Security and Privacy of Machine Learning - Homework 1

Chun-Yao Chang
National Taiwan University
b07902022@csie.ntu.edu.tw

Abstract

There are two phases in this homework. In phase 1, we[1] implemented adversarial attacks against CIFAR-100 image classifiers under black-box setting. We've tried several attacking methods and consider multiple attacking parameters. Based on the evaluation, we selected the adversarial examples that is most powerful on fooling the classifier. In phase 2, we changed our evaluation metric to the website[2] provided by the teacher. And then, we slightly modified the adversarial examples to reach the best attack accuracy.

1 Introduction

Powerful as neural networks are, they are also vulnerable to malicious adversarial examples. The perturbations are imperceptible to human; however, they are likely to make the classifiers output the wrong label with high confidence.

In phase 1, we implemented adversarial attacks against CIFAR-100 classifiers under a black-box setting, which means that we don't know any details about the models. The adversarial perturbation is $l_\infty = 8$ on the 0-255 pixel scale.

In phase 2, we use the website to evaluate the model's performance and slightly modify our choice.

Our code is available at: `https://github.com/ChunYaoChang/SPML-HW1`[3]

2 Attack Methodology

First, we denote the classifier defined by the DNN with softmax output activation as $y' = f(\theta, x)$ for a given image-label pair (x, y). Our target is to find the adversarial image x' by maximizing the loss $L(x', y) = L(f(\theta, x'), y)$, subject to the l_∞ perturbation constraint $||x' - x||_\infty \leq \epsilon$ with ϵ be the attack strength. In this homework, ϵ is set to 8 on the 0-255 pixel scale.

2.1 Variation of Fast Gradient Sign Methods

In this section, we introduce different FGSM-based approaches.

Fast Gradient Sign Method (FGSM)

Under the first order approximation, we have the following formula:

$$L(x', y) \approx L(x, y) + \nabla_x L(x, y)^T \cdot (x' - x)$$

[1] This homework is actually done by myself.
[2] http://clais2.csie.org:8000/
[3] It is made public after the deadline.

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

FGSM find the adversarial perturbations in the direction of the loss gradient $\nabla_x L(x, y)$. The update equation is

$$x' = x + \epsilon \cdot \text{sign}(\nabla_x L(x, y))$$

Iterative Fast Gradient Sign Method (IFGSM)

IFGSM iterates FGSM to generate enhanced attacks. The update equation is

$$x^{(m)} = \text{Clip}\{x^{(m-1)} + \alpha \cdot \text{sign}(\nabla_x L(x^{(m-1)}, y))\},$$

where α is the step size of each iteration, $m = 1, \cdots, M, x^{(0)} = x$ and $x' = x^{(M)}$, with M being the number of iterations.

Momentum Iterative Fast Gradient Sign Method (MIFGSM)

MIFGSM adds the momentum into IFGSM to stabilize the update process and escape from local maxima. The update equation is

$$g^{(m)} = \mu \cdot g^n + \frac{\nabla_x L(x^{(m-1)}, y))}{||\nabla_x L(x^{(m-1)}, y))||_1}$$

$$x^{(m)} = \text{Clip}\{x^{(m-1)} + \alpha \cdot \text{sign}(g^{(m)}\},$$

where μ is the decay factor of momentum.

3 Proxy Model Selection

Selection of proxy models plays an important role in black / gray box attacks. A better selection may lead to better transferability and thus make the attack more successful without knowing the exact model. For the model selection, I follow the following three conclusion

1. From Andrewa et al.[1], we know that stronger model tends to learn more information from non-robust feature, which will theoretically harm the robustness of the model.
2. From Liu et al.[2], we know that ensemble method can result in higher success rate on generating transferable adversarial examples.
3. From Miller et al.[3], we know that shallower models can be seen as a form of undertrained models, and undertrained models have better transferability on generating adversarial examples.

Therefore, our final proxy model ensembles the following 16 models, and we call it Ensemble-16-small:

1. NIN
2. ResNet20
3. PreResNet20
4. ResNext29 (32x4d)
5. Se-ResNet20
6. Se-PreResNet20
7. PyrimidNet110 (a=48)
8. DenseNet40 (k=12)
9. X-DenseNet-BC-40-2 (k=24)
10. WRN-16-10
11. WRN-20-10 (bit=1)
12. RoR-3-56
13. RiR

14. Shake-Shake-ResNet26 (2x32d)
15. DIA-ResNet20
16. DIA-PreResNet20

All the pretrained models are directly imported from pytorchcv[4]. Also, our model ensemble method can be defined as:

$$l(X; \theta_1, \theta_2, \ldots . \theta_N) = \sum_{i=1}^{N} w_i \cdot l_i(X, \theta_i)$$

where $l_i(X, \theta_i)$ is the logits output of i-th model and w_i is the weight with $w_i \geq 0$ and $\sum_{i=1}^{N} w_i = 1$.

4 Evaluation Result

In this section, we compare different attack methods and proxy models to get the best result. In phase 1, we use the top-1 accuracy on our self-construct model (Acc on SC-model) as the benchmark. In phase 2, we use the top-1 accuracy on query system model (Acc on QS-model) as the benchmark. Our SC-model is EfficientNetV2-S[5] with some modification.

4.1 Attack Method Comparison

Here, I choose the three previous mentioned attack method to evaluate their performances. The proxy model is Ensemble-16-small.

Attack Method	Acc on SC-model	Acc on QS-model
FGSM	27.1	47.2
IFGSM (10 iterations)	8.7	42.6
MIFGSM (10 iterations)	9.0	41.8

Table 1: Performance of different attack method

Table 1 shows that the iterative version of FGSM performs better than FGSM, and adding momentum into it actually slightly enhances the performance on QS-model.

Attack Method	Acc on SC-model	Acc on QS-model
MIFGSM (2 iterations)	18.2	45.3
MIFGSM (5 iterations)	14.8	42.9
MIFGSM (10 iterations)	9.0	41.8
MIFGSM (20 iterations)	8.2	41.9

Table 2: Performance of different number of iteration

Table 2 shows that the performance is better when the number of iteration grows from 2 to 5. Surprisingly, the accuracy on QS-model drops when the number of iteration becomes 20. It's probably because of the decrease of transferability.

4.2 Proxy Model Comparison

Here, I choose four different models to evaluate their performances:

1. ResNet20
2. ResNet542-bn
3. Ensemble-16-small
4. Ensemble-16-large (increase the network size of each model in Ensemble-16-small)

The attack method is MIFGSM with 10 iterations.

[4]https://pypi.org/project/pytorchcv/

[5]https://github.com/lukemelas/EfficientNet-PyTorch

Proxy Model	Acc on SC-model	Acc on QS-model
Resnet20	46.9	76.3
ResNet542-bn	38.5	64.2
Ensemble-16-small	9.0	41.8
Ensemble-16-large	13.7	50.6

Table 3: Performance of different proxy model

The table 3 shows the power of ensemble method: easily surpass the single state-of-the-art model. Also, we discover that Ensemble-16-small perform better than Ensemble-16-large, which verifies that shallower models have better transferability.

5 Conclusion

In this homework, we've tried several attack methods and proxy models to perform balck-box attack and gray-box attack. For the attack method, we found that both IFGSM and MIFGSM perform better than FGSM. For the proxy model, we found that adversarial examples generated by ensemble-based model is much more robust than that by single model. Based on the result above, we use **MIFGSM with 10 iterations and Ensemble-16-small** as our final method. And we successfully reduce the accuracy on QS-model to only 41.8%.

References

[1] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, "Adversarial examples are not bugs, they are features," 2019.

[2] Y. Liu, X. Chen, C. Liu, and D. Song, "Delving into transferable adversarial examples and black-box attacks," 2017.

[3] C. Miller and S. Vosoughi, "Query-free adversarial transfer via undertrained surrogates," 2020.