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

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

Кафедра КБ-4 «Интеллектуальные системы информационной безопасности»

Лабораторная работа №2

по дисциплине

«Анализ защищенности систем искусственного интеллекта»

Группа:
ББМО-01-22
Выполнил:
Богомолов В.И.

Проверил:
Спирин А.А.

Москва 2023

Ход выполнения работы

Задание 1

Установим необходимые библиотеки:

```
import cv2
import os
import torch
import random
import pickle
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
from sklearn.model_selection import train_test_split
from keras.utils import to_categorical
from keras.applications import ResNet50
from keras.applications import VGG16
from keras.applications.resnet50 import preprocess_input
from keras.preprocessing import image
from keras.models import load_model, save_model
from keras.layers import Dense, Flatten, GlobalAveragePooling2D
from keras.models import Model
from keras.optimizers import Adam
from keras.losses import categorical_crossentropy
from keras.metrics import categorical_accuracy
from keras.callbacks import ModelCheckpoint, EarlyStopping, TensorBoard
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPool2D, AvgPool2D, BatchNormalization, Reshape, Lambda
from art.estimators.classification import KerasClassifier
from art.attacks.evasion import FastGradientMethod, ProjectedGradientDescent
from google.colab import drive
import zipfile
from PIL import Image
from tensorflow import keras as k
```

Подключим Google Drive и разархивируем файл с датасетом:

Подключим Google диск

```
[ ]  drive.mount('/content/drive/')
```

```
Mounted at /content/drive/
```

Загрузим датасет

```
[ ]  !unzip /content/drive/MyDrive/файлы/archive.zip
```

```
        inflating: train/9/00009_00047_00001.png
        inflating: train/9/00009_00047_00002.png
```

Разделим датасет на обучающую и тестовую выборки:

```
x_train, x_test, y_train, y_test = train_test_split(data, labels, test_size=0.2, random_state=42)
print("training shape: ",x_train.shape, y_train.shape)
print("testing shape: ",x_test.shape, y_test.shape)

y_train = to_categorical(y_train, 43)
y_test = to_categorical(y_test, 43)
```

```
training shape:  (31367, 32, 32, 3) (31367,)
testing shape:  (7842, 32, 32, 3) (7842,)
```

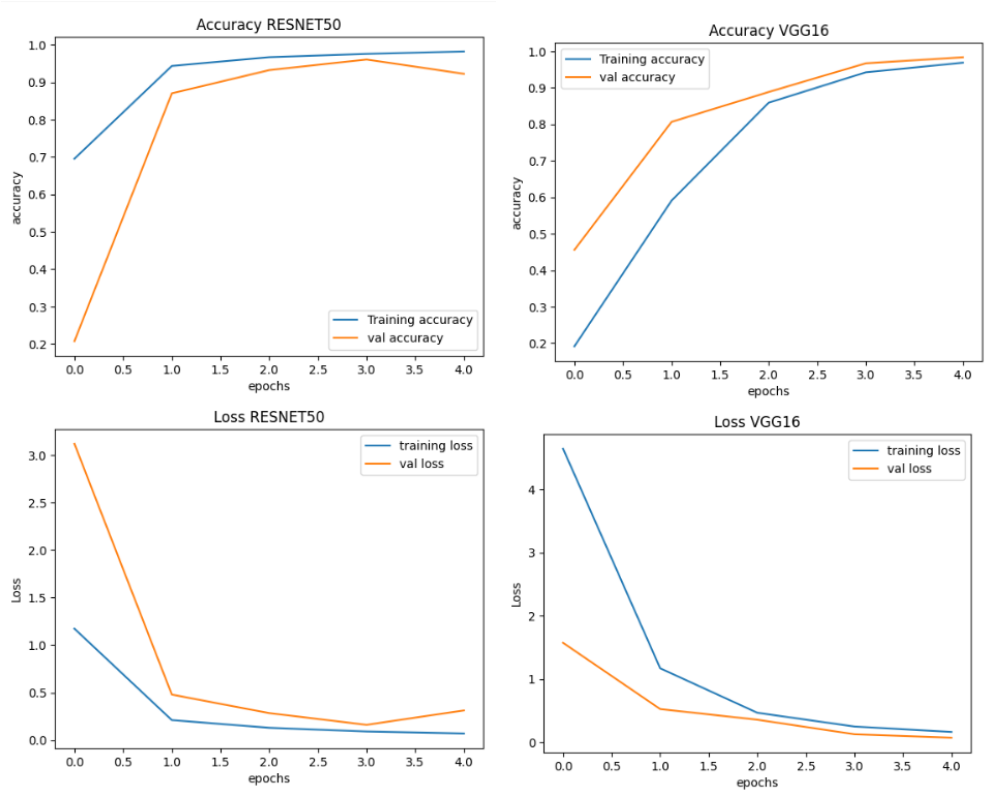
Создадим модель ResNet50:

```
resnet = k.applications.ResNet50(weights='imagenet', include_top=False)

model_2 = k.models.Sequential([
                              resnet,
                              tf.keras.layers.GlobalAveragePooling2D(),
                              k.layers.Dropout(0.2),
                              k.layers.Dense(256, activation='relu'),
                              k.layers.BatchNormalization(),
                              k.layers.Dropout(0.1),
                              k.layers.Dense(512, activation='relu'),
                              k.layers.BatchNormalization(),
                              k.layers.Dropout(0.2),
                              k.layers.Dense(43, activation='softmax')
])
print(model_2.summary())
```

Создадим модель VGG16:

```
vgg16 = k.applications.VGG16(weights='imagenet', include_top=False)


model_3 = k.models.Sequential([
                              vgg16,
                              tf.keras.layers.GlobalAveragePooling2D(),
                              k.layers.Dropout(0.2),
                              k.layers.Dense(256, activation='relu'),
                              k.layers.BatchNormalization(),
                              k.layers.Dropout(0.1),
                              k.layers.Dense(512, activation='relu'),
                              k.layers.BatchNormalization(),
                              k.layers.Dropout(0.2),
                              k.layers.Dense(43, activation='softmax')
])
print(model_3.summary())
```

По завершении обучения были сформированы следующие графики точности для моделей ResNet50, VGG16:



По заданию 1 была получена следующая результирующая таблица:

Модель	Обучение	Валидация	Тест
VGG16	Loss:0.1626 accuracy:0.9687	Loss:0.0712 accuracy:0.9834	Loss:0.2693 accuracy:0.9461
ResNet50	Loss:0.0703 accuracy:0.9821	Loss:0.3128 accuracy:0.9223	Loss:0.6175 accuracy:0.8676

Задание 2

Атаки на изображения проводятся со следующими параметрами искажения:

```
epsilons = [1/255, 2/255, 3/255, 4/255, 5/255, 8/255, 10/255, 20/255, 50/255, 80/255]
```

Атаки проводятся на уменьшенном наборе данных, а именно на первых 1000 элементов.

Проведем атаку FGSM на модель ResNet50.

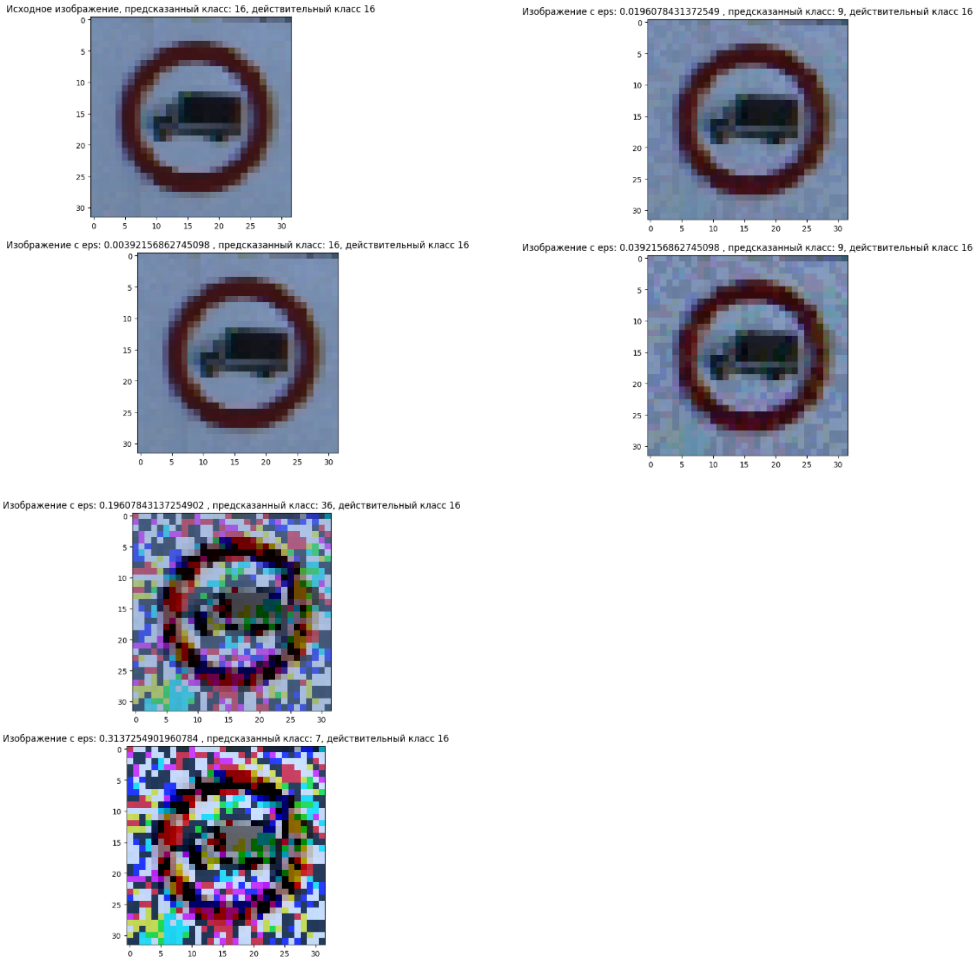
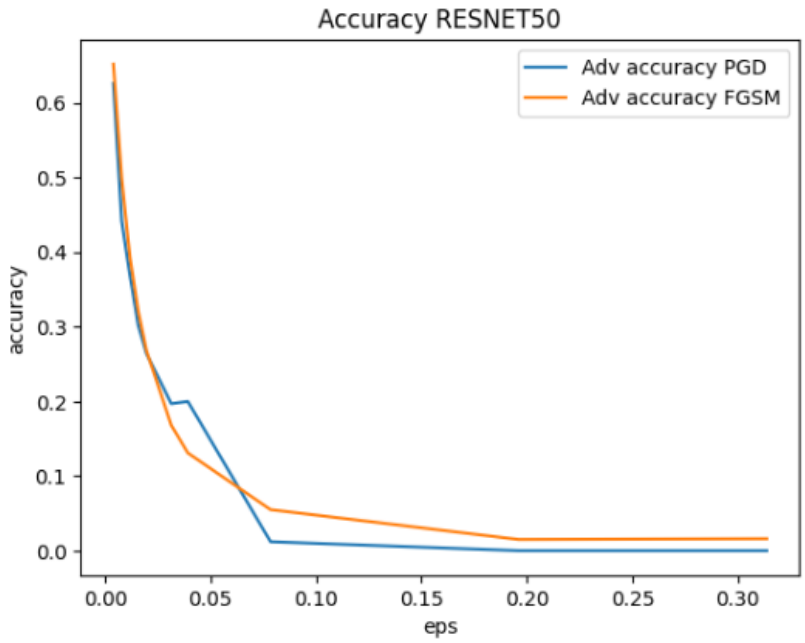


График зависимости точности классификации от параметра искажения (ResNet50).



Проведем атаку FGSM на модель VGG16.

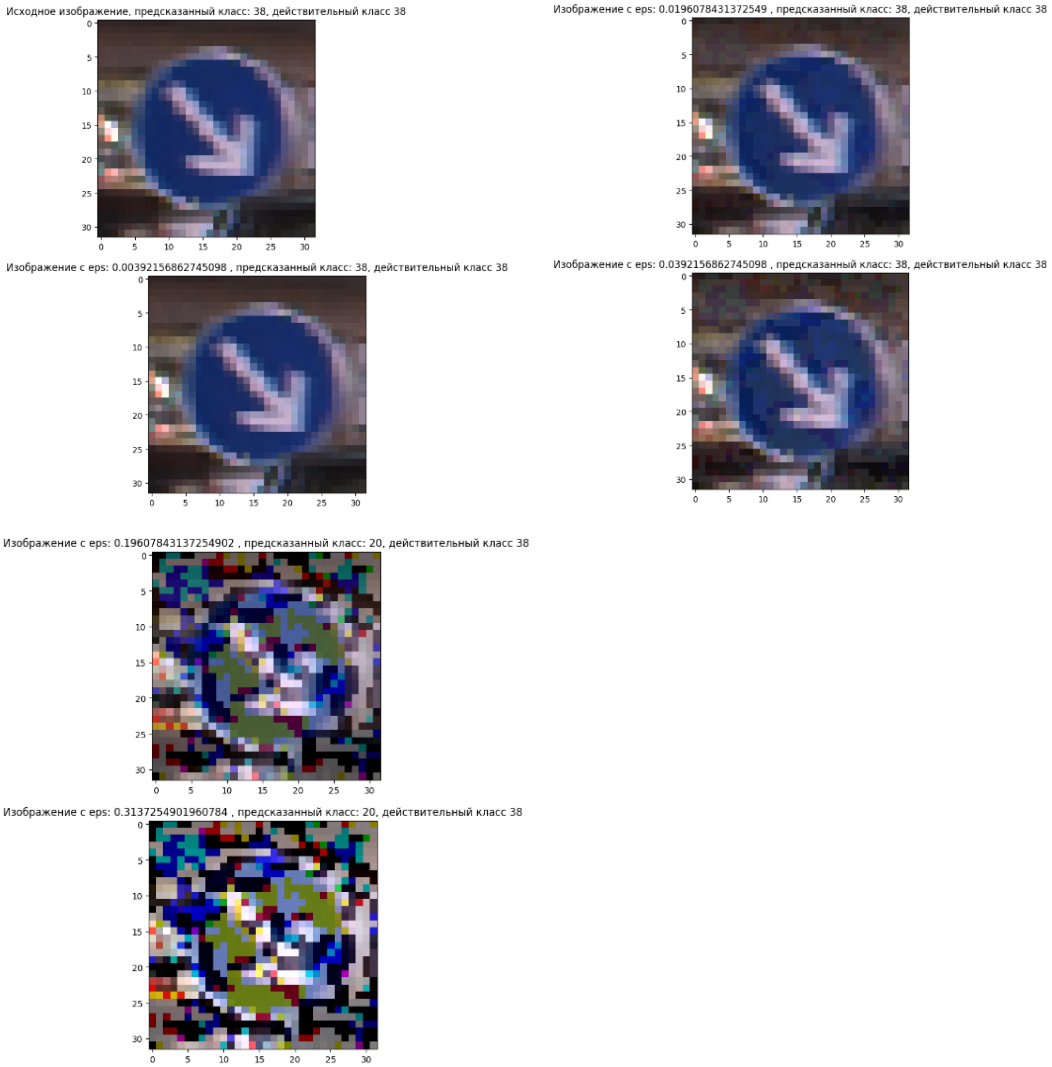
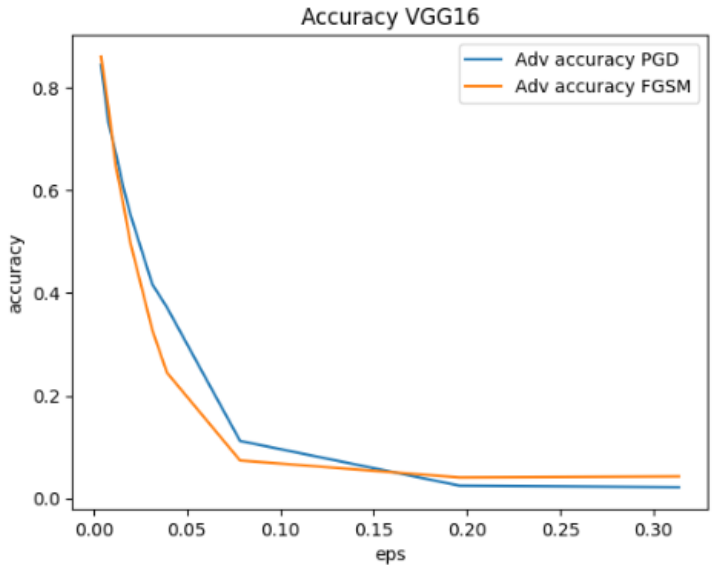


График зависимости точности классификации от параметра искажения (VGG16).

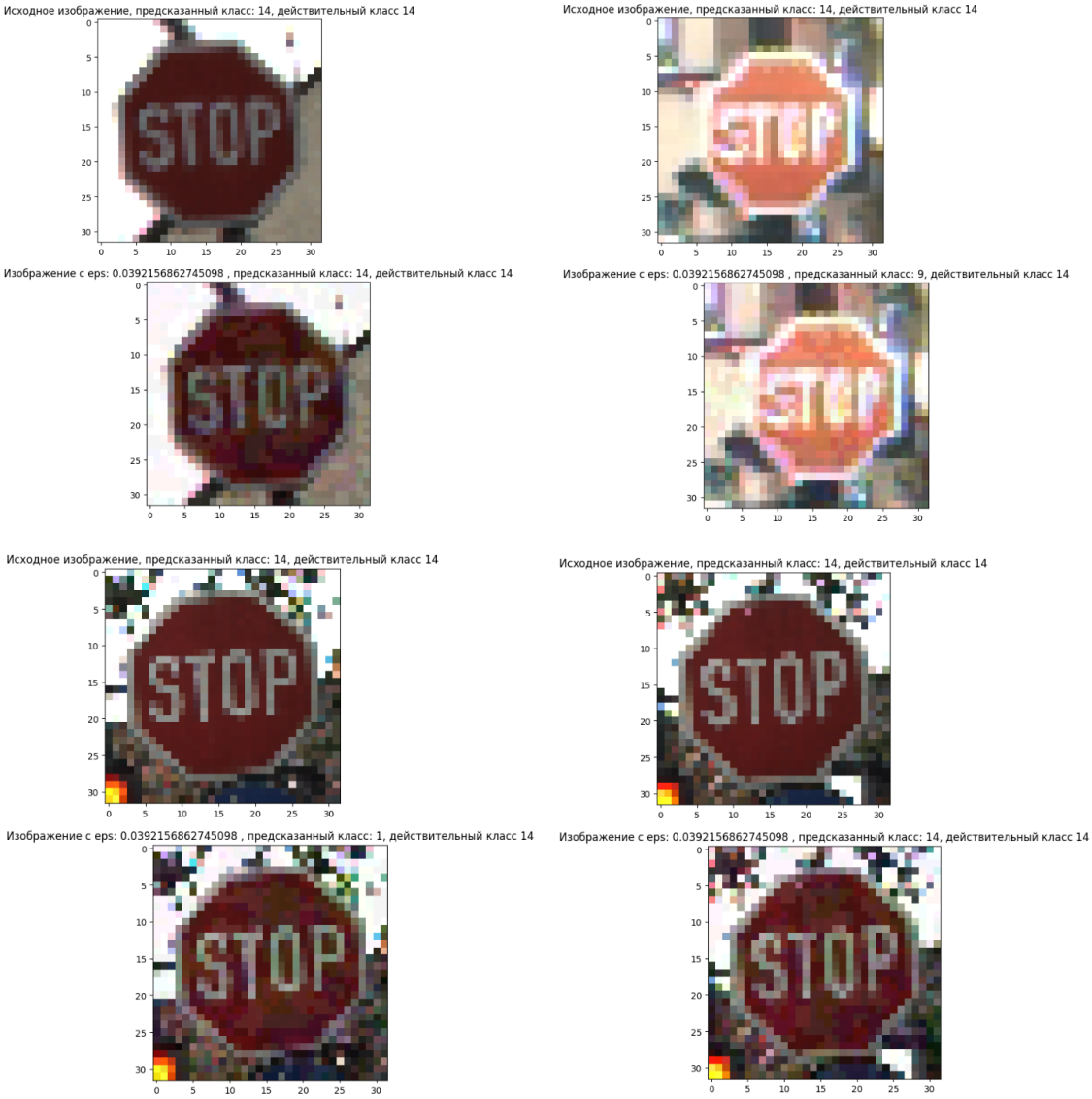


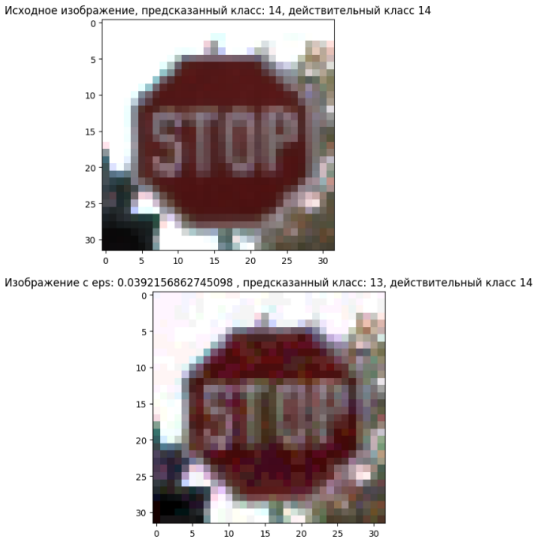
По завершении подготовки атак была получена следующая результирующая таблица:

Модель	Исходные изображения	Adversarial images ϵ=1/255	Adversarial images ϵ=5/255	Adversarial images ϵ=10/255
VGG16-FGSM	94%	86%	49.7%	24.5%
VGG16-PGD	94%	84%	55%	37%
ResNet50-FGSM	86%	65%	27%	13%
ResNet50-PGD	86%	62.5%	26%	20%

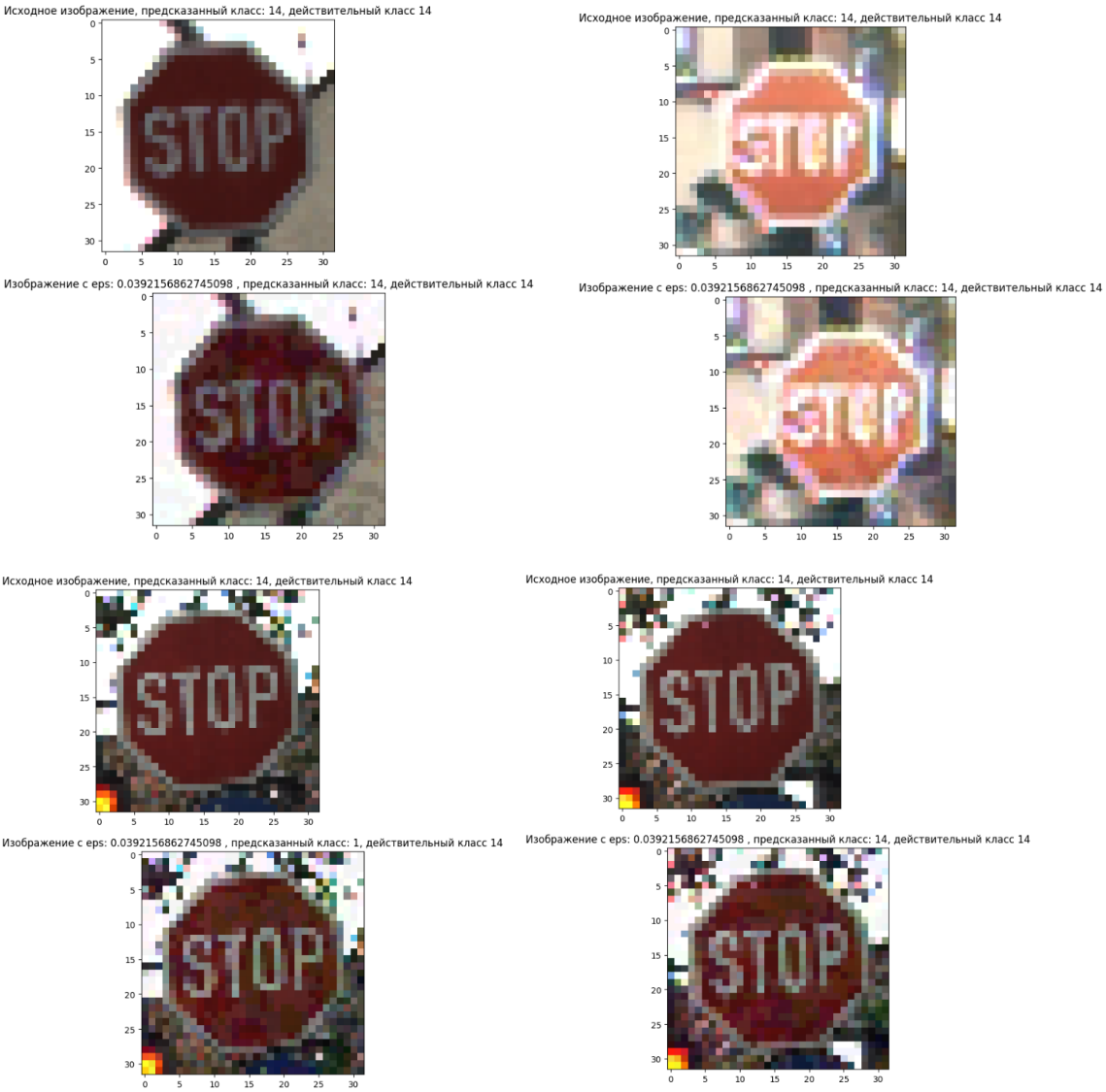
Задание 3

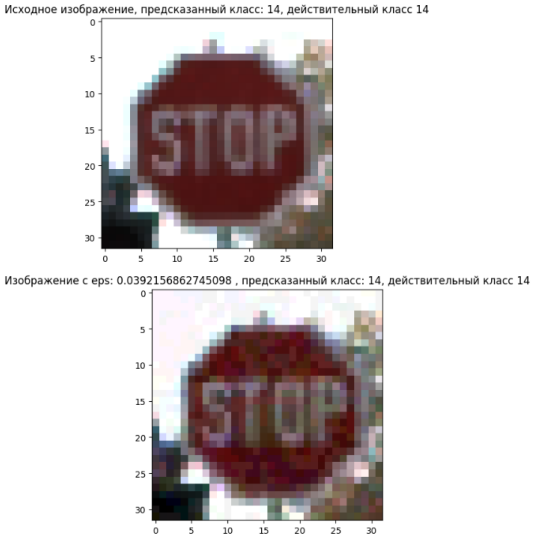
Пример исходных изображений знака «Стоп» и соответствующих атакующих примеров. (FGSM)





Пример исходных изображений знака «Стоп» и соответствующих атакующих примеров. (PGD)





Результирующая таблица по заданию:

Искажение	PGD attack – Stop sign images	FGSM attack – Stop sign images
ϵ=1/255	100%	99%
ϵ=3/255	97%	83%
ϵ=5/255	93%	73%
ϵ=10/255	79%	44%
ϵ=20/255	56%	5%
ϵ=50/255	10%	0%
ϵ=80/255	5%	0%