МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

ИКБ направление «Киберразведка и противодействие угрозам с применением технологий искусственного интеллекта» 10.04.01

Кафедра КБ-4 «Интеллектуальные системы информационной безопасности»

Лабораторная работа №1

по дисциплине

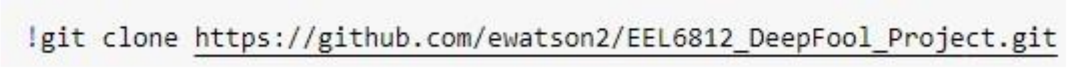
«Анализ защищенности систем искусственного интеллекта»

Группа:
ББМО-01-22
Выполнил:
Орлянский В.Е.

Проверил:
Спирин А.А.

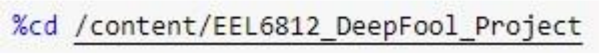
Москва 2023

1. Клонируем репозиторий.



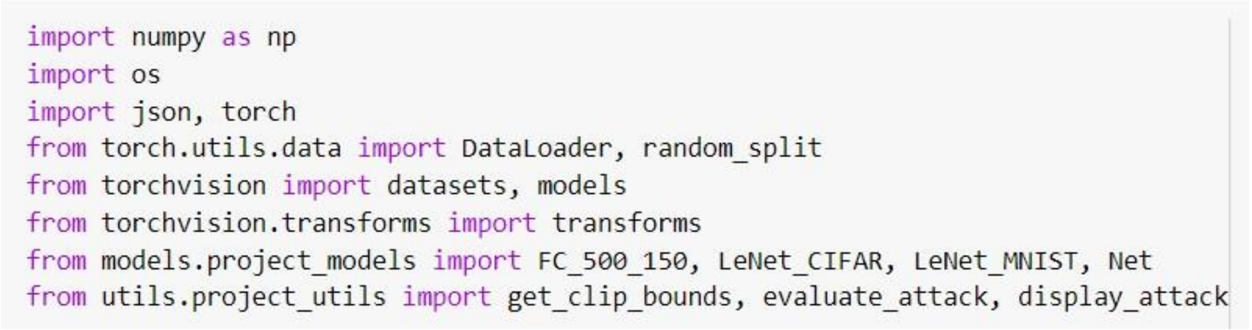
```
!git clone https://github.com/ewatson2/EEL6812_DeepFool_Project.git
```

2. Сменим директорию исполнения на вновь созданную папку "EEL6812_DeepFool_Project" проекта.



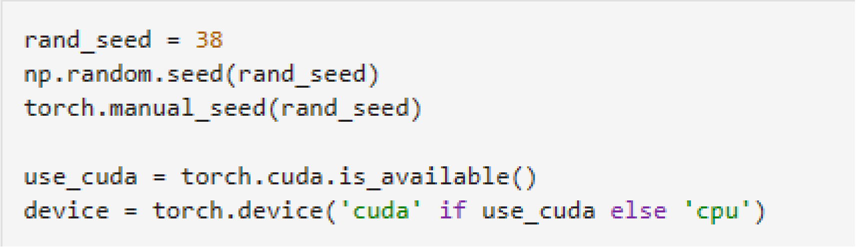
```
%cd /content/EEL6812_DeepFool_Project
```

3. Выполним импорт стандартных и вспомогательных библиотек.



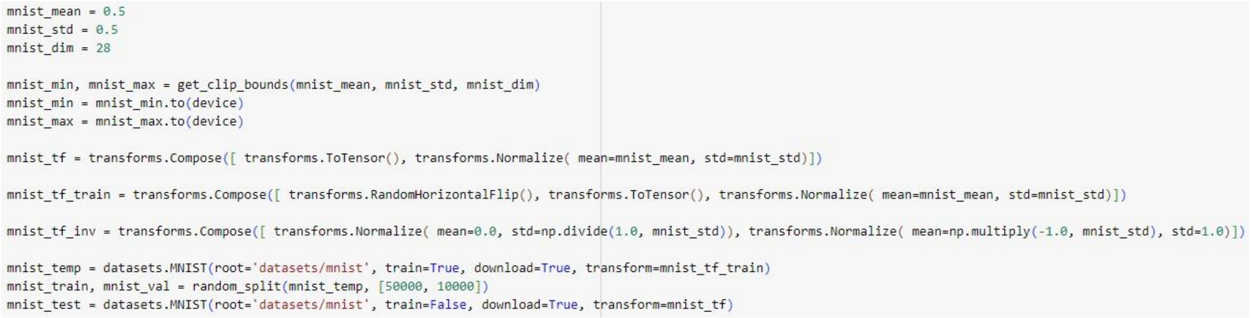
```
import numpy as np
import os
import json, torch
from torch.utils.data import DataLoader, random_split
from torchvision import datasets, models
from torchvision.transforms import transforms
from models.project_models import FC_500_150, LeNet_CIFAR, LeNet_MNIST, Net
from utils.project_utils import get_clip_bounds, evaluate_attack, display_attack
```

4. Установим случайное рандомное значение на номер в списке (38).



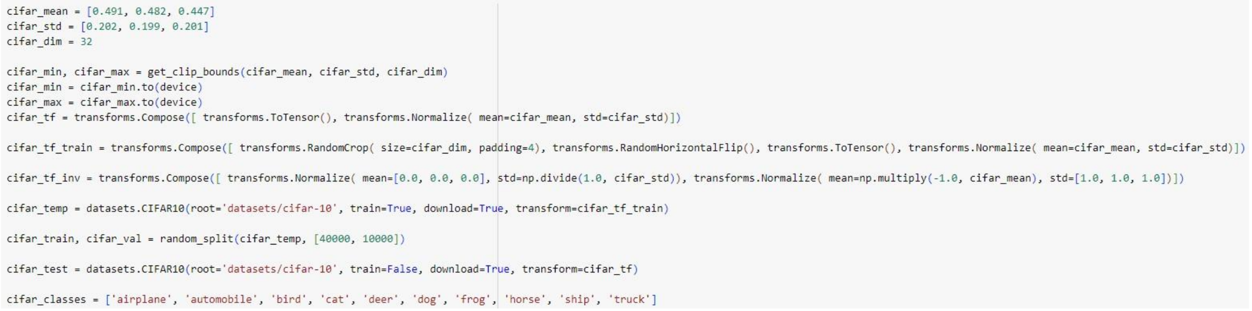
```
rand_seed = 38
np.random.seed(rand_seed)
torch.manual_seed(rand_seed)

use_cuda = torch.cuda.is_available()
device = torch.device('cuda' if use_cuda else 'cpu')
```

5. Загрузим датасет MNIST.



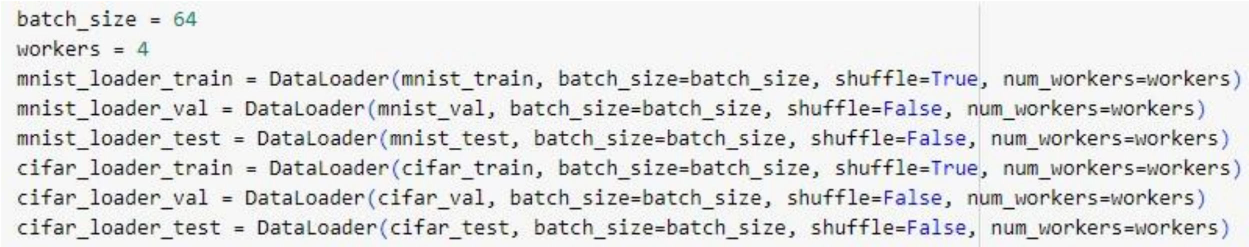
```
mnist_mean = 0.5
mnist_std = 0.5
mnist_dim = 28

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)
```

6. Загрузим датасет CIFAR-10.



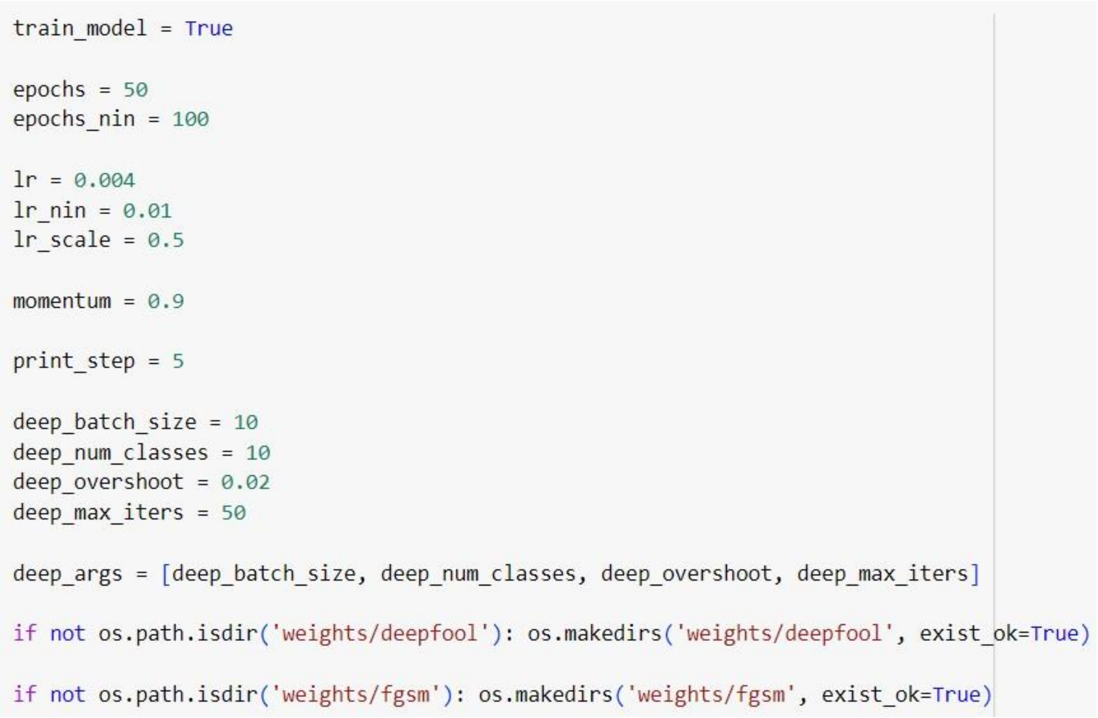
```
cifar_mean = [0.491, 0.482, 0.447]
cifar_std = [0.202, 0.199, 0.201]
cifar_dim = 32

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']
```

7. Выполним настройку и загрузку DataLoader.



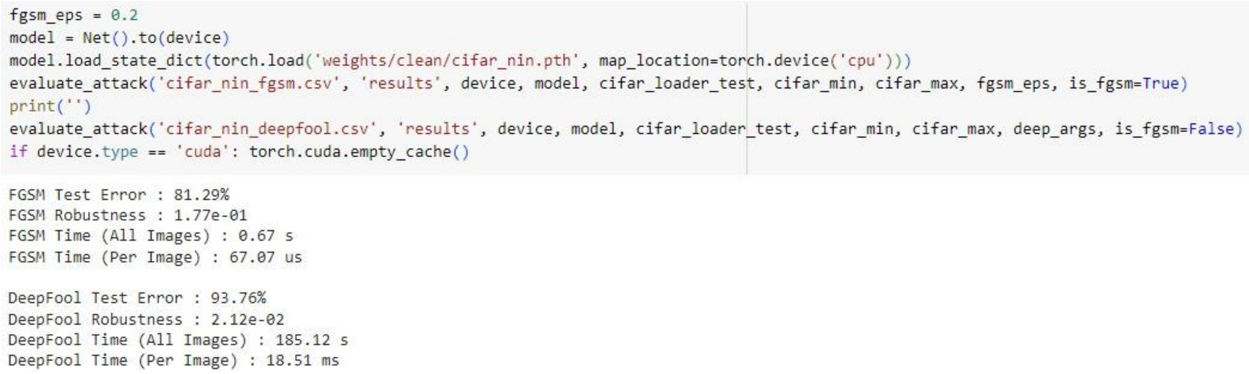
```
batch_size = 64
workers = 4
mnist_loader_train = DataLoader(mnist_train, batch_size=batch_size, shuffle=True, num_workers=workers)
mnist_loader_val = DataLoader(mnist_val, batch_size=batch_size, shuffle=False, num_workers=workers)
mnist_loader_test = DataLoader(mnist_test, batch_size=batch_size, 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)
```

8. Настроим обучающую модель.



```
train_model = True

epochs = 50
epochs_nin = 100

lr = 0.004
lr_nin = 0.01
lr_scale = 0.5

momentum = 0.9

print_step = 5

deep_batch_size = 10
deep_num_classes = 10
deep_overshoot = 0.02
deep_max_iters = 50

deep_args = [deep_batch_size, deep_num_classes, deep_overshoot, deep_max_iters]

if not os.path.isdir('weights/deepfool'): os.makedirs('weights/deepfool', exist_ok=True)

if not os.path.isdir('weights/fgsm'): os.makedirs('weights/fgsm', exist_ok=True)
```

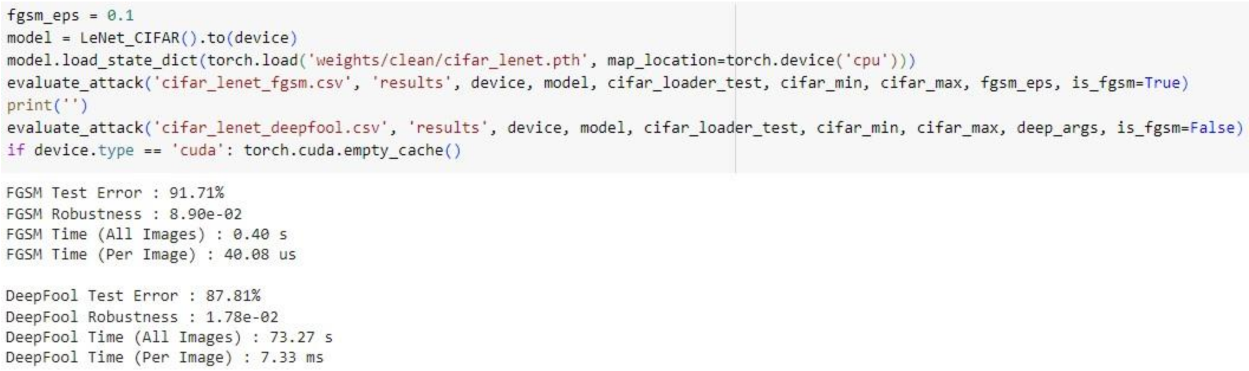
9. Загрузим и оценим стойкость модели Network-In-Network Model к FGSM и DeepFool атакам на основе датасета CIFAR-10.



```
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()
```

```
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
```

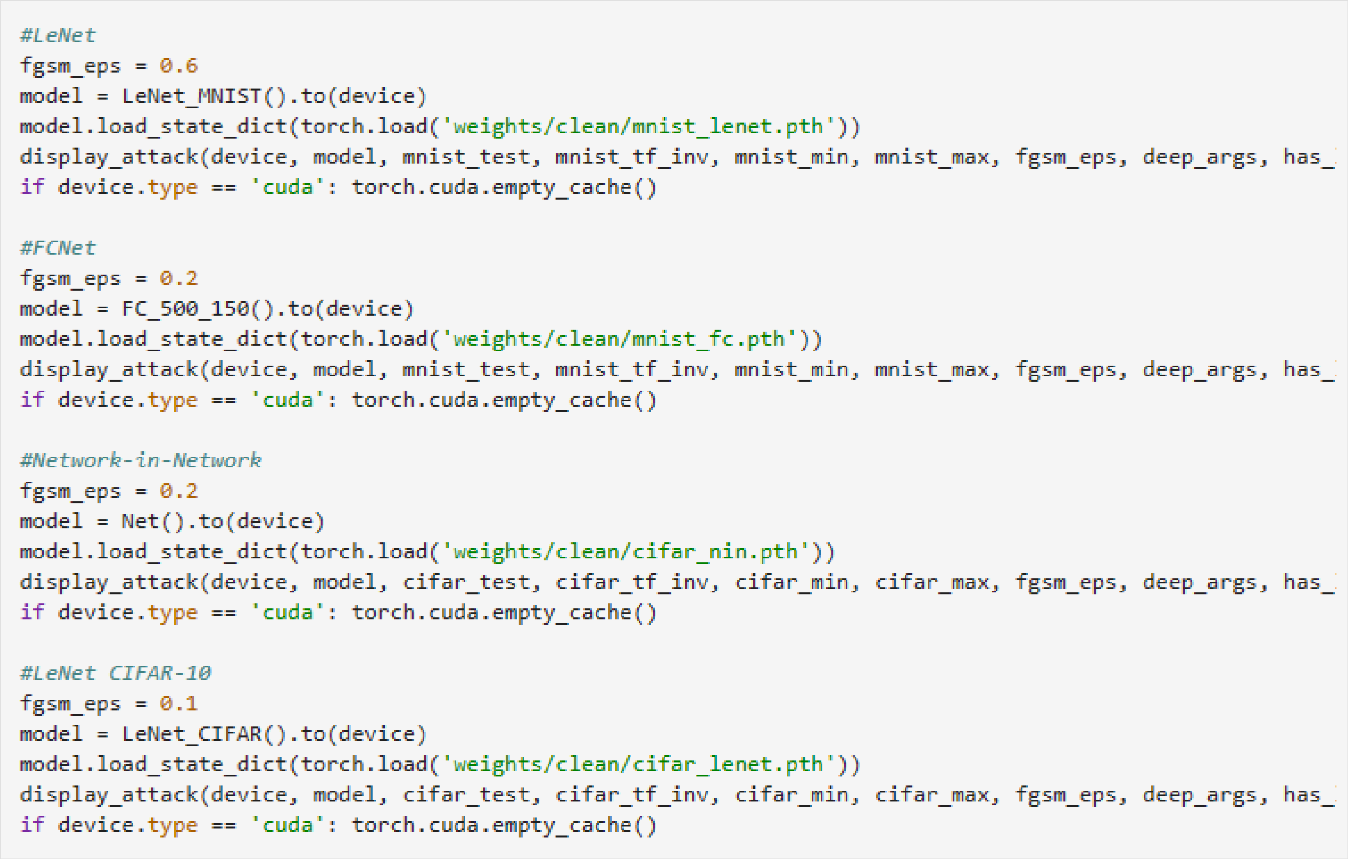
10. Загрузим и оценим стойкость модели LeNet к FGSM и DeepFool атакам на основе датасета CIFAR-10.



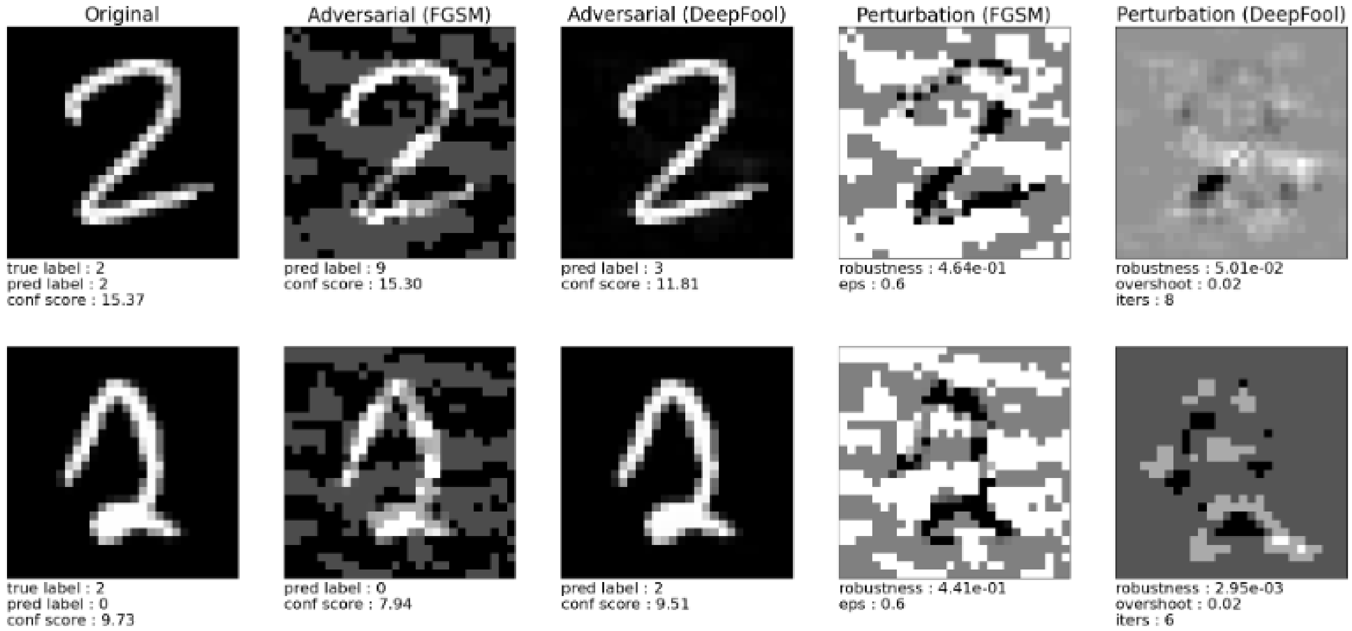
```
fgsm_eps = 0.1
model = LeNet_CIFAR().to(device)
model.load_state_dict(torch.load('weights/clean/cifar_lenet.pth', map_location=torch.device('cpu')))
evaluate_attack('cifar_lenet_fgsm.csv', 'results', device, model, cifar_loader_test, cifar_min, cifar_max, fgsm_eps, is_fgsm=True)
print('')
evaluate_attack('cifar_lenet_deepfool.csv', 'results', device, model, cifar_loader_test, cifar_min, cifar_max, deep_args, is_fgsm=False)
if device.type == 'cuda': torch.cuda.empty_cache()
```

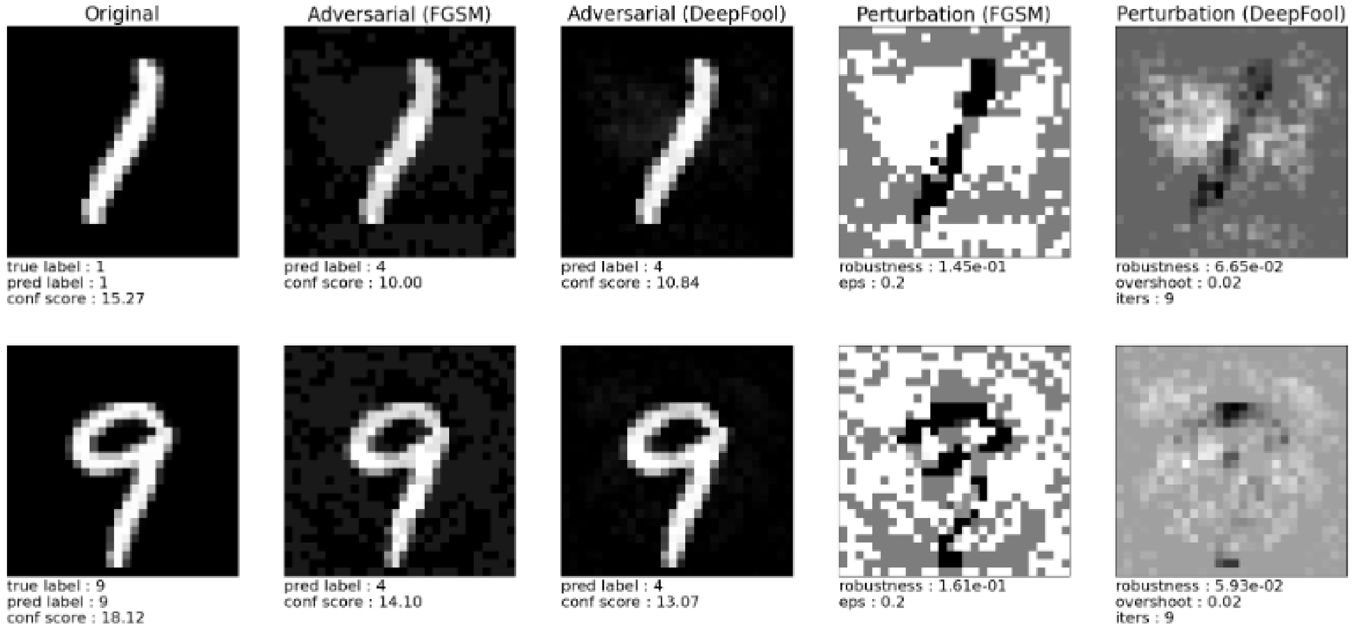
```
FGSM Test Error : 91.71%
FGSM Robustness : 8.90e-02
FGSM Time (All Images) : 0.40 s
FGSM Time (Per Image) : 40.08 us

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

11. Выполним оценку атакующих примеров для сетей.

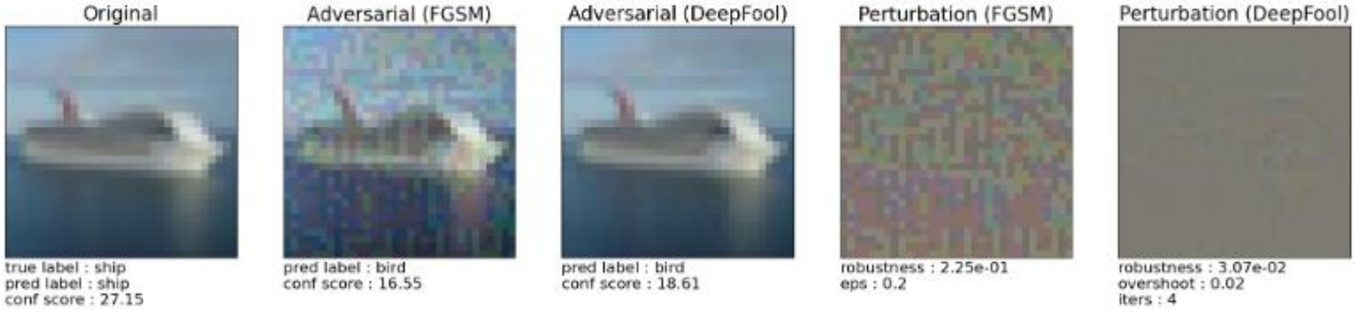


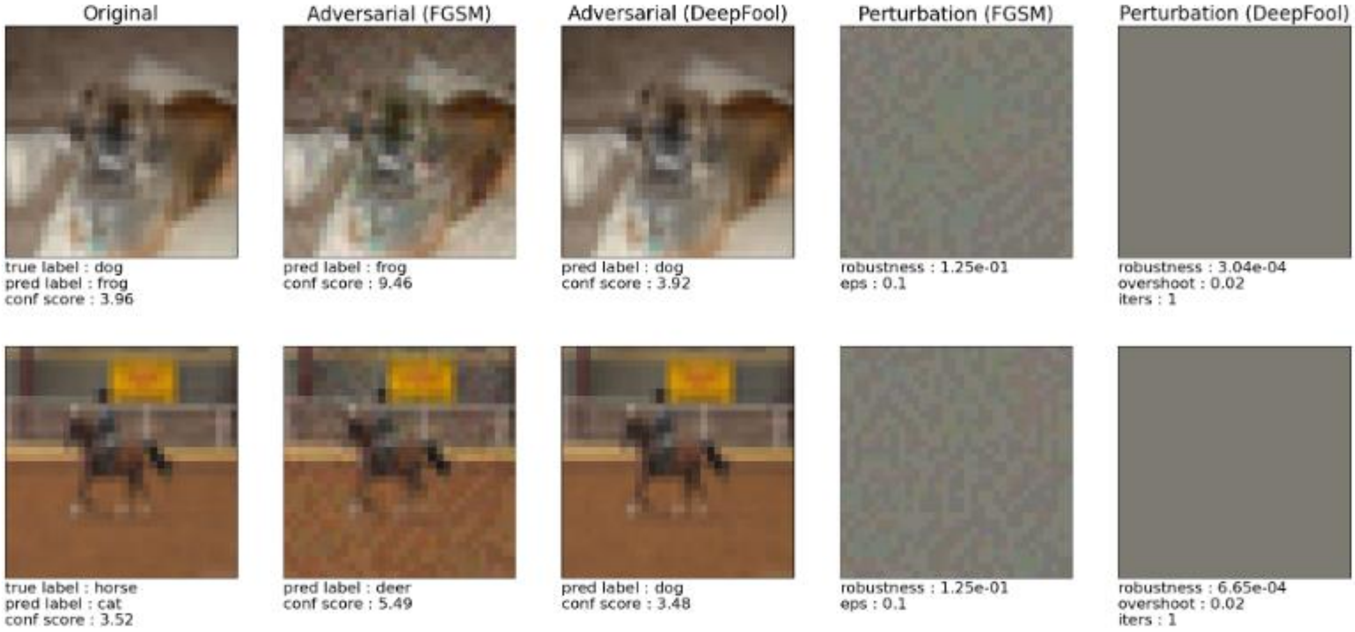
```
#LeNet
fgsm_eps = 0.6
model = LeNet_MNIST().to(device)
model.load_state_dict(torch.load('weights/clean/mnist_lenet.pth'))
display_attack(device, model, mnist_test, mnist_tf_inv, mnist_min, mnist_max, fgsm_eps, deep_args, has_
if device.type == 'cuda': torch.cuda.empty_cache()

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

#Network-in-Network
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_
if device.type == 'cuda': torch.cuda.empty_cache()

#LeNet CIFAR-10
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_
if device.type == 'cuda': torch.cuda.empty_cache()
```


Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : 2
pred label : 2
conf score : 15.37
pred label : 9
conf score : 15.30
pred label : 3
conf score : 11.81
robustness : 4.64e-01
eps : 0.6
robustness : 5.01e-02
overshoot : 0.02
iters : 8
true label : 2
pred label : 0
conf score : 9.73
pred label : 0
conf score : 7.94
pred label : 2
conf score : 9.51
robustness : 4.41e-01
eps : 0.6
robustness : 2.95e-03
overshoot : 0.02
iters : 6


Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : 1
pred label : 1
conf score : 15.27
pred label : 4
conf score : 10.00
pred label : 4
conf score : 10.84
robustness : 1.45e-01
eps : 0.2
robustness : 6.65e-02
overshoot : 0.02
iters : 9
true label : 9
pred label : 9
conf score : 18.12
pred label : 4
conf score : 14.10
pred label : 4
conf score : 13.07
robustness : 1.61e-01
eps : 0.2
robustness : 5.93e-02
overshoot : 0.02
iters : 9

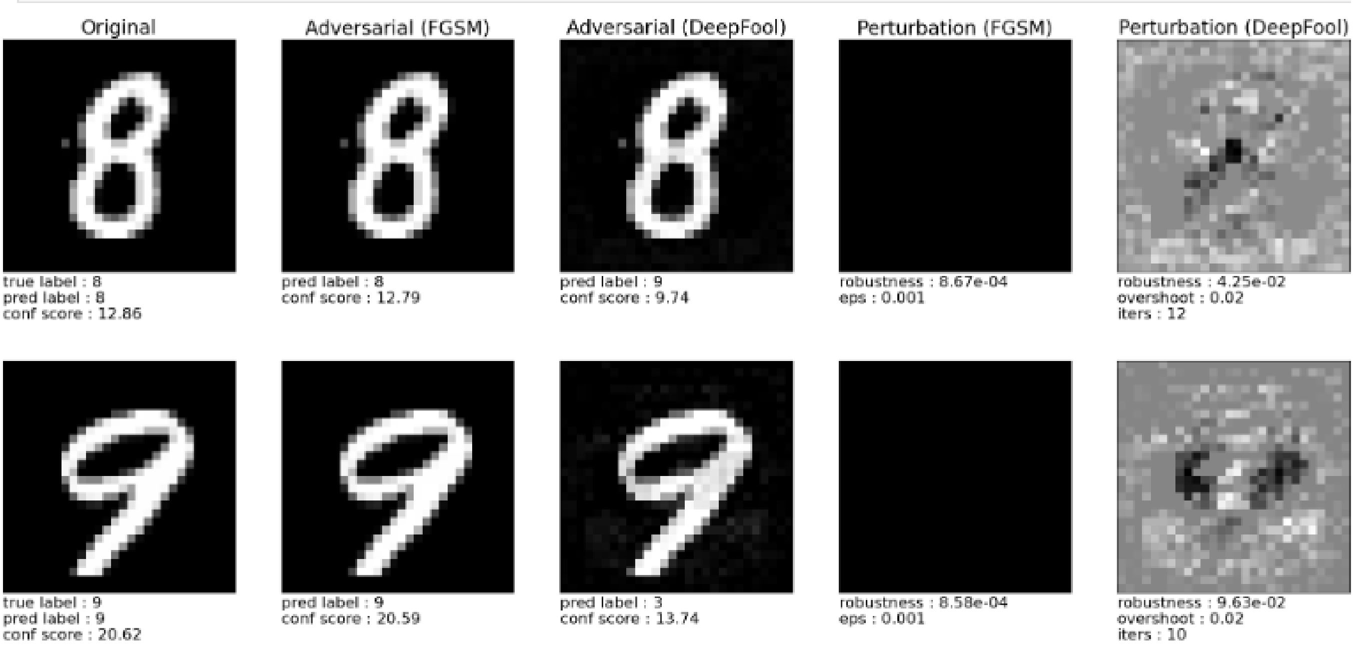
\

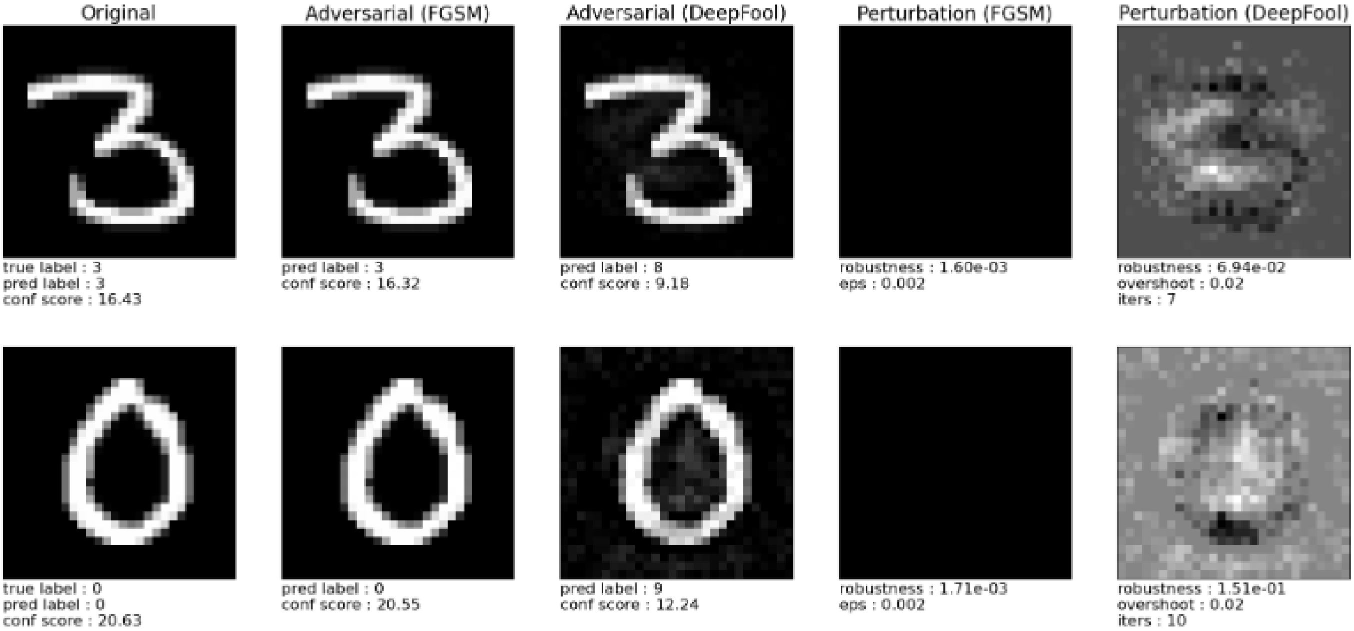


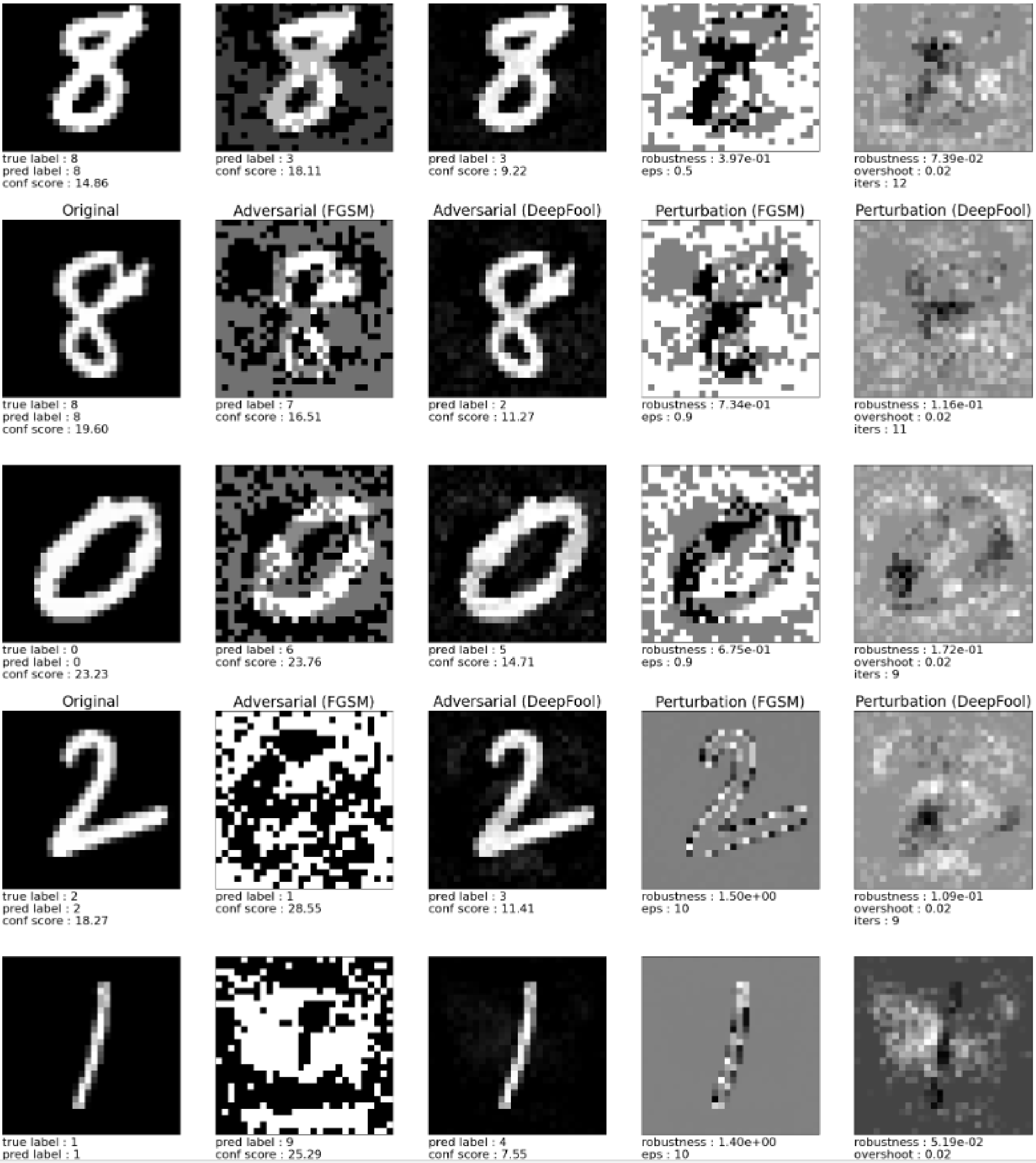


12. Отражаем отличия для fgsm_eps=(0.001, 0.02, 0.5, 0.9, 10).

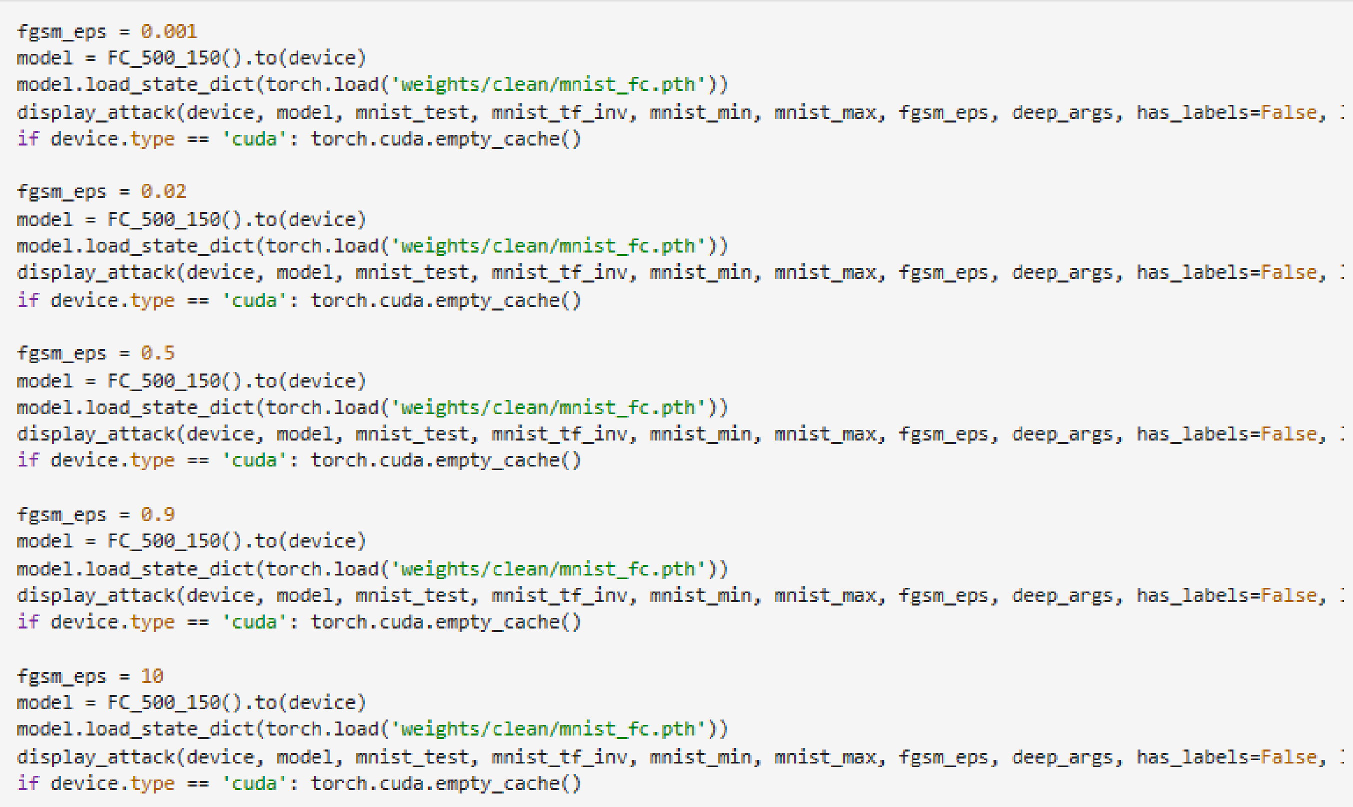
```
fgsm_eps = 0.001
model = FC_500_150().to(device)
model.load_state_dict(torch.load('weights/clean/mnist_fc.pth'))
display_attack(device, model, mnist_test, mnist_tf_inv, mnist_min, mnist_max, fgsm_eps, deep_args, has_
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 0.002
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_
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 0.5
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_
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 0.9
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_
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 10
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_
if device.type == 'cuda': torch.cuda.empty_cache()
```


Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : 8
pred label : 8
conf score : 12.86
pred label : 8
conf score : 12.79
pred label : 9
conf score : 9.74
robustness : 8.67e-04
eps : 0.001
robustness : 4.25e-02
overshoot : 0.02
iters : 12
true label : 9
pred label : 9
conf score : 20.62
pred label : 9
conf score : 20.59
pred label : 3
conf score : 13.74
robustness : 8.58e-04
eps : 0.001
robustness : 9.63e-02
overshoot : 0.02
iters : 10

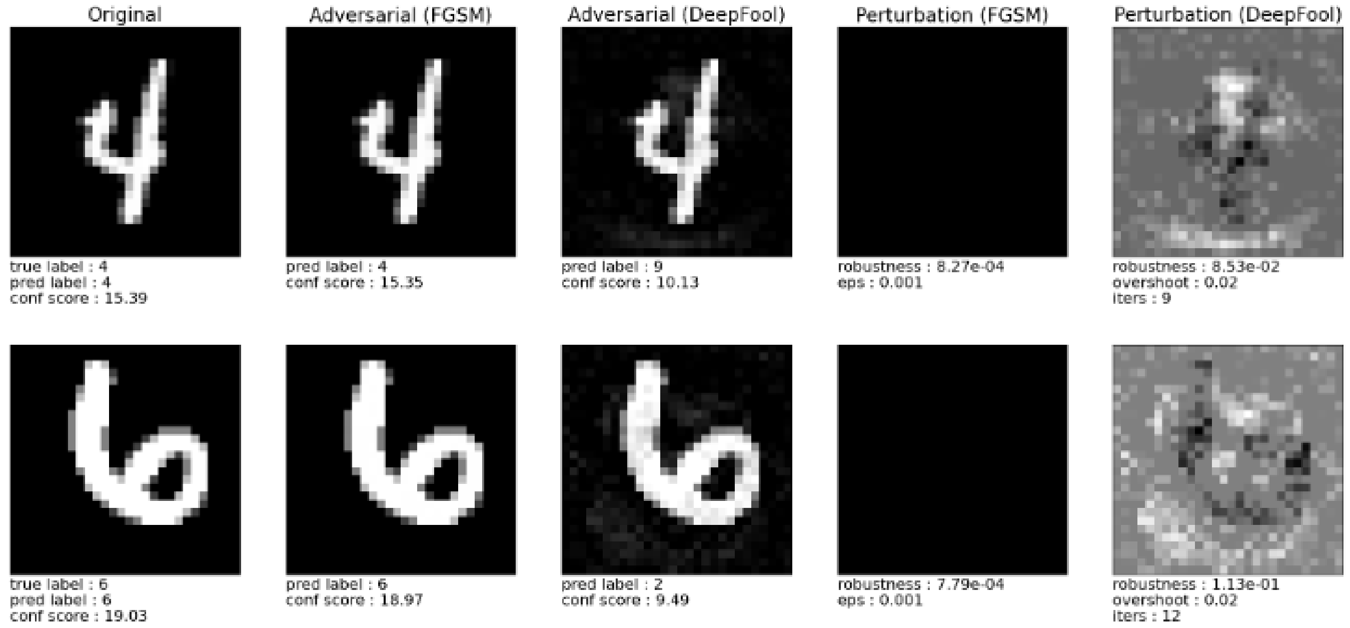

Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : 3
pred label : 3
conf score : 16.43
pred label : 3
conf score : 16.32
pred label : 8
conf score : 9.18
robustness : 1.60e-03
eps : 0.002
robustness : 6.94e-02
overshoot : 0.02
iters : 7
true label : 0
pred label : 0
conf score : 20.63
pred label : 0
conf score : 20.55
pred label : 9
conf score : 12.24
robustness : 1.71e-03
eps : 0.002
robustness : 1.51e-01
overshoot : 0.02
iters : 10

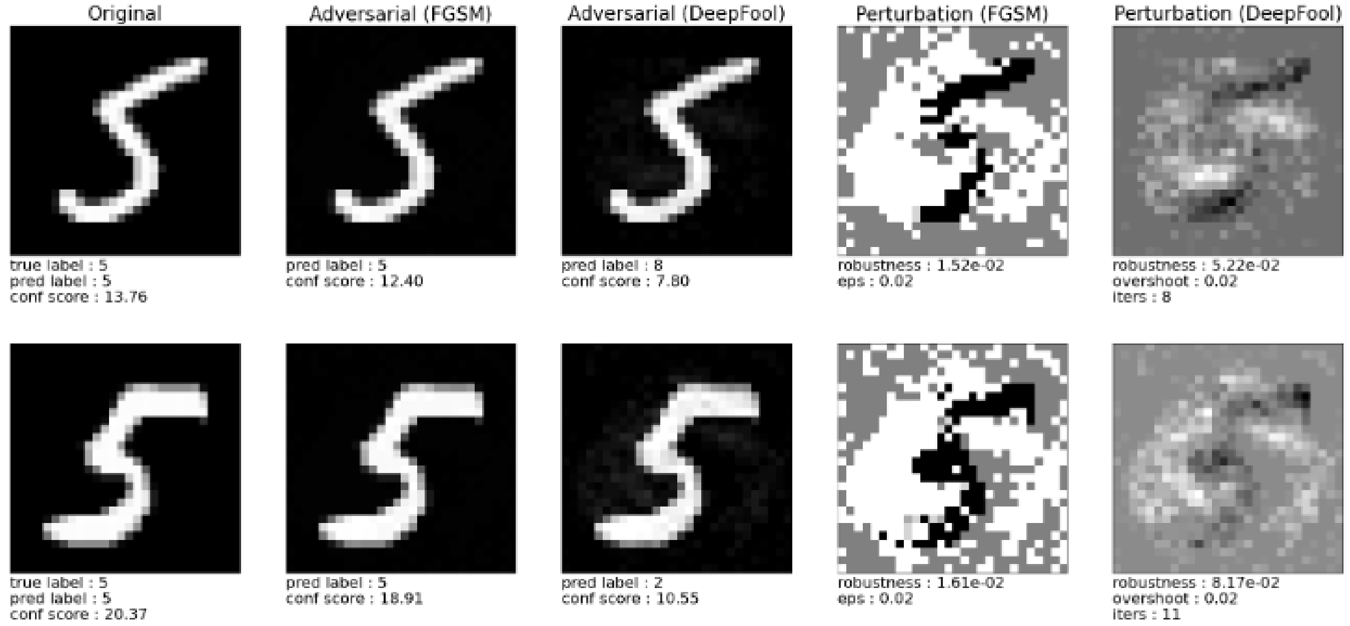


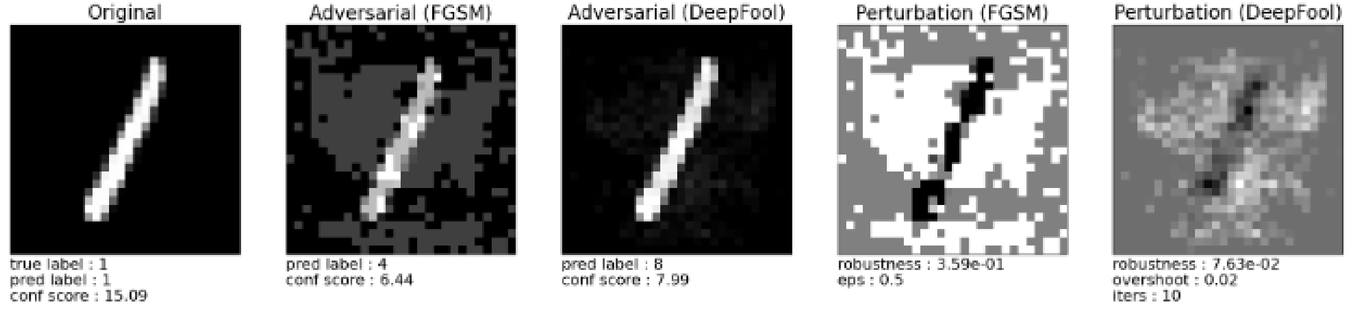
13. Проверим влияние параметра fgsm_eps для FC на датасете MNIST.

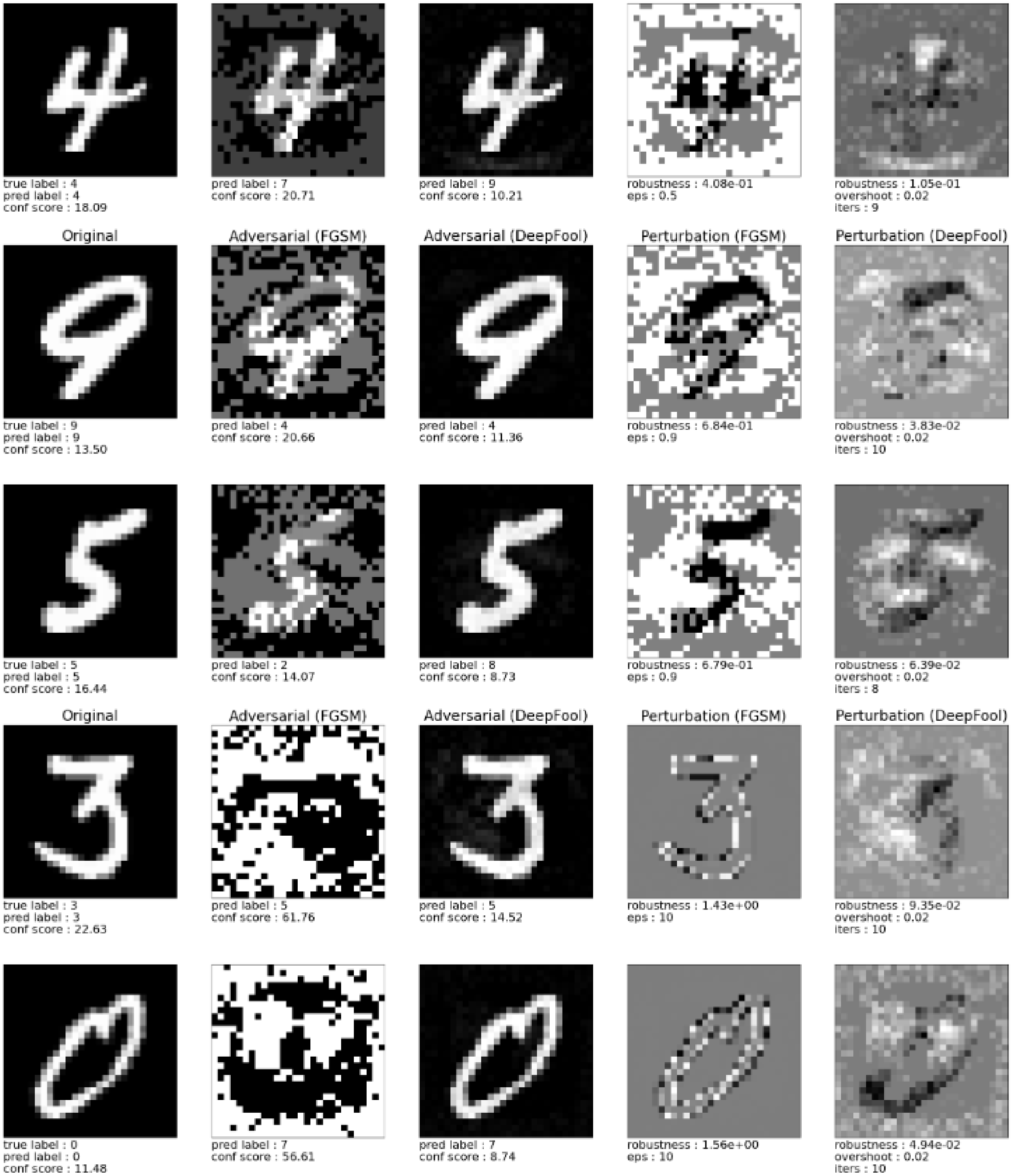


```python
fgsm_eps = 0.001
model = FC_500_150().to(device)
model.load_state_dict(torch.load('weights/clean/mnist_fc.pth'))
display_attack(device, model, mnist_test, mnist_tf_inv, mnist_min, mnist_max, fgsm_eps, deep_args, has_labels=False,
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 0.02
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,
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 0.5
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,
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 0.9
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,
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 10
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,
if device.type == 'cuda': torch.cuda.empty_cache()
```

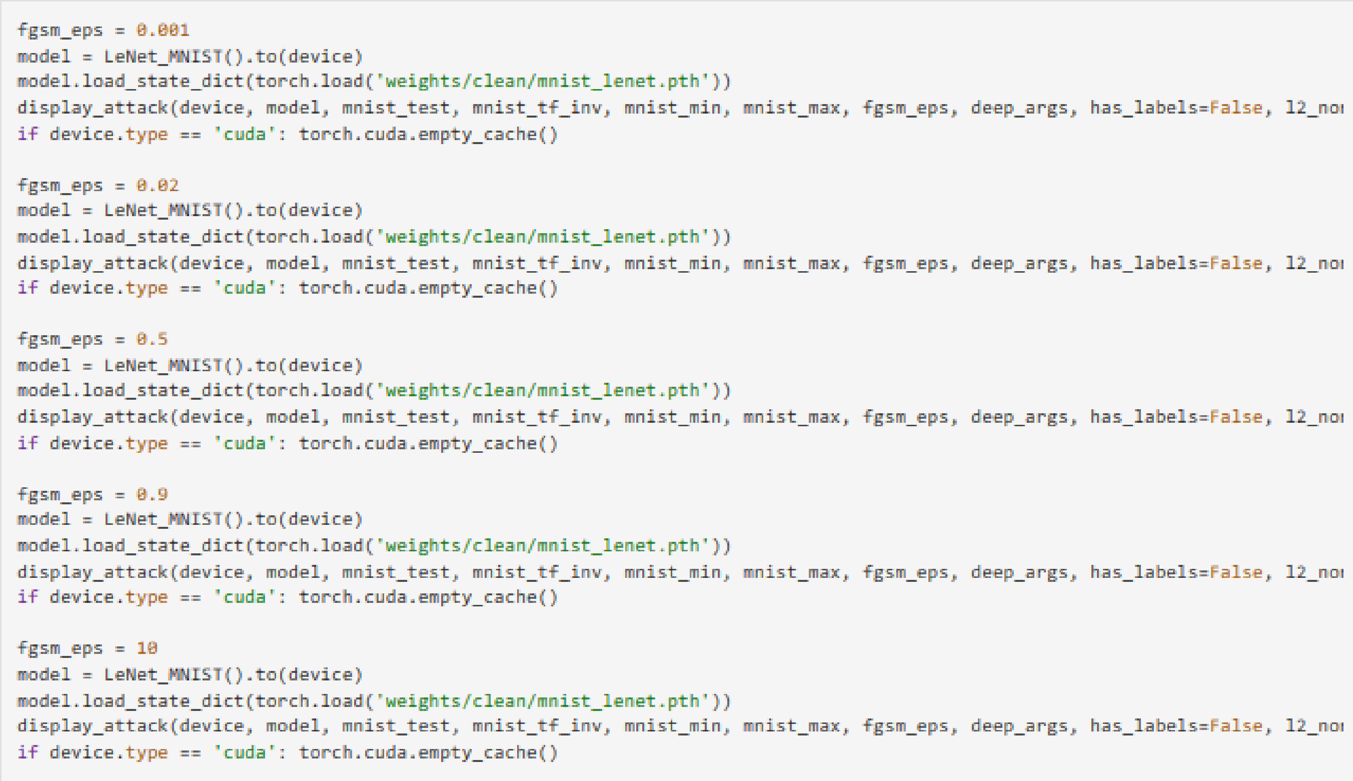

Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : 4
pred label : 4
conf score : 15.39
pred label : 4
conf score : 15.35
pred label : 9
conf score : 10.13
robustness : 8.27e-04
eps : 0.001
robustness : 8.53e-02
overshoot : 0.02
iters : 9
true label : 6
pred label : 6
conf score : 19.03
pred label : 6
conf score : 18.97
pred label : 2
conf score : 9.49
robustness : 7.79e-04
eps : 0.001
robustness : 1.13e-01
overshoot : 0.02
iters : 12


Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : 5
pred label : 5
conf score : 13.76
pred label : 5
conf score : 12.40
pred label : 8
conf score : 7.80
robustness : 1.52e-02
eps : 0.02
robustness : 5.22e-02
overshoot : 0.02
iters : 8
true label : 5
pred label : 5
conf score : 20.37
pred label : 5
conf score : 18.91
pred label : 2
conf score : 10.55
robustness : 1.61e-02
eps : 0.02
robustness : 8.17e-02
overshoot : 0.02
iters : 11

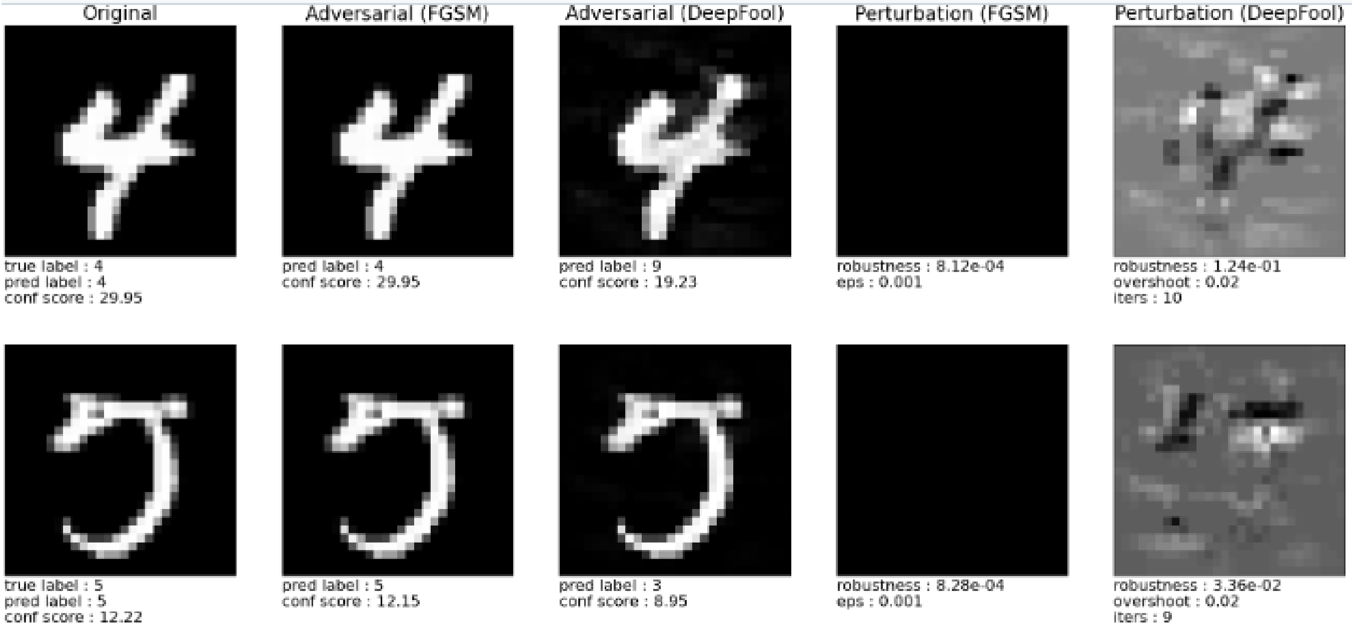

Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : 1
pred label : 1
conf score : 15.09
pred label : 4
conf score : 6.44
pred label : 8
conf score : 7.99
robustness : 3.59e-01
eps : 0.5
robustness : 7.63e-02
overshoot : 0.02
iters : 10

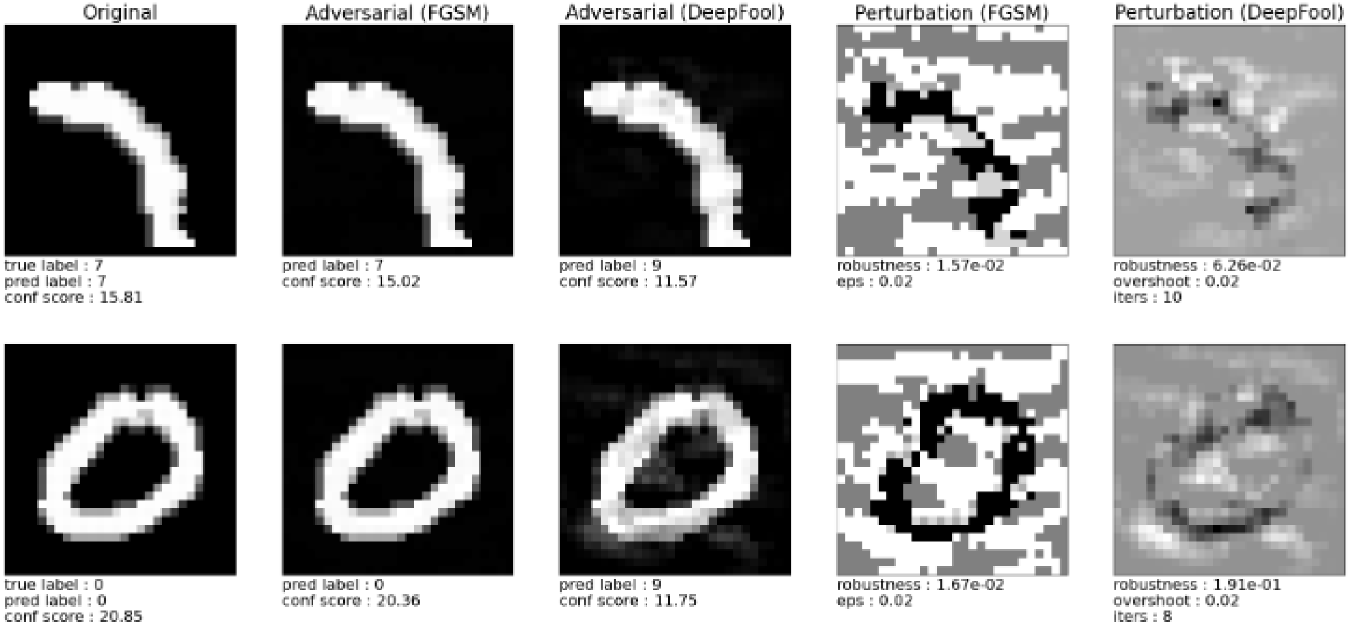

true label : 4
pred label : 4
conf score : 18.09
pred label : 7
conf score : 20.71
pred label : 9
conf score : 10.21
robustness : 4.08e-01
eps : 0.5
robustness : 1.05e-01
overshoot : 0.02
iters : 9
Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : 9
pred label : 9
conf score : 13.50
pred label : 4
conf score : 20.66
pred label : 4
conf score : 11.36
robustness : 6.84e-01
eps : 0.9
robustness : 3.83e-02
overshoot : 0.02
iters : 10
true label : 5
pred label : 5
conf score : 16.44
pred label : 2
conf score : 14.07
pred label : 8
conf score : 8.73
robustness : 6.79e-01
eps : 0.9
robustness : 6.39e-02
overshoot : 0.02
iters : 8
Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : 3
pred label : 3
conf score : 22.63
pred label : 5
conf score : 61.76
pred label : 5
conf score : 14.52
robustness : 1.43e+00
eps : 10
robustness : 9.35e-02
overshoot : 0.02
iters : 10
true label : 0
pred label : 0
conf score : 11.48
pred label : 7
conf score : 56.61
pred label : 7
conf score : 8.74
robustness : 1.56e+00
eps : 10
robustness : 4.94e-02
overshoot : 0.02
iters : 10

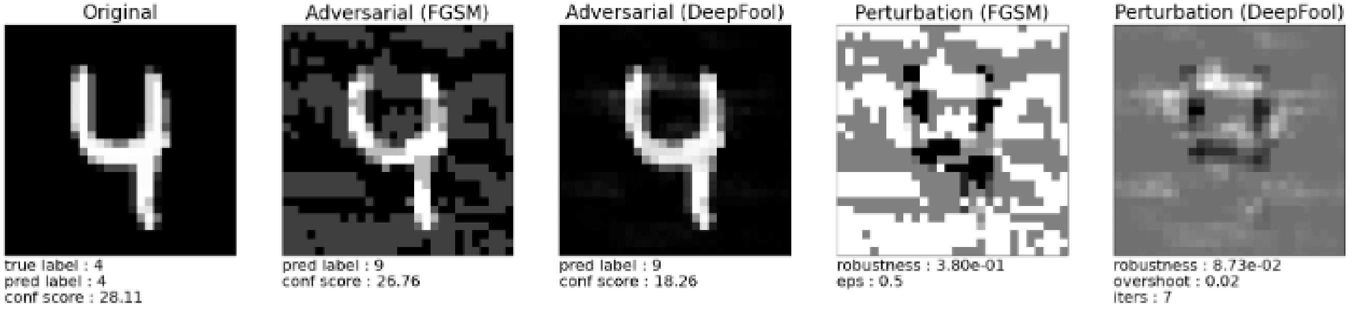
14. Проверим влияние параметра fgsm_esp для LeNet на датасете MNIST.

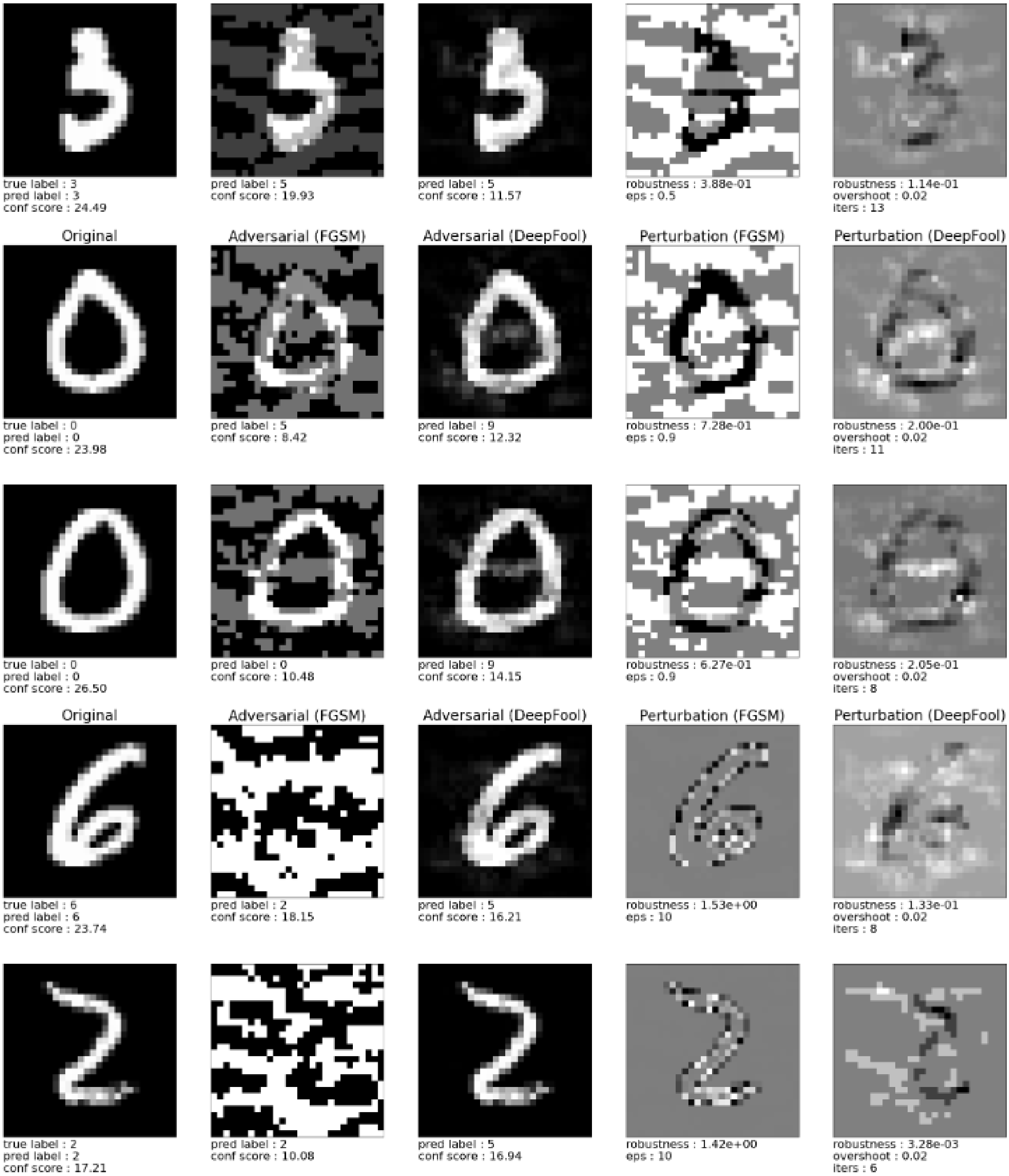


```
fgsm_eps = 0.001
model = LeNet_MNIST().to(device)
model.load_state_dict(torch.load('weights/clean/mnist_lenet.pth'))
display_attack(device, model, mnist_test, mnist_tf_inv, mnist_min, mnist_max, fgsm_eps, deep_args, has_labels=False, l2_no
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 0.02
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_no
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 0.5
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_no
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 0.9
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_no
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 10
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_no
if device.type == 'cuda': torch.cuda.empty_cache()
```

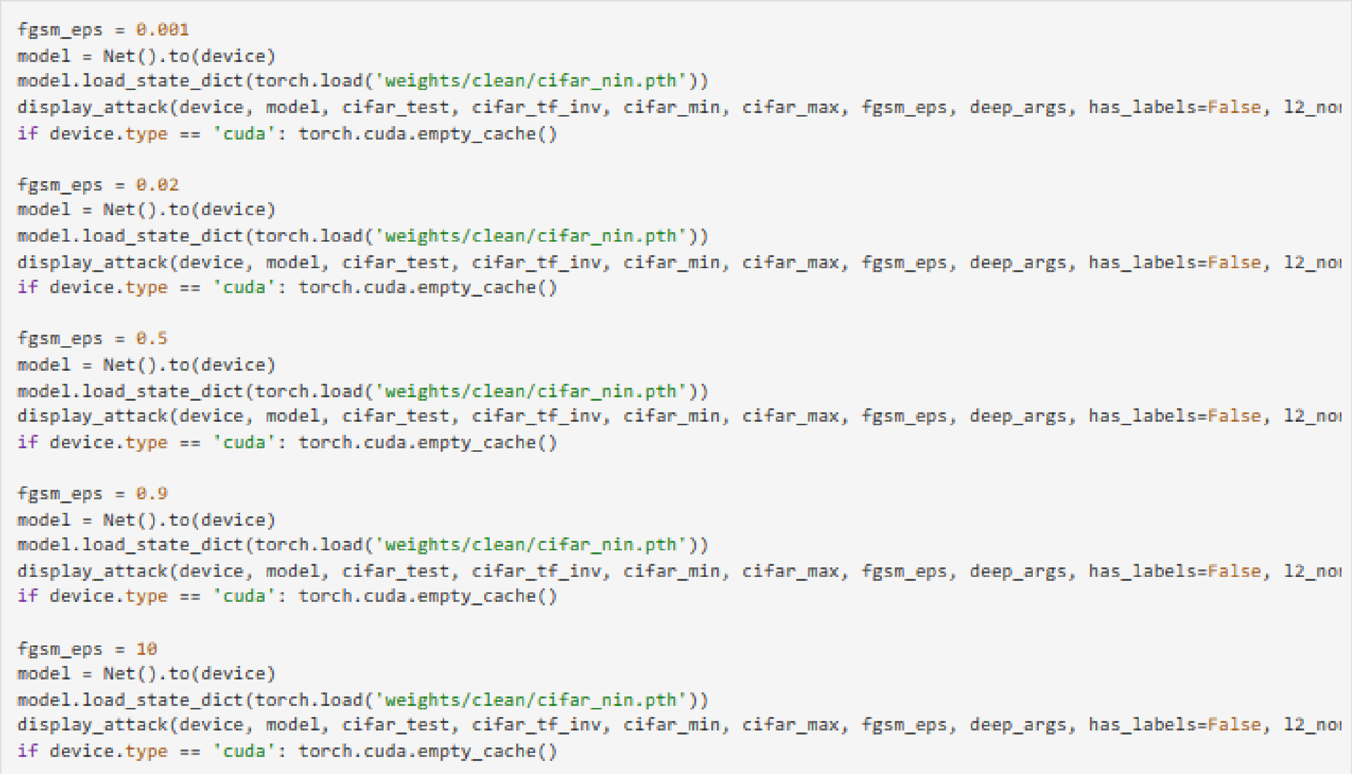

Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : 4
pred label : 4
conf score : 29.95
pred label : 4
conf score : 29.95
pred label : 9
conf score : 19.23
robustness : 8.12e-04
eps : 0.001
robustness : 1.24e-01
overshoot : 0.02
iters : 10
true label : 5
pred label : 5
conf score : 12.22
pred label : 5
conf score : 12.15
pred label : 3
conf score : 8.95
robustness : 8.28e-04
eps : 0.001
robustness : 3.36e-02
overshoot : 0.02
iters : 9


Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : 7
pred label : 7
conf score : 15.81
pred label : 7
conf score : 15.02
pred label : 9
conf score : 11.57
robustness : 1.57e-02
eps : 0.02
robustness : 6.26e-02
overshoot : 0.02
iters : 10
true label : 0
pred label : 0
conf score : 20.85
pred label : 0
conf score : 20.36
pred label : 9
conf score : 11.75
robustness : 1.67e-02
eps : 0.02
robustness : 1.91e-01
overshoot : 0.02
iters : 8

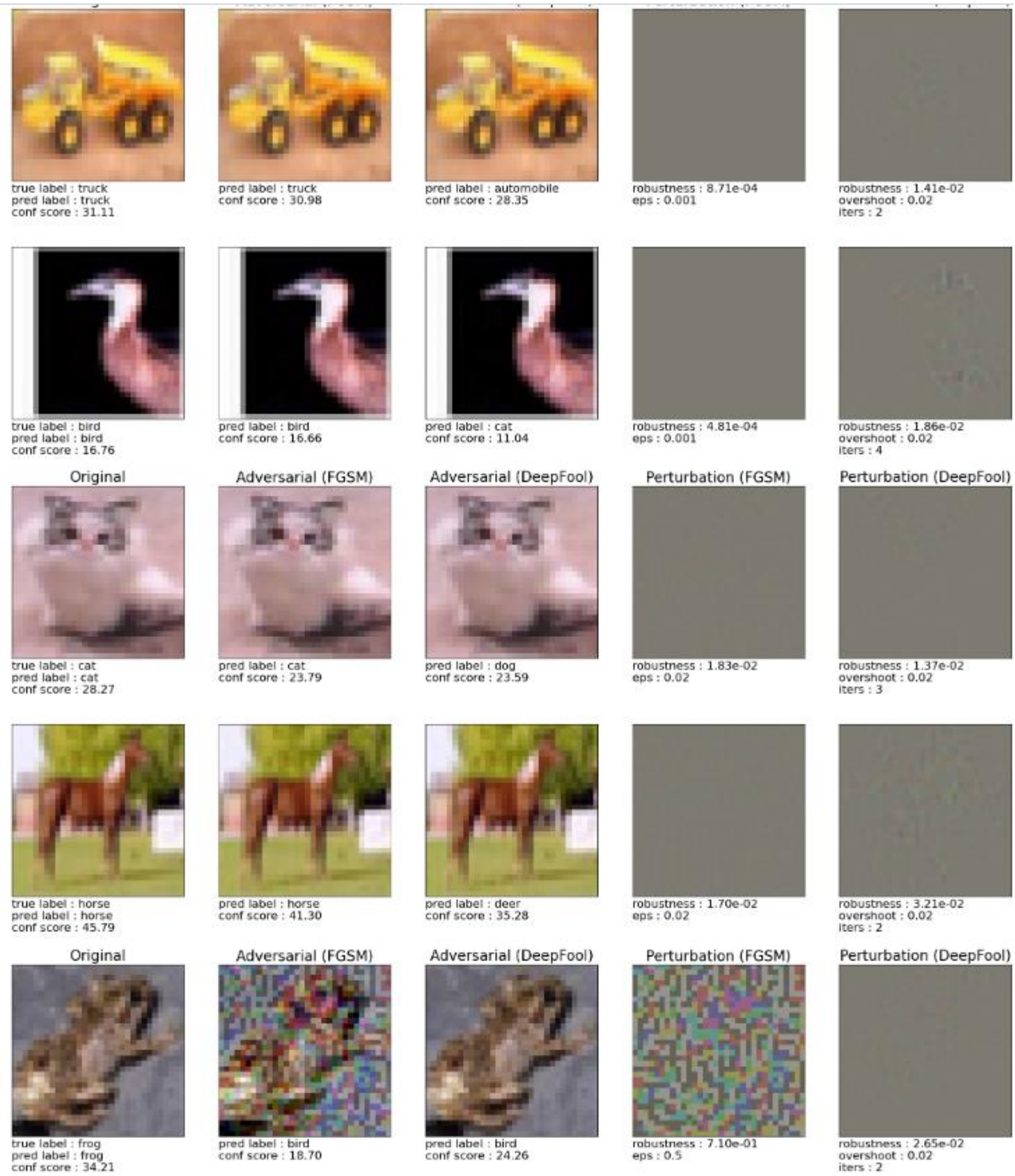

Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : 4
pred label : 4
conf score : 28.11
pred label : 9
conf score : 26.76
pred label : 9
conf score : 18.26
robustness : 3.80e-01
eps : 0.5
robustness : 8.73e-02
overshoot : 0.02
iters : 7



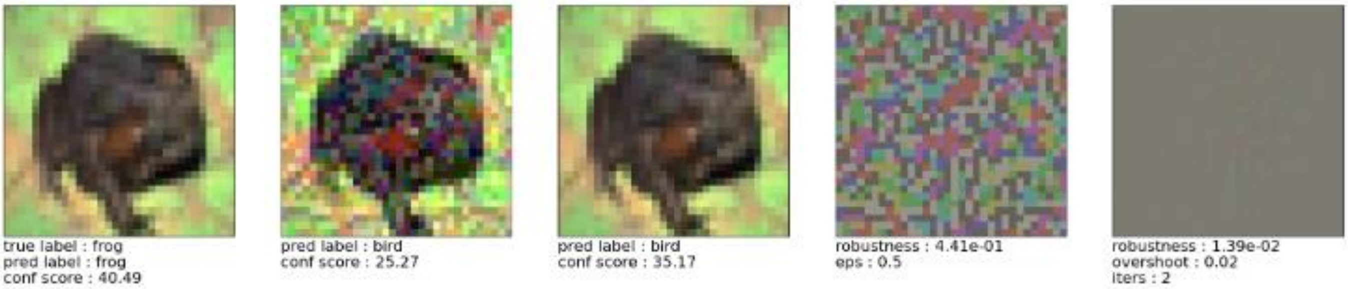
15. Проверим влияние параметра fgsm_esp для NiN на датасете Cifar-10.

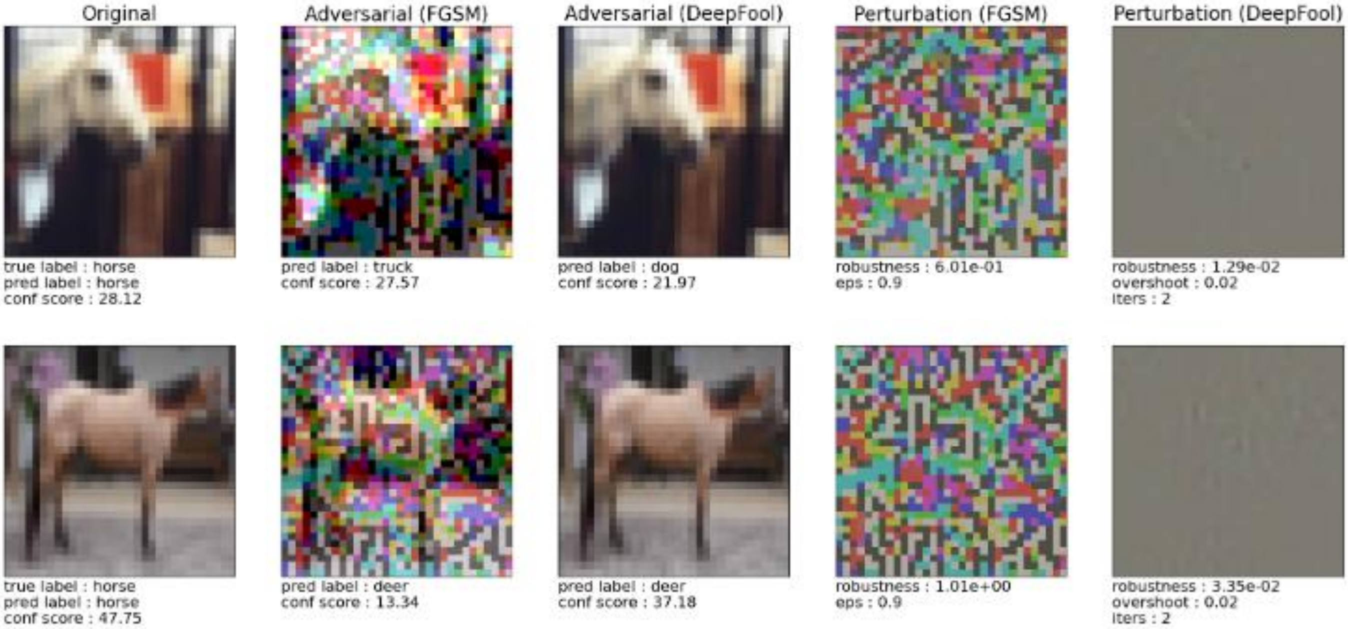


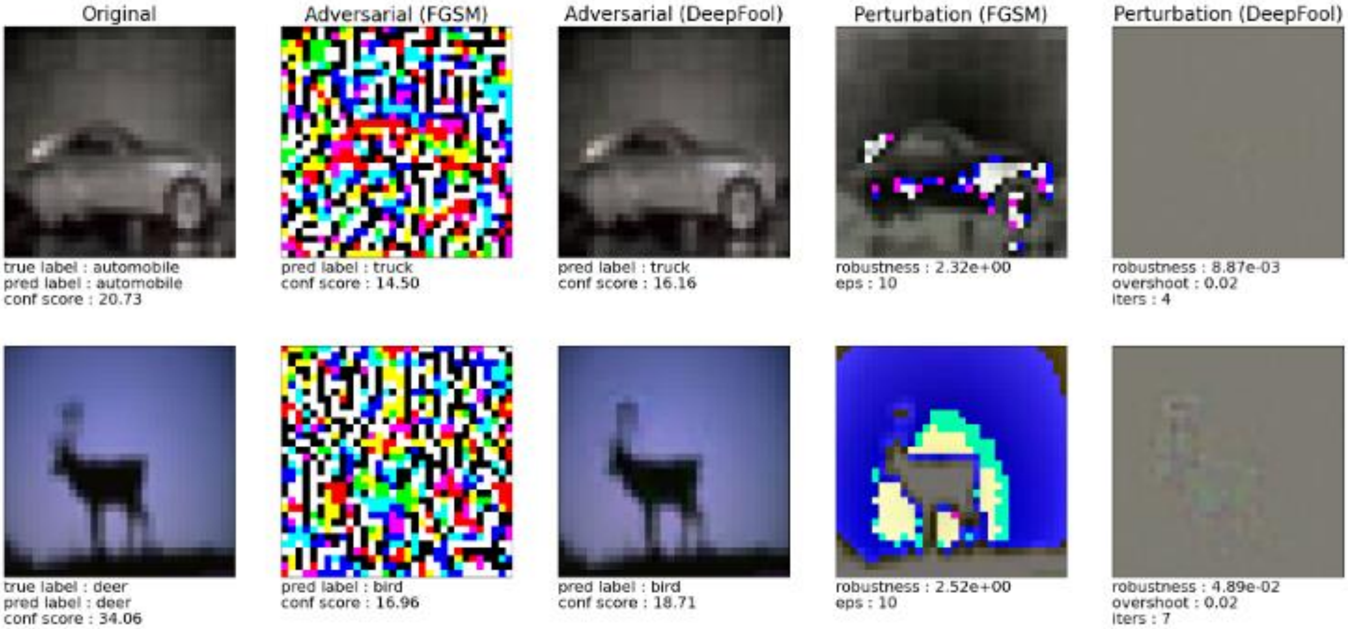
```
fgsm_eps = 0.001
model = Net().to(device)
model.load_state_dict(torch.load('weights/clean/cifar_nin.pth'))
display_attack(device, model, cifar_test, cifar_tf_inv, cifar_min, cifar_max, fgsm_eps, deep_args, has_labels=False, l2_no
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 0.02
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_no
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 0.5
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_no
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 0.9
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_no
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 10
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_no
if device.type == 'cuda': torch.cuda.empty_cache()
```



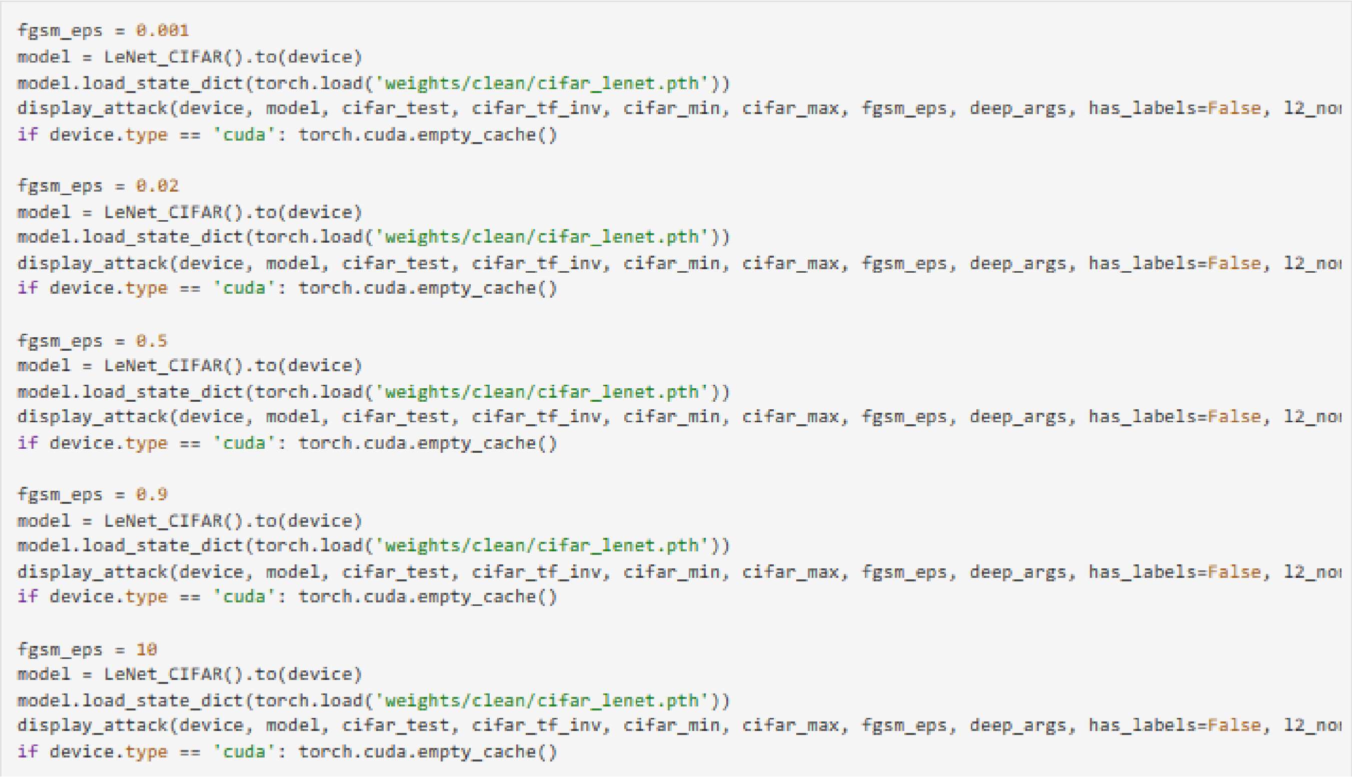
true label : truck
pred label : truck
conf score : 31.11
pred label : truck
conf score : 30.98
pred label : automobile
conf score : 28.35
robustness : 8.71e-04
eps : 0.001
robustness : 1.41e-02
overshoot : 0.02
iters : 2
true label : bird
pred label : bird
conf score : 16.76
pred label : bird
conf score : 16.66
pred label : cat
conf score : 11.04
robustness : 4.81e-04
eps : 0.001
robustness : 1.86e-02
overshoot : 0.02
iters : 4
Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : cat
pred label : cat
conf score : 28.27
pred label : cat
conf score : 23.79
pred label : dog
conf score : 23.59
robustness : 1.83e-02
eps : 0.02
robustness : 1.37e-02
overshoot : 0.02
iters : 3
true label : horse
pred label : horse
conf score : 45.79
pred label : horse
conf score : 41.30
pred label : deer
conf score : 35.28
robustness : 1.70e-02
eps : 0.02
robustness : 3.21e-02
overshoot : 0.02
iters : 2
Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : frog
pred label : frog
conf score : 34.21
pred label : bird
conf score : 18.70
pred label : bird
conf score : 24.26
robustness : 7.10e-01
eps : 0.5
robustness : 2.65e-02
overshoot : 0.02
iters : 2



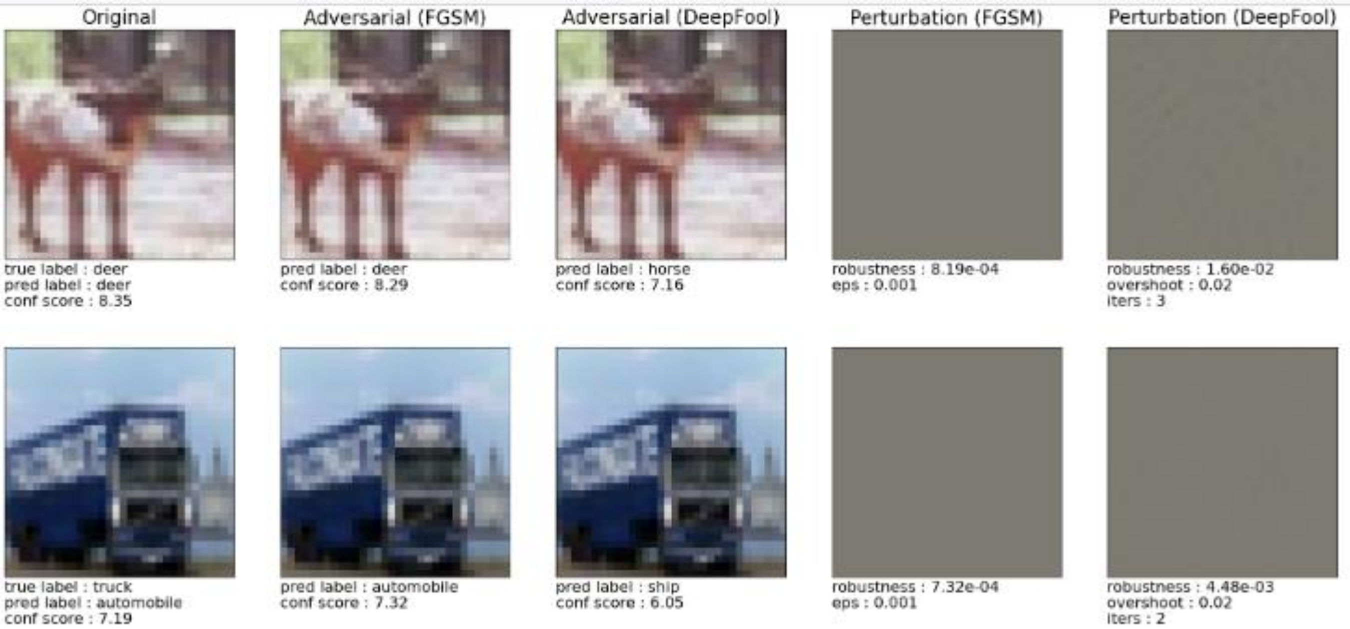


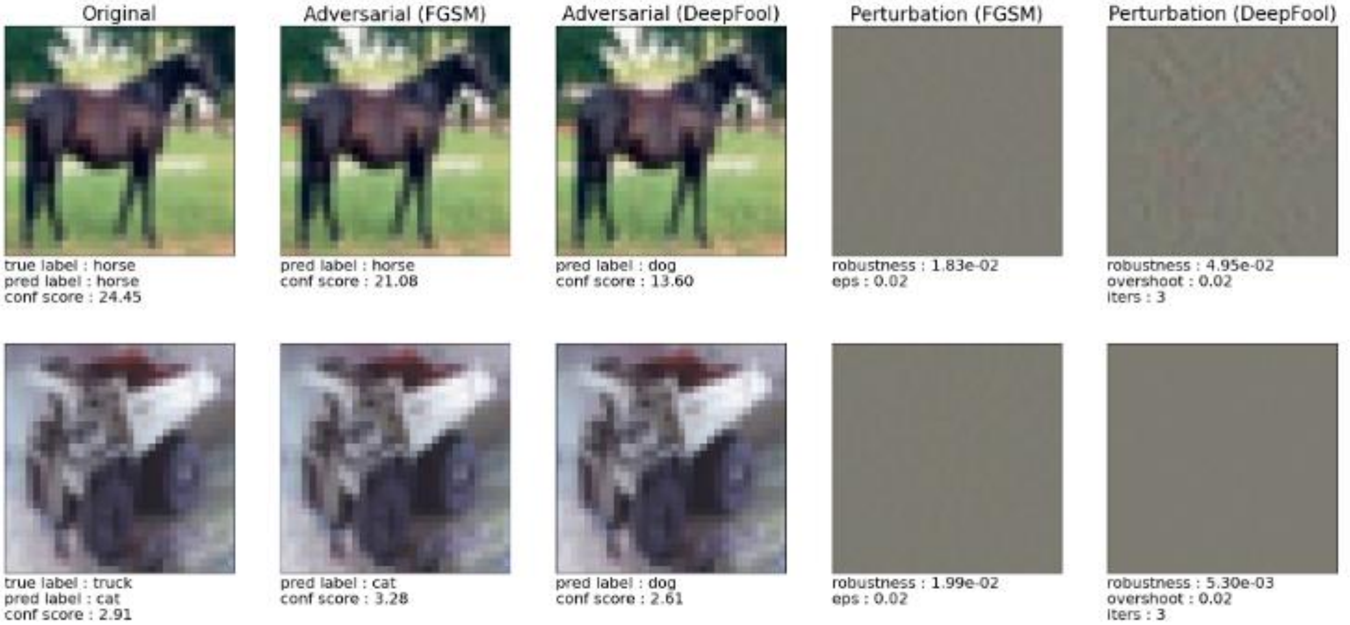


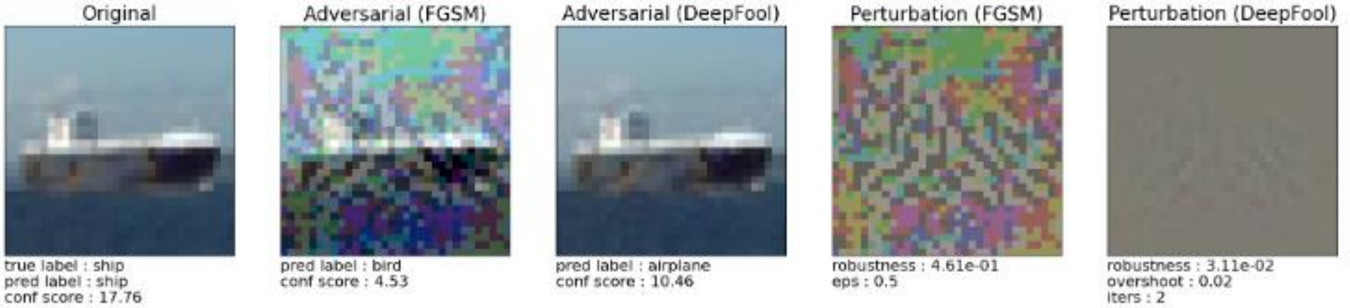
16. Проверим влияние параметра fgsm_esp для LeNet на датасете Cifar-10.

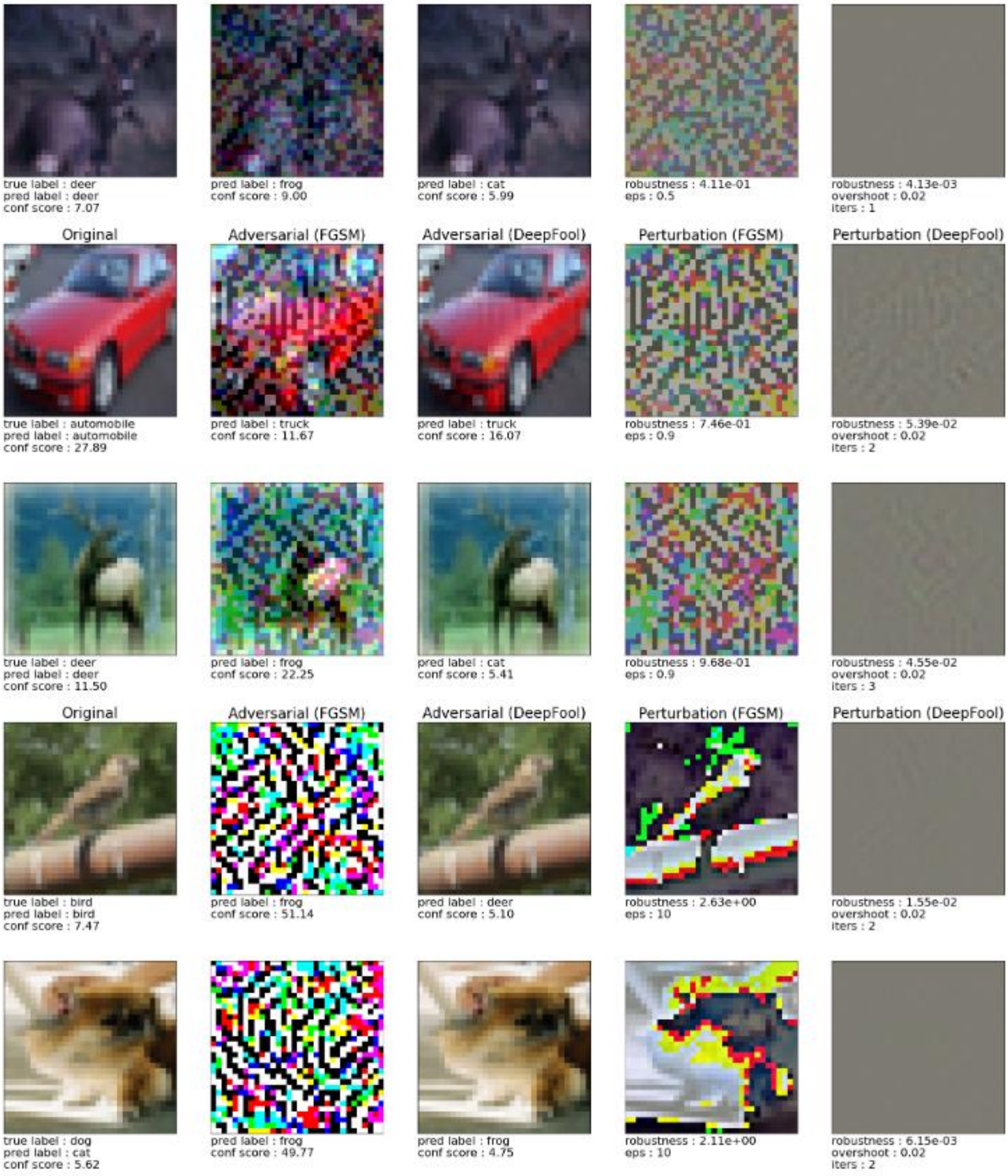


```
fgsm_eps = 0.001
model = LeNet_CIFAR().to(device)
model.load_state_dict(torch.load('weights/clean/cifar_lenet.pth'))
display_attack(device, model, cifar_test, cifar_tf_inv, cifar_min, cifar_max, fgsm_eps, deep_args, has_labels=False, l2_no
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 0.02
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_no
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 0.5
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_no
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 0.9
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_no
if device.type == 'cuda': torch.cuda.empty_cache()

fgsm_eps = 10
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_no
if device.type == 'cuda': torch.cuda.empty_cache()
```


Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : deer
pred label : deer
conf score : 8.35
pred label : deer
conf score : 8.29
pred label : horse
conf score : 7.16
robustness : 8.19e-04
eps : 0.001
robustness : 1.60e-02
overshoot : 0.02
iters : 3
true label : truck
pred label : automobile
conf score : 7.19
pred label : automobile
conf score : 7.32
pred label : ship
conf score : 6.05
robustness : 7.32e-04
eps : 0.001
robustness : 4.48e-03
overshoot : 0.02
iters : 2


Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : horse
pred label : horse
conf score : 24.45
pred label : horse
conf score : 21.08
pred label : dog
conf score : 13.60
robustness : 1.83e-02
eps : 0.02
robustness : 4.95e-02
overshoot : 0.02
iters : 3
true label : truck
pred label : cat
conf score : 2.91
pred label : cat
conf score : 3.28
pred label : dog
conf score : 2.61
robustness : 1.99e-02
eps : 0.02
robustness : 5.30e-03
overshoot : 0.02
iters : 3


Original
Adversarial (FGSM)
Adversarial (DeepFool)
Perturbation (FGSM)
Perturbation (DeepFool)
true label : ship
pred label : ship
conf score : 17.76
pred label : bird
conf score : 4.53
pred label : airplane
conf score : 10.46
robustness : 4.61e-01
eps : 0.5
robustness : 3.11e-02
overshoot : 0.02
iters : 2



Вывод: параметр fgsm_esp влияет на устойчивость сети. При увеличении значения fgsm_esp сети становятся уязвимыми к атакам.