МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА –Российский технологический университет»

РТУ МИРЭА

Институт кибербезопасности и цифровых технологий

Кафедра КБ-4 «Интеллектуальные системы информационной безопасности»

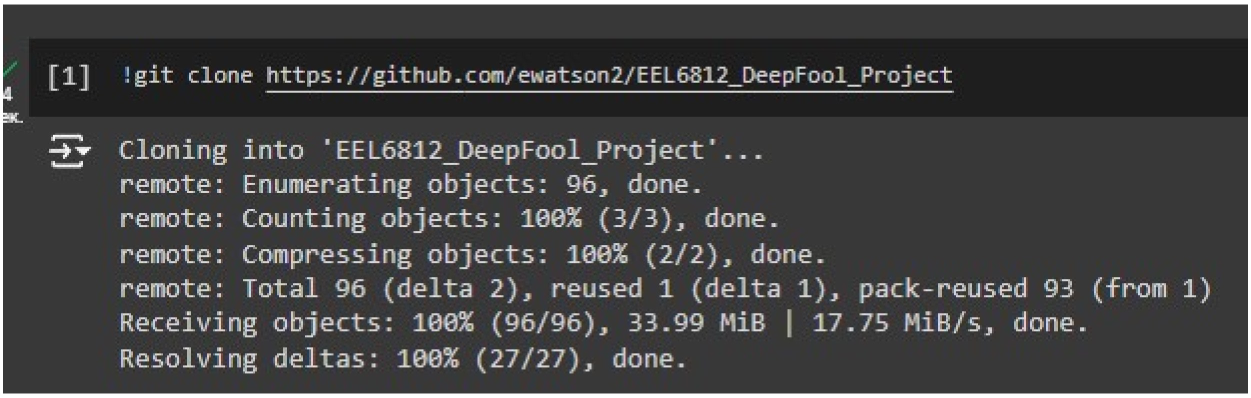
Дисциплина «Анализ защищенности систем искусственного интеллекта»

Отчет
о проделанной лабораторной работе №1
«EEL6812 DeepFool Project»

Выполнил студент 2 курса
Группы: ББМО-01-23
Чурсинов Герман Сергеевич

Москва, 2024

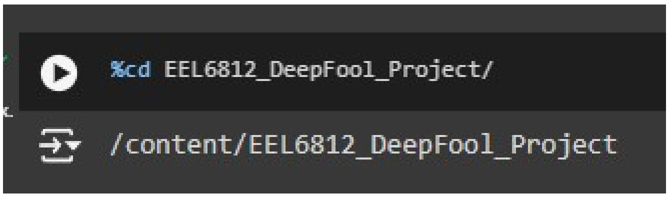
1. Скопировать проект по ссылке в локальную среду выполнения Jupyter (Google Colab)



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

```
Cloning into 'EEL6812_DeepFool_Project'...
remote: Enumerating objects: 96, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 96 (delta 2), reused 1 (delta 1), pack-reused 93 (from 1)
Receiving objects: 100% (96/96), 33.99 MiB | 17.75 MiB/s, done.
Resolving deltas: 100% (27/27), done.
```

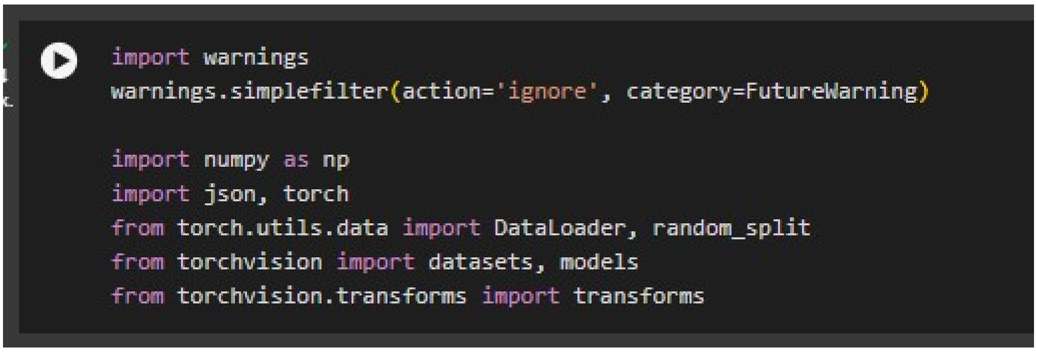
2. Сменить директорию исполнения на вновь созданную папку "EEL6812_DeepFool_Project" проекта.



```
%cd EEL6812_DeepFool_Project/
```

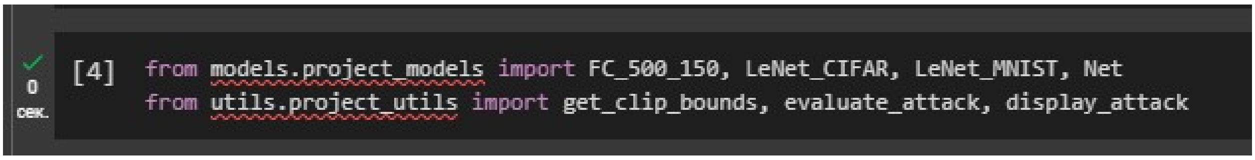
```
/content/EEL6812_DeepFool_Project
```

3. Выполнить импорт библиотек



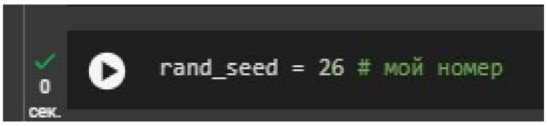
```
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

import numpy as np
import json, torch
from torch.utils.data import DataLoader, random_split
from torchvision import datasets, models
from torchvision.transforms import transforms
```

4. Выполнить импорт вспомогательных библиотек из локальных файлов проекта:



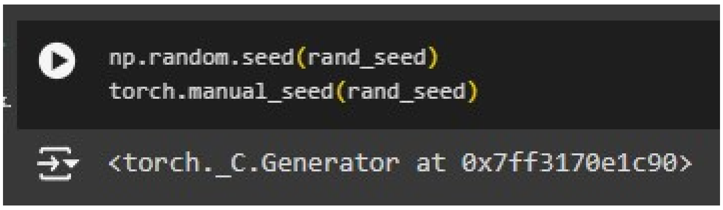
```
from models.project_models import FC_500_150, LeNet_CIFAR, LeNet_MNIST, Net
from utils.project_utils import get_clip_bounds, evaluate_attack, display_attack
```

5. Установить случайное рандомное значение в виде переменной rand_seed



```
rand_seed = 26 # мой номер
```

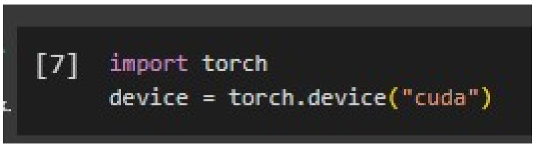
6. Установить указанное значение для np.random.seed и torch.manual_seed



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

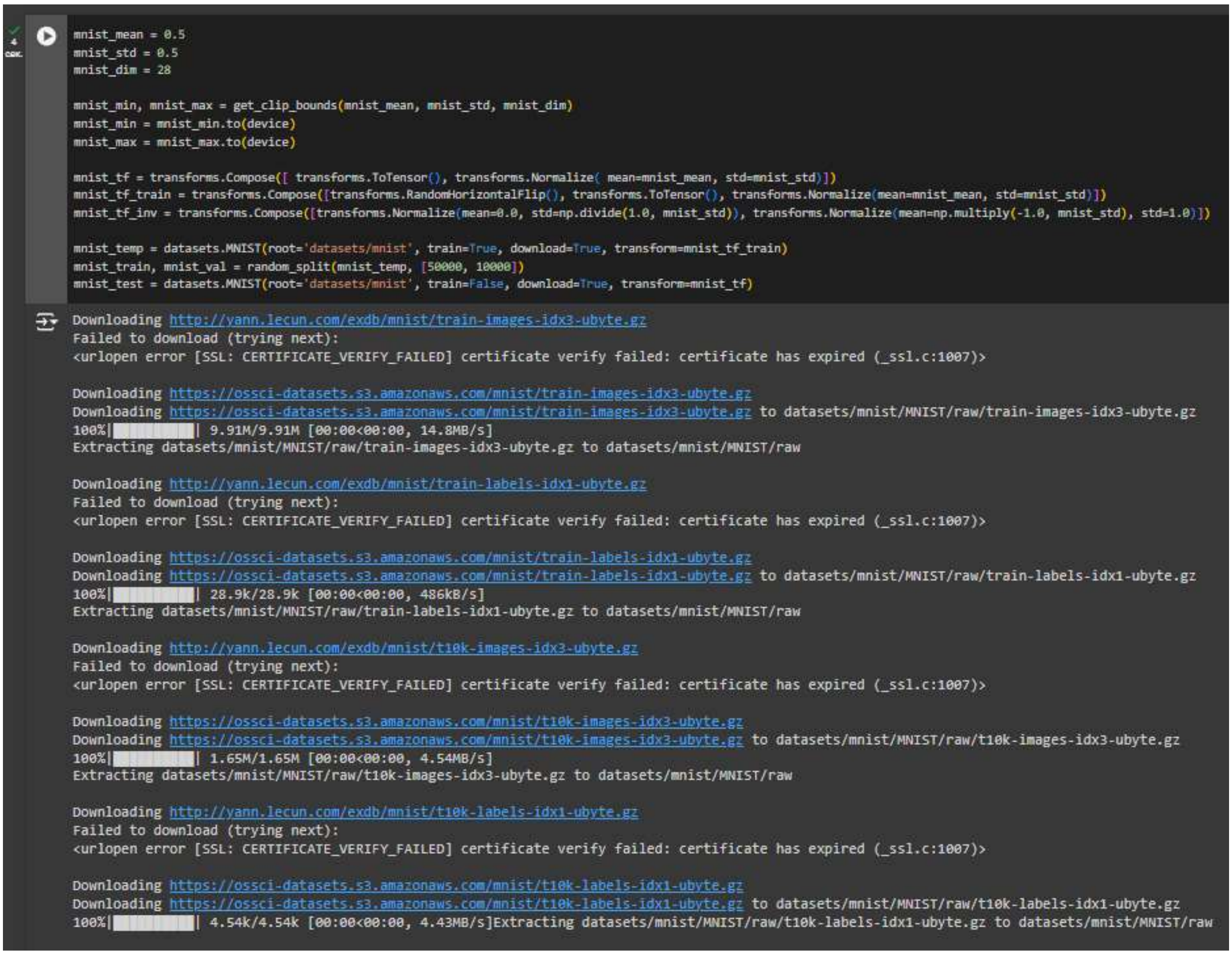
```
<torch._C.Generator at 0x7ff3170e1c90>
```

7. Использовать в качестсве устройства видеокарту



```
import torch
device = torch.device("cuda")
```

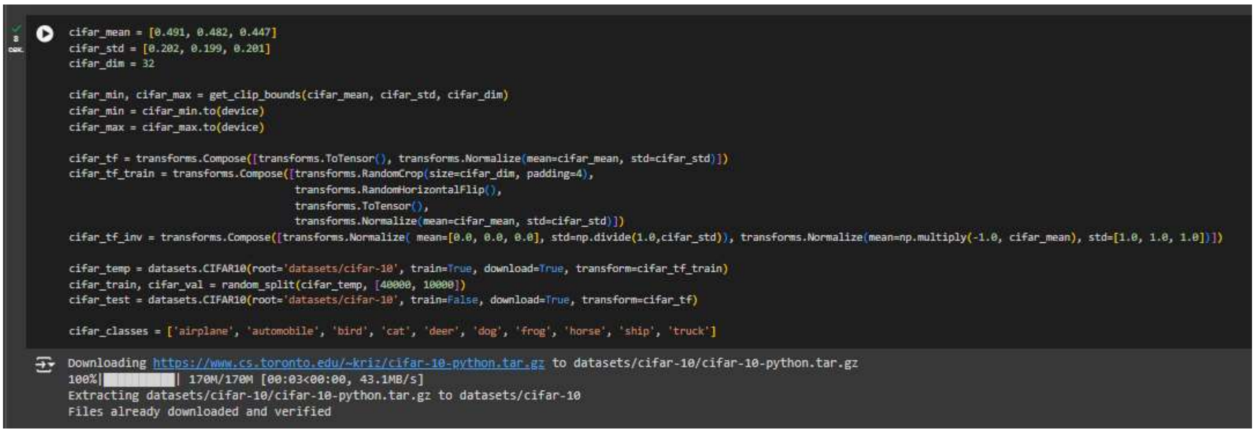
8. Загрузить датасет MNIST с параметрами mnist_mean = 0.5, mnist_std = 0.5, mnist_dim = 28



```
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)
```

```
Downloading http://yann.lecun.com/exdb/mnist/train-images-idx3-ubyte.gz
Failed to download (trying next):
<urlopen error [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: certificate has expired (_ssl.c:1007)>

Downloading https://ossci-datasets.s3.amazonaws.com/mnist/train-images-idx3-ubyte.gz
Downloading https://ossci-datasets.s3.amazonaws.com/mnist/train-images-idx3-ubyte.gz to datasets/mnist/MNIST/raw/train-images-idx3-ubyte.gz
100%|██████████| 9.91M/9.91M [00:00<00:00, 14.8MB/s]
Extracting datasets/mnist/MNIST/raw/train-images-idx3-ubyte.gz to datasets/mnist/MNIST/raw

Downloading http://yann.lecun.com/exdb/mnist/train-labels-idx1-ubyte.gz
Failed to download (trying next):
<urlopen error [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: certificate has expired (_ssl.c:1007)>

Downloading https://ossci-datasets.s3.amazonaws.com/mnist/train-labels-idx1-ubyte.gz
Downloading https://ossci-datasets.s3.amazonaws.com/mnist/train-labels-idx1-ubyte.gz to datasets/mnist/MNIST/raw/train-labels-idx1-ubyte.gz
100%|██████████| 28.9k/28.9k [00:00<00:00, 486kB/s]
Extracting datasets/mnist/MNIST/raw/train-labels-idx1-ubyte.gz to datasets/mnist/MNIST/raw

Downloading http://yann.lecun.com/exdb/mnist/t10k-images-idx3-ubyte.gz
Failed to download (trying next):
<urlopen error [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: certificate has expired (_ssl.c:1007)>

Downloading https://ossci-datasets.s3.amazonaws.com/mnist/t10k-images-idx3-ubyte.gz
Downloading https://ossci-datasets.s3.amazonaws.com/mnist/t10k-images-idx3-ubyte.gz to datasets/mnist/MNIST/raw/t10k-images-idx3-ubyte.gz
100%|██████████| 1.65M/1.65M [00:00<00:00, 4.54MB/s]
Extracting datasets/mnist/MNIST/raw/t10k-images-idx3-ubyte.gz to datasets/mnist/MNIST/raw

Downloading http://yann.lecun.com/exdb/mnist/t10k-labels-idx1-ubyte.gz
Failed to download (trying next):
<urlopen error [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: certificate has expired (_ssl.c:1007)>

Downloading https://ossci-datasets.s3.amazonaws.com/mnist/t10k-labels-idx1-ubyte.gz
Downloading https://ossci-datasets.s3.amazonaws.com/mnist/t10k-labels-idx1-ubyte.gz to datasets/mnist/MNIST/raw/t10k-labels-idx1-ubyte.gz
100%|██████████| 4.54k/4.54k [00:00<00:00, 4.43MB/s]Extracting datasets/mnist/MNIST/raw/t10k-labels-idx1-ubyte.gz to datasets/mnist/MNIST/raw
```

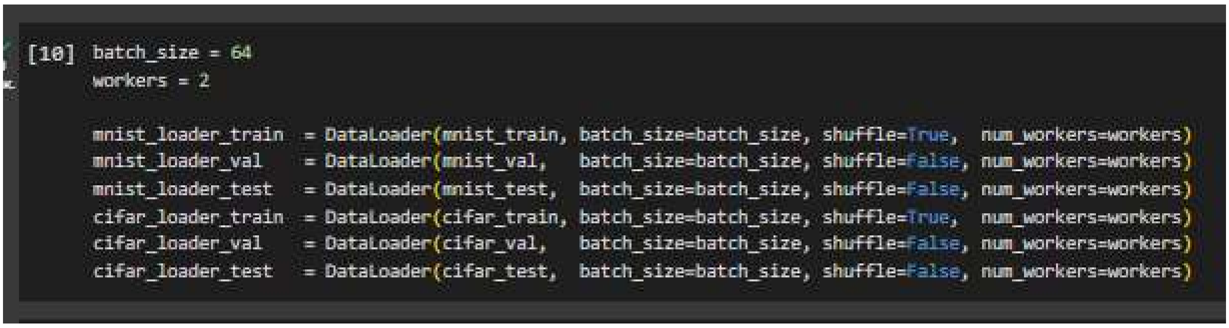
9. Загрузить датасет CIFAR-10 с параметрами cifar_mean = [0.491, 0.482, 0.447] cifar_std = [0.202, 0.199, 0.201] cifar_dim = 32



```
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']
```

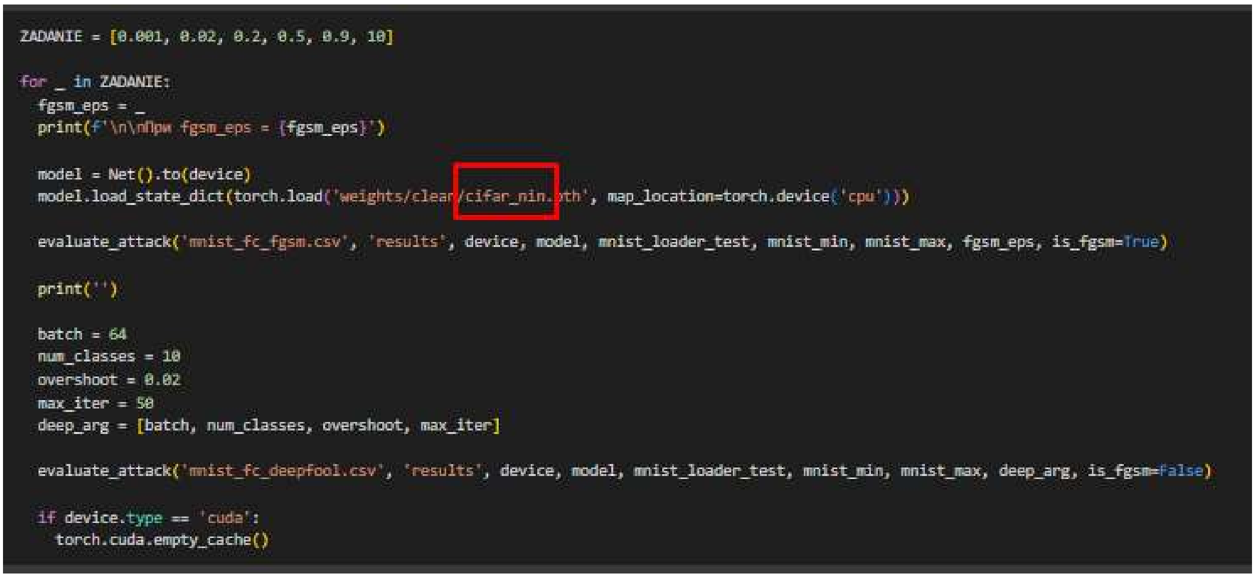
```
Downloading https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz to datasets/cifar-10/cifar-10-python.tar.gz
100%|██████████| 170M/170M [00:03<00:00, 43.1MB/s]
Extracting datasets/cifar-10/cifar-10-python.tar.gz to datasets/cifar-10
Files already downloaded and verified
```

10. Выполнить настройку и загрузку DataLoader batch_size = 64 workers = 2



```
[10] batch_size = 64
     workers = 2

     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)
```

11. Загрузить и оценить стойкость модели Network-In-Network Model к FGSM и DeepFool атакам на основе датасета CIFAR-10

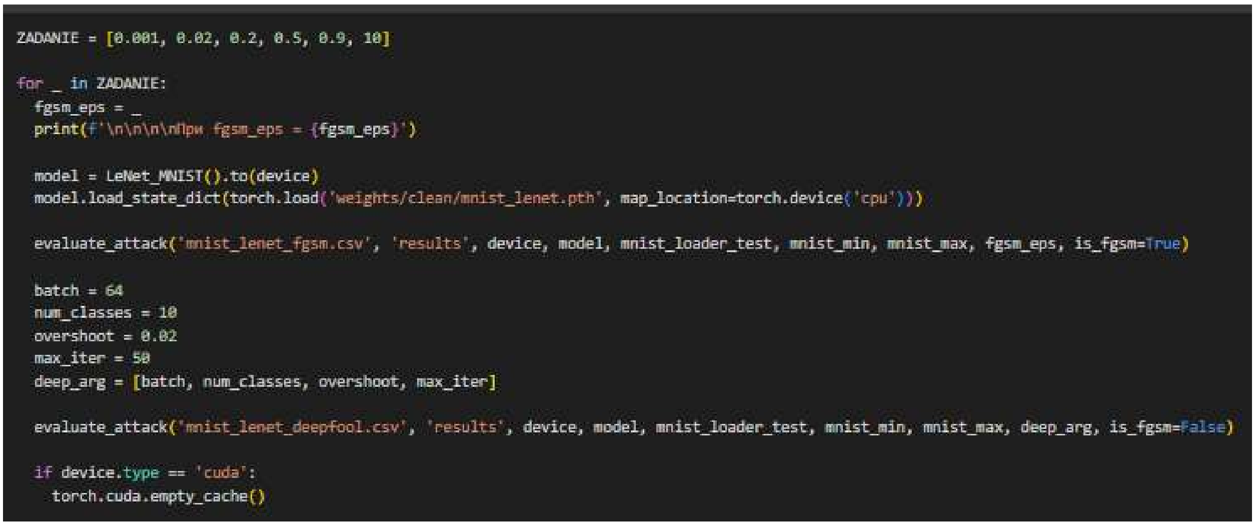


```
ZADANIE = [0.001, 0.02, 0.2, 0.5, 0.9, 10]

for _ in ZADANIE:
  fgsm_eps = _
  print(f'\n\nПри fgsm_eps = {fgsm_eps}')

  model = Net().to(device)
  model.load_state_dict(torch.load('weights/clean/cifar_nin.pth', map_location=torch.device('cpu')))

  evaluate_attack('mnist_fc_fgsm.csv', 'results', device, model, mnist_loader_test, mnist_min, mnist_max, fgsm_eps, is_fgsm=True)

  print('')

  batch = 64
  num_classes = 10
  overshoot = 0.02
  max_iter = 50
  deep_arg = [batch, num_classes, overshoot, max_iter]

  evaluate_attack('mnist_fc_deepfool.csv', 'results', device, model, mnist_loader_test, mnist_min, mnist_max, deep_arg, is_fgsm=False)

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

```
При fgsm_eps = 0.001
FGSM Test Error : 87.08%
FGSM Robustness : 1.56e-01
FGSM Time (All Images) : 0.15 s
FGSM Time (Per Image) : 14.99 us

DeepFool Test Error : 97.92%
DeepFool Robustness : 6.78e-02
DeepFool Time (All Images) : 141.81 s
DeepFool Time (Per Image) : 14.18 ms
```

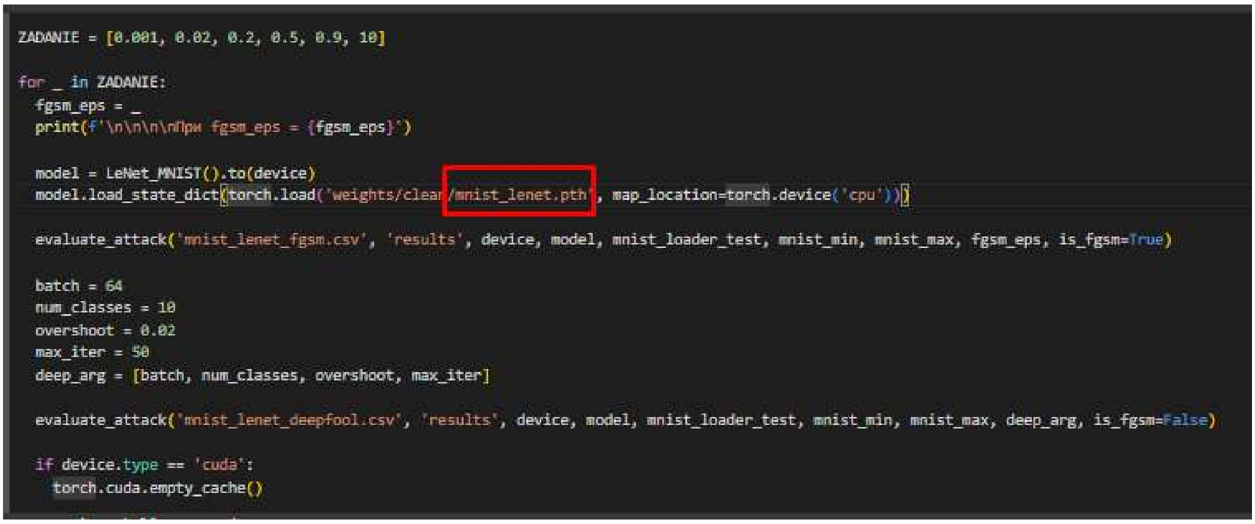
```
При fgsm_eps = 0.02
FGSM Test Error : 87.08%
FGSM Robustness : 1.56e-01
FGSM Time (All Images) : 0.15 s
FGSM Time (Per Image) : 14.99 us

DeepFool Test Error : 97.92%
DeepFool Robustness : 6.78e-02
DeepFool Time (All Images) : 141.81 s
DeepFool Time (Per Image) : 14.18 ms


При fgsm_eps = 0.2
FGSM Test Error : 87.08%
FGSM Robustness : 1.56e-01
FGSM Time (All Images) : 0.15 s
FGSM Time (Per Image) : 14.99 us

DeepFool Test Error : 97.92%
DeepFool Robustness : 6.78e-02
DeepFool Time (All Images) : 141.81 s
DeepFool Time (Per Image) : 14.18 ms


При fgsm_eps = 0.5
FGSM Test Error : 87.08%
FGSM Robustness : 1.56e-01
FGSM Time (All Images) : 0.15 s
FGSM Time (Per Image) : 14.99 us

DeepFool Test Error : 97.92%
DeepFool Robustness : 6.78e-02
DeepFool Time (All Images) : 141.81 s
DeepFool Time (Per Image) : 14.18 ms


При fgsm_eps = 0.9
FGSM Test Error : 87.08%
FGSM Robustness : 1.56e-01
FGSM Time (All Images) : 0.15 s
FGSM Time (Per Image) : 14.99 us

DeepFool Test Error : 97.92%
DeepFool Robustness : 6.78e-02
DeepFool Time (All Images) : 141.81 s
DeepFool Time (Per Image) : 14.18 ms


При fgsm_eps = 10
FGSM Test Error : 87.08%
FGSM Robustness : 1.56e-01
FGSM Time (All Images) : 0.15 s
FGSM Time (Per Image) : 14.99 us

DeepFool Test Error : 97.92%
DeepFool Robustness : 6.78e-02
DeepFool Time (All Images) : 141.81 s
DeepFool Time (Per Image) : 14.18 ms
```

12. Загрузить и оценить стойкость модели LeNet к FGSM и DeepFool атакам на основе датасета CIFAR-10



```
ZADANIE = [0.001, 0.02, 0.2, 0.5, 0.9, 10]

for _ in ZADANIE:
  fgsm_eps = _
  print(f'\n\n\n\nПри fgsm_eps = {fgsm_eps}')

  model = LeNet_MNIST().to(device)
  model.load_state_dict(torch.load('weights/clean/mnist_lenet.pth', map_location=torch.device('cpu')))

  evaluate_attack('mnist_lenet_fgsm.csv', 'results', device, model, mnist_loader_test, mnist_min, mnist_max, fgsm_eps, is_fgsm=True)

  batch = 64
  num_classes = 10
  overshoot = 0.02
  max_iter = 50
  deep_arg = [batch, num_classes, overshoot, max_iter]

  evaluate_attack('mnist_lenet_deepfool.csv', 'results', device, model, mnist_loader_test, mnist_min, mnist_max, deep_arg, is_fgsm=False)

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

13. Выполнить оценку атакующих примеров для сетей: LeNet

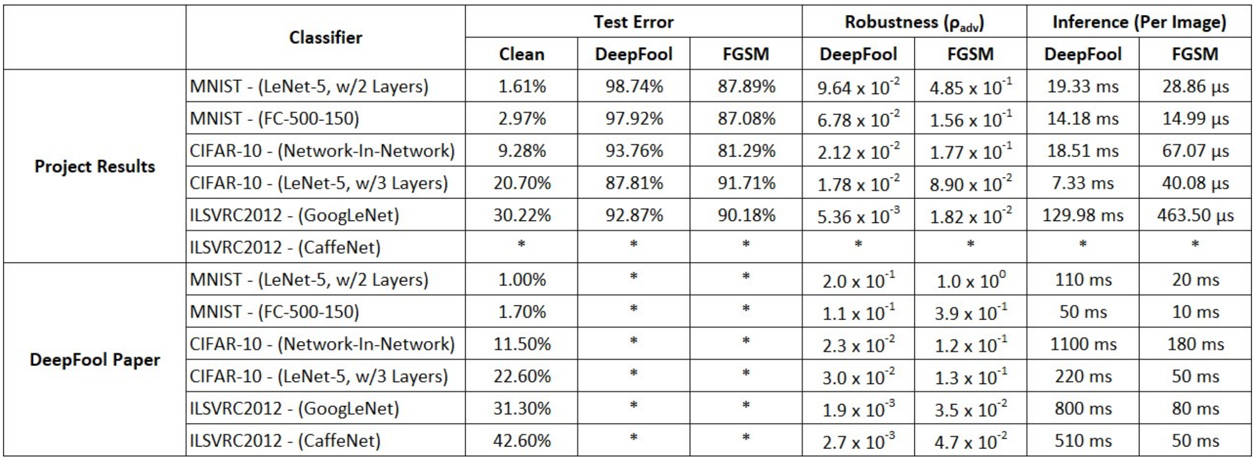


```
ZADANIE = [0.001, 0.02, 0.2, 0.5, 0.9, 10]

for _ in ZADANIE:
  fgsm_eps = _
  print(f'\n\n\n\nПри fgsm_eps = {fgsm_eps}')

  model = LeNet_MNIST().to(device)
  model.load_state_dict(torch.load('weights/clean/mnist_lenet.pth', map_location=torch.device('cpu')))

  evaluate_attack('mnist_lenet_fgsm.csv', 'results', device, model, mnist_loader_test, mnist_min, mnist_max, fgsm_eps, is_fgsm=True)

  batch = 64
  num_classes = 10
  overshoot = 0.02
  max_iter = 50
  deep_arg = [batch, num_classes, overshoot, max_iter]

  evaluate_attack('mnist_lenet_deepfool.csv', 'results', device, model, mnist_loader_test, mnist_min, mnist_max, deep_arg, is_fgsm=False)

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

```
При fgsm_eps = 0.001
FGSM Test Error : 87.89%
FGSM Robustness : 4.58e-01
FGSM Time (All Images) : 0.29 s
FGSM Time (Per Image) : 28.86 us
DeepFool Test Error : 98.74%
DeepFool Robustness : 9.64e-02
DeepFool Time (All Images) : 193.32 s
DeepFool Time (Per Image) : 19.33 ms

При fgsm_eps = 0.02
FGSM Test Error : 87.89%
FGSM Robustness : 4.58e-01
FGSM Time (All Images) : 0.29 s
FGSM Time (Per Image) : 28.86 us
DeepFool Test Error : 98.74%
DeepFool Robustness : 9.64e-02
DeepFool Time (All Images) : 193.32 s
DeepFool Time (Per Image) : 19.33 ms

При fgsm_eps = 0.2
```

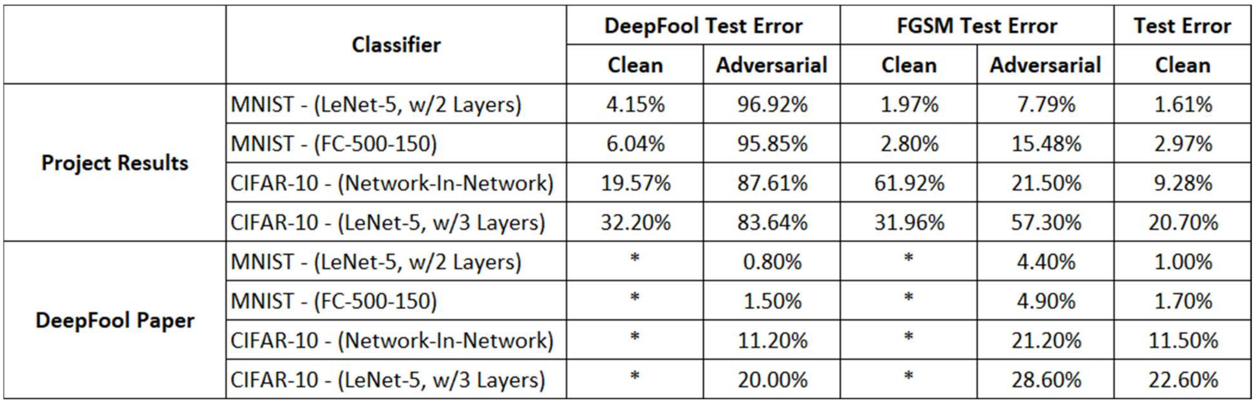
```
FGSM Test Error : 87.89%
FGSM Robustness : 4.58e-01
FGSM Time (All Images) : 0.29 s
FGSM Time (Per Image) : 28.86 us
DeepFool Test Error : 98.74%
DeepFool Robustness : 9.64e-02
DeepFool Time (All Images) : 193.32 s
DeepFool Time (Per Image) : 19.33 ms

При fgsm_eps = 0.5
FGSM Test Error : 87.89%
FGSM Robustness : 4.58e-01
FGSM Time (All Images) : 0.29 s
FGSM Time (Per Image) : 28.86 us
DeepFool Test Error : 98.74%
DeepFool Robustness : 9.64e-02
DeepFool Time (All Images) : 193.32 s
DeepFool Time (Per Image) : 19.33 ms

При fgsm_eps = 0.9
FGSM Test Error : 87.89%
FGSM Robustness : 4.58e-01
FGSM Time (All Images) : 0.29 s
FGSM Time (Per Image) : 28.86 us
DeepFool Test Error : 98.74%
DeepFool Robustness : 9.64e-02
DeepFool Time (All Images) : 193.32 s
DeepFool Time (Per Image) : 19.33 ms

При fgsm_eps = 10
FGSM Test Error : 87.89%
FGSM Robustness : 4.58e-01
FGSM Time (All Images) : 0.29 s
FGSM Time (Per Image) : 28.86 us
DeepFool Test Error : 98.74%
DeepFool Robustness : 9.64e-02
DeepFool Time (All Images) : 193.32 s
DeepFool Time (Per Image) : 19.33 ms
```

14. Подготовить отчет в формате pdf (отразить отличия для `fgsm_eps=(0.001, 0.02, 0.5, 0.9, 10)` и выявить закономерность/обнаружить отсутсвие влияние параметра eps для сетей FC LeNet на датасете MNIST, NiN LeNEt на датасете CIFAR



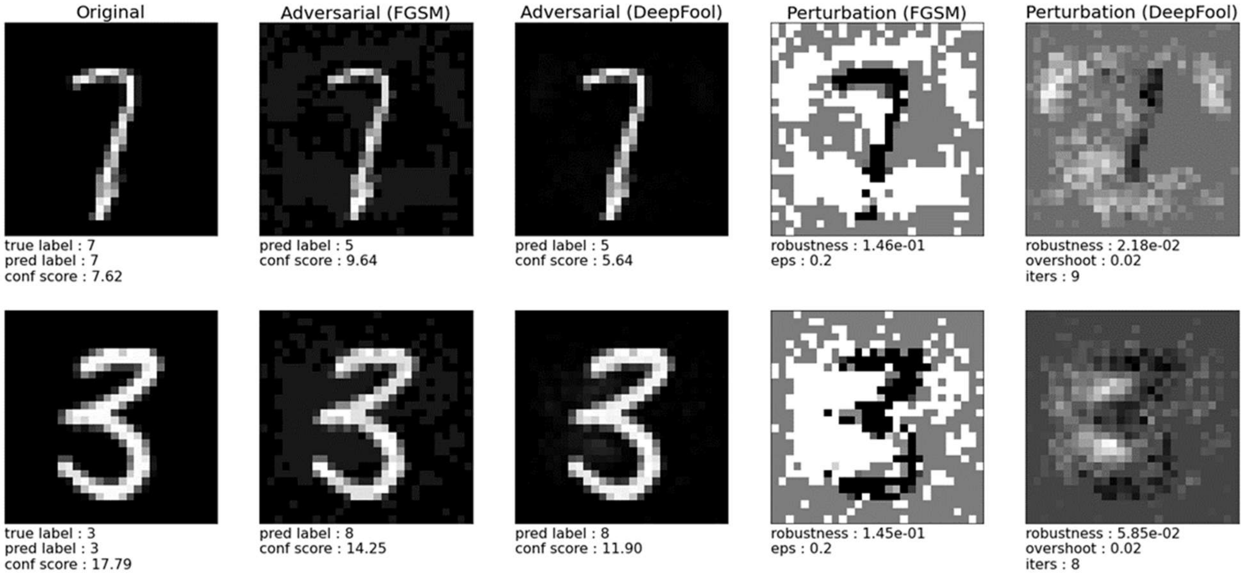
	Classifier	Test Error			Robustness (ρ_{adv})		Inference (Per Image)	
		Clean	DeepFool	FGSM	DeepFool	FGSM	DeepFool	FGSM
Project Results	MNIST - (LeNet-5, w/2 Layers)	1.61%	98.74%	87.89%	9.64×10^{-2}	4.85×10^{-1}	19.33 ms	28.86 μs
	MNIST - (FC-500-150)	2.97%	97.92%	87.08%	6.78×10^{-2}	1.56×10^{-1}	14.18 ms	14.99 μs
	CIFAR-10 - (Network-In-Network)	9.28%	93.76%	81.29%	2.12×10^{-2}	1.77×10^{-1}	18.51 ms	67.07 μs
	CIFAR-10 - (LeNet-5, w/3 Layers)	20.70%	87.81%	91.71%	1.78×10^{-2}	8.90×10^{-2}	7.33 ms	40.08 μs
	ILSVRC2012 - (GoogLeNet)	30.22%	92.87%	90.18%	5.36×10^{-3}	1.82×10^{-2}	129.98 ms	463.50 μs
	ILSVRC2012 - (CaffeNet)	*	*	*	*	*	*	*
DeepFool Paper	MNIST - (LeNet-5, w/2 Layers)	1.00%	*	*	2.0×10^{-1}	1.0×10^{0}	110 ms	20 ms
	MNIST - (FC-500-150)	1.70%	*	*	1.1×10^{-1}	3.9×10^{-1}	50 ms	10 ms
	CIFAR-10 - (Network-In-Network)	11.50%	*	*	2.3×10^{-2}	1.2×10^{-1}	1100 ms	180 ms
	CIFAR-10 - (LeNet-5, w/3 Layers)	22.60%	*	*	3.0×10^{-2}	1.3×10^{-1}	220 ms	50 ms
	ILSVRC2012 - (GoogLeNet)	31.30%	*	*	1.9×10^{-3}	3.5×10^{-2}	800 ms	80 ms
	ILSVRC2012 - (CaffeNet)	42.60%	*	*	2.7×10^{-3}	4.7×10^{-2}	510 ms	50 ms

Таблица, демонстрирующая экспериментальные результаты, состязательный вывод и надежность

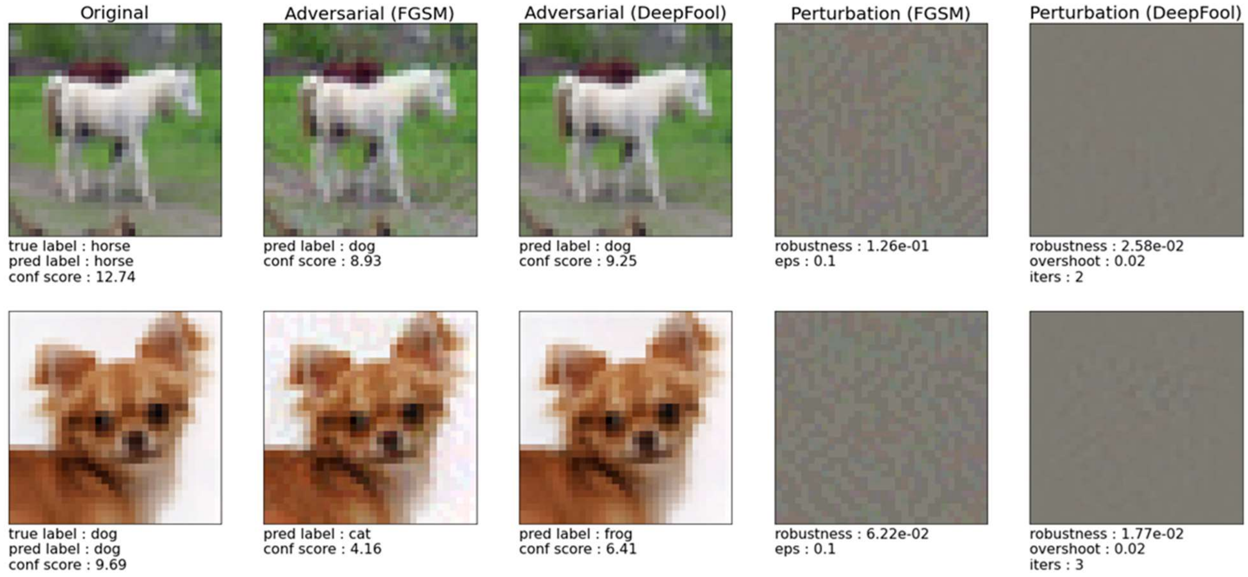


	Classifier	DeepFool Test Error		FGSM Test Error		Test Error
		Clean	Adversarial	Clean	Adversarial	Clean
Project Results	MNIST - (LeNet-5, w/2 Layers)	4.15%	96.92%	1.97%	7.79%	1.61%
	MNIST - (FC-500-150)	6.04%	95.85%	2.80%	15.48%	2.97%
	CIFAR-10 - (Network-In-Network)	19.57%	87.61%	61.92%	21.50%	9.28%
	CIFAR-10 - (LeNet-5, w/3 Layers)	32.20%	83.64%	31.96%	57.30%	20.70%
DeepFool Paper	MNIST - (LeNet-5, w/2 Layers)	*	0.80%	*	4.40%	1.00%
	MNIST - (FC-500-150)	*	1.50%	*	4.90%	1.70%
	CIFAR-10 - (Network-In-Network)	*	11.20%	*	21.20%	11.50%
	CIFAR-10 - (LeNet-5, w/3 Layers)	*	20.00%	*	28.60%	22.60%

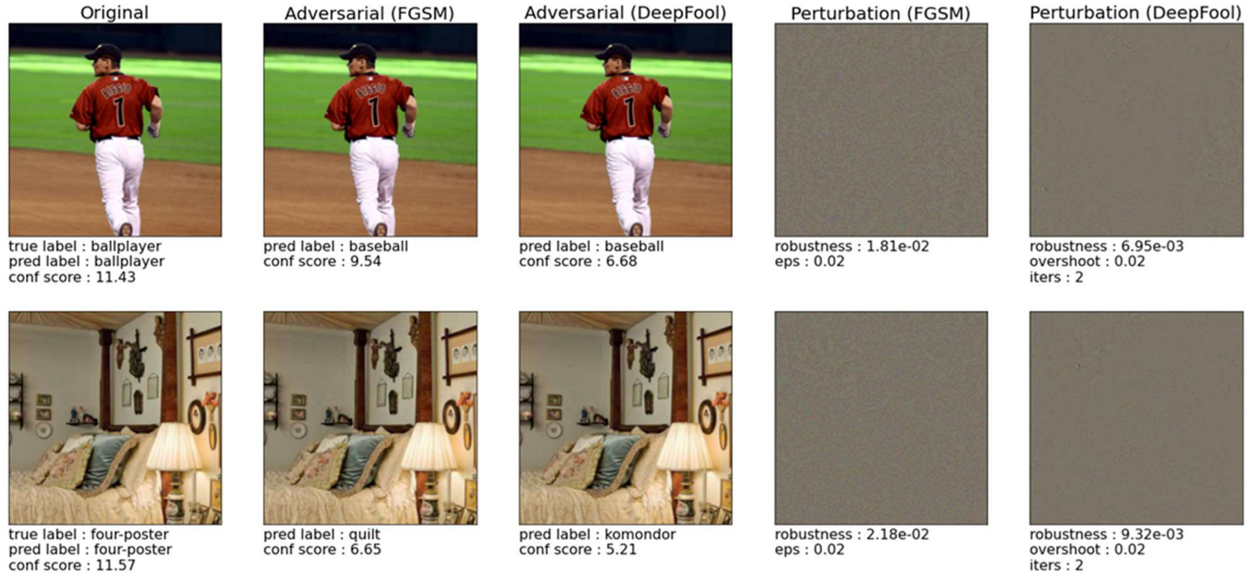
Обучение состязательности



Примеры состязательных программ MNIST (FC-500-150)

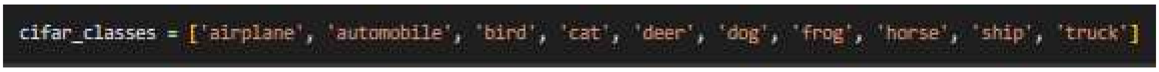


Примеры состязательных программ CIFAR-10 (LeNet-5)



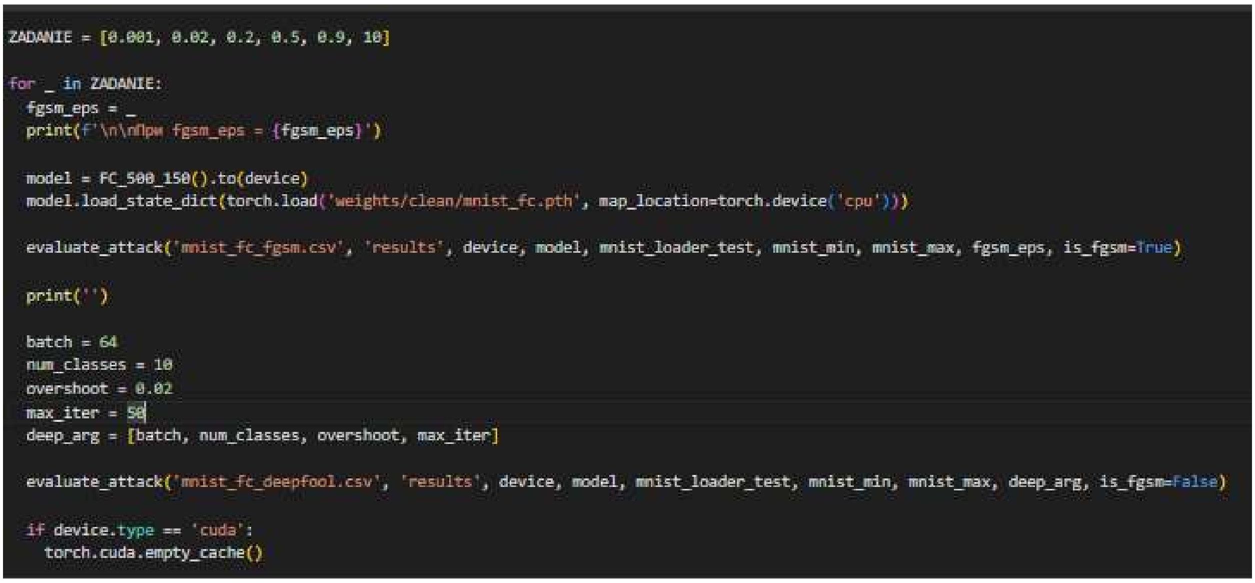
Примеры состязательных атак ILSVRC2012 (GoogLeNet)

Для этого мы указывали заранее классы, по которым будет проходить распределение моделью изображений.



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

eps – эпсилон, параметр, отвечающий за максимальную степень изменчивости отдельно взятого пикселя



```
ZADANIE = [0.001, 0.02, 0.2, 0.5, 0.9, 10]

for _ in ZADANIE:
  fgsm_eps = _
  print(f'\n\nПри fgsm_eps = {fgsm_eps}')

  model = FC_500_150().to(device)
  model.load_state_dict(torch.load('weights/clean/mnist_fc.pth', map_location=torch.device('cpu')))

  evaluate_attack('mnist_fc_fgsm.csv', 'results', device, model, mnist_loader_test, mnist_min, mnist_max, fgsm_eps, is_fgsm=True)

  print('')

  batch = 64
  num_classes = 10
  overshoot = 0.02
  max_iter = 50
  deep_arg = [batch, num_classes, overshoot, max_iter]

  evaluate_attack('mnist_fc_deepfool.csv', 'results', device, model, mnist_loader_test, mnist_min, mnist_max, deep_arg, is_fgsm=False)

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

```
При fgsm_eps = 0.001
FGSM Test Error : 87.08%
FGSM Robustness : 1.56e-01
FGSM Time (All Images) : 0.15 s
FGSM Time (Per Image) : 14.99 us

DeepFool Test Error : 97.92%
DeepFool Robustness : 6.78e-02
DeepFool Time (All Images) : 141.81 s
DeepFool Time (Per Image) : 14.18 ms


При fgsm_eps = 0.02
FGSM Test Error : 87.08%
FGSM Robustness : 1.56e-01
FGSM Time (All Images) : 0.15 s
FGSM Time (Per Image) : 14.99 us

DeepFool Test Error : 97.92%
DeepFool Robustness : 6.78e-02
DeepFool Time (All Images) : 141.81 s
DeepFool Time (Per Image) : 14.18 ms


При fgsm_eps = 0.2
FGSM Test Error : 87.08%
FGSM Robustness : 1.56e-01
FGSM Time (All Images) : 0.15 s
FGSM Time (Per Image) : 14.99 us

DeepFool Test Error : 97.92%
DeepFool Robustness : 6.78e-02
DeepFool Time (All Images) : 141.81 s
DeepFool Time (Per Image) : 14.18 ms


При fgsm_eps = 0.5
FGSM Test Error : 87.08%
FGSM Robustness : 1.56e-01
FGSM Time (All Images) : 0.15 s
FGSM Time (Per Image) : 14.99 us

DeepFool Test Error : 97.92%
DeepFool Robustness : 6.78e-02
DeepFool Time (All Images) : 141.81 s
DeepFool Time (Per Image) : 14.18 ms


При fgsm_eps = 0.9
FGSM Test Error : 87.08%
FGSM Robustness : 1.56e-01
FGSM Time (All Images) : 0.15 s
FGSM Time (Per Image) : 14.99 us

DeepFool Test Error : 97.92%
DeepFool Robustness : 6.78e-02
DeepFool Time (All Images) : 141.81 s
DeepFool Time (Per Image) : 14.18 ms


При fgsm_eps = 10
FGSM Test Error : 87.08%
FGSM Robustness : 1.56e-01
FGSM Time (All Images) : 0.15 s
FGSM Time (Per Image) : 14.99 us

DeepFool Test Error : 97.92%
DeepFool Robustness : 6.78e-02
DeepFool Time (All Images) : 141.81 s
DeepFool Time (Per Image) : 14.18 ms
```