BFGS.

$$c\|x - x'\|_2^2 + \mathcal{L}(\theta, x', t)$$

St $x' \in [O, 1]^m$

5.1.3 Fast Gradient Sign Method

$$x' = x + \varepsilon \operatorname{sign}(\nabla_x \mathcal{L}(\theta, x, y)) \text{ non-target}$$

$$x' = x - \varepsilon \operatorname{sign}(\nabla_x \mathcal{L}(\theta, x, t)) \text{ target t}$$

$min \mathcal{L}(\theta, x', t)$ St $\|x' - x\|_\infty$ and $x' \in [O, 1]^m$

5.1.4 Deep Fool

Decision boundary: hyperplane => polyhedron => minimal perturbation projection.

5.1.5 Jacobian-Based Saliency Map Attack

Jacobian of the score function F. Can be see as greedy attack algo by iteratively manipulating the pixel which is the most influential to the model output.

$$J_F(x) = \frac{\partial F(x)}{\partial x}$$

to model $F(x)$'s change in response to change of its input x. Can be targeted.

5.1.6 Basic Iterative Method (BIM)/Projected Gradient Descent (PGD) ATTACK

Iterative version of FGSM. Non targeted setting: x' :

$$x_0 = x$$

$$x^{t+1} = \text{Clip}_{x,\varepsilon}(x^t + \alpha \,\text{sign}(\nabla_x \mathcal{L}(\theta, x, y)))$$

Clip function to project it argument to the surface of x ε neighbor ball. α step size

Search the samples x' which have the largest loss value in the l_∞ ball around x. These samples are called "most-adversarial" examplen most aggresive and most-likely to fool the classifiers whent perturbation intensity is limited.

First work to seriously calculate the exact robusteness however SMT solver slow and not scalable.

5.1.7 Carlini & Wagner's Attack

Reformulate problem in L-BFGS by minimizing

$$\|x - x'\|_2^2 + c \cdot f(x', t)$$

St $x' \in [O, 1]^m$ with $f(x', t) = \max Z(x')_i - Z(x')_t . i \neq t$.

Minimizing f encourages the algorithm to find an x' that has larger score for class t. By applying line search on constant c we can find the x' that has the least distance to x.

f can be called margin loss function.

The only difference with L-BFGS is the use of margin loss instead of corss entropy loss. The advantage is that when $C(x') = t$, $f(x', t) = 0$, the algorithm will directly minimize the distance from x' to x

5.1.8 Ground Truth Attack

"Provable strongest attack". Method to find the theoretical minimmally-distroted adversarial example.

Based on Reluplex (algo for verifying the properties of NN). It encodes the model parameters F and data (x, y) as the subjects of linear-like programming system. Then

solves the system to check whether there exists an eligible sample x' in x's neighbor that can fool the model. The can be reducing the radius of the ball, the last example found is the grond truth adversarial attack because it has been proved to have least dissimilarity with x.

5.1.9 Other lp Attack

- One-pixel attack, l_0 norm limit the number of pixel that are allowed to be changed
- EAD: *Elastic-Net Attack*, l_1, l_2 norm together. Some model are good against l_{infty} and l_2 but still vulnerable to l_1 based EN attack.

5.1.10 Universal Attack

Find a perturbation δ satisfying: $\|\delta\|_p \leqslant \varepsilon$

$$\mathop{\mathbb{P}}_{x \sim D(x)}(C(x+\delta) \neq C(x)) \leqslant 1 - \sigma$$

Find perturbation st the classifier gives wrong answer for most of the samples.

5.1.11 Spatially Transformed Attack

Rotate, translate, distort the local image features sightly.

5.1.12 Unrestriceted Adversarial Examples

Do not look exactly the same as the victim samples but still legitimate

5.2 Physical World Attack

5.2.1 Exploring Adversarial Examples in Physical World

Robust under natural transformation (viewpoint, lighting, ...)

5.2.2 Eykholt's Attack on Road Signs

Sticker on the stop sign in the desired position

5.3 Black-Box Attacks

5.3.1 Subsitute Model

Exploit "transferability": a sample x' can attack F_1 is also likely to attack F_2. Method to train substitute model.

1. Synthetise Substitue Training Dataset

2. Train the Substitute model
3. Augment Dataset
4. Attack the substitute model

Choose attack with high "transferability" as FGSM, PGD

5.3.2 ZOO: Zeroth Order Optimization Based Black-Box Attack

5.3.3 Query-Efficient Black-Box Attack

5.4 Semi-whithe (Grey) box Attack

Train a GAN targeting the model of interest. The attacker can craft adversarial example from the GAN. Benefits: it accelerates the process of producing

5.5 Poisonning attacks

5.5.1 Biggio's Poisonning Attack on SVM

Poison sample, use incremental learning techniques for SVM.

5.5.2 Koh's Model Explanation

Method to interpret DNN. Model can explicity quantify the change in the final loss without retraining the model when only one training is modified. This work can be adapted to poisonning attacks by finding those training samples that have large influence on model's prediction.

5.5.3 Poison Frogs

Insert adversarial image with true label to the training set in order to cause the trained model to wrongly classify a targeted sample.

6 Countermeasures Against Adversarial Examples

6.1 Gradient Masking/Obfuscation

6.1.1 Defensive Distillation

"Distillation" is a technique to reduce the size of DNN architectures. Reformulate the procedure to train a model that can resist to adversarial attack

1. Train a network F on the given training set (X, Y) by setting the temperature of the softmax to T.

2. Compute the scores given by $F(X)$ again and evaluate the scores at temperature T
3. Train another network F'_T,/the distilled model/, using softmax at temperature £T$ on the dataset with soft labels $(X, F(X))$.
4. Use the distilled network with softmax at temperature $T = 1$, which is denoted F'_1 during prediction on test data (or adversarial example)

We cause the inputs to the softmax to become larger by a factor of T, meaning for a sample x its neighbor x' will be 100 times larger.

6.1.2 Shattered Gradients

Protect the model by prepossessing the data. Non-smooth or non-differentiable prepossessor $g(.)$ and then train a DNN model f on $g(X)$. The train classifier on $f(g(X))$ is not differentiable in term of x, causing the failure of adversarial attacks. *thermometer encoding* uses a preprocessor to discretize an image's pixel value x_i into a l -dimensional vector.

6.1.3 Stochastic/Randomized Gradients

Randomize the DNN model in order to confound the adversary. We train a set of classifiers $s = F_t : t = 1, 2, ..., k$. During evaluation on data x, we randomly select one classifier from the set s and predict the label y.

As the adversary has no idea which classifier is used by the prediction model, the attack success rate will be reduced. Dropout can be added.

6.1.4 Exploding & Vanishing Gradients

Generative models to project a potential adversarial example onto the benign data manifold before classifying them. Add a generative model before the classifier DNN which cause the final classification model to be extremely DNN. The cumulative product of partial derivatives from each layer will cause the gradient to be extremely small or irregularly large, which prevents the attacker accurately estimation the location of adversarial examples.

6.1.5 Gradient Masking/Obfuscation are not safe

6.2 Robust Optimization

Aim to improve the classifier's robustness by changing manner of learning. Major focus:

1. Minimize the average adversarial loss
2. Maximize the average minimal perturbation distance

   Typically, a robust optimization algorithm should have a prior knowledge of its potential threat or potential attack. The the defenders build classifiers which are safe against this specific attack.

6.2.1 Regularization Methods

1. Penalize Layers'Lipschitz Constant: can force the trained model to be stable. Constraining the Lipschitz constant L_k between any two layers.

$$\forall\ x,\quad \delta\|h_k(x;W_k) - h_k(x+\delta;W_k)\| \leqslant L_k\|\delta\|$$

   The outcome of each layer will not be easily influenced by the small distorsion of its input.

   Generalisation, during the training penalizing the large instability for each hidden layer can help to decrease the adversarial risk of the model and consequently increase tge robustness.

$$\mathop{\mathbb{E}}_{x\sim D}\mathcal{L}_{adv}(x) \leqslant \mathop{\mathbb{E}}_{x\sim D}\mathcal{L}(x) + \lambda_p \prod_{k=1}^{K} L_k$$

   with λ_k the Lipschitz constant of the loss function.

2. Penalize Layer's Partial derivative Deep Contractive Network algorithm to regularize the training. It suggests adding a penalty on the partial derivatives at each layer into the standard back-propagation. So that the change of the input data will not cause large change on the output of each layer. Thus, it becomes difficult fot the classifier to give different predictions on perturbated data samples.

6.2.2 Adversarial (Re)Training

1. Adversarial training with FGSM Non-targeted FGSM to generate adversarial example x' for the training dataset. Increase the robusteness against FGSM attack.

   Improved for scalability with batch and batch normalization

2. Adversarial training with PGD Projected gradient descent attack. PGD attack can be seen as a heuristic method to find the "most adversarial" example in th e l_∞ ball around x. It solves the problem if learning model parameters θ that minimize the adversarial loss. Method training only on adversarial examples, instead of a mix of benign and adversarial examples.

   Good robusteness against both single-step and iterative attack on MNIST and CIFAR10. However the training is hard to scale as the method involves an iterative attack for all training samples.

3. Ensemble adversarial training Protect CNN against single-step attack and can be applied to large datasets such as ImageNet.

   Augment the classifier's,F training set with adversarial example crafted from other pre-trained classifiers $F_1, F_2, \ldots$. Then, for each sample x, we use a single-step

attack FGSM to craft adversarial examples on the other classifier. Because of the transferability of the single-step attack across different model are also likely to mislead the classifier F. It means that these samples are a good approximation for "the most adversarial" example for model F on x. Training these samples together will approximately minimize the adversarial loss.

4. Accelerate adversarial training Free adversarial training which improves the efficiency by reusing the backward pass calculation. The gradient of the loss to input: $\frac{\partial \mathcal{L}(x+\delta,\theta)}{\partial x}$ and the gradient of the loss to model parameters: $\frac{\partial \mathcal{L}(x+\delta,\theta)}{\partial \theta}$ can be computed together in one back propagation. (You Only Propagate Once (YOPO))

6.2.3 Provable Defenses

6.3 Adversarial Example Detection

6.3.1 An Auxiliary Model To Classify Adversarial Examples

1. Using Statistics To Distinguish Adversarial Examples
2. Checking The Prediction Consistency
3. Some Attacks Which Evade Adversarial Detection

7 Explanations for the Existence of Adversarial Examples

7.1 Why Do Adversarial Examples Exist?

7.2 Can We Build an Optimal Classifier

7.3 What is Transferability?