Homework Assignment 2(Programming Category)

Student Name: Pei-Lun Tai

Student Session: cs6220-A

Github link: https://github.com/victai/GTAttackPod

Problem 1. Understanding new security vulnerabilities in deep learning.

Option 1: Understanding security vulnerabilities of using a well-trained DNN models for image classification

Input analysis

1. Provide a summary of your pre-trained models and datasets. For each dataset, provide 10 example inputs under five different classes, 2 per class.
    a. Model (ResNet)

    ResNet is a well-known convolution neural network that contains **Residual blocks**. Residual block contains an identity shortcut that allows tensors to skip several layers. This architecture can effectively deal with gradient vanishing problem, and thus allows us to build very deep neural networks, which is often considered to be more powerful.

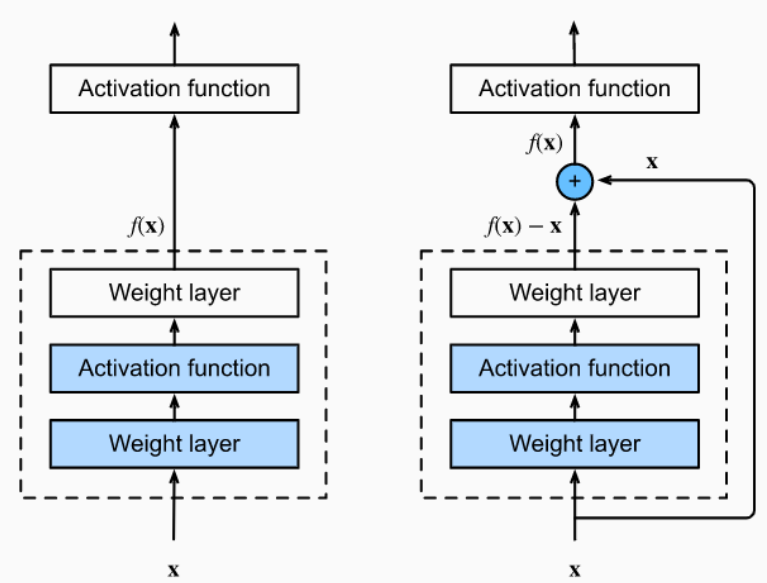


Fig. 7.6.2 A regular block (left) and a residual block (right).

reference: https://d2l.ai/chapter_convolutional-modern/resnet.html

b. Dataset (Cifar10)

The CIFAR-10 dataset consists of 60000 32x32 color images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images.

The 10 classes:

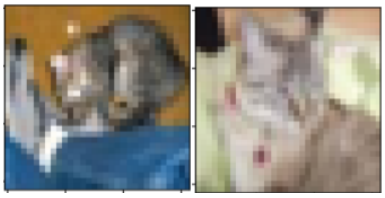
airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck

Examples:

- airplane

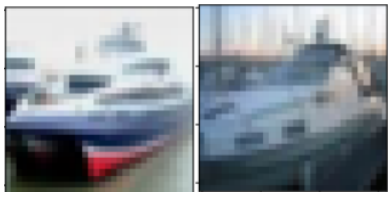
- bird

- cat



- frog

- ship



2. Provide a summary of the two attack algorithms of your choice.
    a. Fast Gradient Sign Method (FGSM)

        The intuition of FGSM is similar to that of Gradient Descent. However, rather than updating the weight to make the output loss smaller, FGSM updates the input to make the output loss larger.

        The calculation of the updated parameters are the same, except that

        - Gradient calculation
            - FGSM calculates the gradient over the input

- Gradient Descent calculates the gradient over the weights

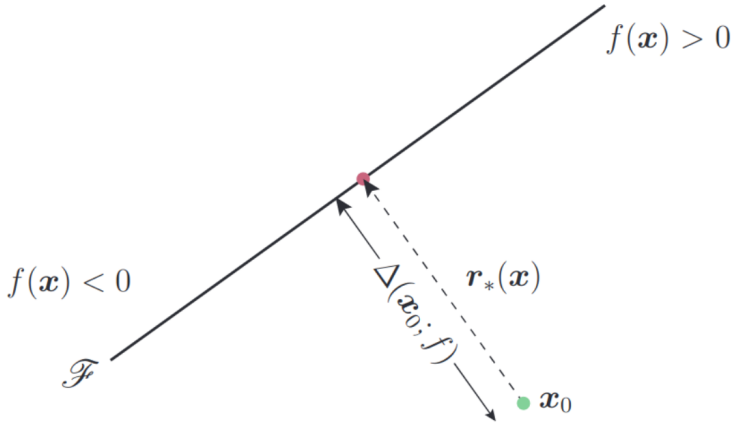
- Update direction
  - FGSM adds the calculated gradient to the input
  - Gradient Descent subtracts the calculated gradient to the weights

b. DeepFool

The essence of DeepFool algorithm is to find the minimal perturbation that makes the classifier goes wrong. Therefore, the algorithm is

1. Find a classification boundary that is closest to the current input.
2. Take a step towards the orthogonal direction with respect to the closest boundary.
3. Check if the prediction is still the same as the original prediction
4. Repeat step 1~3 until the prediction changes.

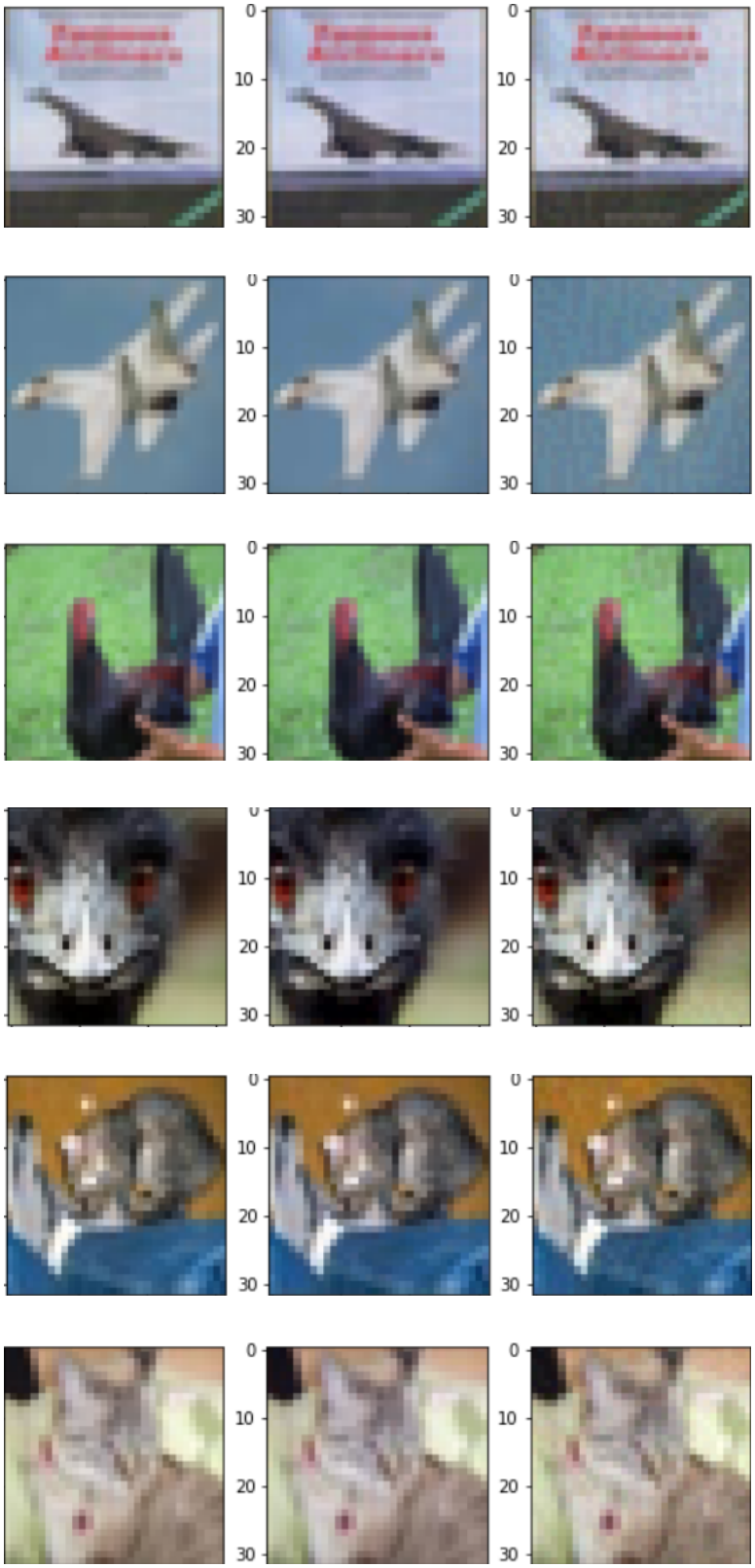
Reference: https://towardsdatascience.com/deepfool-a-simple-and-accurate-method-to-fool-deep-neural-networks-17e0d0910ac0

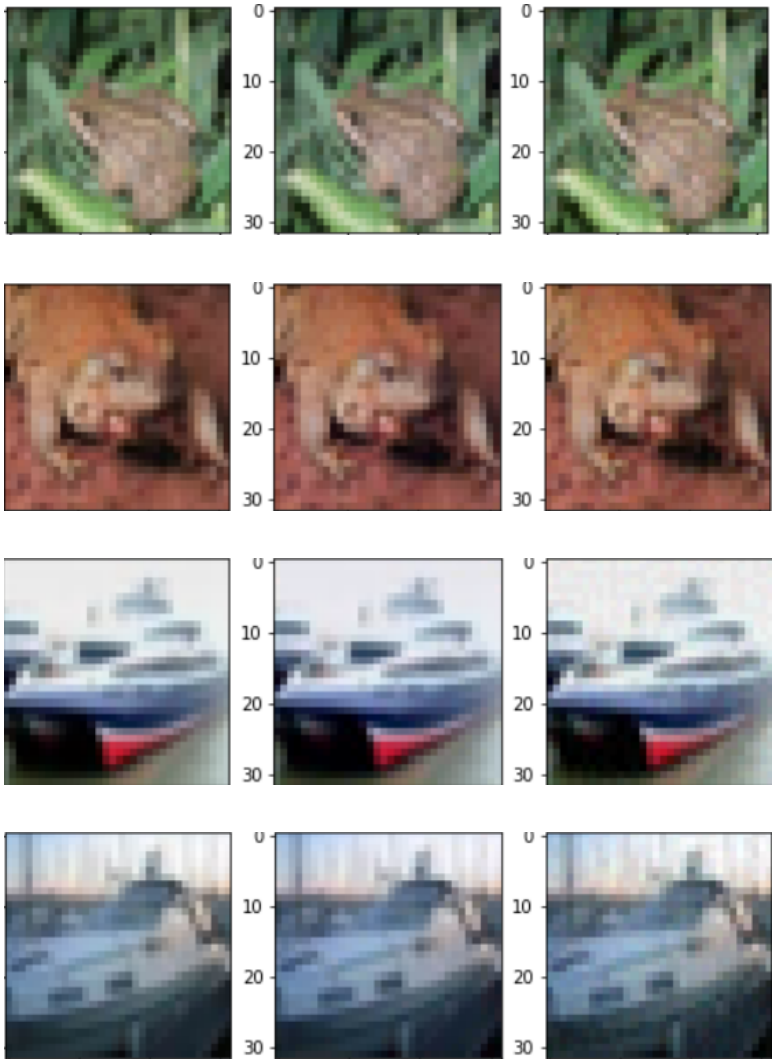


3. Provide the attack examples you generated for the 10 examples you listed in 1).

The images below are FGSM attack results. The DeepFool results are shown in the fifth section in Output Analysis.

From left to right are Original, DenseNet40 result, ResNet110 result, respectively


0
10
20
30


0
10
20
30

Output analysis

1. Include a note that you have provided your answer to the questionnaire posted as quiz 1 on Canvas (under quizzes), refer requirement 2) above.

   I have finished the quiz 1 questionnaire.

2. Provide test accuracy measurement and average test time per example of the two well trained models under no attack.

```
Prediction time: 128.3379340171814; 10000 examples
Prediction time per example: 0.01283379340171814
Test accuracy on benign examples 94.84%
Mean confidence on ground truth classes 92.15%
Selected 100 examples.
Test accuracy on selected benign examples 100.00%
Mean confidence on ground truth classes, selected 95.55%
```

Cifar10-DenseNet40

```
Evaluating the target model...
Prediction time: 11.998623132705688; 10000 examples
Prediction time per example: 0.0011998623132705689
Test accuracy on benign examples 75.77%
Mean confidence on ground truth classes 73.92%
Selected 100 examples.
Test accuracy on selected benign examples 100.00%
Mean confidence on ground truth classes, selected 95.19%
```

Cifar10-ResNet20

```
Prediction time: 72.1700484752655; 10000 examples
Prediction time per example: 0.007217004847526551
Test accuracy on benign examples 92.08%
Mean confidence on ground truth classes 87.71%
Selected 100 examples.
Test accuracy on selected benign examples 100.00%
Mean confidence on ground truth classes, selected 93.21%
```

Cifar10-ResNet110

3. Compare the two models under attacks with the two models under no attack on test accuracy and time.

```
---Statistics of DeepFool Attack (2.180704 seconds per sample)
Success rate: 100.00%, Misclassification rate: 100.00%, Mean confidence: 85.81%
Li dist: 0.0275, L2 dist: 0.2307, L0 dist: 99.1%
```

Cifar10-DenseNet40-attack

```
---Statistics of DeepFool Attack (3.245221 seconds per sample)
Success rate: 74.00%, Misclassification rate: 74.00%, Mean confidence: 87.92%
Li dist: 0.2206, L2 dist: 2.6155, L0 dist: 99.9%
```

Cifar10-ResNet20-attack

```
---Statistics of DeepFool Attack (5.353405 seconds per sample)
Success rate: 99.00%, Misclassification rate: 99.00%, Mean confidence: 85.36%
Li dist: 0.1219, L2 dist: 1.1628, L0 dist: 99.5%
```

Cifar10-ResNet110-attack

4. Plot the results from 1) + 2) into a table for easy comparison.

   Test Results

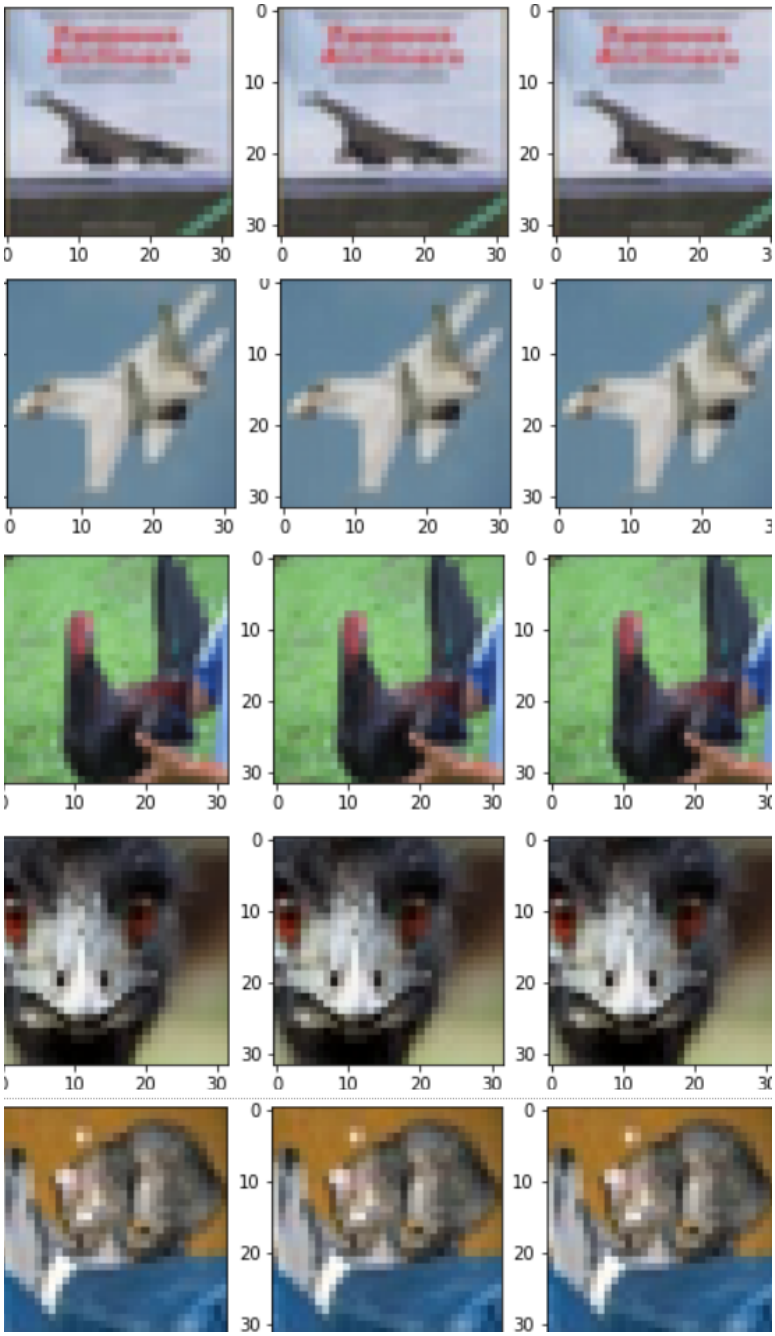
Aa Name	Prediction time	Prediction Acc	Prediction time after attack	Success rate
DenseNet40	0.0128s/img	94.84%	2.181s/img	100%
ResNet20	0.001s/img	75.77%	3.245s/img	74%
ResNet110	0.007s/img	92.08%	5.353s/img	99%

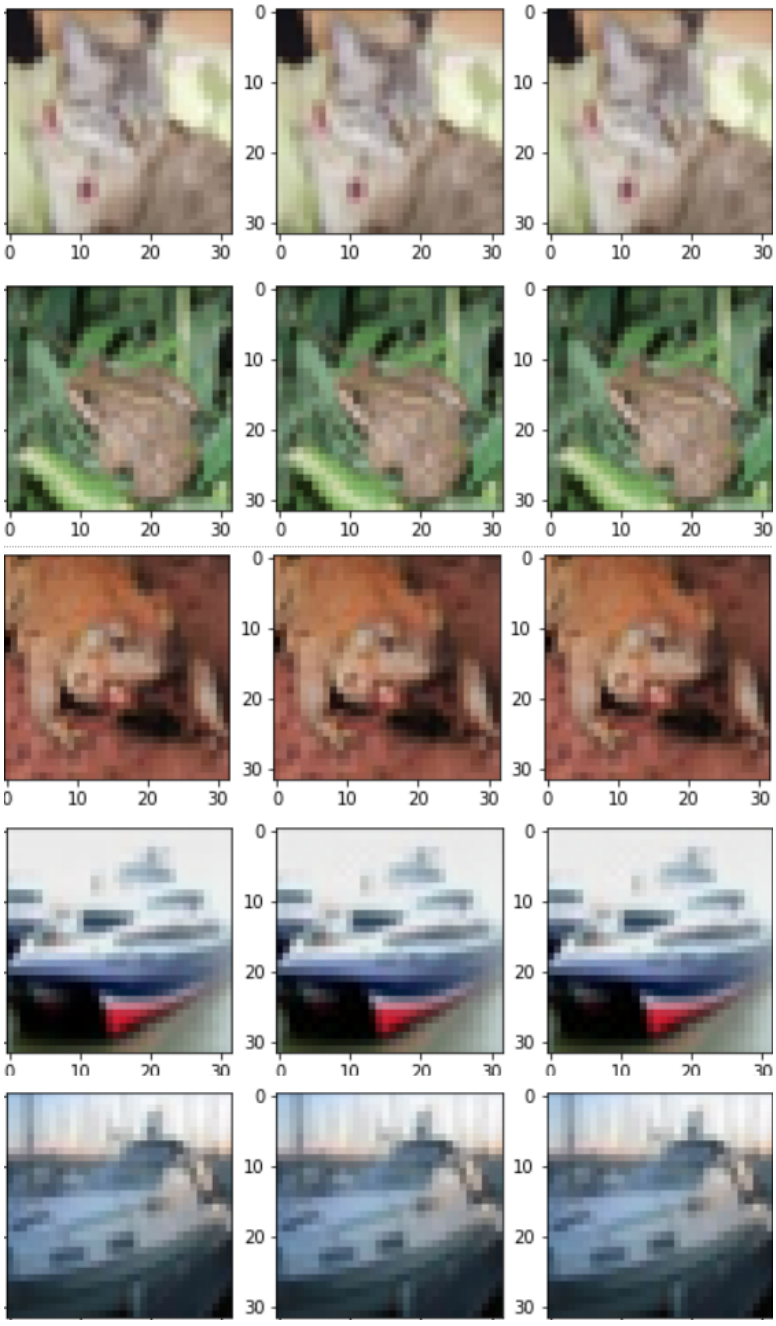
5. Provide 10 visual examples for both datasets to illustrate the adverse effects of the attacks: clean input image, the amount of perturbation added by attack algorithm via iterative learning (ideally show visual results under 2-3 iterations), and the final input image under attack to compare with the clean image without adversarial perturbation.

   The images below are DeepFool attack results.

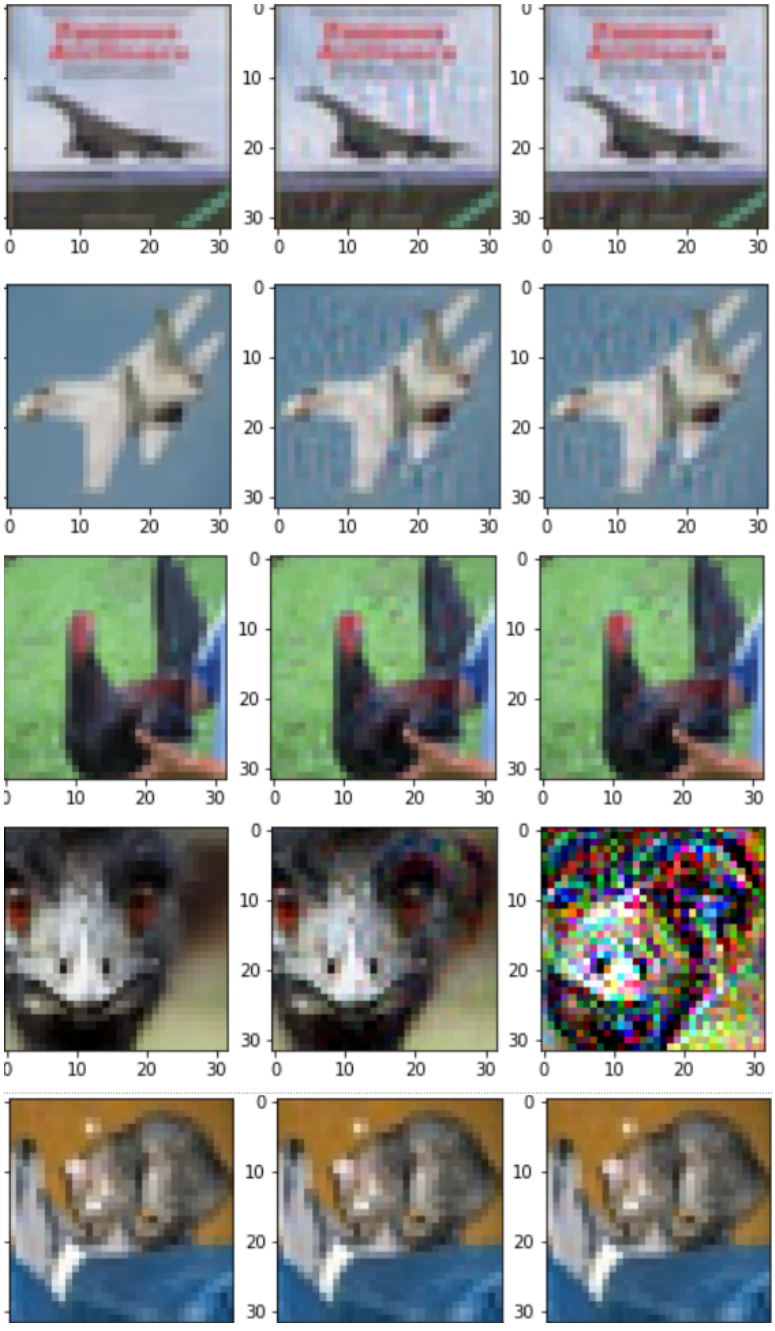
   From left to right are Original, Attack result after 3 iterations, Final Attack result, respectively

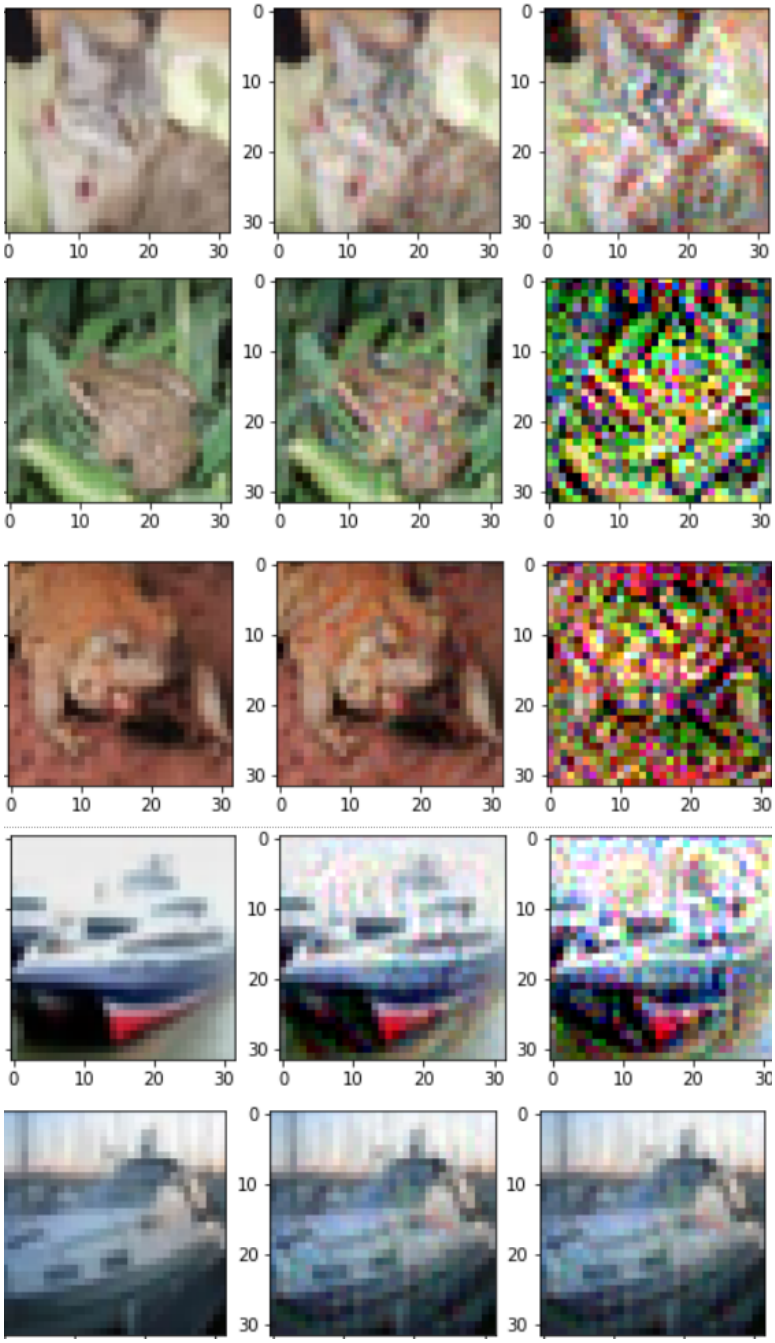
- **DenseNet40**



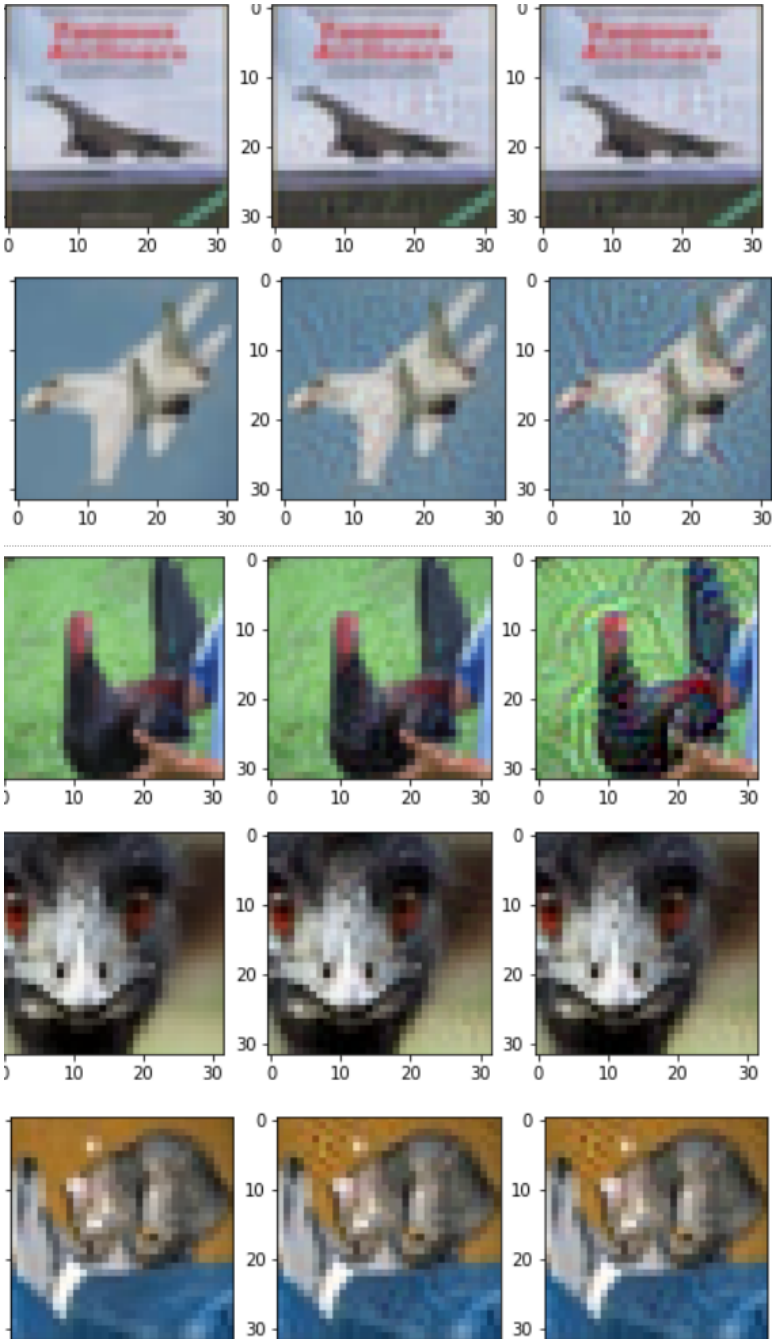


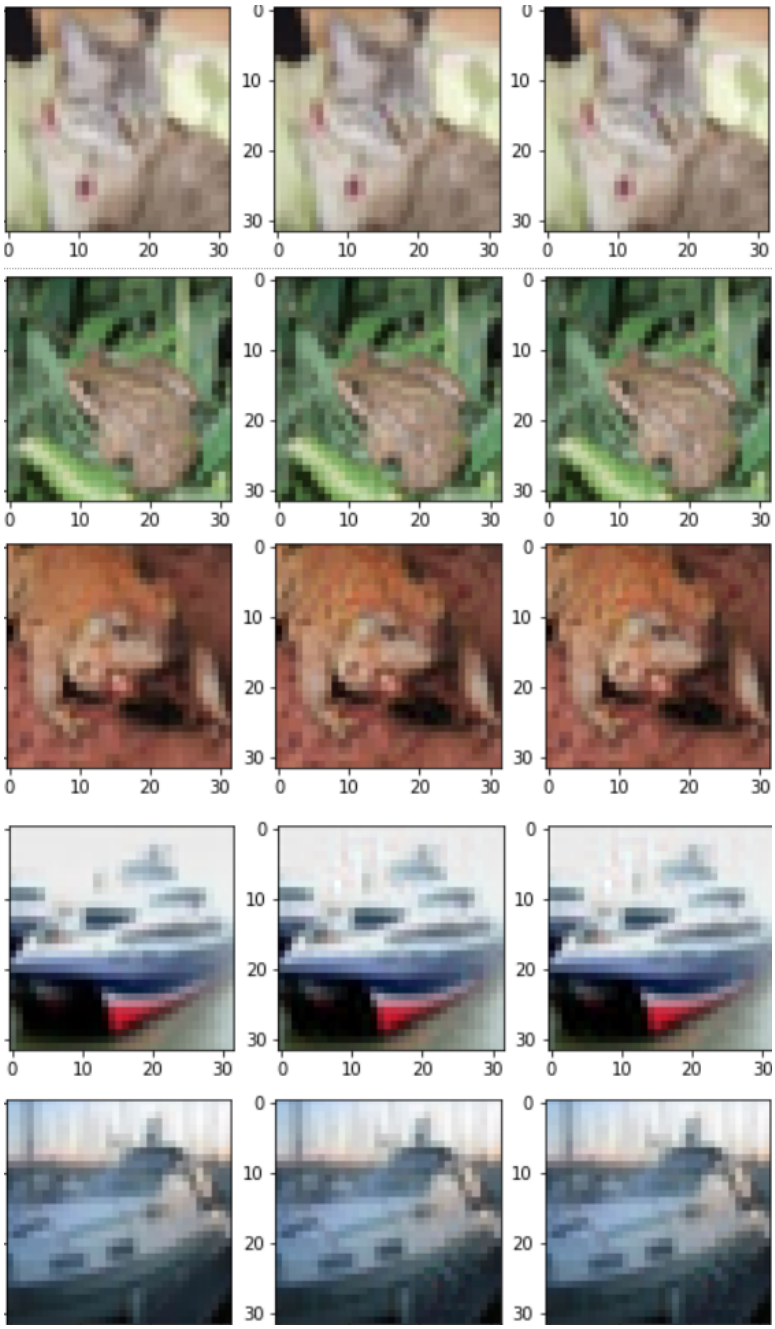
- **ResNet20**





- **ResNet110**





6. Include a note that you have provided your answer to the questionnaire posted as quiz 2 on Canvas (under quizzes), refer requirement 8) above. This questionnaire summarizes your learning experience by combining lecture, the API visual learning and hand-on attack code programming experience.

   I have finished the quiz 2 questionnaire.

7. Provide additional analysis and comments on the learning experience with this problem (optional).
    a. Overall I think it's a comprehensive homework. There are a lot of attack methods to choose and the Github repository is pretty well organized. However I think it's a little bit too open-ended, and I think it's a little difficult to really understand the goal and what to do by simply reading through the spec. I think it would be of great help if TAs can go through the homework once released.
    b. Another comment is about the Tensorflow version. Since the given code is well structured, we are basically restricted to use Keras, and the code base uses Tensorflow 1.0. However there is a huge difference between Tensorflow 1.0 and Tensorflow 2.0. They are mostly not compatible, and a lot of resources I found on the web is based on TF2.0. I have had a hard time trying to find pre-trained TF1.0 models with different backbone algorithms and I believe plenty of students have met the same problems (from the discussions on Piazza). I chose to train it on my own in the end, which I did not expect by simply reading through the spec. It would be nice if the projects use up to date libraries.

Requirement 6

(a) Adverse effect on different depths of CNNs. Choose another two or more pre-trained models on CIFAR-10 with very different DNN layers in terms of depths (ranging from 10s to 100s for example), using the attack algorithm to examine whether adversarial examples are sensitive to deeper NNs. Report your results.

Dataset: CIFAR-10

Models:

- ResNet20
- ResNet110

I chose ResNet20 and ResNet110 for this experiments. The testing results and output images are shown in the **Output Analysis.**

Unfortunately, I could not fit ResNet20 well on Cifar10, only achieves 76% accuracy, while ResNet110 achieves 92% accuracy. It might be a variation factor in this experiment.

Through this experiment. I found that ResNet20 is more robust than ResNet110 (74% vs 99% attack success rate). I find this result pretty reasonable, as complex models tend to have higher variance, which may perform well on certain datasets, but performs worse on others. With much higher accuracy, the attack success rate is also extremely high since some noise on the inputs can cause ResNet110 predicts wrongly.

The result is also in line with DenseNet40, with even higher accuracy of 94%, and 100% attack success rate.

The result also reflects on the output images. The perturbed images has lower artifacts on models with higher accuracy. That said, by applying slight noises on the input, although not perceivable by human eyes, can fool the neural network.

(b) Test transferability of your generated adversarial examples on a pre-trained CIFAR-10 (say trained by DenseNet 40) or MNIST (say trained by CNN-7) model, ideally trained using different backbone algorithms. Using 100 test adversarial examples to produce average transferability of your attack algorithm to this second CIFAR-10 or MNIST model.

Dataset: CIFAR-10

Models:

- DenseNet40
- ResNet110

```
Success rate: 10.00%, Misclassification rate: 10.00%, Mean confidence: 76.07%
Li dist: 0.0224, L2 dist: 0.1877, L0 dist: 100.0%
```

Samples attacked against DenseNet40, tested on ResNet110

```
Success rate: 14.00%, Misclassification rate: 14.00%, Mean confidence: 87.26%
Li dist: 0.1930, L2 dist: 1.9453, L0 dist: 99.8%
```

Samples attacked against ResNet110, tested on DenseNet40

The experiment shows that ResNet110 is more robust than DenseNet40, which is the same as induced by other experiments.