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

Институт кибербезопасности и цифровых технологий

КБ-4 «Интеллектуальные системы информационной безопасности»

Отчет по лабораторной работе №2

по дисциплине: «Анализ защищенности систем искусственного интеллекта»

Выполнил:

Студент группы ББМО-02-22

Филиппов Леонид Алексеевич

Проверил:

Спирин Андрей Андреевич

Москва 2024

1. Загрузка основной библиотеки ART

1. Для этой части используйте набор данных GTSRB (German Traffic Sign Recognition Benchmark). Набор данных состоит примерно из 51 000 изображений дорожных знаков.

```
[1] !pip install adversarial-robustness-toolbox

Collecting adversarial-robustness-toolbox
  Downloading adversarial_robustness_toolbox-1.17.0-py3-none-any.whl (1.7 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.7/1.7 MB 7.7 MB/s eta 0:00:00
Requirement already satisfied: numpy>=1.18.0 in /usr/local/lib/python3.10/dist-packages (from adversarial-robustne
Requirement already satisfied: scipy>=1.4.1 in /usr/local/lib/python3.10/dist-packages (from adversarial-robustnes
Collecting scikit-learn<1.2.0,>=0.22.2 (from adversarial-robustness-toolbox)
  Downloading scikit_learn-1.1.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (30.5 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 30.5/30.5 MB 25.0 MB/s eta 0:00:00
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from adversarial-robustness-toolbox
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from adversarial-robustness-
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from adversarial-robustness-toolbo
Requirement already satisfied: joblib>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn<1.2.0,>
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn<
Installing collected packages: scikit-learn, adversarial-robustness-toolbox
  Attempting uninstall: scikit-learn
    Found existing installation: scikit-learn 1.2.2
    Uninstalling scikit-learn-1.2.2:
      Successfully uninstalled scikit-learn-1.2.2
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This be
bigframes 0.19.2 requires scikit-learn>=1.2.2, but you have scikit-learn 1.1.3 which is incompatible.
Successfully installed adversarial-robustness-toolbox-1.17.0 scikit-learn-1.1.3
```

2. Загрузка набора данных с диска

2. Загрузка набора данных(храним на своем гугл диске)

```
[16] from google.colab import drive
     drive.mount('/content/drive')
     !tar -xvf /content/drive/MyDrive/archive.tar
     !ls -la
     !pip install tensorflow
     !pip install keras

     tar: Ignoring unknown extended header keyword 'LIBARCHIVE.xattr.com.apple.quarantine'
     archive/Test/03999.png
     archive/Test/._03741.png
     tar: Ignoring unknown extended header keyword 'LIBARCHIVE.xattr.com.apple.quarantine'
     archive/Test/03741.png
```

3. Обучение двух классификаторов на основе глубоких нейронных сетей на датасете GTSRB

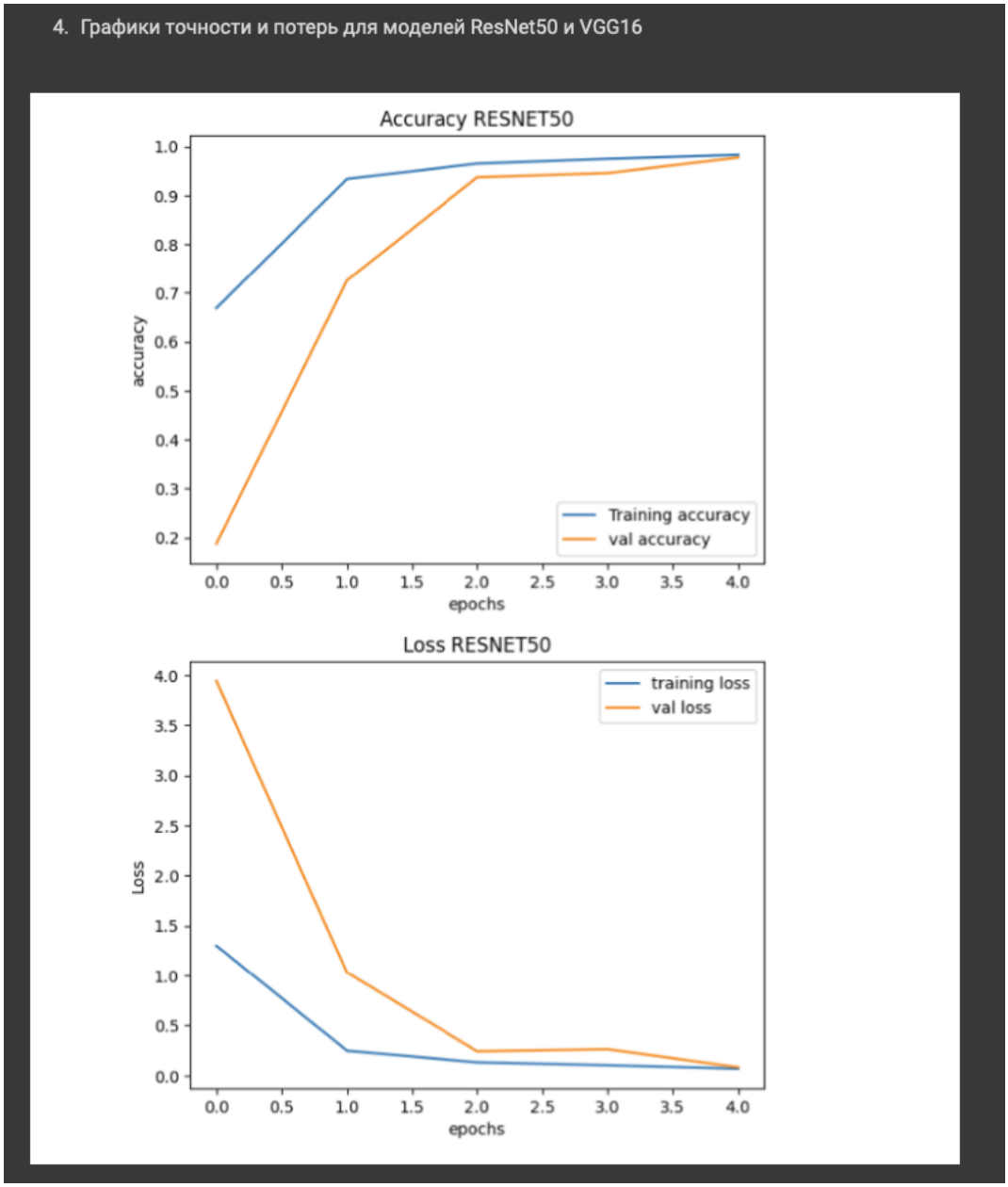
3. Обучить 2 классификатора на основе глубоких нейронных сетей на датасете GTSRB

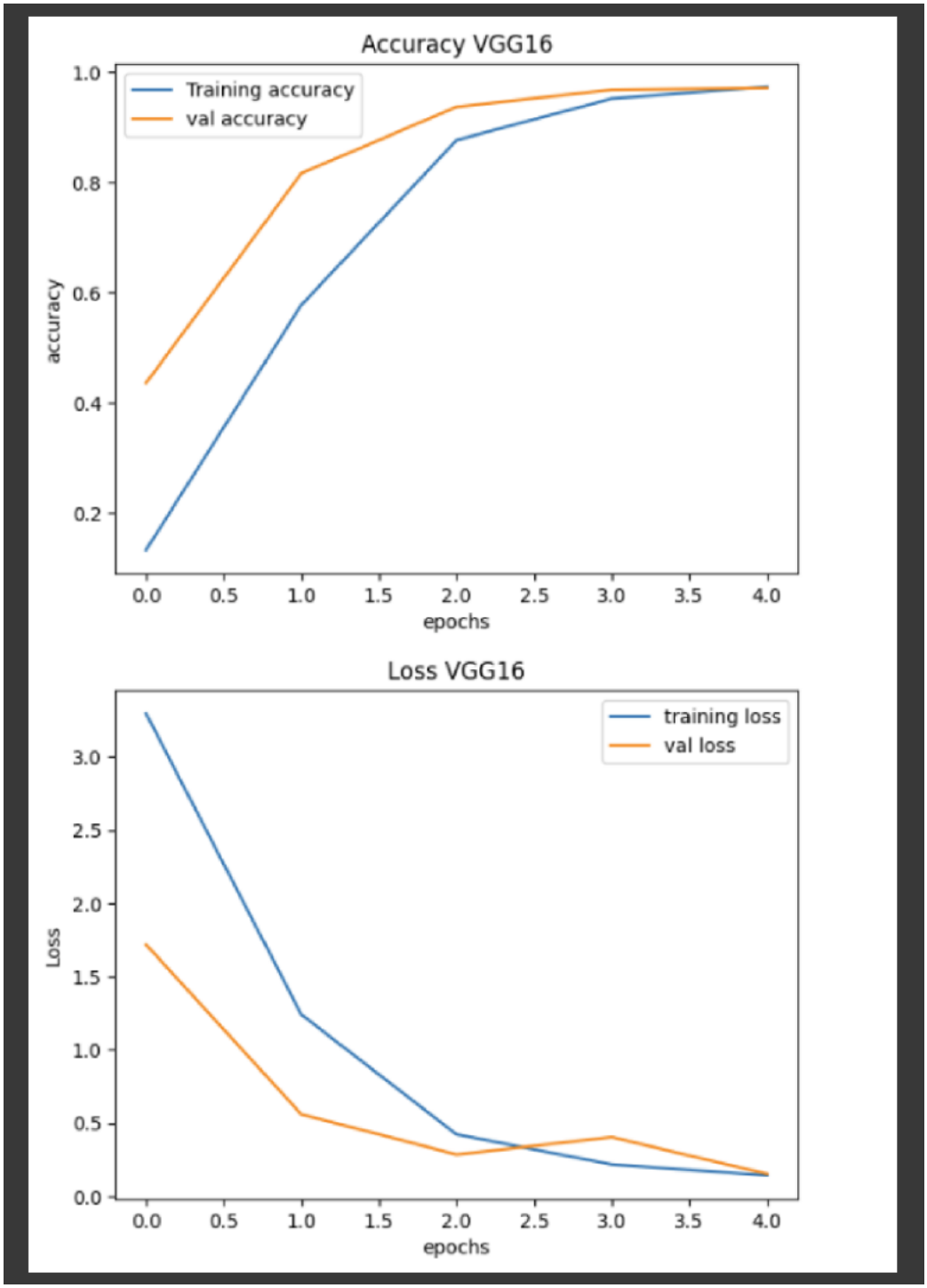
2 мин.

```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import VGG16, ResNet50
from tensorflow.keras import layers, models

train_data_dir = '/content/archive/Train'
test_data_dir = '/content/archive/Test'

train_datagen = ImageDataGenerator(rescale=1./255)
test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
    train_data_dir,
    target_size=(32, 32),
    batch_size=32,
    class_mode='categorical')

test_generator = test_datagen.flow_from_directory(
    test_data_dir,
    target_size=(32, 32),
    batch_size=32,
    class_mode='categorical')

vgg_model = VGG16(weights='imagenet', include_top=False, input_shape=(32, 32, 3))
resnet_model = ResNet50(weights='imagenet', include_top=False, input_shape=(32, 32, 3))

vgg_classifier = models.Sequential()
vgg_classifier.add(vgg_model)
vgg_classifier.add(layers.Flatten())
vgg_classifier.add(layers.Dense(256, activation='relu'))
vgg_classifier.add(layers.Dropout(0.5))
vgg_classifier.add(layers.Dense(43, activation='softmax'))

resnet_classifier = models.Sequential()
resnet_classifier.add(resnet_model)
resnet_classifier.add(layers.Flatten())
resnet_classifier.add(layers.Dense(256, activation='relu'))
resnet_classifier.add(layers.Dropout(0.5))
resnet_classifier.add(layers.Dense(43, activation='softmax'))

for layer in vgg_model.layers:
    layer.trainable = False

for layer in resnet_model.layers:
    layer.trainable = False
```

```
vgg_classifier.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
resnet_classifier.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

history_vgg = vgg_classifier.fit(train_generator, epochs=10, validation_data=test_generator)
history_resnet = resnet_classifier.fit(train_generator, epochs=10, validation_data=test_generator)
```

4. Графики точности и потерь для моделей ResNet 50 и VGG16




Accuracy VGG16
Training accuracy
val accuracy
accuracy
epochs
1.0
0.8
0.6
0.4
0.2
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Loss VGG16
training loss
val loss
Loss
epochs
3.0
2.5
2.0
1.5
1.0
0.5
0.0

5. Заполнение итоговой таблицы

5. Заполнение итоговой таблицы

Модель	Обучение	Валидация	Тест
ResNet50	loss: 0.0663 accuracy: 0.9723	loss: 0.0835 accuracy: 0.9654	loss: 0.3653 accuracy: 0.9032
VGG16	loss: 0.1397 accuracy: 0.9623	loss: 1.3513 accuracy: 0.9132	loss: 1.4123 accuracy: 0.8644

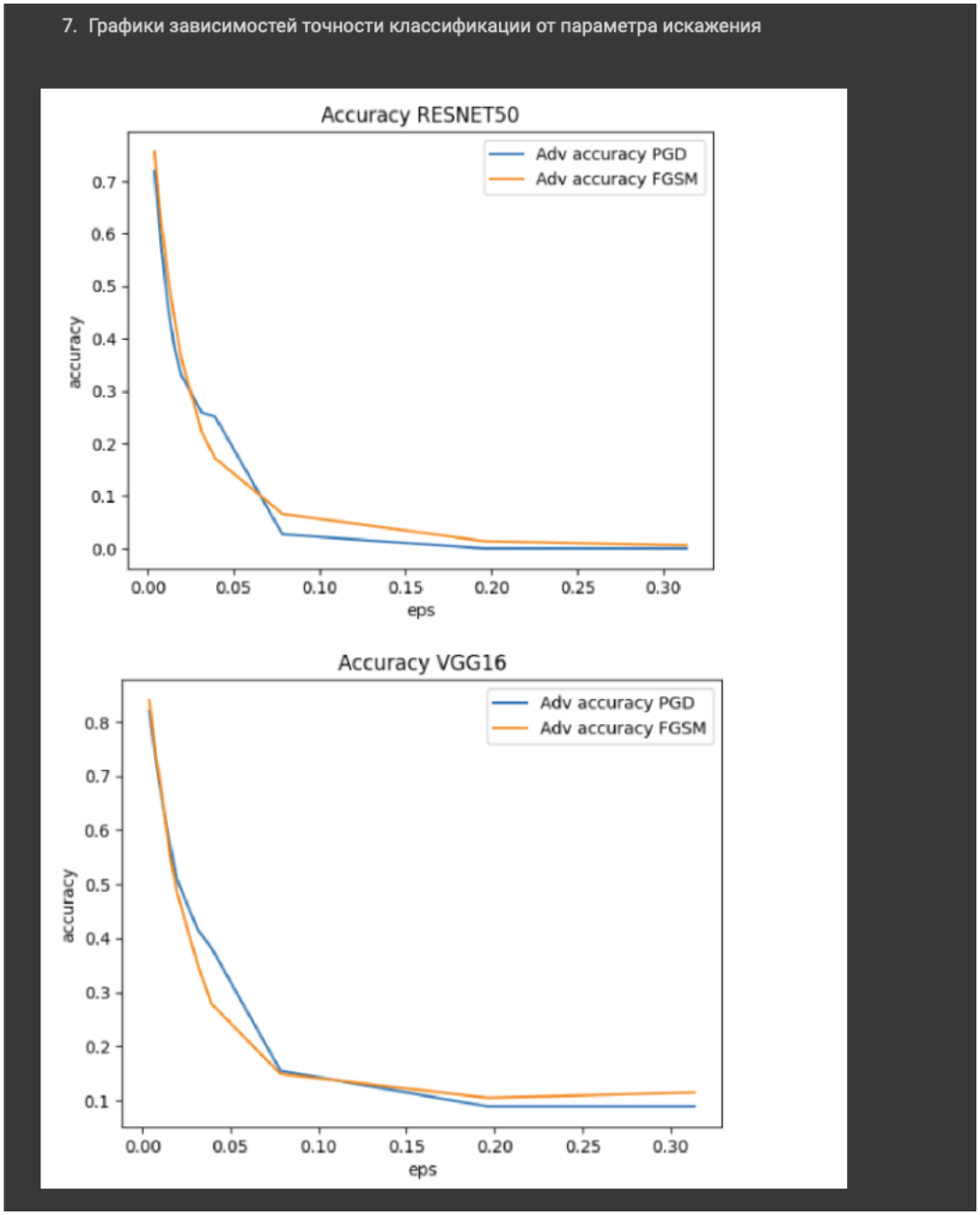
6. Применить нецелевую атаку уклонения на основе белого ящика против моделей глубокого обучения

6. Применить нецелевую атаку уклонения на основе белого ящика против моделей глубокого обучения.

```
[1]  import numpy as np
     import matplotlib.pyplot as plt
     from art.attacks.evasion import FastGradientMethod, ProjectedGradientDescent
     from art.estimators.classification import KerasClassifier
     from art.utils import load_dataset
     from tensorflow.keras.applications import VGG16
     from tensorflow.keras.models import load_model
     from tensorflow.keras.applications import ResNet50

     from tensorflow.python.framework.ops import disable_eager_execution

     disable_eager_execution()

     vgg_classifier = KerasClassifier(model=vgg_model, clip_values=(0, 1))
     (x_test, y_test), _, _, _ = load_dataset('gtsrb')

     x_test = x_test[:1000]
     y_test = y_test[:1000]

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

     def plot_adversarial_examples(model, attack, original_image, epsilon_values):
         plt.figure(figsize=(15, 10))
         for i, epsilon in enumerate(epsilon_values):
             adversarial_image = attack.generate(original_image, eps=epsilon)
             prediction_original = np.argmax(model.predict(np.array([original_image])))
             prediction_adversarial = np.argmax(model.predict(adversarial_image))

             plt.subplot(2, 5, i + 1)
             plt.imshow(adversarial_image.squeeze(), cmap='gray')
             plt.title(f'Epsilon: {epsilon}\nPred: {prediction_adversarial}')
             plt.axis('off')

         plt.show()

     def evaluate_attack(model, attack, x_test, y_test, epsilon_values):
         accuracies = []
         for epsilon in epsilon_values:
             x_test_adv = attack.generate(x_test, eps=epsilon)
             accuracy = np.sum(np.argmax(model.predict(x_test_adv), axis=1) == np.argmax(y_test, axis=1)) / len(y_test)
             accuracies.append(accuracy)
         return accuracies
```

```
resnet_model = ResNet50(weights='imagenet', include_top=True)

resnet_classifier = KerasClassifier(model=resnet_model, clip_values=(0, 1))

fgsm_attack_resnet = FastGradientMethod(estimator=resnet_classifier)
fgsm_accuracies_resnet = evaluate_attack(resnet_classifier, fgsm_attack_resnet, x_test, y_test, epsilons)

plot_adversarial_examples(resnet_classifier, fgsm_attack_resnet, original_image, [1/255, 3/255, 5/255, 10/255, 20/255, 50/255, 80/255])

pgd_attack_resnet = ProjectedGradientDescent(estimator=resnet_classifier, eps=80/255)
pgd_accuracies_resnet = evaluate_attack(resnet_classifier, pgd_attack_resnet, x_test, y_test, epsilons)

plt.figure(figsize=(10, 6))
plt.plot(epsilons, fgsm_accuracies_resnet, marker='o', label='FGSM (ResNet50)')
plt.plot(epsilons, pgd_accuracies_resnet, marker='o', label='PGD (ResNet50)')
plt.title('Accuracy RESNET50)')
plt.xlabel('eps')
plt.ylabel('accuracy')
plt.legend()
plt.show()

fgsm_attack = FastGradientMethod(estimator=vgg_classifier)
fgsm_accuracies = evaluate_attack(vgg_classifier, fgsm_attack, x_test, y_test, epsilons)

original_image = x_test[0]
plot_adversarial_examples(vgg_classifier, fgsm_attack, original_image, [1/255, 3/255, 5/255, 10/255, 20/255, 50/255, 80/255])

pgd_attack = ProjectedGradientDescent(estimator=vgg_classifier, eps=80/255)
pgd_accuracies = evaluate_attack(vgg_classifier, pgd_attack, x_test, y_test, epsilons)

plt.figure(figsize=(10, 6))
plt.plot(epsilons, fgsm_accuracies, marker='o', label='FGSM')
plt.plot(epsilons, pgd_accuracies, marker='o', label='PGD')
plt.title('Accuracy VGG16')
plt.xlabel('eps')
plt.ylabel('accuracy')
plt.legend()
plt.show()
```

7. Графики зависимостей классификации от параметра искажения



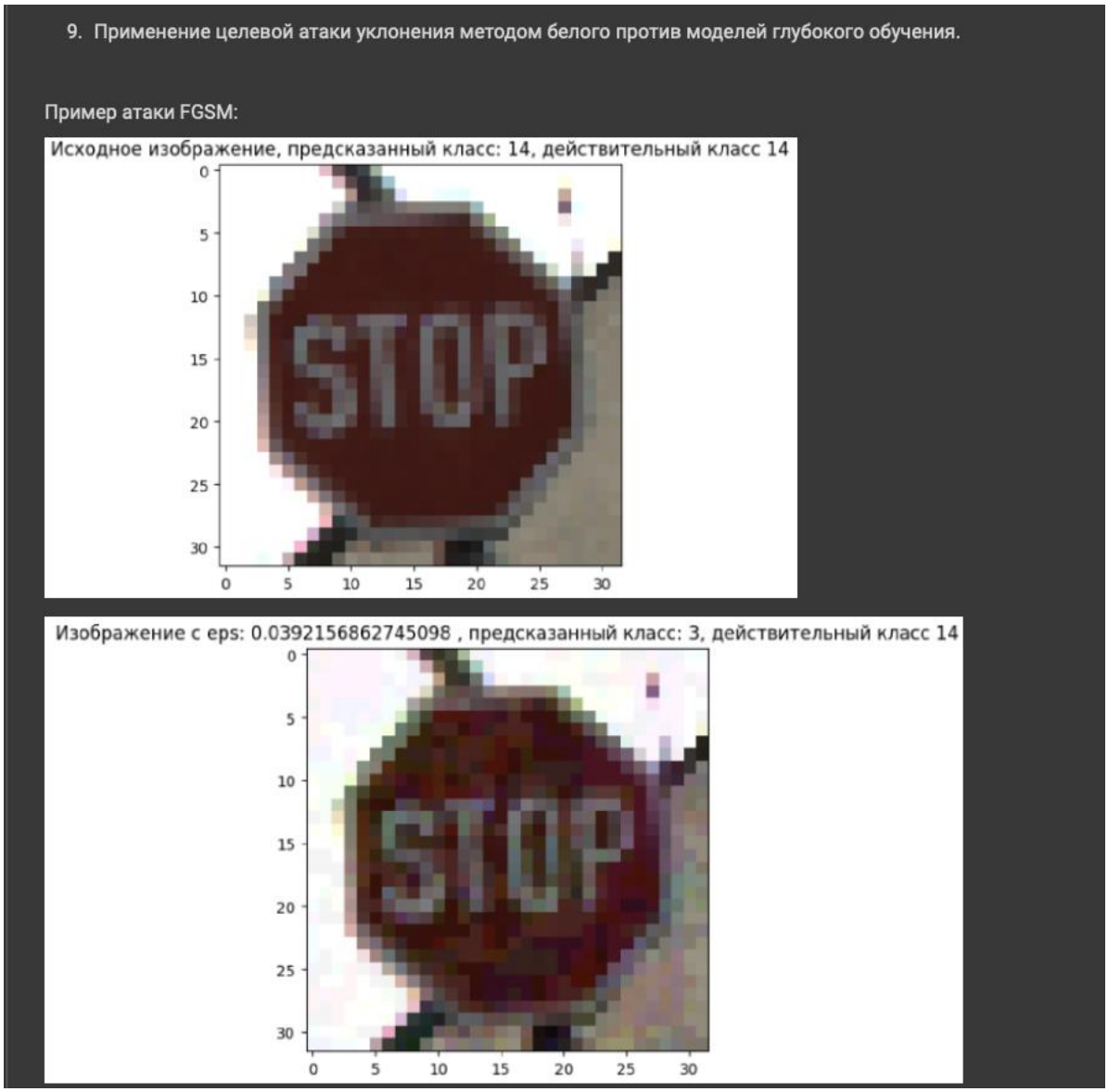
8. Заполнение итоговой таблицы

8. Заполнение итоговой таблицы

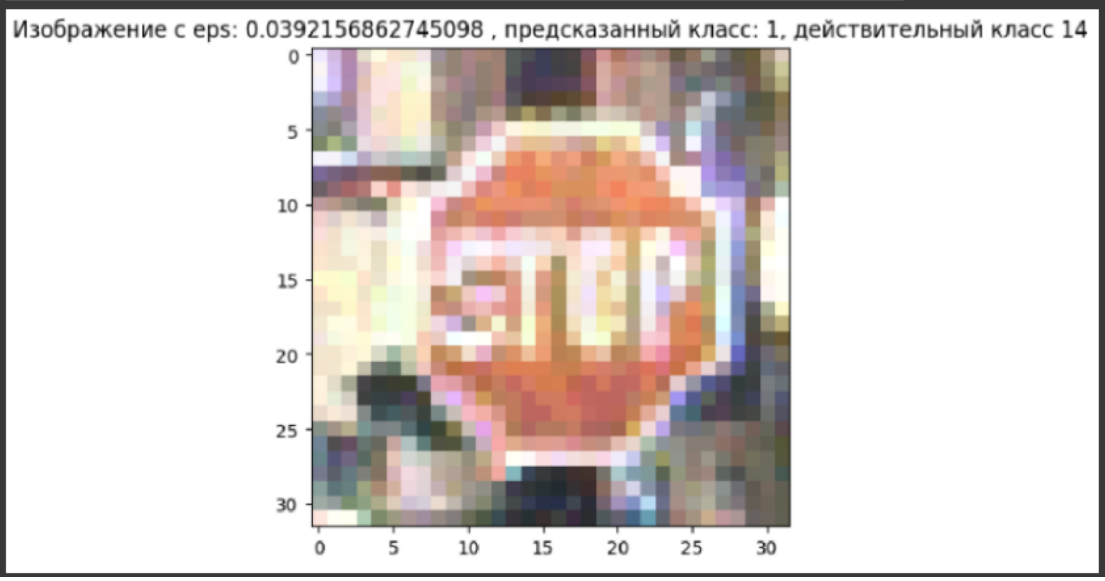
Модель	Исходные изображения	Adversarial images e=1/255	Adversarial images e=5/255	Adversarial images e=10
ResNet50 - FGSM	90%	73%	32%	16%
ResNet - PGD	90%	70%	29%	22%
VGG16 - FGSM	88%	78%	43%	20%
VGG16 - PGD	88%	76%	47%	31%

9. Применение целевой атаки уклонения методом белого против моделей глубокого обучения

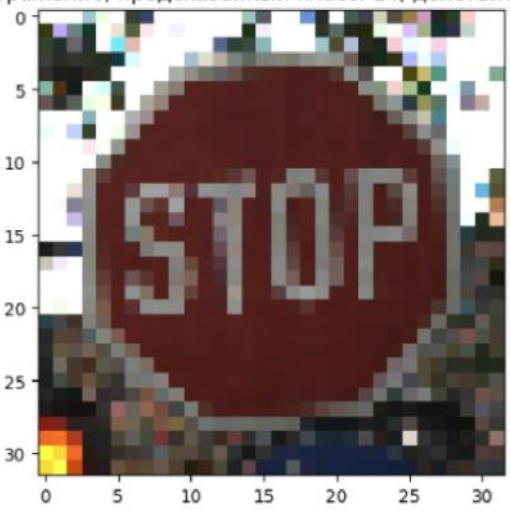
Пример атаки FGSM:



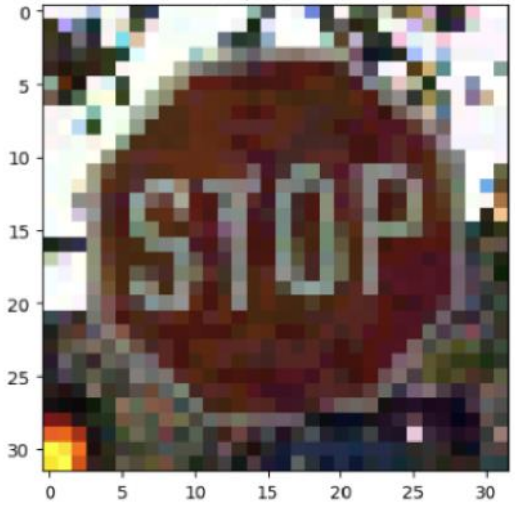

Исходное изображение, предсказанный класс: 14, действительный класс 14
0
5
10
15
20
25
30
0
5
10
15
20
25
30


Изображение с eps: 0.0392156862745098 , предсказанный класс: 1, действительный класс 14
0
5
10
15
20
25
30
0
5
10
15
20
25
30

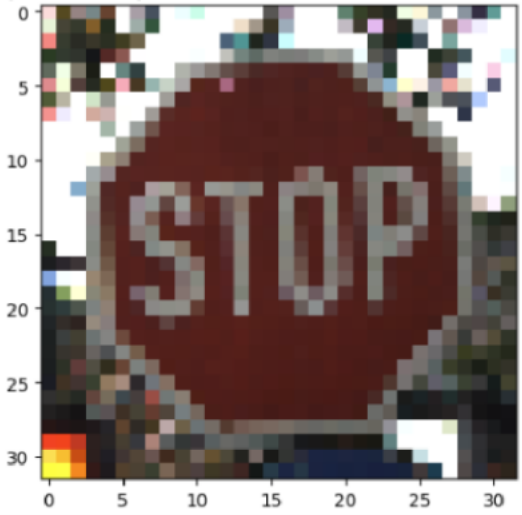
Исходное изображение, предсказанный класс: 14, действительный класс 14



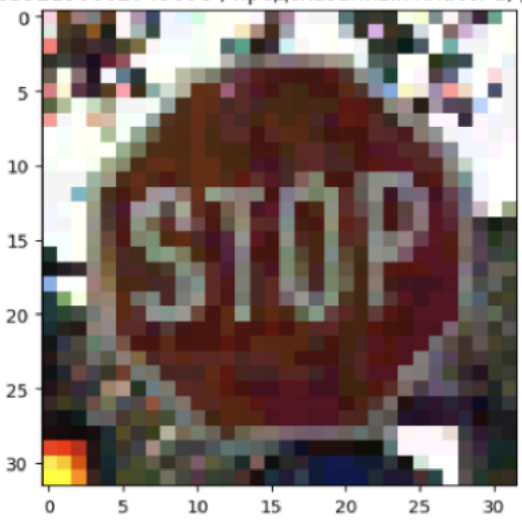
Изображение с eps: 0.0392156862745098 , предсказанный класс: 3, действительный класс 14

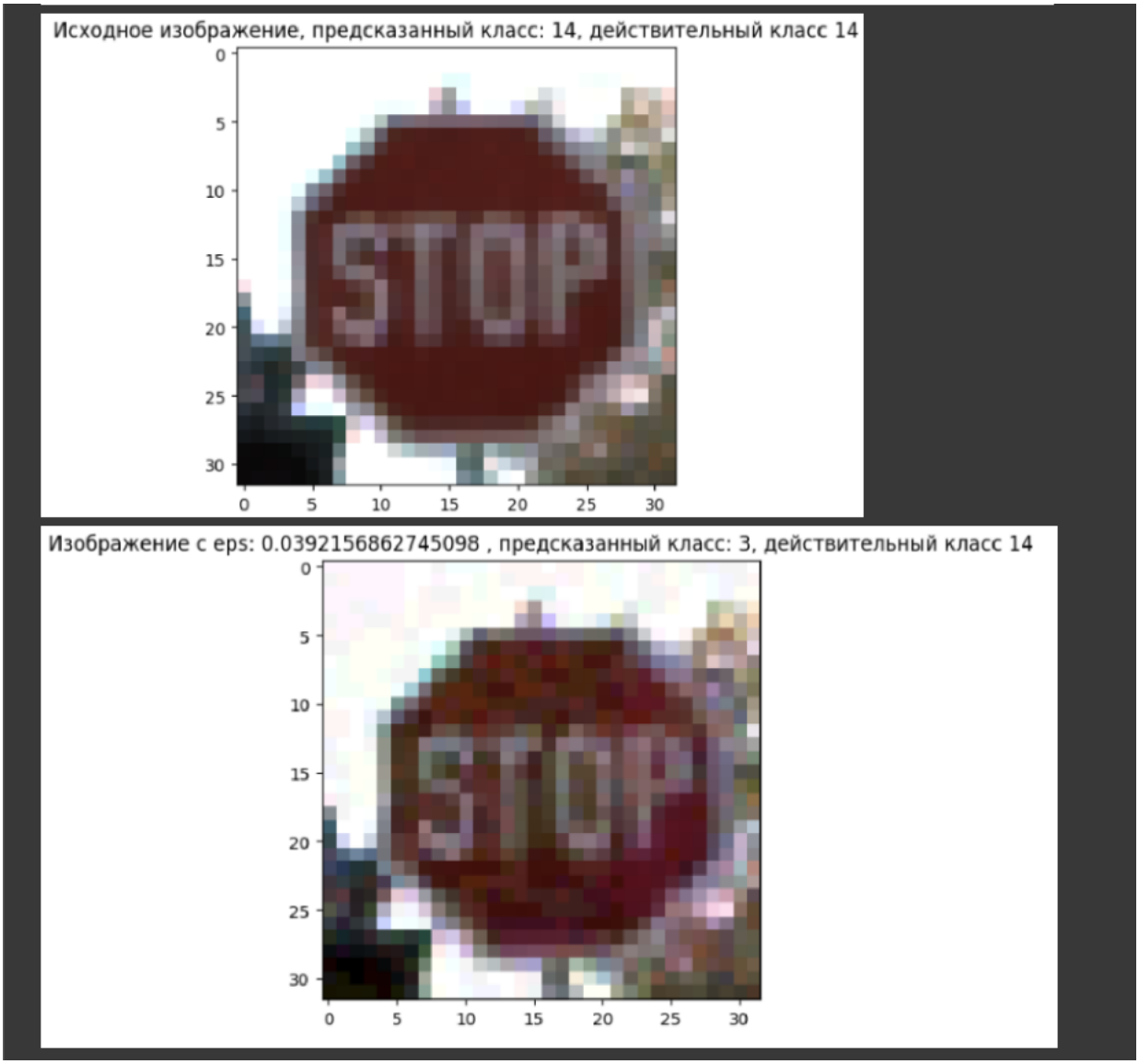


Исходное изображение, предсказанный класс: 14, действительный класс 14



Изображение с eps: 0.0392156862745098 , предсказанный класс: 1, действительный класс 14

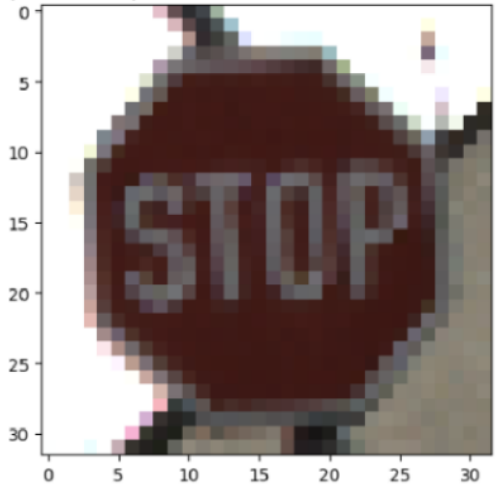




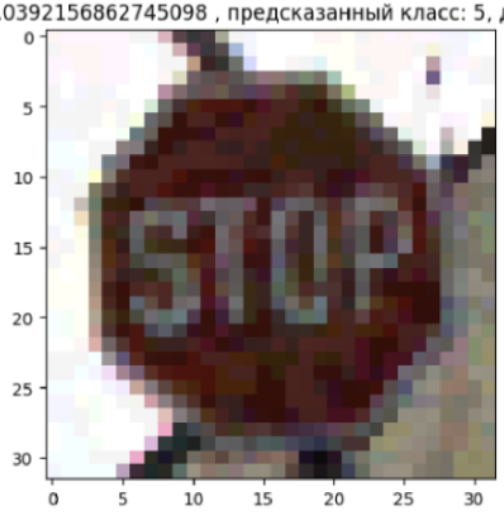
Пример атаки PGD:

Пример атаки PGD:

Исходное изображение, предсказанный класс: 14, действительный класс 14

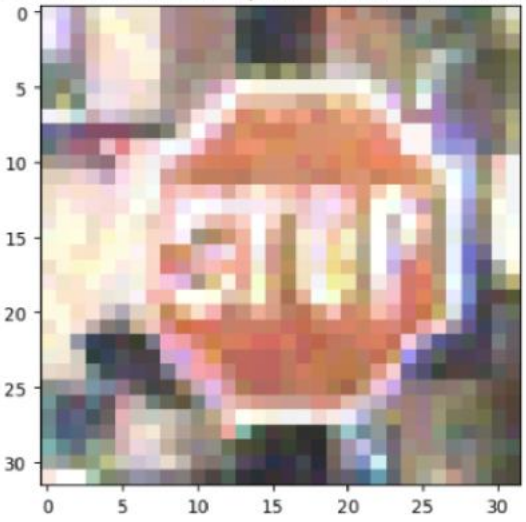


Изображение с eps: 0.0392156862745098 , предсказанный класс: 5, действительный класс 14

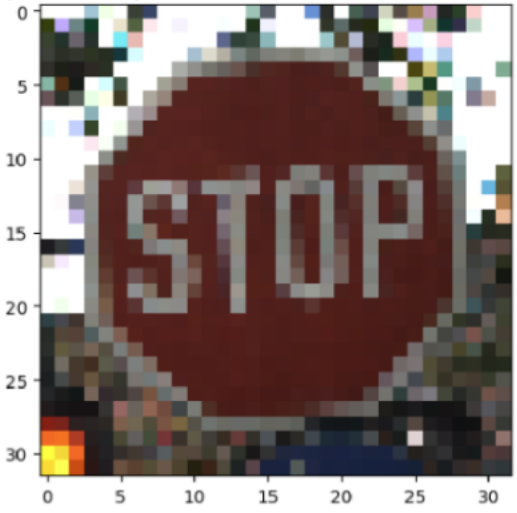


Исходное изображение, предсказанный класс: 14, действительный класс 14

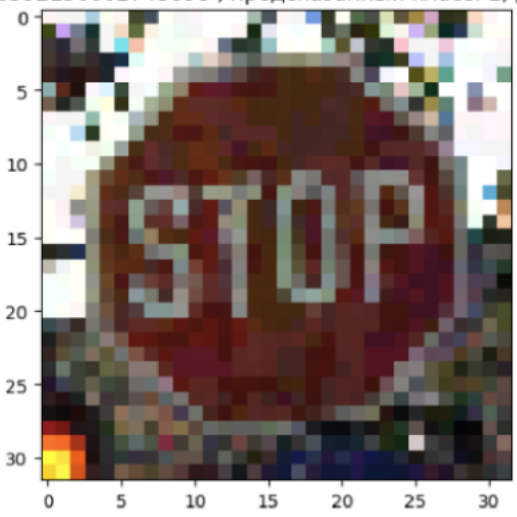
Изображение с eps: 0.0392156862745098 , предсказанный класс: 1, действительный класс 14

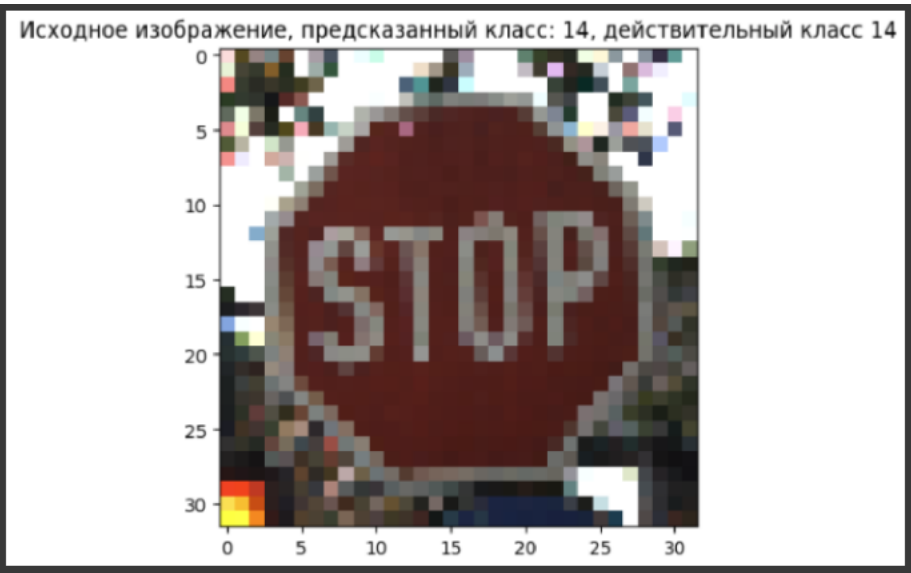


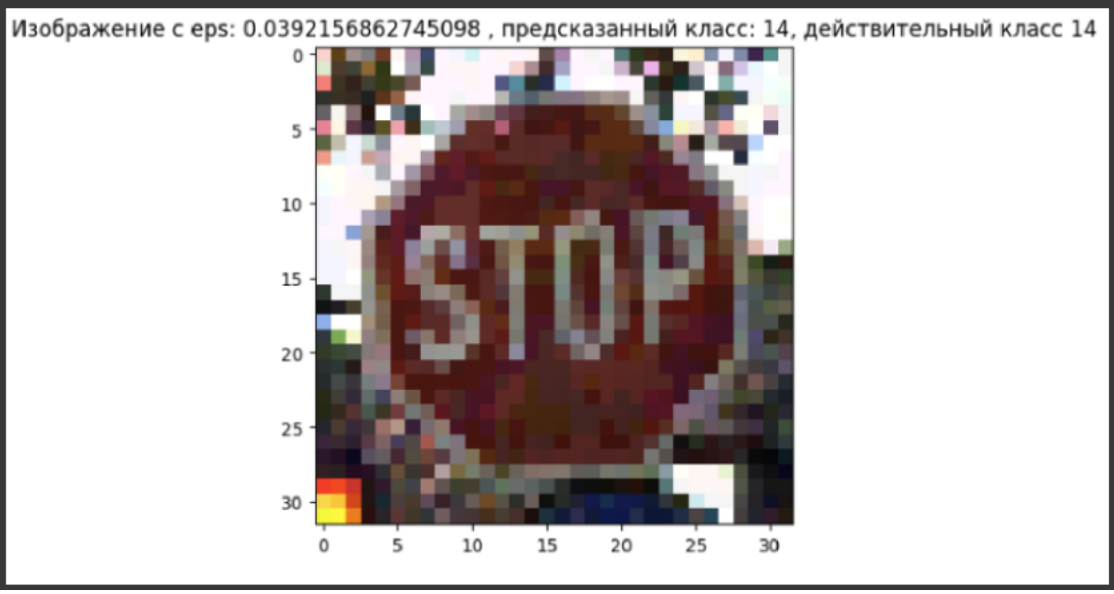
Исходное изображение, предсказанный класс: 14, действительный класс 14

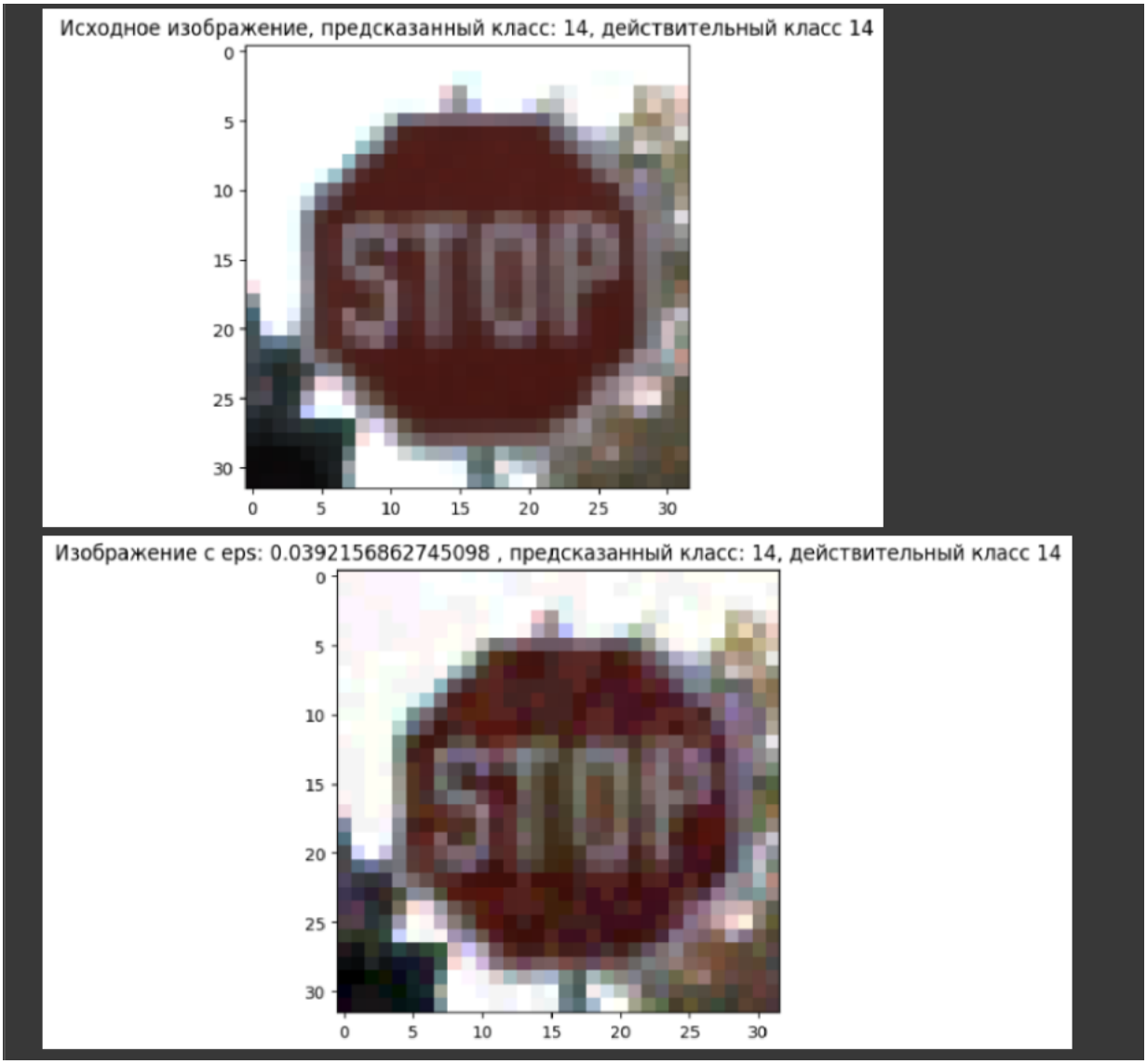


Изображение с eps: 0.0392156862745098 , предсказанный класс: 1, действительный класс 14




Исходное изображение, предсказанный класс: 14, действительный класс 14
0
5
10
15
20
25
30
STOP


Изображение с eps: 0.0392156862745098 , предсказанный класс: 14, действительный класс 14
0
5
10
15
20
25
30
STOP



10. Заполнение итоговой таблицы

10. Заполнение итоговой таблицы:

Искажение	PGD-атака	FGSM-атака
e=1/255	98%	99%
e=3/255	89%	84%
e=5/255	91%	74%
e=10/255	69%	26%
e=20/255	58%	3%
e=50/255	1%	0%
e=80/255	1%	0%