



МИНОБРНАУКИ РОССИИ

**Федеральное государственное бюджетное образовательное
учреждение**

высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт комплексной безопасности и специального приборостроения

Отчет по лабораторной работе №1

по дисциплине: «Анализ защищенности систем искусственного

интеллекта»

Выполнил:

Студент группы БМО-02-23

ФИО: Дурягин М.Р.

Москва 2024

1. Клонировать репозиторий и загружать нужные библиотеки.
(Перехожу монтирую гугл диск, перехожу в рабочую папку)

```
[1] from google.colab import drive
drive.mount('/content/drive')

# Переход в рабочую директорию

import os
os.chdir('/content/drive/MyDrive/EEL6812_DeepFool_Project-main') # Укажите путь к вашей рабочей директории

Mounted at /content/drive

[2] # Скачаем репозиторий с github
!git clone https://github.com/ewatson2/EEL6812_DeepFool_Project.git

fatal: destination path 'EEL6812_DeepFool_Project' already exists and is not an empty directory.

[3] # Перейдём в директорию /content/EEL6812_DeepFool_Project
%cd /content/EEL6812_DeepFool_Project

[Errno 2] No such file or directory: '/content/EEL6812_DeepFool_Project'
/content

# Выполним импорт библиотек
import numpy as np
import json, torch
import os
from torch.utils.data import DataLoader, random_split
from torchvision import datasets, models
from torchvision.transforms import transforms

[5] # Выполним импорт вспомогательных библиотек из локальных файлов проекта
from models.project_models import FC_500_150, LeNet_CIFAR, LeNet_MNIST, Net
from utils.project_utils import get_clip_bounds, evaluate_attack, display_attack
```

2. Устанавливаем случайное число (Мой вариант) и выбираем устройство выполнения.

```
[6] # Установим случайное случайное значение в виде переменной (сюда необходимо
# записать порядковый номер в списке, в моём случае это 4)
rand_seed = 4

# Установим указанное значение для np.random.seed и torch.manual_seed
np.random.seed(rand_seed)
torch.manual_seed(rand_seed)

<torch._C.Generator at 0x7a1b92179730>

[8] # Используем в качестве устройства видеокарту (T4 GPU)
use_cuda = torch.cuda.is_available()
device = torch.device('cuda' if use_cuda else 'cpu')
```

3. Загружаем dataset «MNIST».

```
# Загрузим датасет MNIST со следующими параметрами
mnist_mean = 0.5
mnist_std = 0.5
mnist_dim = 28

mnist_min, mnist_max = get_clip_bounds(mnist_mean, mnist_std, mnist_dim)
mnist_min = mnist_min.to(device)
mnist_max = mnist_max.to(device)

mnist_tf = transforms.Compose([transforms.ToTensor(),
                              transforms.Normalize(mean=mnist_mean, std=mnist_std)])

mnist_tf_train = transforms.Compose([transforms.RandomHorizontalFlip(),
                                     transforms.ToTensor(),
                                     transforms.Normalize(mean=mnist_mean, std=mnist_std)])

mnist_tf_inv = transforms.Compose([transforms.Normalize(mean=0.0, std=np.divide(1.0, mnist_std)),
                                   transforms.Normalize(mean=np.multiply(-1.0, mnist_std), std=1.0)])

mnist_temp = datasets.MNIST(root='datasets/mnist', train=True, download=True, transform=mnist_tf_train)
mnist_train, mnist_val = random_split(mnist_temp, [50000, 10000])
mnist_test = datasets.MNIST(root='datasets/mnist', train=False, download=True, transform=mnist_tf)
```

4. Загружаем dataset «CIFAR-10».

```
# Загрузим датасет CIFAR-10 со следующими параметрами
cifar_mean = [0.491, 0.482, 0.447]
cifar_std = [0.202, 0.199, 0.201]
cifar_dim = 32

cifar_min, cifar_max = get_clip_bounds(cifar_mean, cifar_std, cifar_dim)
cifar_min = cifar_min.to(device)
cifar_max = cifar_max.to(device)

cifar_tf = transforms.Compose([transforms.ToTensor(),
                              transforms.Normalize(mean=cifar_mean, std=cifar_std)])

cifar_tf_train = transforms.Compose([transforms.RandomCrop(size=cifar_dim, padding=4),
                                     transforms.RandomHorizontalFlip(),
                                     transforms.ToTensor(),
                                     transforms.Normalize(mean=cifar_mean, std=cifar_std)])

cifar_tf_inv = transforms.Compose([transforms.Normalize(mean=[0.0, 0.0, 0.0], std=np.divide(1.0, cifar_std)),
                                   transforms.Normalize(mean=np.multiply(-1.0, cifar_mean), std=[1.0, 1.0, 1.0])])

cifar_temp = datasets.CIFAR10(root='datasets/cifar-10', train=True, download=True, transform=cifar_tf_train)
cifar_train, cifar_val = random_split(cifar_temp, [40000, 10000])
cifar_test = datasets.CIFAR10(root='datasets/cifar-10', train=False, download=True, transform=cifar_tf)

cifar_classes = ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']

Files already downloaded and verified
Files already downloaded and verified
```

5. Выполним настройку и загрузку DataLoader.

```
# Выполним настройку и загрузку DataLoader
batch_size = 64
workers = 4

mnist_loader_train = DataLoader(mnist_train, batch_size=batch_size, shuffle=True, num_workers=workers)
mnist_loader_val = DataLoader(mnist_val, batch_size=batch_size, shuffle=False, num_workers=workers)
mnist_loader_test = DataLoader(mnist_test, batch_size=batch_size, shuffle=False, num_workers=workers)

cifar_loader_train = DataLoader(cifar_train, batch_size=batch_size, shuffle=True, num_workers=workers)
cifar_loader_val = DataLoader(cifar_val, batch_size=batch_size, shuffle=False, num_workers=workers)
cifar_loader_test = DataLoader(cifar_test, batch_size=batch_size, shuffle=False, num_workers=workers)
```

6. Зададим параметры на модель.

```
# Зададим параметры deep_args
deep_batch_size = 64
deep_num_classes = 10
deep_overshoot = 0.02
deep_max_iters = 100

deep_args = [deep_batch_size, deep_num_classes, deep_overshoot, deep_max_iters]
```

7. Загрузим и оценим стойкость модели.

```
# Загрузим и оценим стойкость модели Network-In-Network Model к FGSM и DeepFool атакам на основе датасета CIFAR-10
fgsm_eps = 0.2
model = Net().to(device)
model.load_state_dict(torch.load('weights/clean/cifar_nin.pth', map_location=torch.device('cpu')))

evaluate_attack('cifar_nin_fgsm.csv', 'results', device, model, cifar_loader_test, cifar_min, cifar_max, fgsm_eps, is_fgsm=True)
print('')
evaluate_attack('cifar_nin_deepfool.csv', 'results', device, model, cifar_loader_test, cifar_min, cifar_max, deep_args, is_fgsm=False)

if device.type == 'cuda': torch.cuda.empty_cache()

<ipython-input-13-a5434b034ce5>:4: FutureWarning: You are using 'torch.load' with 'weights_only=False' (the current default value), which uses the d
model.load_state_dict(torch.load('weights/clean/cifar_nin.pth', map_location=torch.device('cpu')))
FGSM Test Error : 81.29%
FGSM Robustness : 1.77e-01
FGSM Time (All Images) : 0.67 s
FGSM Time (Per Image) : 67.07 us

DeepFool Test Error : 93.76%
DeepFool Robustness : 2.12e-02
DeepFool Time (All Images) : 185.12 s
DeepFool Time (Per Image) : 18.51 ms

[14] # Загрузим и оценим стойкость модели LeNet к FGSM и DeepFool атакам на основе датасета CIFAR-10
fgsm_eps = 0.1
model = LeNet_CIFAR().to(device)
model.load_state_dict(torch.load('weights/clean/cifar_lenet.pth', map_location=torch.device('cpu')))

evaluate_attack('cifar_lenet_fgsm.csv', 'results', device, model, cifar_loader_test, cifar_min, cifar_max, fgsm_eps, is_fgsm=True)
print('')
evaluate_attack('cifar_lenet_deepfool.csv', 'results', device, model, cifar_loader_test, cifar_min, cifar_max, deep_args, is_fgsm=False)

if device.type == 'cuda': torch.cuda.empty_cache()

<ipython-input-14-08045a679969>:4: FutureWarning: You are using 'torch.load' with 'weights_only=False' (the current default value), which uses the d
model.load_state_dict(torch.load('weights/clean/cifar_lenet.pth', map_location=torch.device('cpu')))
FGSM Test Error : 91.71%
FGSM Robustness : 8.90e-02
FGSM Time (All Images) : 0.40 s
FGSM Time (Per Image) : 40.08 us

DeepFool Test Error : 87.81%
DeepFool Robustness : 1.78e-02
DeepFool Time (All Images) : 73.27 s
DeepFool Time (Per Image) : 7.33 ms
```

8. Выполним оценку атакующих примеров для сетей.

```
# Выполним оценку атакующих примеров для сетей:

# LeNet на датасете MNIST
print("Running attacks for: LeNet on MNIST dataset")
fgsm_eps = 0.6
model = LeNet_MNIST().to(device)
model.load_state_dict(torch.load('weights/clean/mnist_lenet.pth'))
display_attack(device, model, mnist_test, mnist_tf_inv, mnist_min, mnist_max, fgsm_eps, deep_args,
               has_labels=False, l2_norm=True, pert_scale=1.0, fig_rows=2, fig_width=25, fig_height=11)

if device.type == 'cuda': torch.cuda.empty_cache()

# FCNet на датасете MNIST
print("Running attacks for: FCNet on MNIST dataset")
fgsm_eps = 0.2
model = FC_500_150().to(device)
model.load_state_dict(torch.load('weights/clean/mnist_fc.pth'))
display_attack(device, model, mnist_test, mnist_tf_inv, mnist_min, mnist_max, fgsm_eps, deep_args,
               has_labels=False, l2_norm=True, pert_scale=1.0, fig_rows=2, fig_width=25, fig_height=11)

if device.type == 'cuda': torch.cuda.empty_cache()

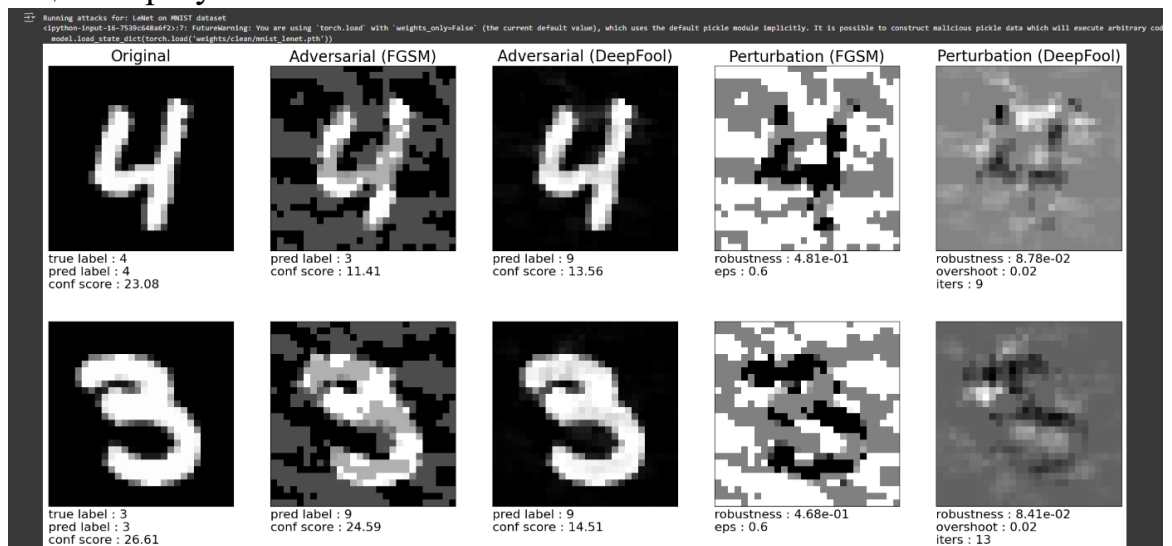
# Network-in-Network на датасете CIFAR-10
print("Running attacks for: Network-in-Network on CIFAR-10 dataset")
fgsm_eps = 0.2
model = Net().to(device)
model.load_state_dict(torch.load('weights/clean/cifar_nin.pth'))
display_attack(device, model, cifar_test, cifar_tf_inv, cifar_min, cifar_max, fgsm_eps, deep_args,
               has_labels=False, l2_norm=True, pert_scale=1.0, fig_rows=2, fig_width=25, fig_height=11, label_map=cifar_classes)

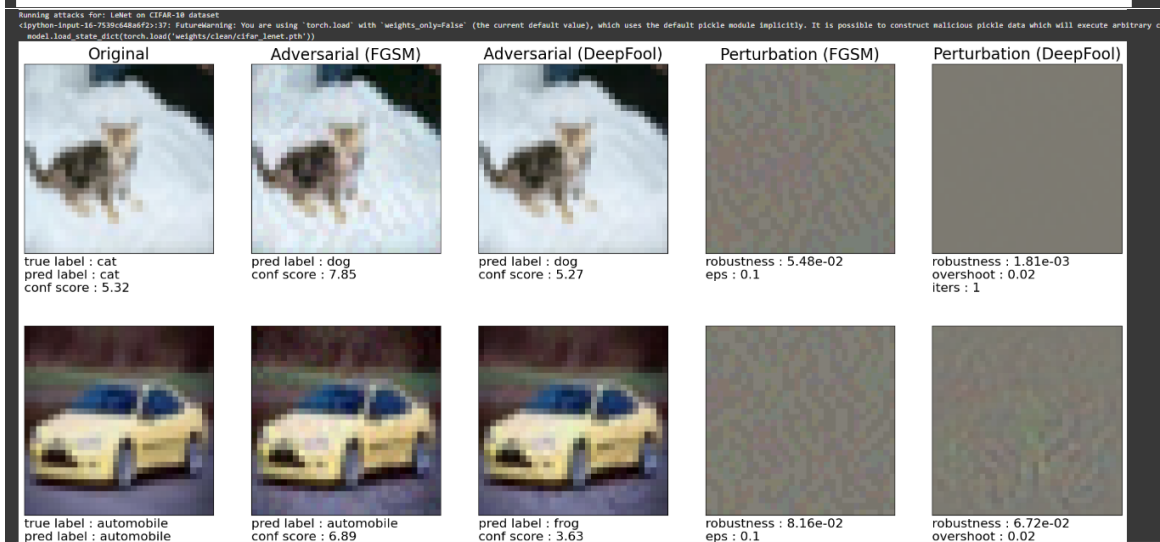
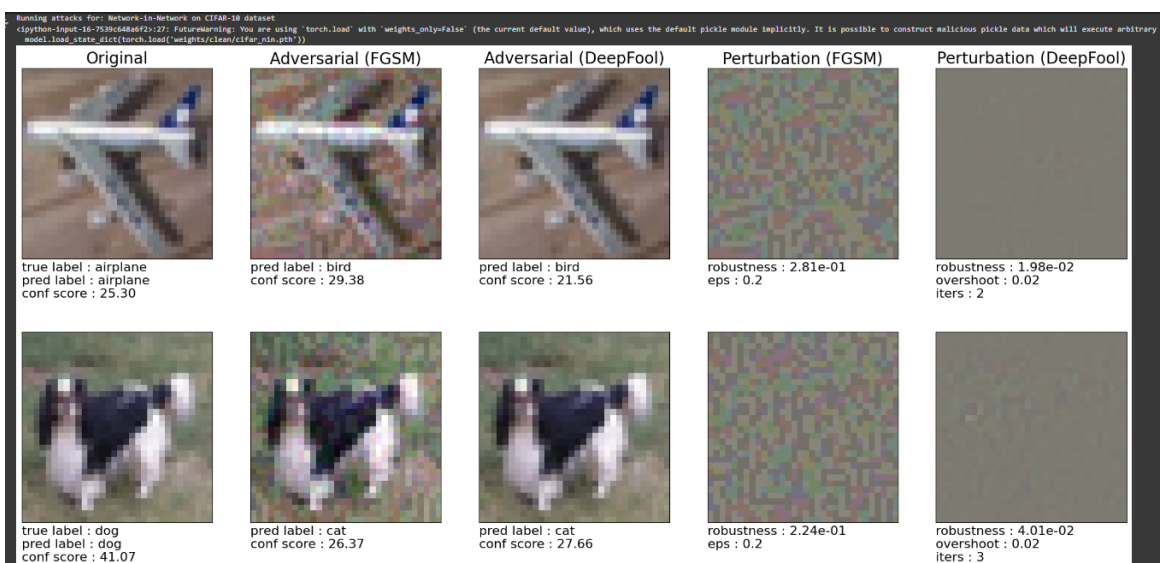
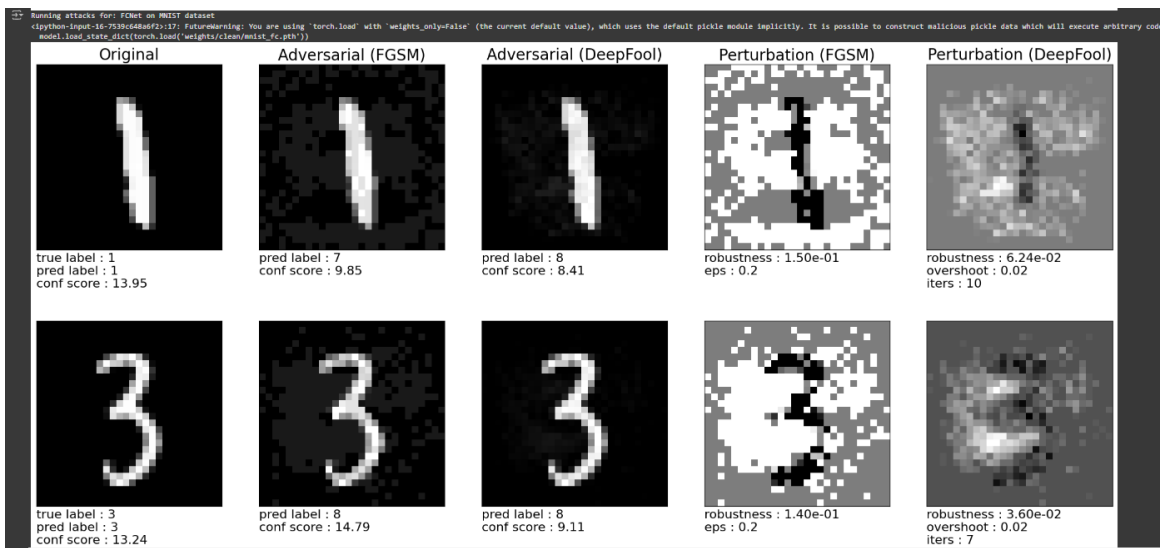
if device.type == 'cuda': torch.cuda.empty_cache()

# LeNet на датасете CIFAR-10
print("Running attacks for: LeNet on CIFAR-10 dataset")
fgsm_eps = 0.1
model = LeNet_CIFAR().to(device)
model.load_state_dict(torch.load('weights/clean/cifar_lenet.pth'))
display_attack(device, model, cifar_test, cifar_tf_inv, cifar_min, cifar_max, fgsm_eps, deep_args,
               has_labels=False, l2_norm=True, pert_scale=1.0, fig_rows=2, fig_width=25, fig_height=11, label_map=cifar_classes)

if device.type == 'cuda': torch.cuda.empty_cache()
```

9. Оценим результат атаки.





10. Создадим список со значениями ϵ s, которые мы хотим исследовать.

```
# Создадим список со значениями eps для FGSM атаки, которые мы хотим исследовать
fgsm_eps_list = [0.001, 0.02, 0.5, 0.9, 10]
```

11. Настраиваем отображение итоговой таблицы сравнения.

```
import io
import sys
import pandas as pd

# Создаем список со значениями eps для FGSM атаки, которые мы хотим исследовать
fgsm_eps = [0.001, 0.02, 0.5, 0.9, 10]

# FC на MNIST
for eps in fgsm_eps:
    model = FC_500_150().to(device)
    model.load_state_dict(torch.load('weights/clean/mnist_fc.pth'))
    display_attack(
        device, model, mnist_test, mnist_tf_inv, mnist_min, mnist_max, eps, deep_args,
        has_labels=False, l2_norm=True, pert_scale=1.0
    )
    if device.type == 'cuda':
        torch.cuda.empty_cache()

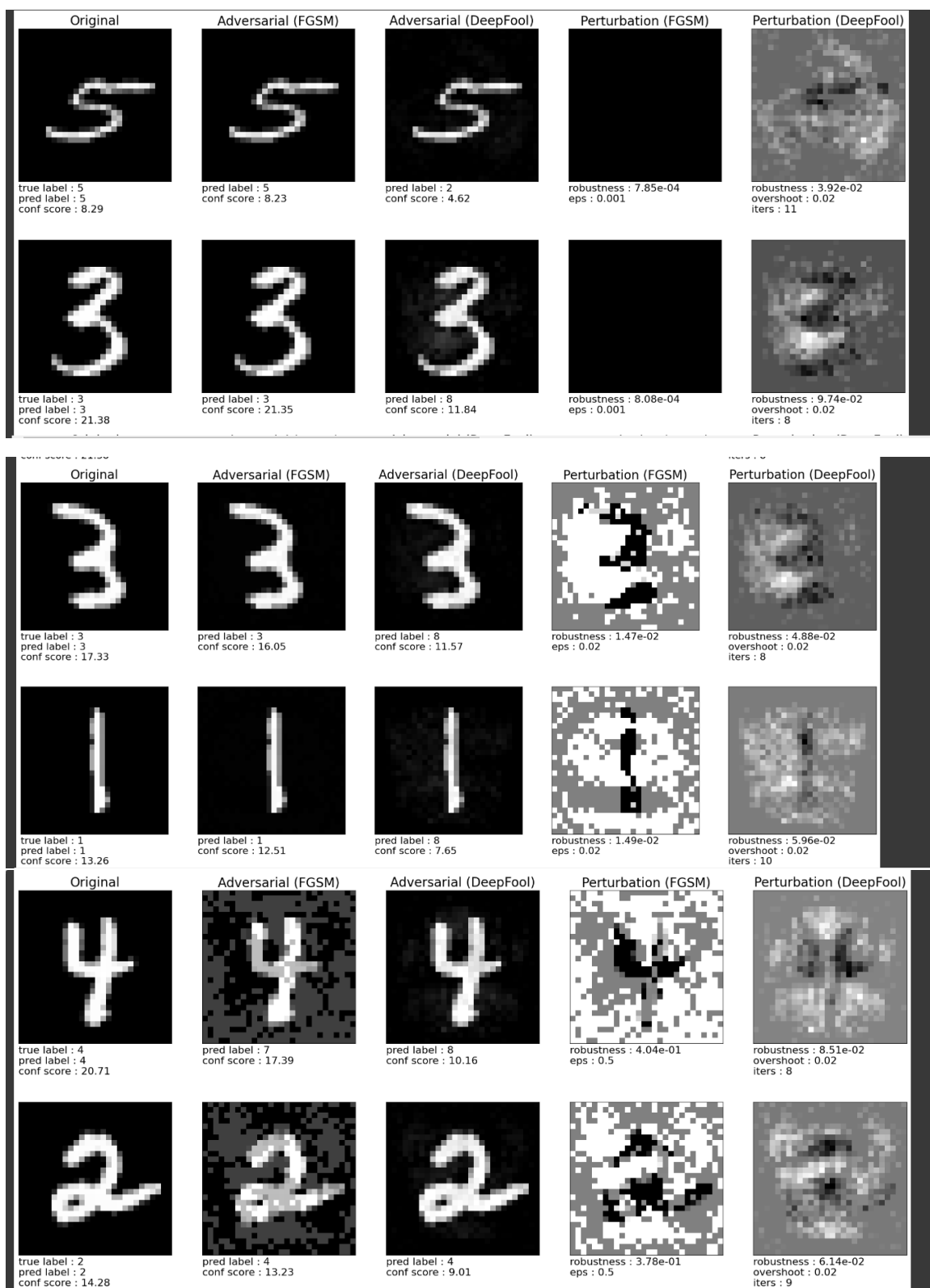
# LeNet на MNIST
for eps in fgsm_eps:
    model = LeNet_MNIST().to(device)
    model.load_state_dict(torch.load('weights/clean/mnist_lenet.pth'))
    display_attack(
        device, model, mnist_test, mnist_tf_inv, mnist_min, mnist_max, eps, deep_args,
        has_labels=False, l2_norm=True, pert_scale=1.0
    )
    if device.type == 'cuda':
        torch.cuda.empty_cache()


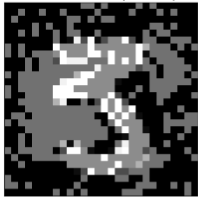

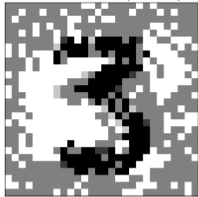
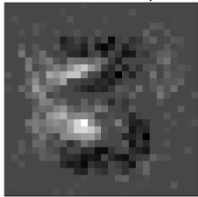
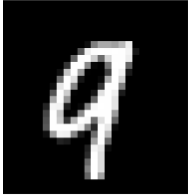

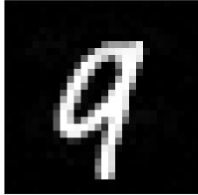
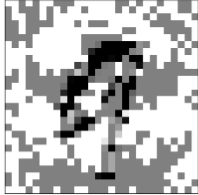
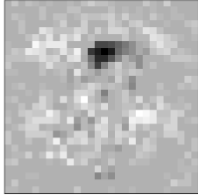




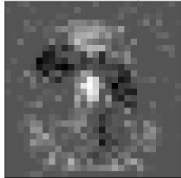
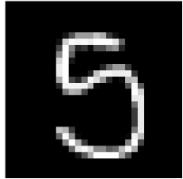


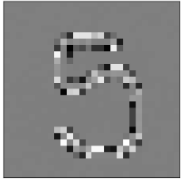
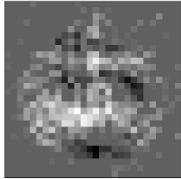
# NIN на CIFAR-10
for eps in fgsm_eps:
    model = Net().to(device)
    model.load_state_dict(torch.load('weights/clean/cifar_nin.pth'))
    display_attack(
        device, model, cifar_test, cifar_tf_inv, cifar_min, cifar_max, eps, deep_args,
        has_labels=False, l2_norm=True, pert_scale=1.0, label_map=cifar_classes
    )
    if device.type == 'cuda':
        torch.cuda.empty_cache()

# LeNet на CIFAR-10
for eps in fgsm_eps:
    model = LeNet_CIFAR().to(device)
    model.load_state_dict(torch.load('weights/clean/cifar_lenet.pth'))
    display_attack(
        device, model, cifar_test, cifar_tf_inv, cifar_min, cifar_max, eps, deep_args,
        has_labels=False, l2_norm=True, pert_scale=1.0, label_map=cifar_classes
    )
    if device.type == 'cuda':
        torch.cuda.empty_cache()
```

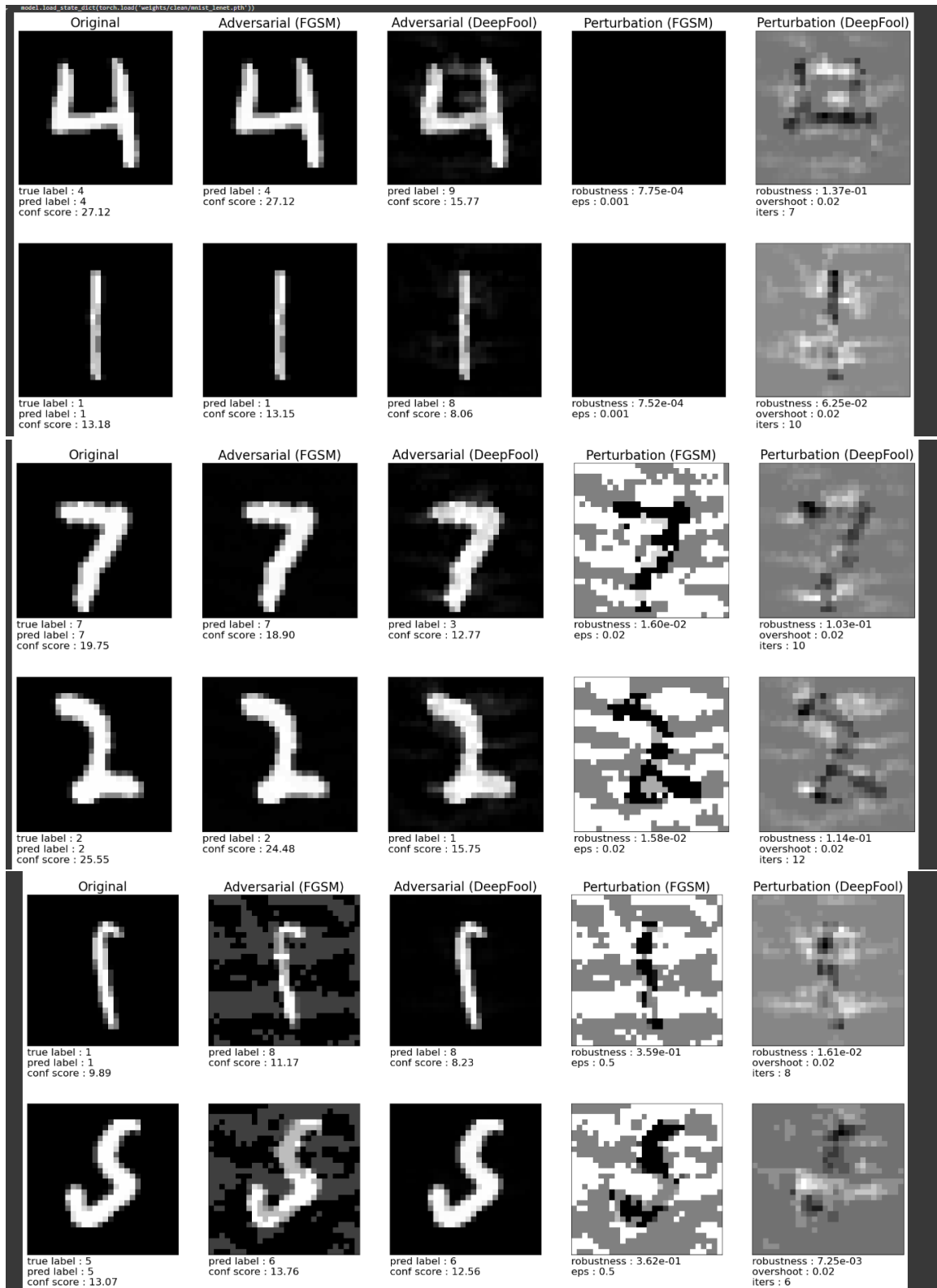

12. Смотрим вывод блока кода.

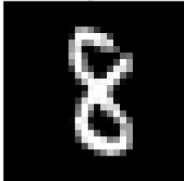



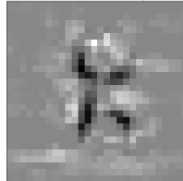




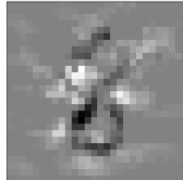
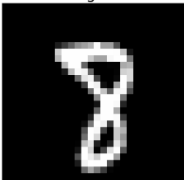

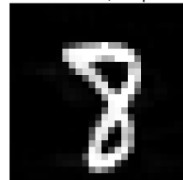

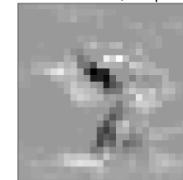




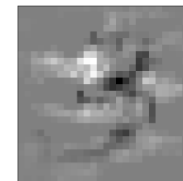
а. FC на MNIST






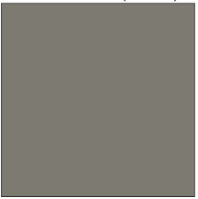




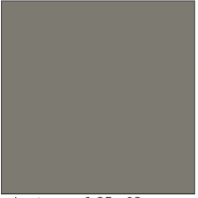
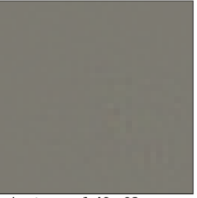




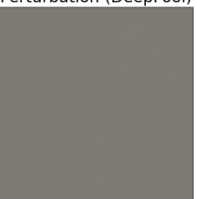



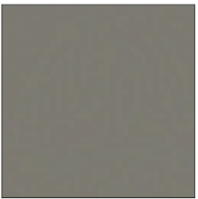


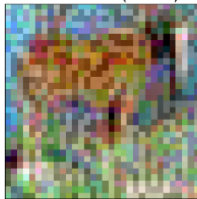

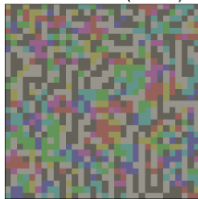
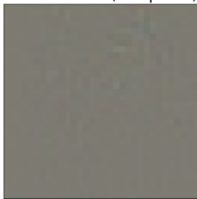

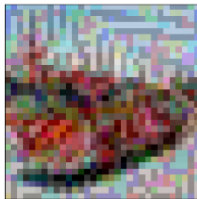
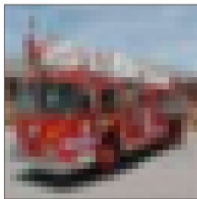
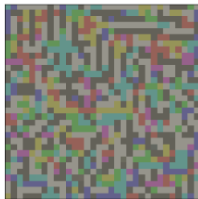
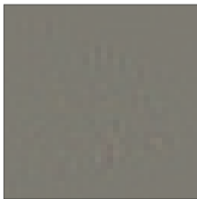
Original	Adversarial (FGSM)	Adversarial (DeepFool)	Perturbation (FGSM)	Perturbation (DeepFool)
 <p> true label : 3 red label : 3 conf score : 18.21 </p>	 <p> pred label : 8 conf score : 15.67 </p>	 <p> pred label : 8 conf score : 12.23 </p>	 <p> robustness : 6.58e-01 eps : 0.9 </p>	 <p> robustness : 7.08e-02 overshoot : 0.02 iters : 8 </p>
 <p> true label : 9 red label : 9 conf score : 7.97 </p>	 <p> pred label : 1 conf score : 21.57 </p>	 <p> pred label : 4 conf score : 5.66 </p>	 <p> robustness : 7.03e-01 eps : 0.9 </p>	 <p> robustness : 2.89e-02 overshoot : 0.02 iters : 10 </p>
 <p> true label : 7 pred label : 7 conf score : 17.04 </p>	 <p> pred label : 3 conf score : 20.77 </p>	 <p> pred label : 9 conf score : 9.28 </p>	 <p> robustness : 1.41e+00 eps : 10 </p>	 <p> robustness : 8.52e-02 overshoot : 0.02 iters : 9 </p>
 <p> true label : 5 pred label : 5 conf score : 10.75 </p>	 <p> pred label : 2 conf score : 23.65 </p>	 <p> pred label : 8 conf score : 7.41 </p>	 <p> robustness : 1.43e+00 eps : 10 </p>	 <p> robustness : 3.73e-02 overshoot : 0.02 iters : 8 </p>


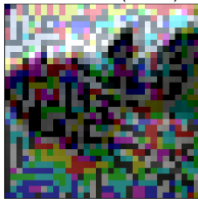

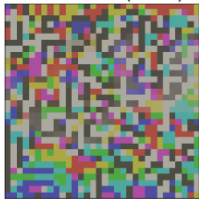


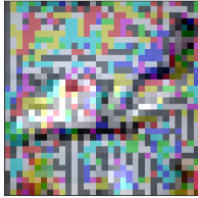

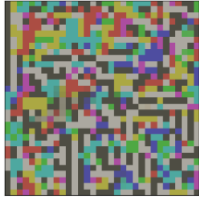
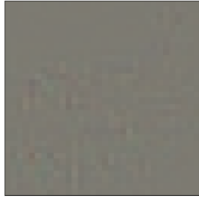






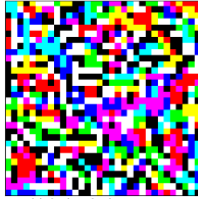

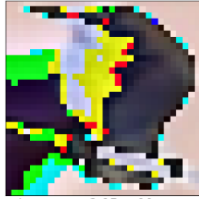
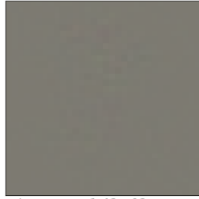
b. LeNet на MNIST











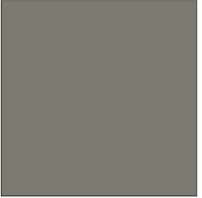
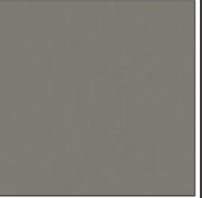









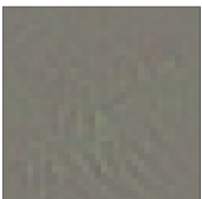



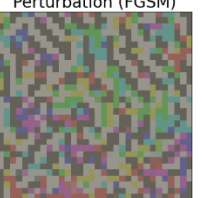

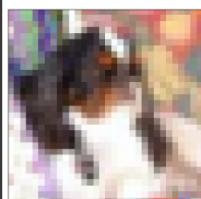
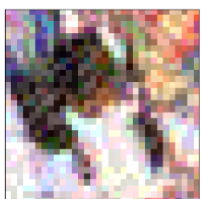
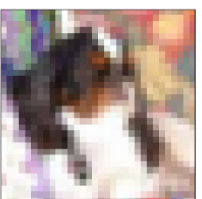
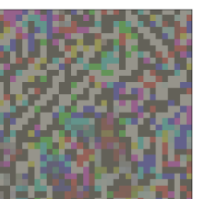
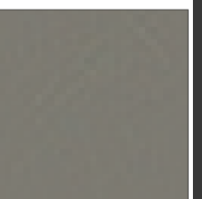
Original	Adversarial (FGSM)	Adversarial (DeepFool)	Perturbation (FGSM)	Perturbation (DeepFool)
 <p>true label : 8 pred label : 8 conf score : 32.97</p>	 <p>pred label : 3 conf score : 25.56</p>	 <p>pred label : 3 conf score : 20.22</p>	 <p>robustness : 6.74e-01 eps : 0.9</p>	 <p>robustness : 1.12e-01 overshoot : 0.02 iters : 9</p>
 <p>true label : 8 pred label : 8 conf score : 28.12</p>	 <p>pred label : 8 conf score : 10.56</p>	 <p>pred label : 4 conf score : 18.27</p>	 <p>robustness : 6.58e-01 eps : 0.9</p>	 <p>robustness : 1.05e-01 overshoot : 0.02 iters : 9</p>
Original	Adversarial (FGSM)	Adversarial (DeepFool)	Perturbation (FGSM)	Perturbation (DeepFool)
 <p>true label : 8 pred label : 8 conf score : 21.92</p>	 <p>pred label : 3 conf score : 14.26</p>	 <p>pred label : 3 conf score : 16.60</p>	 <p>robustness : 1.48e+00 eps : 10</p>	 <p>robustness : 3.63e-02 overshoot : 0.02 iters : 8</p>
 <p>true label : 3 pred label : 3 conf score : 34.94</p>	 <p>pred label : 5 conf score : 25.54</p>	 <p>pred label : 9 conf score : 14.51</p>	 <p>robustness : 1.45e+00 eps : 10</p>	 <p>robustness : 1.45e-01 overshoot : 0.02 iters : 9</p>


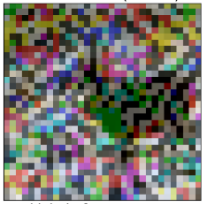

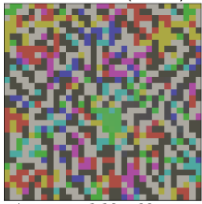


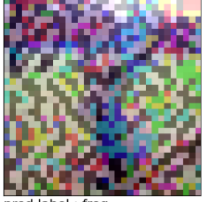

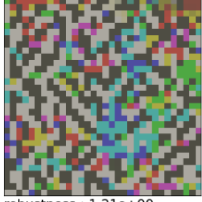

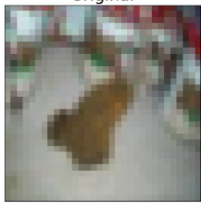

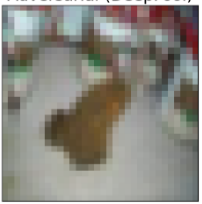





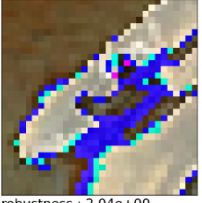

c. NiN на CIFAR-10

Original	Adversarial (FGSM)	Adversarial (DeepFool)	Perturbation (FGSM)	Perturbation (DeepFool)
 true label : cat pred label : cat conf score : 39.07	 pred label : cat conf score : 38.88	 pred label : dog conf score : 28.91	 robustness : 7.76e-04 eps : 0.001	 robustness : 2.57e-02 overshoot : 0.02 iters : 3
 true label : bird pred label : bird conf score : 21.50	 pred label : bird conf score : 21.27	 pred label : frog conf score : 17.17	 robustness : 1.25e-03 eps : 0.001	 robustness : 1.49e-02 overshoot : 0.02 iters : 2
Original	Adversarial (FGSM)	Adversarial (DeepFool)	Perturbation (FGSM)	Perturbation (DeepFool)
 true label : bird pred label : bird conf score : 17.95	 pred label : frog conf score : 18.83	 pred label : frog conf score : 17.44	 robustness : 1.42e-02 eps : 0.02	 robustness : 1.66e-03 overshoot : 0.02 iters : 1
 true label : automobile pred label : automobile conf score : 64.21	 pred label : automobile conf score : 56.97	 pred label : truck conf score : 53.37	 robustness : 1.10e-02 eps : 0.02	 robustness : 1.31e-02 overshoot : 0.02 iters : 1
Original	Adversarial (FGSM)	Adversarial (DeepFool)	Perturbation (FGSM)	Perturbation (DeepFool)
 true label : horse pred label : horse conf score : 44.91	 pred label : frog conf score : 21.28	 pred label : deer conf score : 36.35	 robustness : 6.42e-01 eps : 0.5	 robustness : 2.86e-02 overshoot : 0.02 iters : 2
 true label : truck pred label : truck conf score : 50.34	 pred label : ship conf score : 17.85	 pred label : ship conf score : 39.99	 robustness : 4.79e-01 eps : 0.5	 robustness : 2.27e-02 overshoot : 0.02 iters : 2

Original	Adversarial (FGSM)	Adversarial (DeepFool)	Perturbation (FGSM)	Perturbation (DeepFool)
 true label : truck pred label : truck conf score : 18.08	 pred label : frog conf score : 19.22	 pred label : airplane conf score : 14.67	 robustness : 6.23e-01 eps : 0.9	 robustness : 4.92e-03 overshoot : 0.02 iters : 3
 true label : airplane pred label : airplane conf score : 34.21	 pred label : frog conf score : 18.06	 pred label : ship conf score : 23.75	 robustness : 1.04e+00 eps : 0.9	 robustness : 4.17e-02 overshoot : 0.02 iters : 2
Original	Adversarial (FGSM)	Adversarial (DeepFool)	Perturbation (FGSM)	Perturbation (DeepFool)
 true label : automobile pred label : automobile conf score : 61.77	 pred label : automobile conf score : 31.47	 pred label : truck conf score : 36.19	 robustness : 2.18e+00 eps : 10	 robustness : 4.84e-02 overshoot : 0.02 iters : 2
 true label : bird pred label : bird conf score : 26.25	 pred label : airplane conf score : 23.71	 pred label : cat conf score : 24.36	 robustness : 2.95e+00 eps : 10	 robustness : 1.42e-02 overshoot : 0.02 iters : 3

d. LeNet на CIFAR-10

Original	Adversarial (FGSM)	Adversarial (DeepFool)	Perturbation (FGSM)	Perturbation (DeepFool)
 true label : bird pred label : bird conf score : 4.67	 pred label : bird conf score : 4.60	 pred label : frog conf score : 4.04	 robustness : 6.95e-04 eps : 0.001	 robustness : 6.32e-03 overshoot : 0.02 iters : 1
 true label : frog pred label : dog conf score : 7.16	 pred label : dog conf score : 7.29	 pred label : frog conf score : 5.77	 robustness : 8.25e-04 eps : 0.001	 robustness : 6.80e-03 overshoot : 0.02 iters : 2
Original	Adversarial (FGSM)	Adversarial (DeepFool)	Perturbation (FGSM)	Perturbation (DeepFool)
 true label : horse pred label : bird conf score : 4.51	 pred label : bird conf score : 5.96	 pred label : horse conf score : 4.40	 robustness : 1.79e-02 eps : 0.02	 robustness : 6.90e-04 overshoot : 0.02 iters : 1
 true label : automobile pred label : automobile conf score : 11.49	 pred label : automobile conf score : 9.69	 pred label : bird conf score : 2.35	 robustness : 1.62e-02 eps : 0.02	 robustness : 4.19e-02 overshoot : 0.02 iters : 3
Original	Adversarial (FGSM)	Adversarial (DeepFool)	Perturbation (FGSM)	Perturbation (DeepFool)
 true label : automobile pred label : automobile conf score : 9.32	 pred label : truck conf score : 9.63	 pred label : truck conf score : 8.27	 robustness : 5.04e-01 eps : 0.5	 robustness : 6.90e-03 overshoot : 0.02 iters : 2
 true label : dog pred label : dog conf score : 9.34	 pred label : cat conf score : 7.70	 pred label : cat conf score : 8.00	 robustness : 3.14e-01 eps : 0.5	 robustness : 1.35e-02 overshoot : 0.02 iters : 2

Original	Adversarial (FGSM)	Adversarial (DeepFool)	Perturbation (FGSM)	Perturbation (DeepFool)
 <p>true label : deer pred label : deer conf score : 8.11</p>	 <p>pred label : frog conf score : 19.12</p>	 <p>pred label : bird conf score : 5.87</p>	 <p>robustness : 1.10e+00 eps : 0.9</p>	 <p>robustness : 1.65e-02 overshoot : 0.02 iters : 3</p>
 <p>true label : deer pred label : deer conf score : 7.68</p>	 <p>pred label : frog conf score : 21.41</p>	 <p>pred label : bird conf score : 6.47</p>	 <p>robustness : 1.21e+00 eps : 0.9</p>	 <p>robustness : 9.30e-03 overshoot : 0.02 iters : 2</p>
Original	Adversarial (FGSM)	Adversarial (DeepFool)	Perturbation (FGSM)	Perturbation (DeepFool)
 <p>true label : frog pred label : frog conf score : 5.26</p>	 <p>pred label : frog conf score : 37.97</p>	 <p>pred label : cat conf score : 4.50</p>	 <p>robustness : 3.34e+00 eps : 10</p>	 <p>robustness : 4.95e-03 overshoot : 0.02 iters : 2</p>
 <p>true label : ship pred label : ship conf score : 10.51</p>	 <p>pred label : frog conf score : 38.04</p>	 <p>pred label : automobile conf score : 6.60</p>	 <p>robustness : 2.04e+00 eps : 10</p>	 <p>robustness : 3.82e-02 overshoot : 0.02 iters : 3</p>

Выводы

В ходе выполнения лабораторной работы была изучена устойчивость различных моделей к атакам FGSM и DeepFool с использованием наборов данных MNIST и CIFAR-10. Основное внимание уделялось сравнению влияния параметра ϵ на качество классификации и поведение моделей. Были получены следующие результаты:

FGSM атака:

Увеличение значения ϵ значительно искажает изображение. Это искажение становится видимым невооружённым глазом при высоких значениях ϵ .

При низких значениях ϵ модели демонстрируют высокую точность, однако уже небольшое увеличение ϵ приводит к заметному снижению точности классификации.

Чем больше значение ϵ , тем выше вероятность того, что модель ошибочно классифицирует изображение.

DeepFool атака:

DeepFool генерирует такие искажения, которые практически не различимы для человека даже при значениях ϵ , вызывающих высокую ошибку классификации.

Независимо от параметра ϵ , DeepFool-атака всегда приводит к ошибкам классификации, показывая высокую эффективность против моделей.

Сравнение моделей:

Модель на основе MNIST оказалась более устойчивой к FGSM-атакам по сравнению с моделью на основе CIFAR-10. Однако для атак DeepFool существенной разницы в устойчивости не наблюдается.

Устойчивость моделей к FGSM-атакам напрямую зависит от сложности и размера набора данных: CIFAR-10 продемонстрировал большую уязвимость по сравнению с MNIST.

Заключение: Данная лабораторная работа наглядно показала важность учета устойчивости моделей к различным типам атак при разработке систем машинного обучения. FGSM и DeepFool показали свои сильные и слабые стороны. FGSM более наглядна в демонстрации искажений, однако DeepFool эффективнее в том, чтобы скрывать искажения от человека, при этом вводя модель в заблуждение. Эти результаты подчеркивают необходимость разработки более защищенных моделей для практического использования.