Defending DNN Adversarial Attacks with Pruning and Logits Augmentation

Siyue Wang[1*], Xiao Wang[2*], Shaokai Ye[3], Pu Zhao[1] & Xue Lin[1]
1. Department of ECE, Northeastern University
2. Department of Computer Science, Boston university
3. Department of EECS, Syracuse University
* Equal Contribution

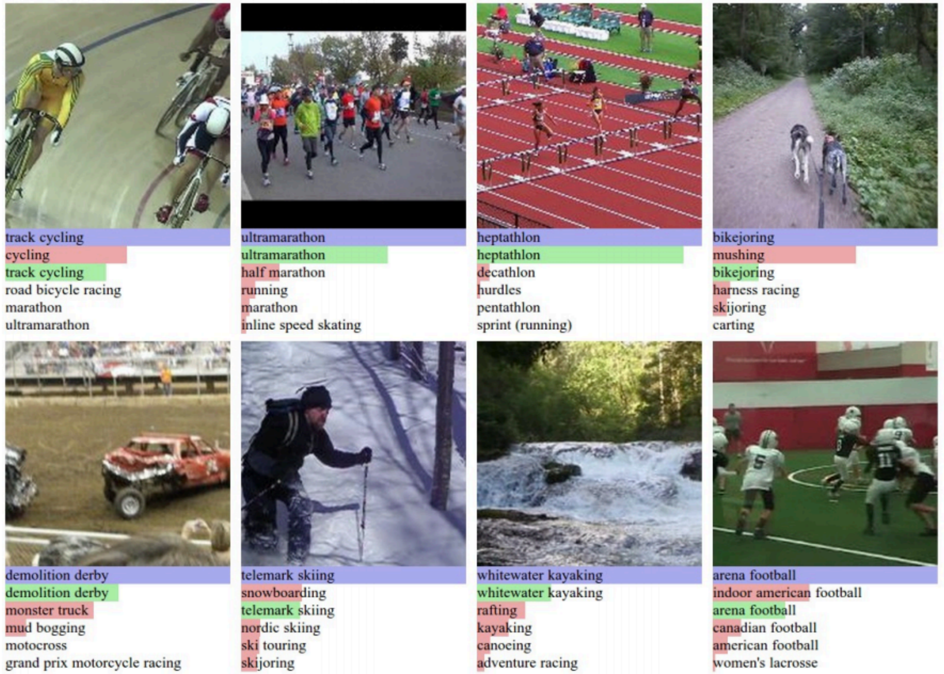
GlobalSip 2018

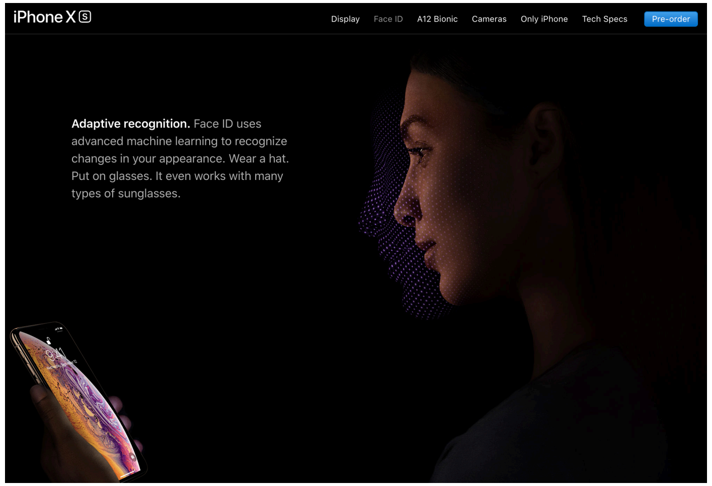
Outline

- Background
- Introduction to Adversarial Attacks
- Related work
- Our defense techniques
  - ❑Pruning + Logits Augmentation
- Conclusion

Background

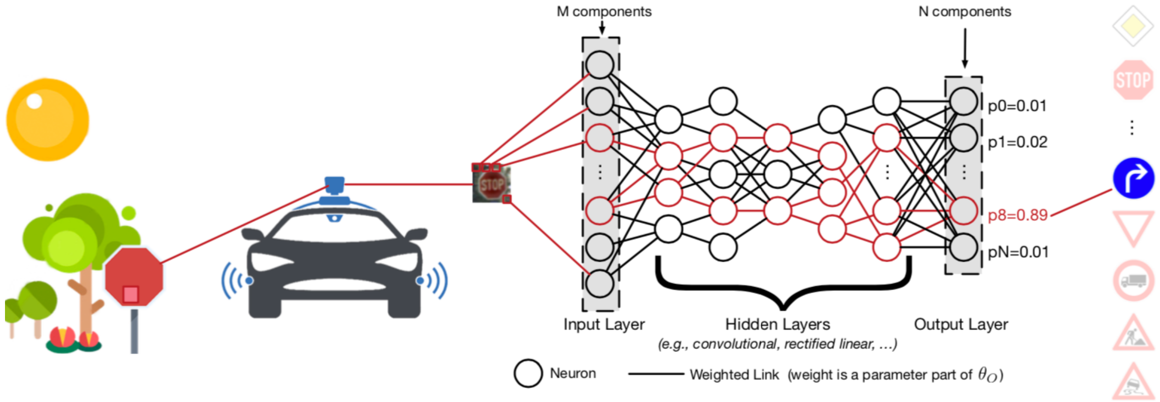
- Deep neural networks (DNNs) have been shown to be powerful models and perform extremely well on many complicated artificial intelligent tasks.
- Some are security critical like facial recognition and self-driving cars.



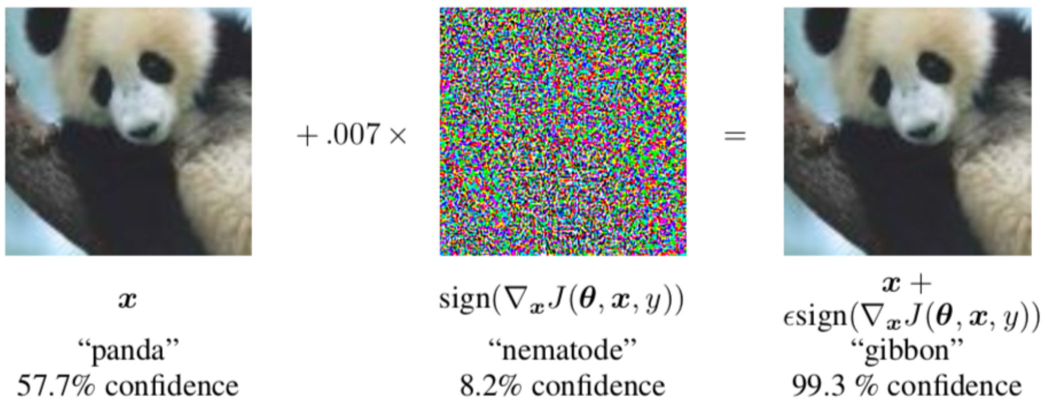


❖ Krizhevsky, A., Sutskever, I. and Hinton, G. E.. "ImageNet Classification with Deep Convolutional Neural Networks". NIPS 2012.
❖ Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". arXiv 2015.
❖ Shaoqing Ren, Kaiming He, Ross Girshick, & Jian Sun. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". NIPS 2015.
❖ https://selfdrivingcars.mit.edu/
❖ https://www.apple.com/iphone-xs/face-id/

Adversarial Attacks



- DNN models are vulnerable to adversarial attacks.
- Intentionally added imperceptible perturbations to DNN inputs can easily mislead the DNNs with extremely high confidence.



❖ J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," ICLR, 2015.

❖ N. Papernot, P. McDaniel, X. Wu, et al., "Distillation as a defense to adversarial perturbations against deep neural networks," IEEE Symposium on Security and Privacy (SP), 2016.

Problem Formulation

Suppose: a neural network has the model $F(\boldsymbol{x}) = \boldsymbol{y}$ and is an m-class classifier;
the neural network classifies input $\boldsymbol{x}$ according to the maximum probability, i.e., $C(\boldsymbol{x}) = \arg\max_i y_i$.

The initial problem of generating adversarial examples:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \mathscr{D}(\boldsymbol{x} - \boldsymbol{x_0}) \\ s.t. \quad & C(\boldsymbol{x}) = t \\ & \boldsymbol{x} \in [0,1]^n \end{aligned}$$

$\boldsymbol{x_0}$ is the original legal input;
$\boldsymbol{x}$ is the adversarial example;
$\mathscr{D}(\boldsymbol{x} - \boldsymbol{x_0})$ is a measure of the distortion $\delta = \boldsymbol{x} - \boldsymbol{x_0}$;
t is the target label to mislead the DNN.

L_p ***norms are the most commonly used measures in the literature***, defined as:

$$\|\boldsymbol{x} - \boldsymbol{x_0}\|_p = \left(\sum_{i=1}^{n} |\boldsymbol{x}_i - \boldsymbol{x}_{0i}|^p \right)^{\frac{1}{p}}$$

L_0 measures the number of mismatched elements;
L_1 measures the sum of the absolute values of the differences;
L_2 measures the standard Euclidean distance;
L_∞ measures the maximum difference between $\boldsymbol{x}$ and $\boldsymbol{x_0}$.

Adversarial attacks use L_0, L_1, L_2, and L_∞ norms to measure the distortions are namely L_0, L_1, L_2, and L_∞ attacks, respectively.

❖ Wang, Siyue et al. "Defensive Dropout for Hardening Deep Neural Networks under Adversarial Attacks.", ICCAD, 2018.

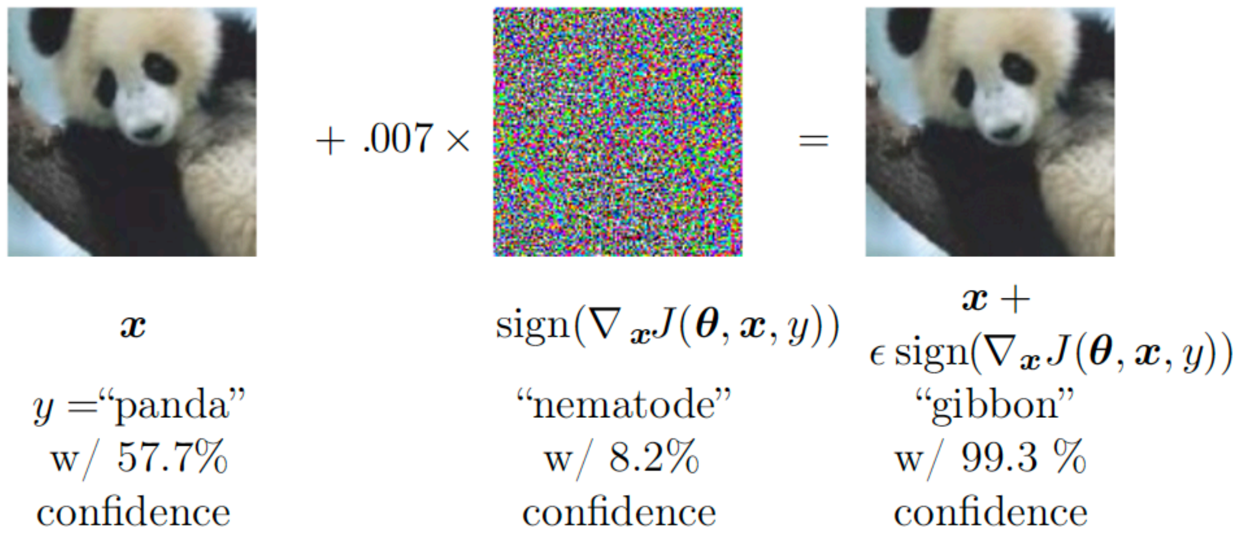
Fast Gradient Sign Method (FGSM)

- Adversarial examples are generated directly as

$$\boldsymbol{x} = \boldsymbol{x_0} - \epsilon \cdot \mathrm{sign}(\nabla(loss_{F,t}(\boldsymbol{x_0})))$$

ϵ is the magnitude of the added distortion.

- Designed to be fast, not optimal



❖ J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” ICLR, 2015.

Basic Iterative Method (BIM)

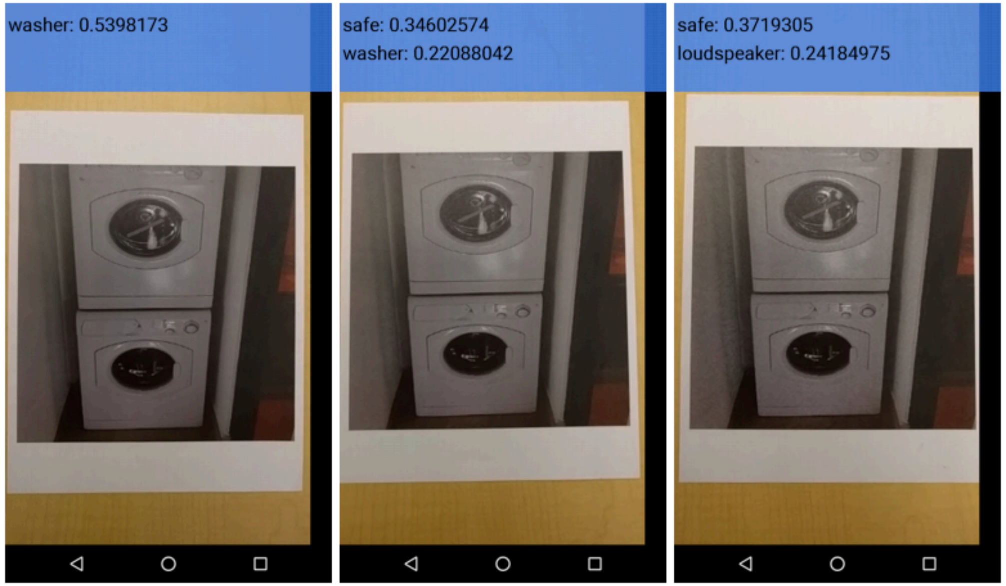
- BIM gives a further modify of FGSM. Instead of taking a single step , BIM takes multiple steps a. Given an initial setting:

$$x'_0 = x$$

- for each iteration, it calculates:

$$x'_i = x'_{i-1} - clip_\epsilon(\alpha sign(\nabla(loss_{F,y}(x'_{i-1}))))$$

- Notice that here BIM clips pixel values of intermediate results after each step to ensure that they are in an epsilon-bounded neighbourhood of the original image.



(b) Clean image (c) Adv. image, $\epsilon = 4$ (d) Adv. image, $\epsilon = 8$

❖ A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world,"

Carlini and Wagner Attack (CW)

- Solve an optimization problem :

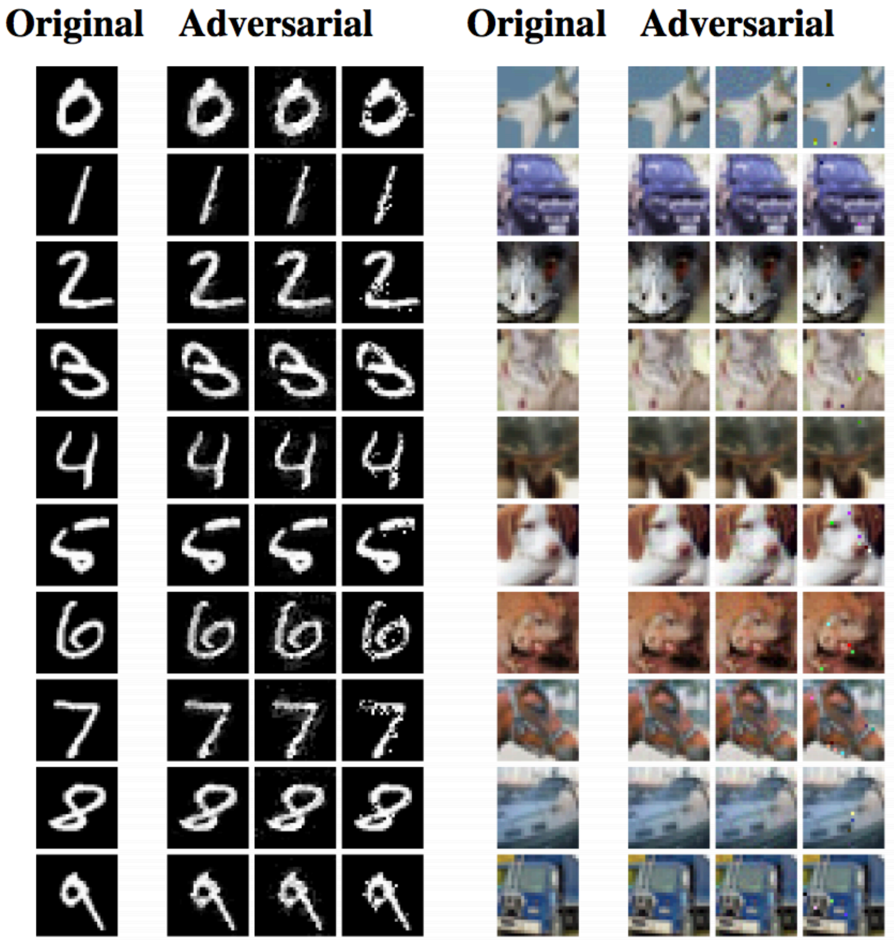
$$\begin{array}{ll} \text{minimize} & D(\boldsymbol{x} - \boldsymbol{x_0}) + c \cdot f(x) \\ \text{subject to} & x \in [0,1]^n \end{array}$$

$c > 0$ is a constant to be chosen;
objective function f has the following form:

$$f(x) = \max(\ \max\{Z(x)_i : i \neq t\} - Z(x)_t, -\kappa)$$

κ is a parameter that controls the confidence in attacks;
$Z(x)$ the input to the softmax, i.e., logits.

- L_0, L_2, and L_infinity attacks
- The strongest iterative attack in the literature



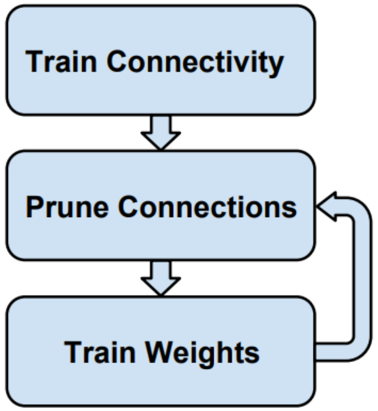
❖ N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," IEEE Symposium on Security and Privacy (SP), 2017.

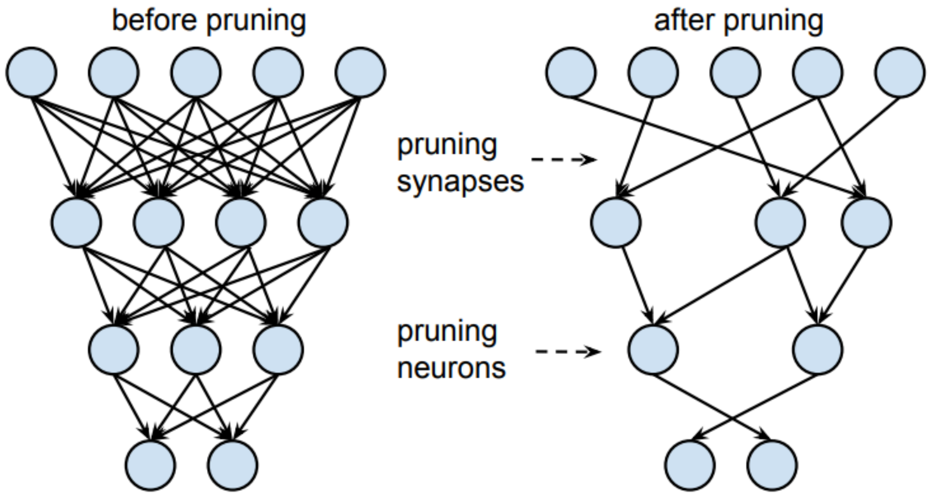
Motivations: Network Pruning

- DNN pruning method reduces the number of weights while preserving the accuracy of the compressed DNN models.

- We prune 10% nonzero weights for fully connected layers and 5% nonzero weights for convolutional layers.

- The network model can be compressed by 7 times after pruning.





❖ S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network,", NIPS 2015

Logits Augmentation

- To further improve the robustness of DNNs under adversarial attacks, we propose to use the logits augmentation on top of the pruning method.

- Inspired by the gradient inhibition method, which changes the weights in the last few layers as:

$$w = w + \tau * sign(w).$$

- In our logits augmentation, we modify the weights in the last fully-connected layer by

$$w = \tau \times w$$

❖ Q. Liu, T. Liu, Z. Liu, Y. Wang, Y. Jin, and W. Wen, "Security analysis and enhancement of model compressed deep learning systems under adversarial attacks," ASP-DAC, 2018.

Defense Models

- **M0** and **C0**: unprotected neural network models that achieve near state-of-the-art accuracy, i.e., 99.4% and 80%, respectively, on MNIST and CIFAR-10.

- **M1** and **C1**: defense level one exploits only the pruning method.

- **M2** and **C2**: defense level two exploits both pruning and logits augmentation as defense.

❖ N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," IEEE Symposium on Security and Privacy (SP), 2017.

Experimental Results

- Results using M0, M1 and M2 on MNIST

TABLE I: Adversarial attack successful rate (and distortion) of the unprotected model M0, Level One model M1, and Level Two model M2 under four attacks (untargeted FGSM, targeted FGSM, targeted BIM, and C&W) using MNIST dataset.

Attack Method	Untargeted FGSM			Targeted FGSM			Targeted BIM			C&W
Parameters	ε= 0.1	ε = 0.15	ε = 0.25	ε = 0.1	ε= 0.15	ε = 0.25	ε= 0.1	ε = 0.15	ε = 0.25	iter = 100
M0	9.0% (2.19)	17.0% (3.28)	45.6% (5.45)	1.97% (2.17)	4.52% (3.25)	12.0% (5.39)	3.89% (2.11)	14.81% (3.11)	39.64% (5.28)	99.6% (2.03)
M1	7.4% (2.16)	8.7% (3.25)	20.2% (5.38)	1.17% (2.15)	1.68% (3.22)	4.04% (5.35)	3.14% (2.14)	9.9% (3.13)	31.26% (5.07)	96.97% (2.28)
M2	1.1% (2.28)	1.1% (3.41)	1.1% (5.65)	1.04% (2.15)	1.5% (3.22)	3.87% (5.35)	2.71% (2.15)	7.9% (3.1)	21.12% (5.1)	95.93% (2.5)

The experiment is evaluated on 1000 source samples from MNIST. We set the search step for line search in C&W as 10.

Experimental Results

- Results using M0, M1 and M2 on CIFAR-10

TABLE II: Adversarial attack successful rates (and distortion) of the unprotected model C0, Level One model C1, and Level Two model C2 under four attacks using CIFAR-10 dataset.

Attack Method	Untargeted FGSM			Targeted FGSM			Targeted BIM			C&W
Parameters	ε= 0.1	ε = 0.15	ε = 0.25	ε = 0.1	ε= 0.15	ε = 0.25	ε= 0.1	ε = 0.15	ε = 0.25	iter = 100
C0	84.6% (5.43)	86.3% (8.05)	87.1% (13.0)	17.71% (5.43)	14.78% (8.05)	11.49% (13.0)	63.59% (4.48)	65.83% (6.66)	65.73% (10.8)	99.54% (2.06)
C1	70.3% (5.43)	75.3% (8.05)	80.9% (13.0)	11.2% (5.42)	10.5% (8.05)	10.1% (13.03)	25.3% (4.47)	23.8% (6.64)	19.3% (10.8)	85.0% (3.55)
C2	24.6% (1.42)	24.5% (2.11)	25% (3.41)	11.12% (5.33)	11.25% (7.91)	11.16% (12.8)	43.41% (4.43)	44.9% (6.5)	41.2% (10.7)	83.9% (4.31)

The experiment is evaluated on 1000 source samples from CIFAR-10. We set the search step for line search in C&W as 10.

Conclusion

- Enhance the robustness of DNNs by using pruning method and logits augmentation
- We achieve DNN model compression by **7** times while maintaining the test accuracy
- Our method can effectively defend against both targeted and untargeted FGSM and BIM attacks under grey-box attack assumption

Thank you!