Summer Research Internship (PC334)

Topic: Adversarial Machine Learning

Mentor: Prof. Manjunath Joshi

Team Members
Nishtha Chaudhary (201801235)
Ritika Lakdawala (201801226)

Acknowledgement

We undertook this research internship project and completed the internship report under the guidance of prof. Manjunath Joshi. We are also grateful to the senior PhD. student Amruta Kalgutkar for her assistance during internship. It was a good learning experience for us to work in the area of adversarial machine learning.

Motivation

Nowadays Machine Learning and deep learning models are being used in various fields but they are also vulnerable to attacks. To ensure the robustness of these models, many adversarial attacks and defence techniques are developed. So we tried to implement some of the attacks to learn about adversarial robustness.

Background

An adversarial attack consists of subtly modifying an original image in such a way that the changes are almost undetectable to the human eye. The modified image is called an adversarial image, and when submitted to a classifier is misclassified, while the original one is correctly classified. Here three different white-box(where the attacker has complete access to the model being attacked) adversarial attacks are implemented: *Fast Gradient Sign Method(FGSM), Projected Gradient Descent(PGD) and Carlini-Wagner(CW) attack*.

Implementation

For the implementation of the attack algorithms, we have used pytorch (python machine learning library). The attacks are performed on the Resnet50 CNN model; pre-trained over the imagenet dataset and the optimizer used is Stochastic gradient descent(SGD).

FGSM attack

The fast gradient sign method works by using the gradients of the neural network to create an adversarial example. For an input image, the method uses the gradients of the loss with respect to the input image to create a new image that maximises the loss. This new image is called the adversarial image. This can be summarised using the following expression:

$$adv\_x = x + \epsilon * \mathrm{sign}(\nabla_x J(\theta, x, y))$$

Where, adv_x: Adversarial image, x: Original input image, y: Original input label, ϵ: Multiplier to ensure the perturbations are small, θ: Model parameters, J: Loss.

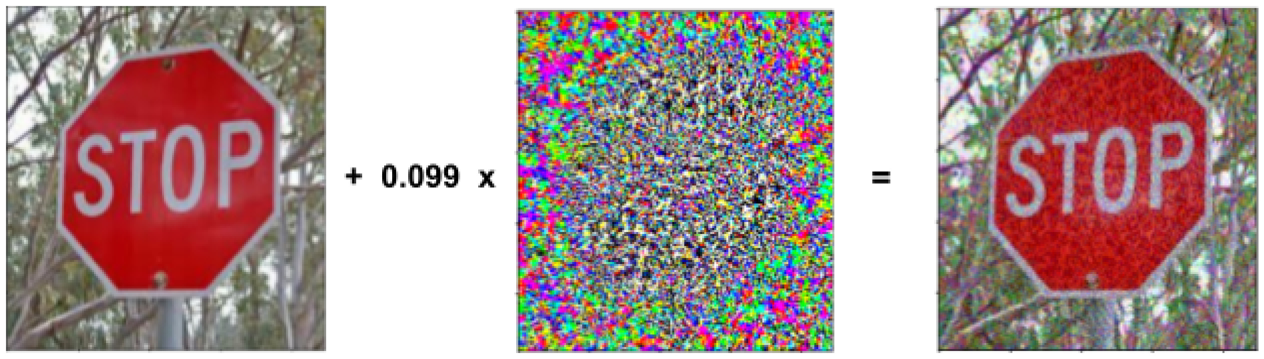
- **Algorithm:**

```
epsilon=0.12
img_tensor.requires_grad=True

prediction = model(norm(img_tensor))
loss = -nn.CrossEntropyLoss()(pred1, torch.LongTensor([true_class]))
loss.backward()

adv_img = img_tensor + epsilon*torch.sign(img_tensor.grad)
```

- **Results:**



Original Image
"street_sign"
96.57% Confidence

Perturbation

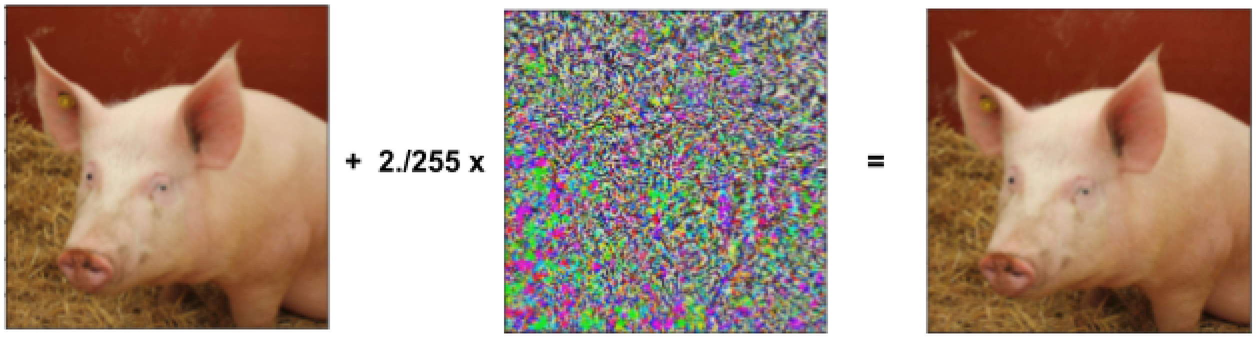
Adversarial Image
"bottlecap"
44.24% confidence

PGD attack

Projected Gradient Descent Attack is also known as I-FGSM which expands for Iterative - Fast Gradient Sign Method. PGD attempts to find the perturbation that iteratively maximises the loss of a model on a particular input while keeping the size of the perturbation smaller than a specified amount referred to as epsilon.

$$\delta_{t+1} = \text{clip}_{\varepsilon}(\delta_t - \alpha * \nabla_{\delta_t} J(\theta, x, y))$$

In every iteration, we add this δ (perturbation) to the original image to get adversarial image and this image is used to calculate the updated loss function. This process is repeated for a finite number of iterations to get the optimized adversarial image.



Original Image
"hog"
99.61% Confidence

Perturbation

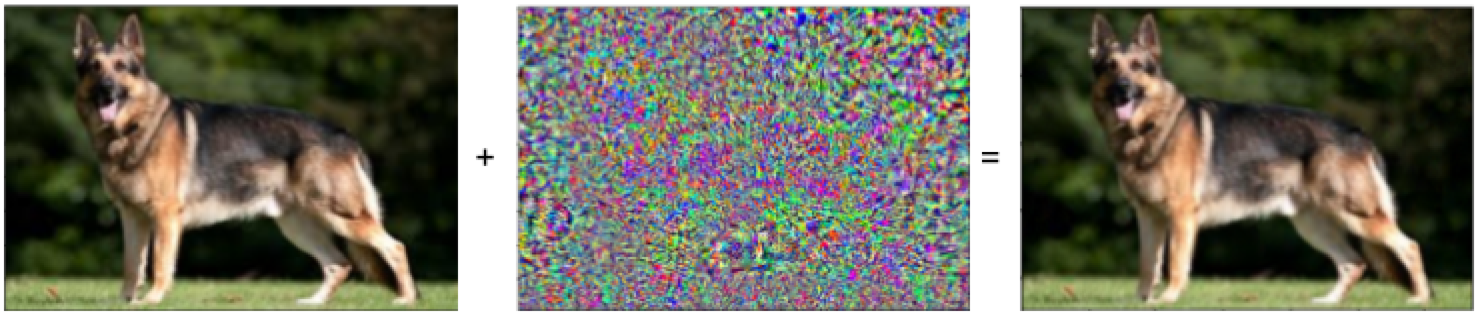
Adversarial Image
"wombat"
99.99% confidence

Targeted(CW) attack

Below is the constrained optimization problem which used to generate the targeted adversarial examples:

$$\begin{aligned} &\textbf{minimize: } D\,(x, x+\delta) \\ &\textbf{such that: } C\,(x+\delta) = t \quad \text{----- constraint 1} \\ &\qquad\qquad x+\delta \in [\,0, 1\,]^n \quad \text{----- constraint 2} \end{aligned}$$

x: input image, δ: perturbations, D: distance metric between the adversarial and real image, C: Classifier function, t: target class. Constraint 1 makes sure that the image is indeed misclassified and constraint 2 makes sure that the adversarial image is valid i.e. it lies within the normalized dimensions of x.
For the optimization purpose, loss related to true class is maximized and loss related to targeted class is minimized so that the adversarial image will be misclassified as the targeted class.



Original Image
"German_shepherd"
98% confidence

Perturbation

Adversarial Image
Targeted class - "Teddy"
Predicted class - "Teddy"
100% confidence

References

https://ieeexplore.ieee.org/document/7958570
https://adversarial-ml-tutorial.org/introduction/
https://pytorch.org/
https://medium.com/@iambibek/explanation-of-the-carlini-wagner-c-w-attack-algorithm-to-generate-adversarial-examples-6c1db8669fa2
https://www.tensorflow.org/tutorials/generative/adversarial_fgsm
https://towardsdatascience.com/understanding-backpropagation-algorithm-7bb3aa2f95fd
https://towardsdatascience.com/know-your-enemy-7f7c5038bdf3