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-02-22
Выполнил:
Шитов А.В.

Проверил:
к.т.н Спирин А.А.

Москва 2023

Задачи

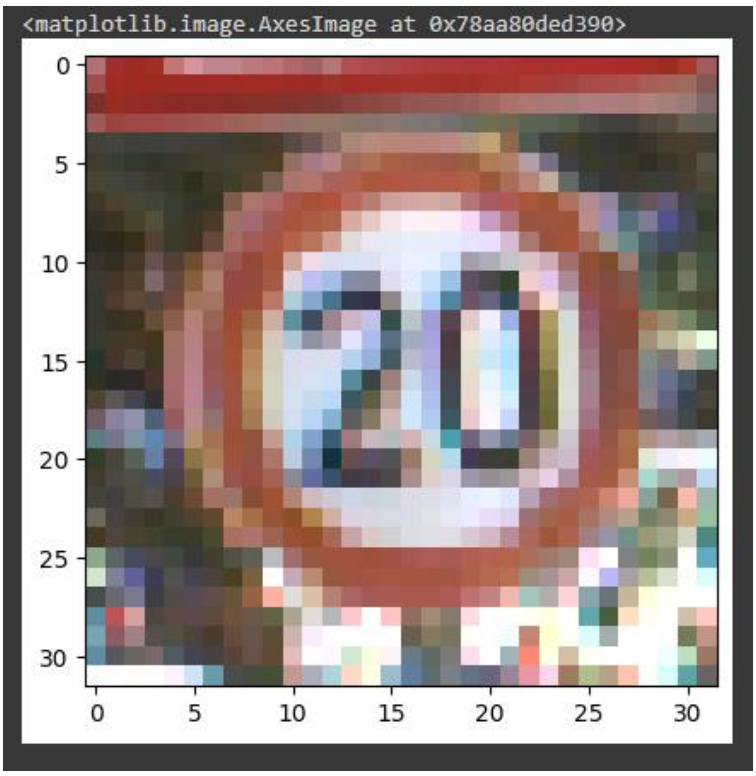
- Реализовать атаки уклонения на основе белого ящика против классификационных моделей на основе глубокого обучения.
- Получить практические навыки переноса атак уклонения на основе черного ящика против моделей машинного обучения.

Набор данных

Для этой части используйте набор данных GTSRB (German Traffic Sign Recognition Benchmark). Набор данных состоит примерно из 51 000 изображений дорожных знаков. Существует 43 класса дорожных знаков, а размер изображений составляет 32×32 пикселя.

Задание 1

Обучим 2 классификатора на основе глубоких нейронных сетей на датасете GTSRB. При извлечении картинок для создания тренировочной выборки, получим матричное представление картинки. Для восприятия моделями нейронных сетей, данные были масштабированы.



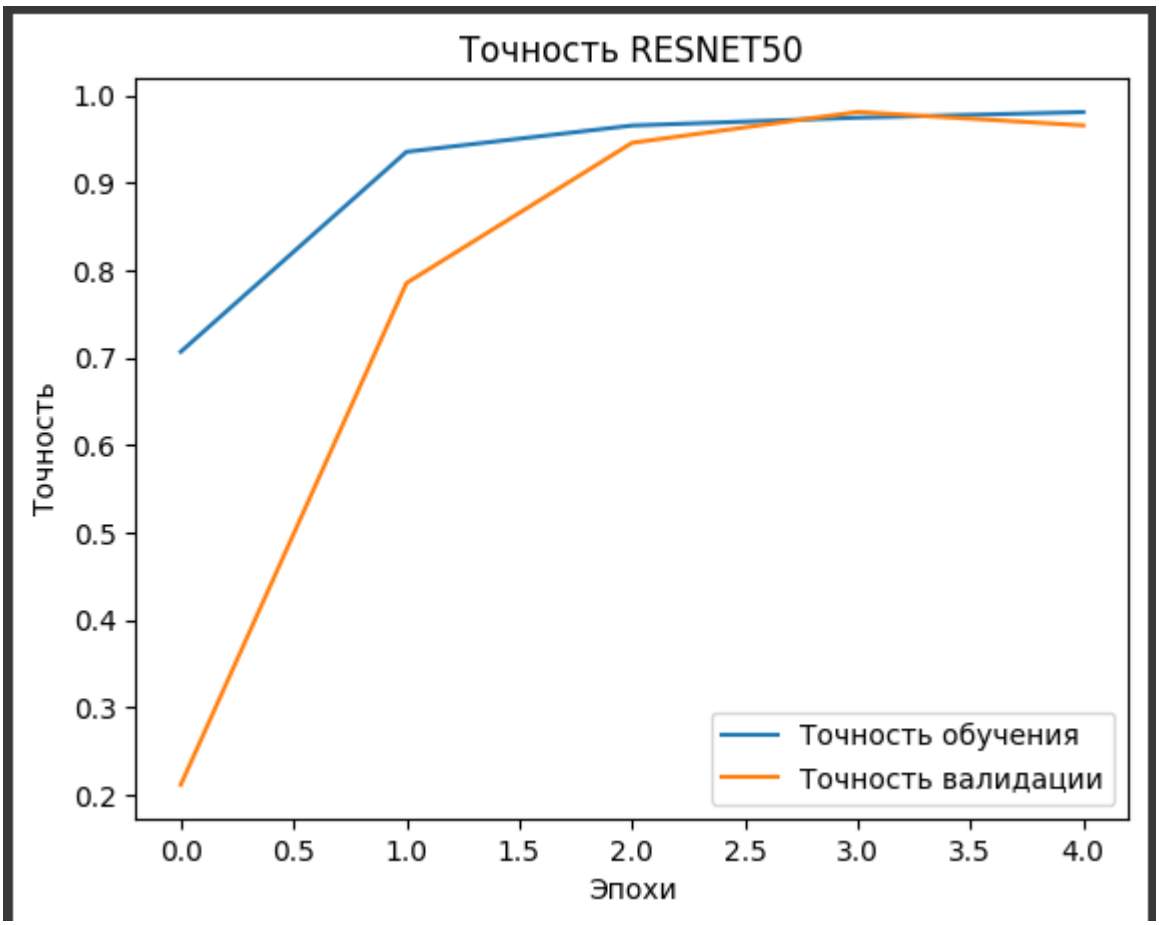
В качестве первой модель будет использоваться ResNet50. Опытным путём были выбраны лучшие значения количества эпох обучения и размера пакета.

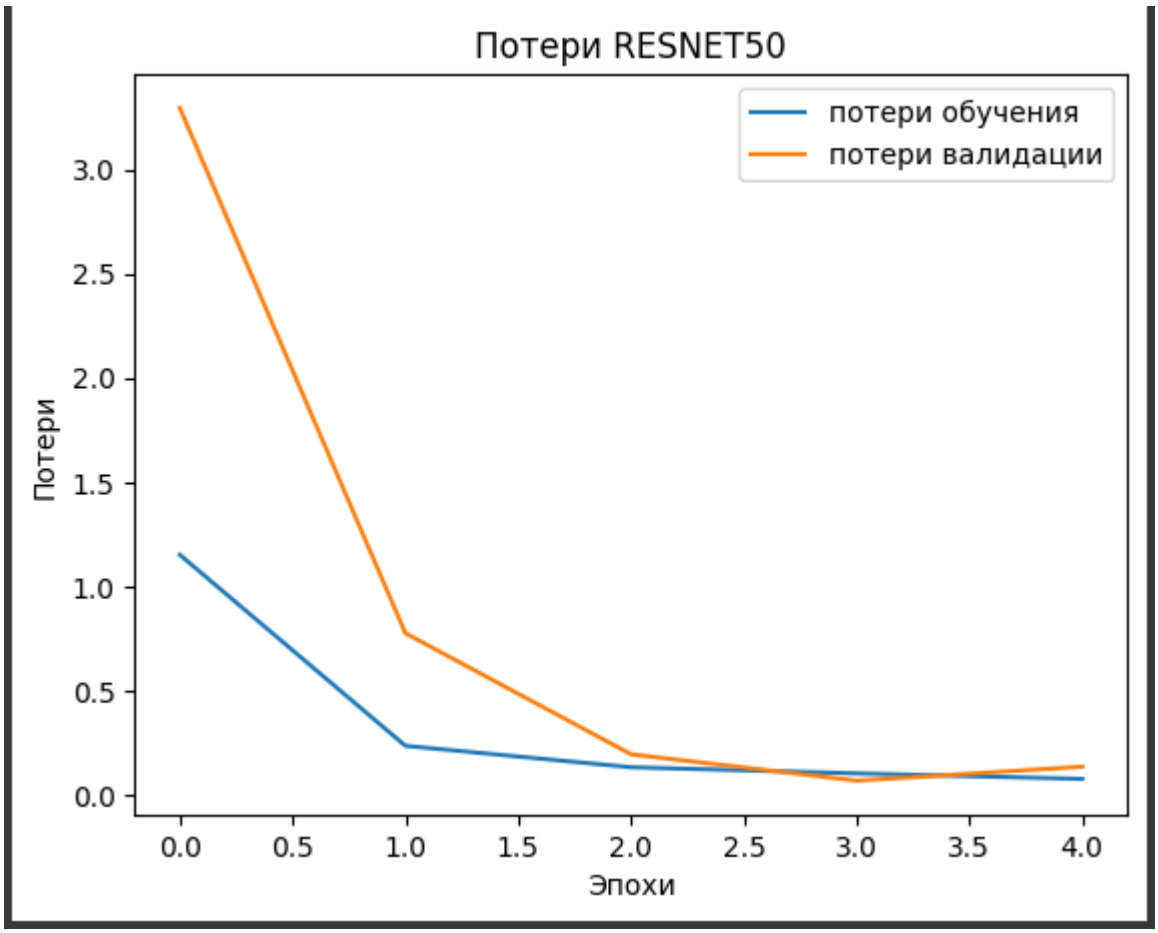
```
[ ]  model.compile(loss = 'categorical_crossentropy', metrics = ['accuracy'])
     history = model.fit(x_train, y_train, validation_data =(x_val, y_val), epochs = 5, batch_size = 64)

     Epoch 1/5
     429/429 [==============================] - 57s 63ms/step - loss: 1.1523 - accuracy: 0.7070 - val_loss: 3.2945 - val_accuracy: 0.2119
     Epoch 2/5
     429/429 [==============================] - 22s 50ms/step - loss: 0.2361 - accuracy: 0.9356 - val_loss: 0.7760 - val_accuracy: 0.7855
     Epoch 3/5
     429/429 [==============================] - 23s 53ms/step - loss: 0.1334 - accuracy: 0.9657 - val_loss: 0.1955 - val_accuracy: 0.9459
     Epoch 4/5
     429/429 [==============================] - 22s 50ms/step - loss: 0.1039 - accuracy: 0.9747 - val_loss: 0.0691 - val_accuracy: 0.9812
     Epoch 5/5
     429/429 [==============================] - 22s 52ms/step - loss: 0.0775 - accuracy: 0.9811 - val_loss: 0.1360 - val_accuracy: 0.9658
```

RestNet50

Далее необходимо построить графики, отражающие качество обучения модели ResNet50. Было принято решение остановиться на 5 эпохах, так как итоговая точность увеличилась по мере роста числа эпох.


Точность RESNET50
Точность
Эпохи
Точность обучения
Точность валидации
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0


Потери RESNET50
Потери
Эпохи
потери обучения
потери валидации
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

После этого необходимо проверить модель на тестовом наборе данных

```
[ ]  loss, accuracy = model.evaluate(data, y_test)
     print(f"Потери тестовой выборки: {loss}")
     print(f"Точность тестовой выборки: {accuracy}")

     395/395 [==============================] - 7s 16ms/step - loss: 0.4720 - accuracy: 0.9129
     Потери тестовой выборки: 0.4720468819141388
     Точность тестовой выборки: 0.912905752658844
```

Точность составила 91%

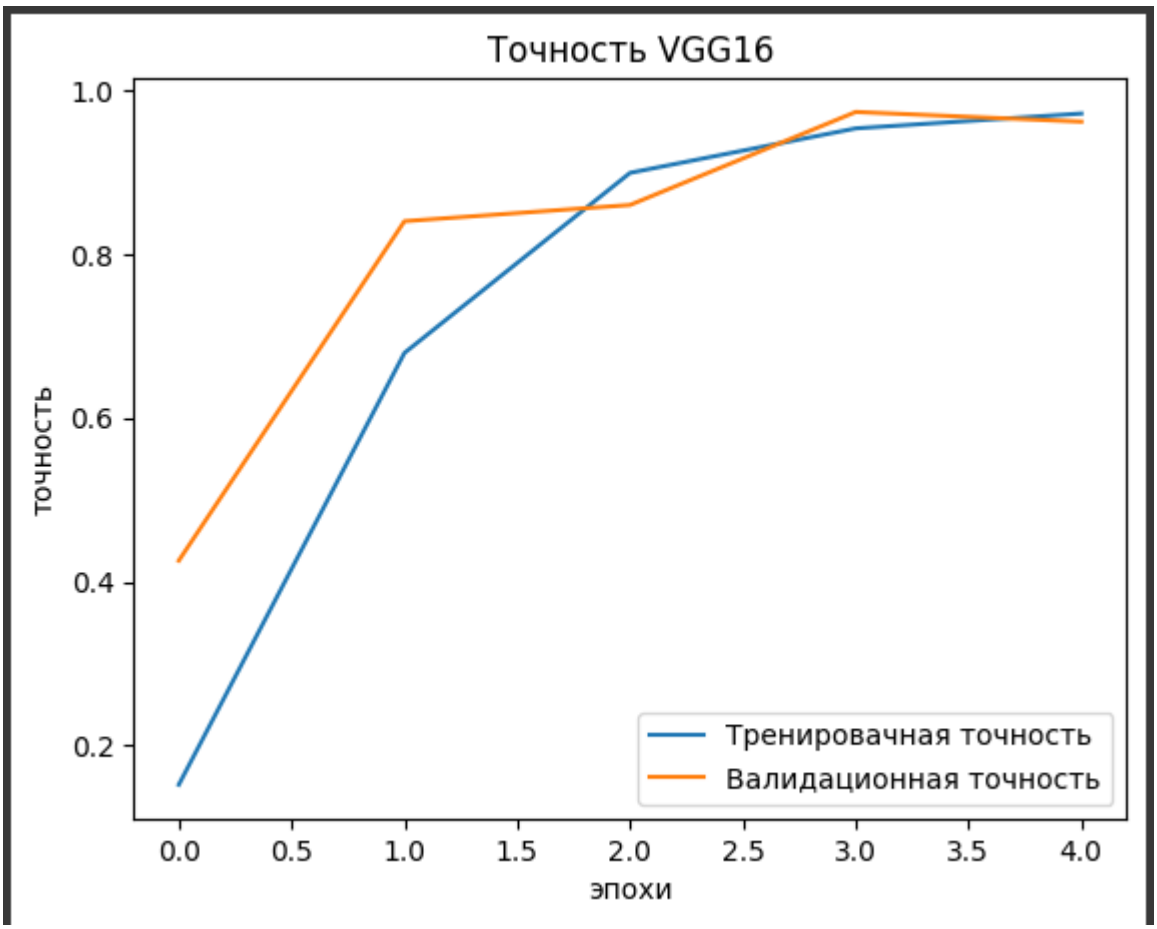
VGG16

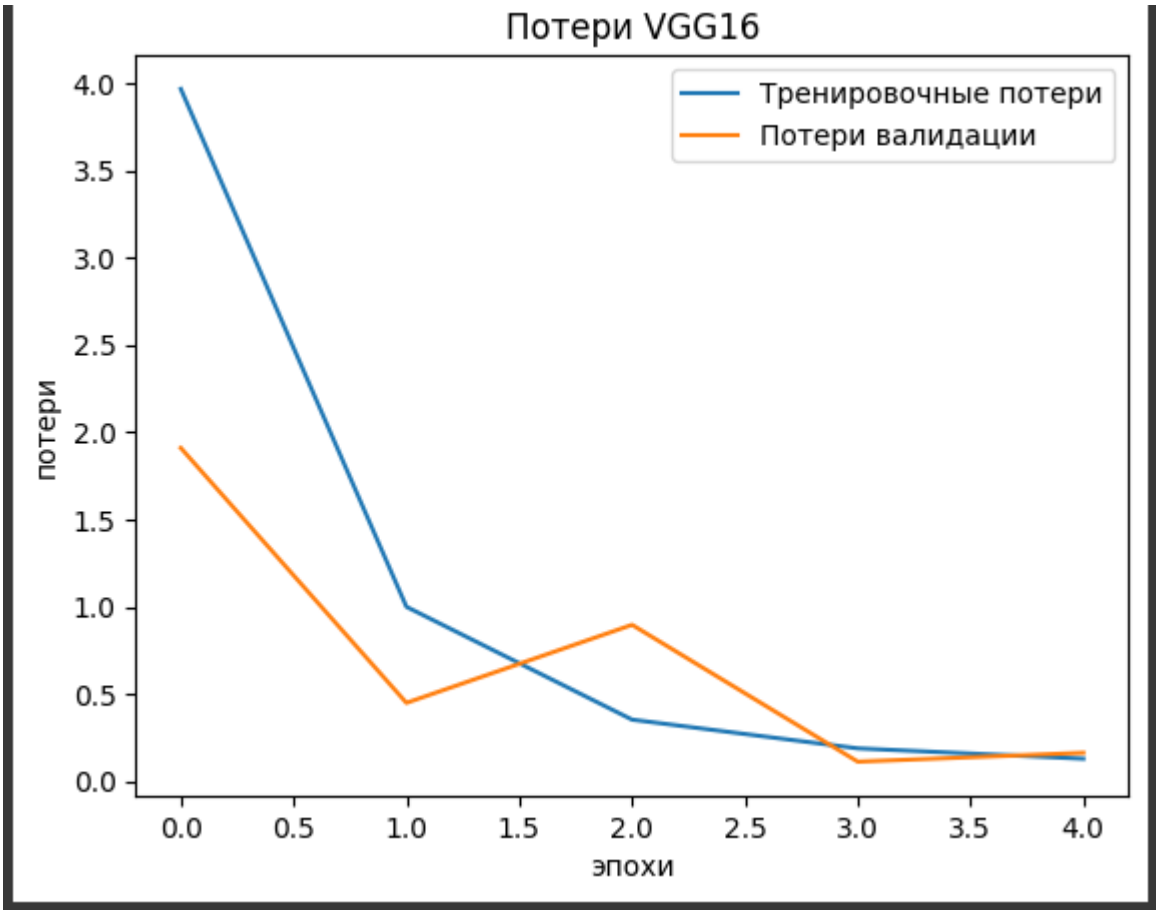
Следующей будет модель VGG16, необходимо её обучить

```
[ ]  model.compile(loss = 'categorical_crossentropy', metrics = ['accuracy'])
     history = model.fit(x_train, y_train, validation_data =(x_val, y_val), epochs = 5, batch_size = 64)

     Epoch 1/5
     429/429 [==============================] - 26s 48ms/step - loss: 3.9668 - accuracy: 0.1523 - val_loss: 1.9099 - val_accuracy: 0.4257
     Epoch 2/5
     429/429 [==============================] - 18s 42ms/step - loss: 0.9996 - accuracy: 0.6792 - val_loss: 0.4491 - val_accuracy: 0.8399
     Epoch 3/5
     429/429 [==============================] - 17s 39ms/step - loss: 0.3530 - accuracy: 0.8989 - val_loss: 0.8963 - val_accuracy: 0.8597
     Epoch 4/5
     429/429 [==============================] - 18s 41ms/step - loss: 0.1885 - accuracy: 0.9532 - val_loss: 0.1123 - val_accuracy: 0.9732
     Epoch 5/5
     429/429 [==============================] - 17s 40ms/step - loss: 0.1292 - accuracy: 0.9713 - val_loss: 0.1629 - val_accuracy: 0.9614
```

Далее необходимо построить графики, аналогичные графикам для модели ResNet50


Точность VGG16
точность
эпохи
Тренировачная точность
Валидационная точность
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


Потери VGG16
потери
эпохи
Тренировочные потери
Потери валидации
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

После этого необходимо проверить модель на тестовом наборе данных

```
[ ]  loss, accuracy = model.evaluate(data, y_test)
     print(f"Тестовые потери: {loss}")
     print(f"Тестовая точность: {accuracy}")

     395/395 [==============================] - 5s 10ms/step - loss: 0.3652 - accuracy: 0.9309
     Тестовые потери: 0.36521872878074646
     Тестовая точность: 0.9308788776397705
```

Точность составила 93%

Таблица 1. Сравнительная таблица моделей ResNet50 и VGG16

Модель	**Обучение**	**Валидация**	**Тест**
ResNet50	Потери: 0.08	Потери: 0.14	Потери: 0.47
	Точность: 0.98	Точность: 0.97	Точность: 0.91
VGG16	Потери: 0.13	Потери: 0.16	Потери: 0.37
	Точность: 0.97	Точность: 0.96	Точность: 0.93

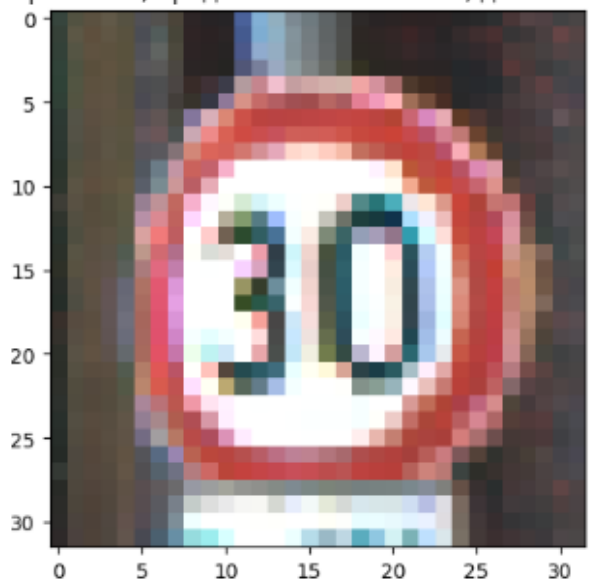
Задание 2

Применить нецелевую атаку уклонения на основе белого ящика против моделей глубокого обучения. Реализовать следующие типы атак:

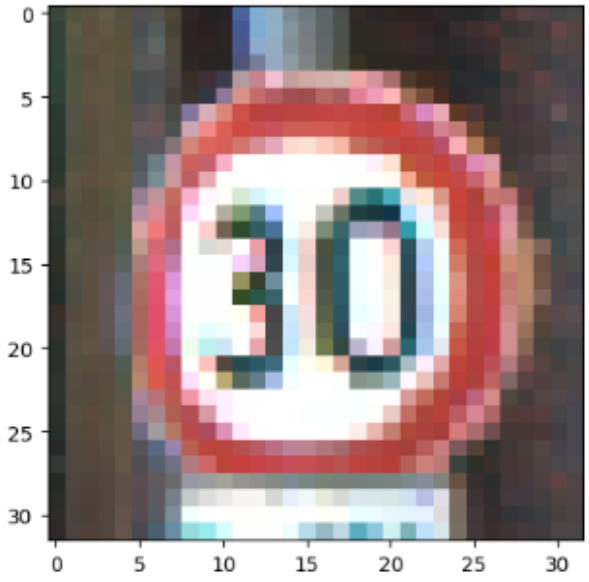
- Fast Gradient Sign Method (FGSM)
- Projected Gradient Descent (PGD)

Необходимо создать модель атаки, которая основывается на классификаторе для внесения шума в изображение. Ниже представлено отображение исходного и атакующих изображений для атаки FGSM

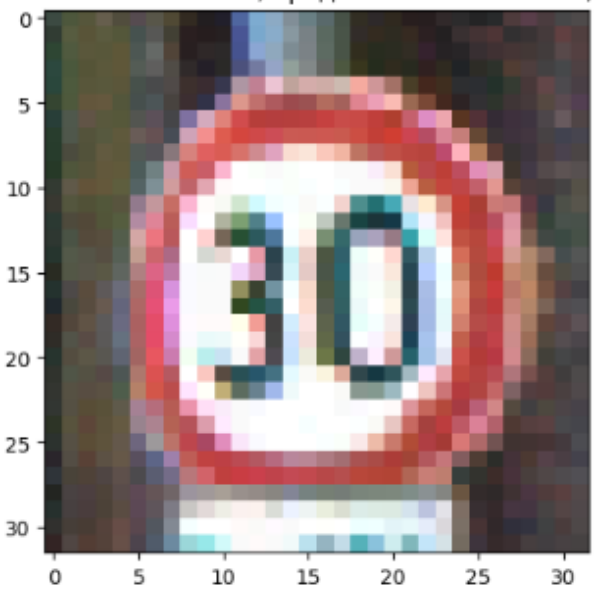
Исходное изображение, предсказанный класс: 1, действительный класс 1



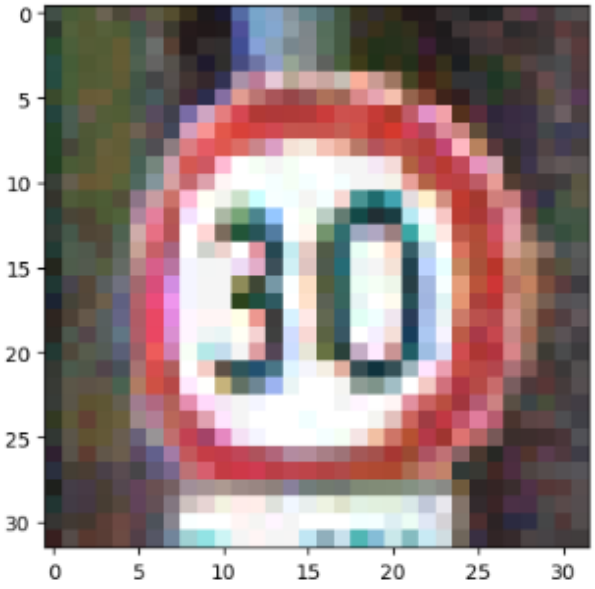
Изображение с eps: 0.00392156862745098 , предсказанный класс: 1, действительный класс 1

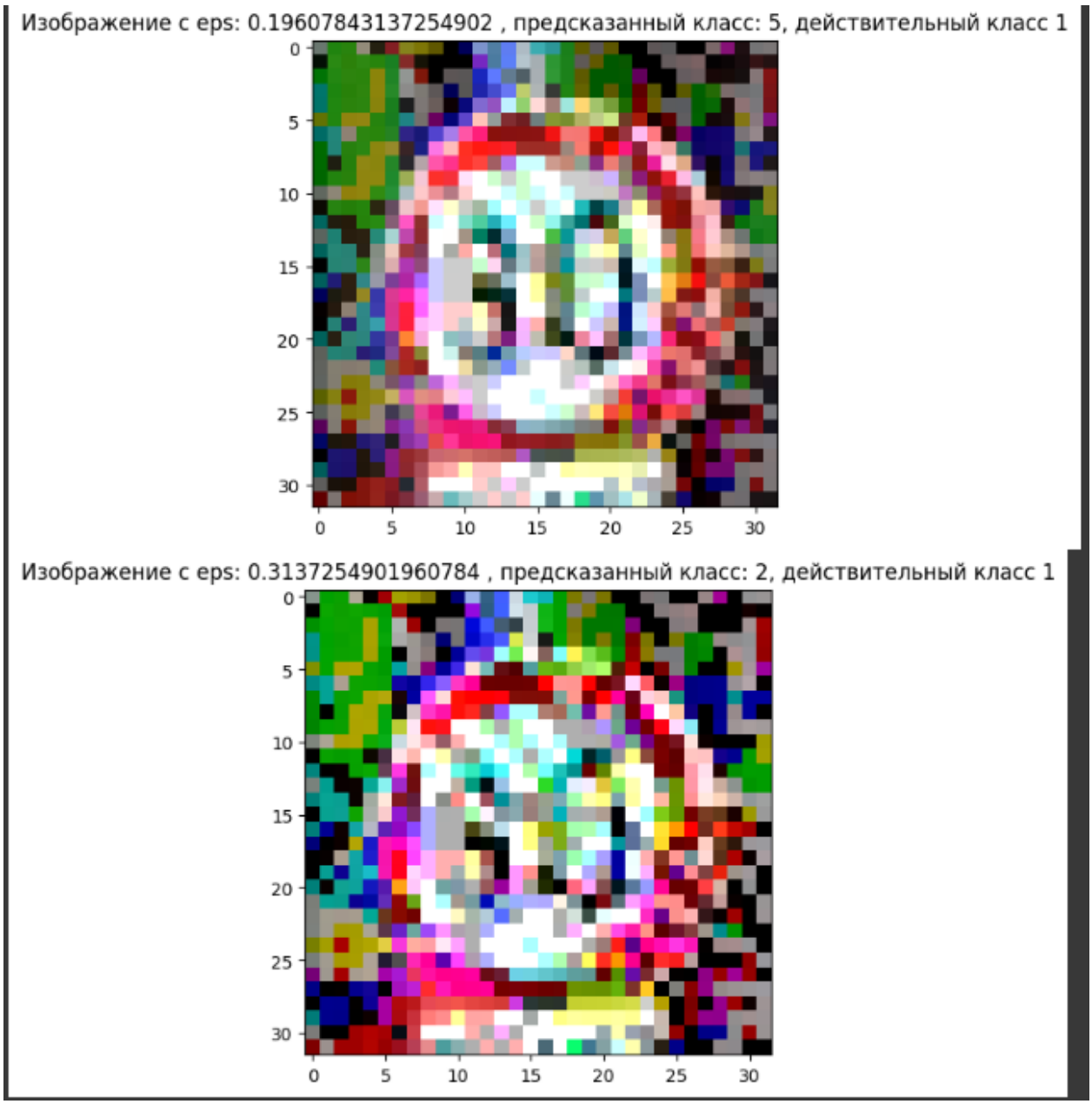


Изображение с eps: 0.0196078431372549 , предсказанный класс: 5, действительный класс 1



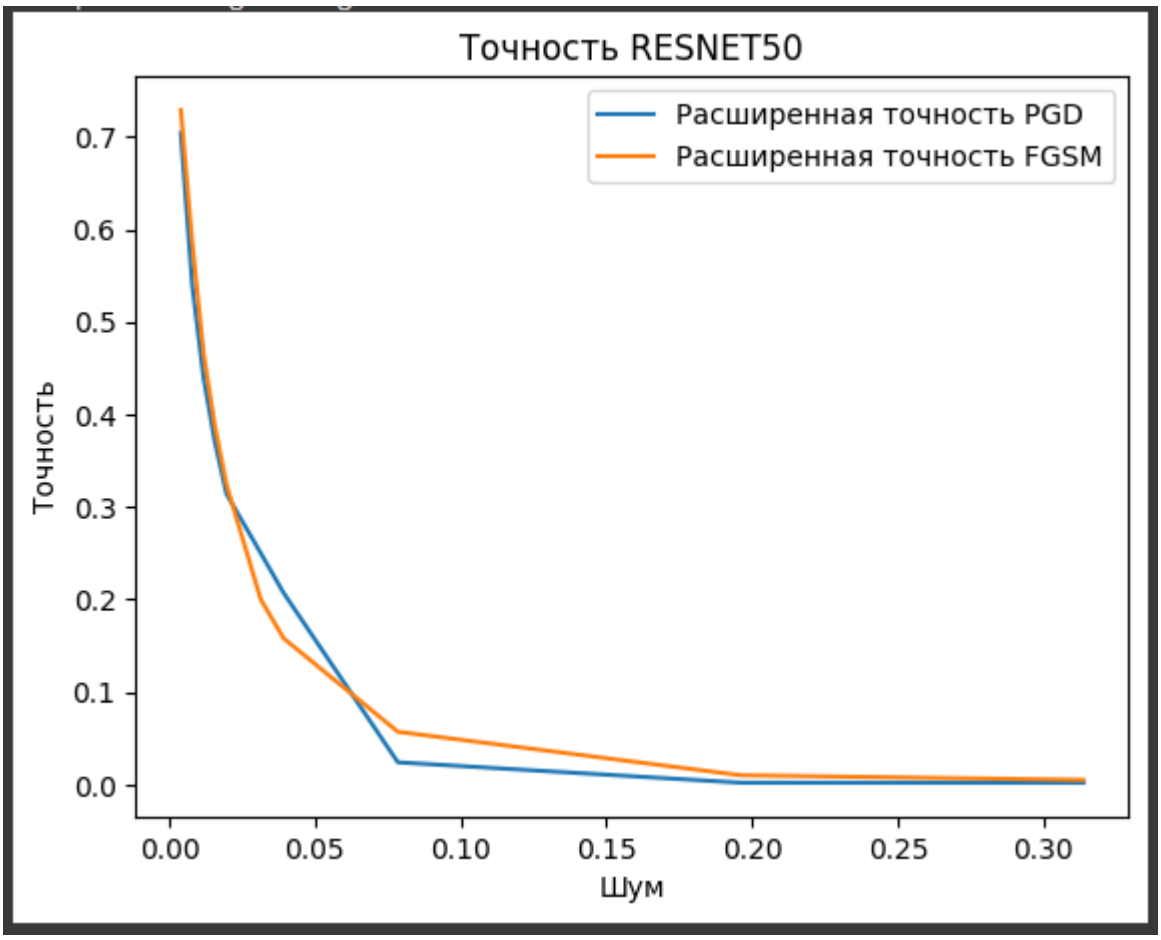
Изображение с eps: 0.0392156862745098 , предсказанный класс: 5, действительный класс 1





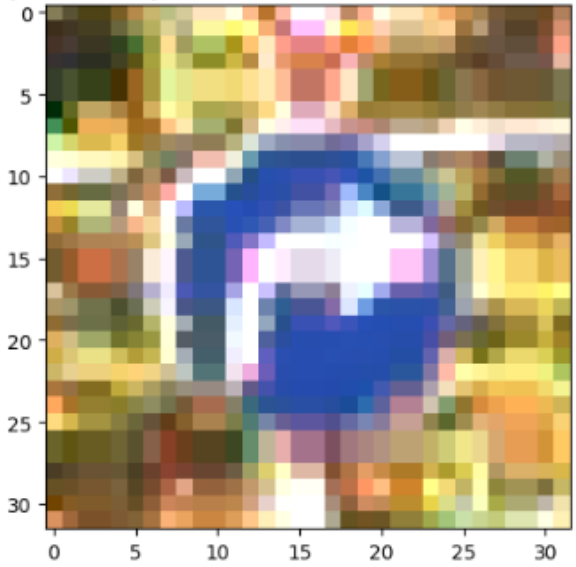
Далее нужно построить график зависимости точности предсказания модели на атакованных изображениях от параметра искажения.

Исходя из графика можно сделать вывод, что данные методы имеют примерно одинаковую эффективность.

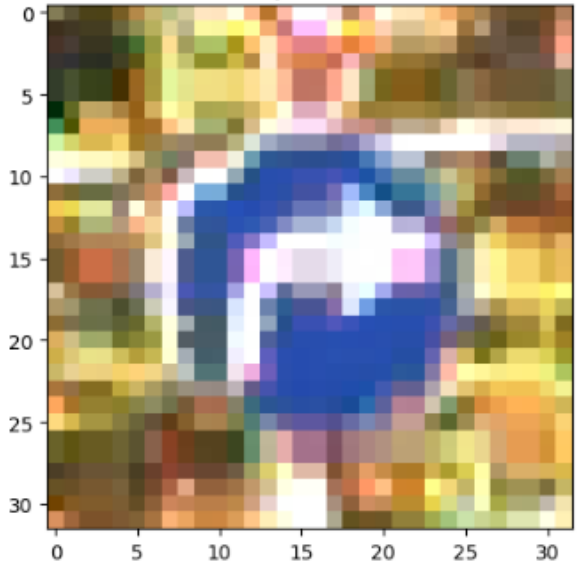


Теперь необходимо повторить эксперимент с атаками FGSM и PGD на базе модели VGG16. Ниже представлено отображение исходного и атакующих изображений для атаки FGSM

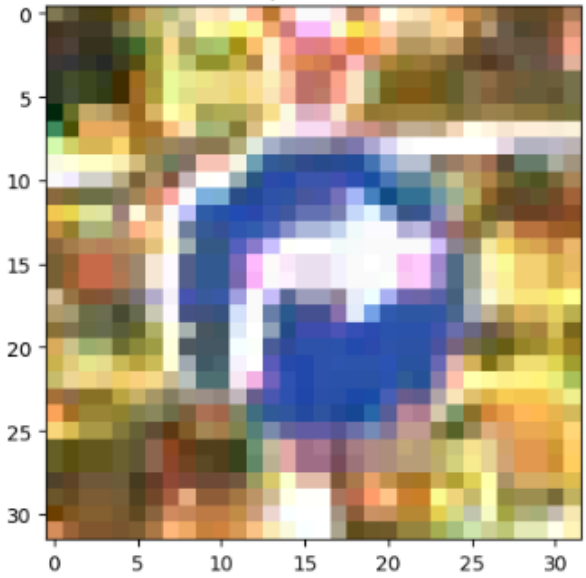
Исходное изображение, предсказанный класс: 33, действительный класс 33



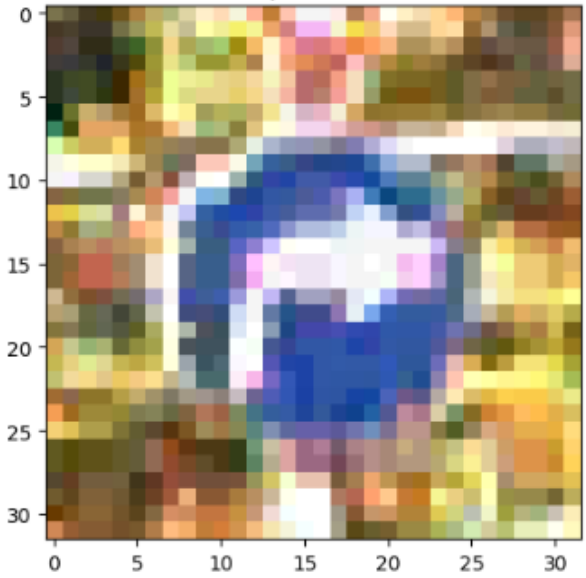
Изображение с eps: 0.00392156862745098 , предсказанный класс: 33, действительный класс 33



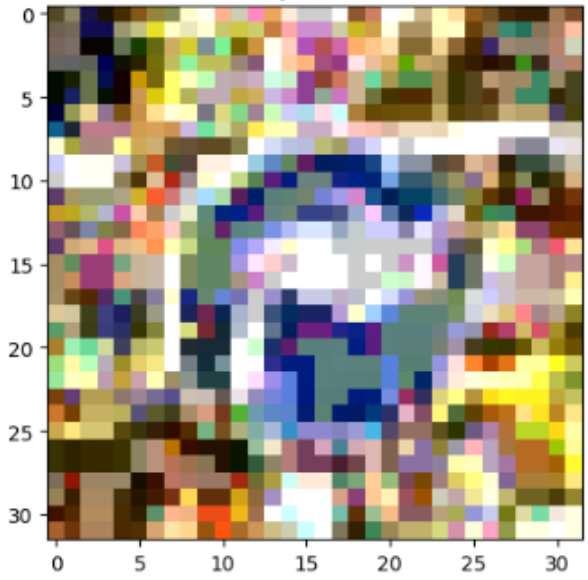
Изображение с eps: 0.0196078431372549 , предсказанный класс: 33, действительный класс 33



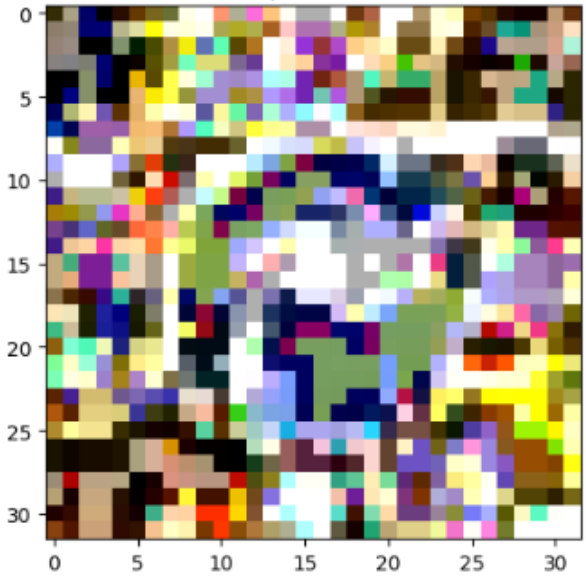
Изображение с eps: 0.0392156862745098 , предсказанный класс: 10, действительный класс 33



Изображение с eps: 0.19607843137254902 , предсказанный класс: 10, действительный класс 33



Изображение с eps: 0.3137254901960784 , предсказанный класс: 10, действительный класс 33



Далее нужно построить график зависимости точности предсказания модели на атакованных изображениях от параметра искажения.

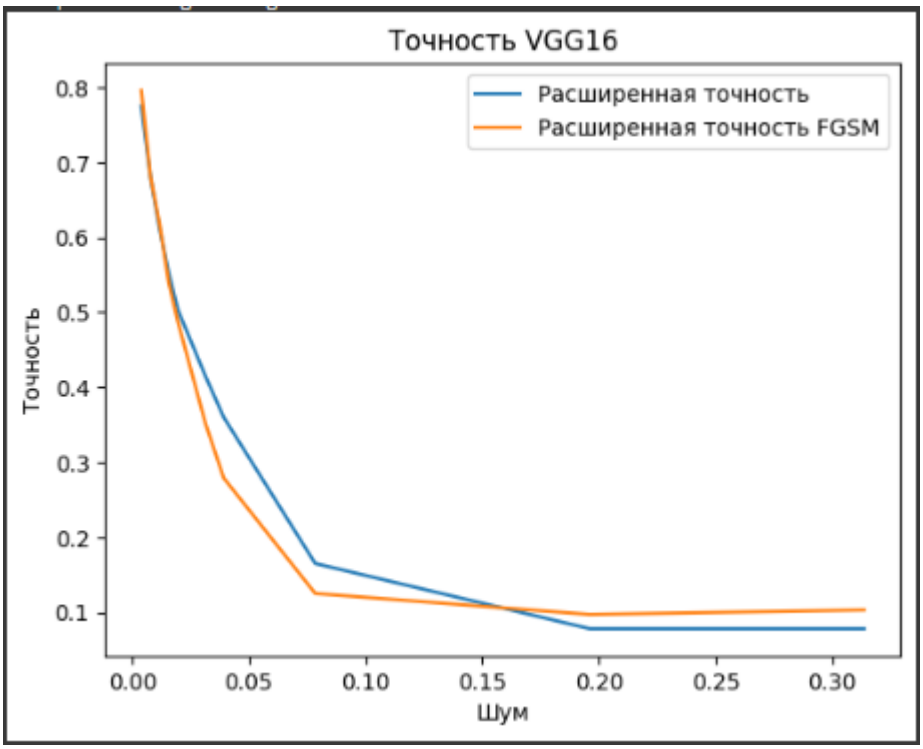


Таблица 2. Задание 2

Модель	**Исходные изображения, %**	**Adversarial images ϵ=1/255, %**	**Adversarial images ϵ=5/255, %**	**Adversarial images ϵ=10/255, %**
ResNet50 FGSM	91	72.9	32.4	15.8
ResNet50 PGD	91	70.3	31.4	20.7
VGG16 FGSM	93	79.6	48.7	27.9
VGG16 PGD	93	77.5	50.3	36

Задание 3

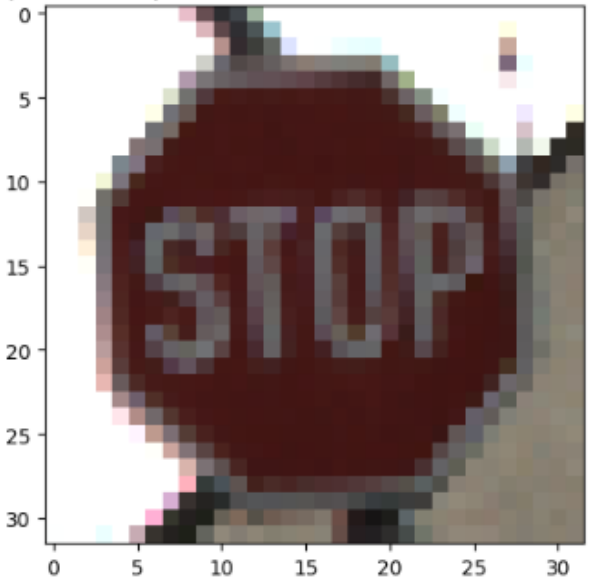
Применение целевой атаки уклонения методом белого против моделей глубокого обучения.

Шаг 1: Используйте изображения знака «Стоп» (label class 14) из тестового набора данных. Примените атаку PGD на знак «Стоп» с целью классификации его как знака «Ограничение скорости 30» (target label class = 1). Изменяйте значения искажений $\epsilon\epsilon$= [1/255, 3/255, 5/255, 10/255, 20/255, 50/255, 80/255], и заполните отчёт значениями точности классификации изображений знаков "Стоп" и "Ограничение скорости 30".

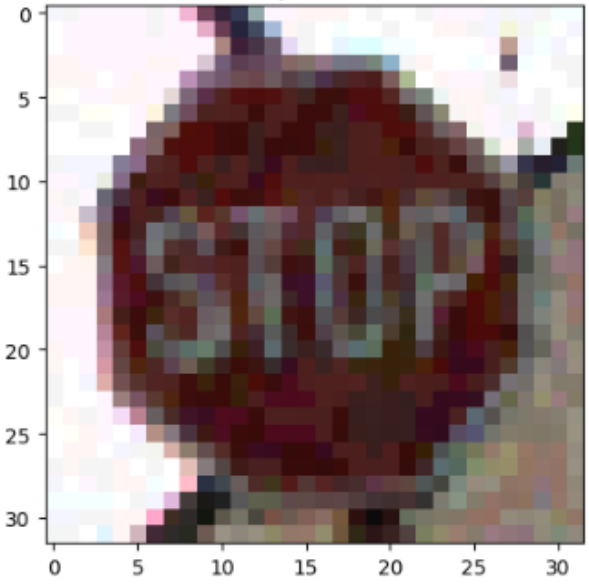
Шаг 2: Повторите атаку методом FGSM, и объясните производительность по сравнению с PGD. Сравните результаты атак PGD и FGSM между собой.

Шаг 1. Применение атаки PGD на знак «STOP» с целью классификации его как знака «Ограничение скорости 30».

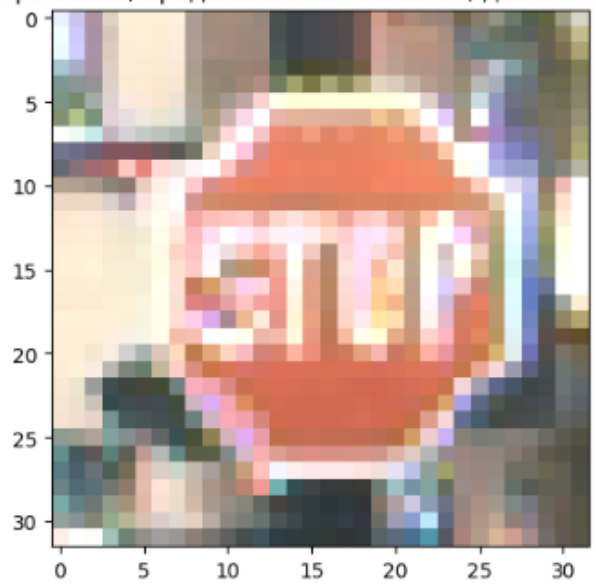
Исходное изображение, предсказанный класс: 14, действительный класс 14



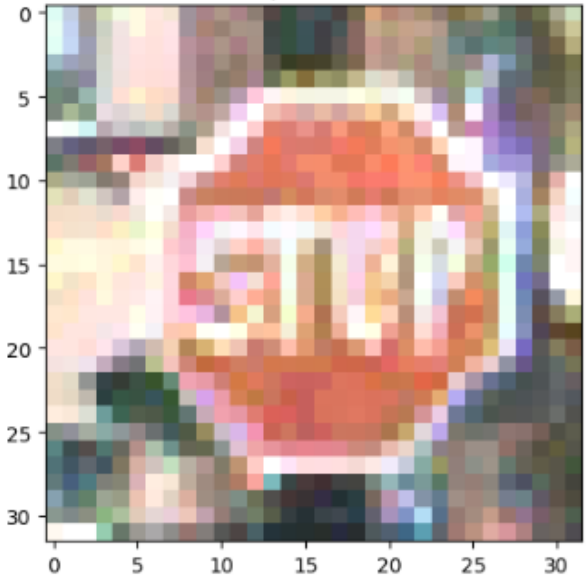
Изображение с eps: 0.0392156862745098 , предсказанный класс: 1, действительный класс 14



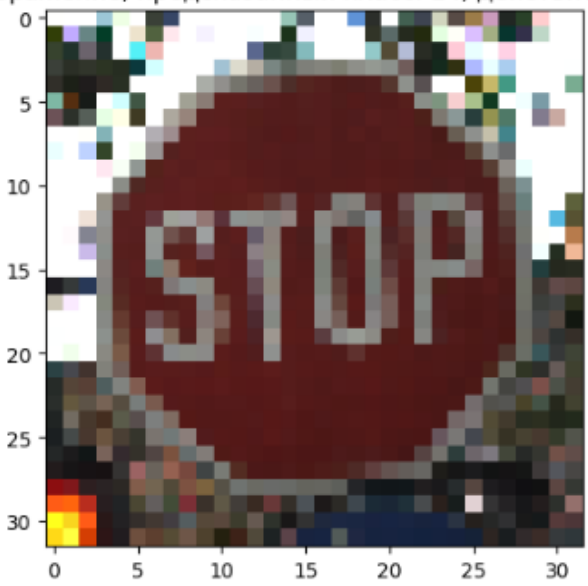
Исходное изображение, предсказанный класс: 14, действительный класс 14



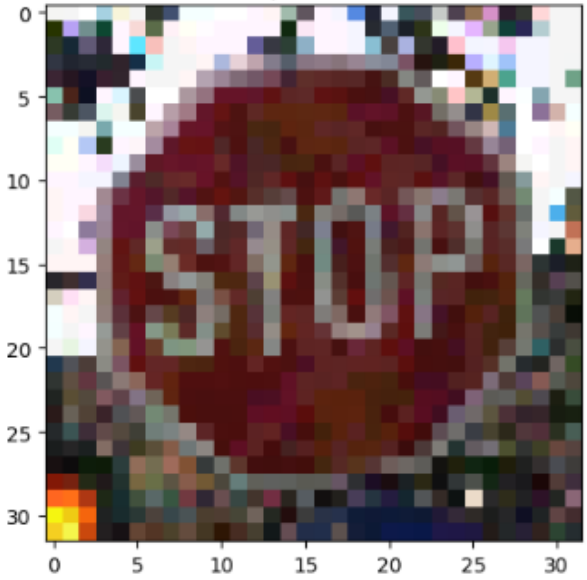
Изображение с eps: 0.0392156862745098 , предсказанный класс: 14, действительный класс 14



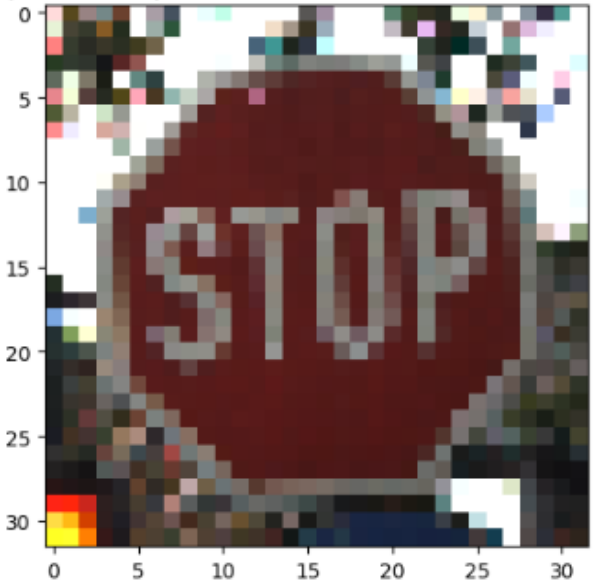
Исходное изображение, предсказанный класс: 14, действительный класс 14



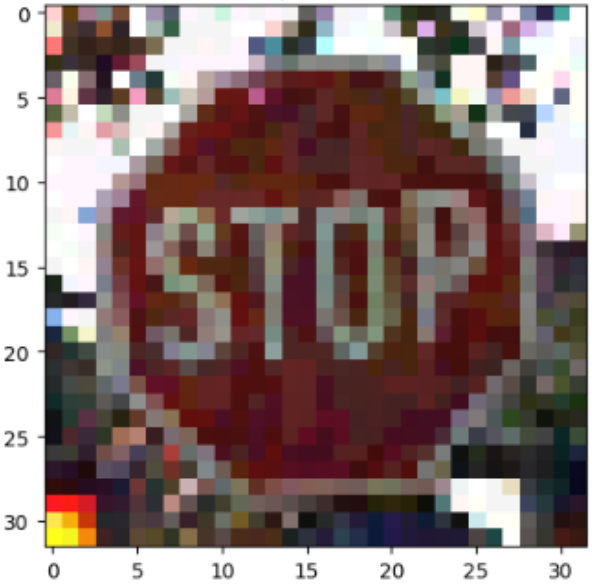
Изображение с eps: 0.0392156862745098 , предсказанный класс: 14, действительный класс 14



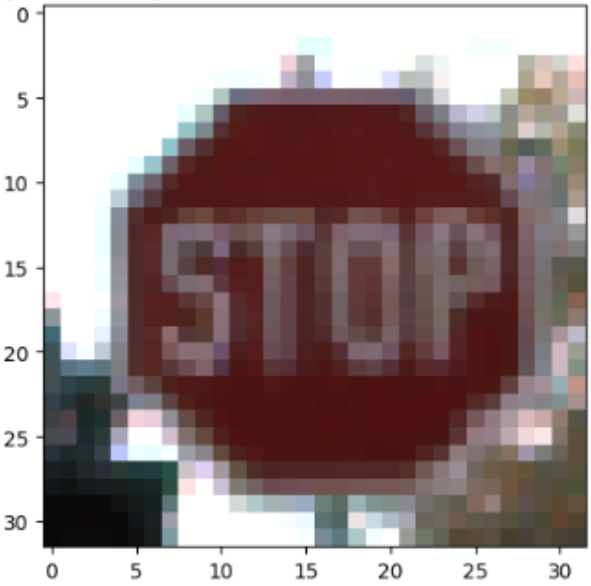
Исходное изображение, предсказанный класс: 14, действительный класс 14



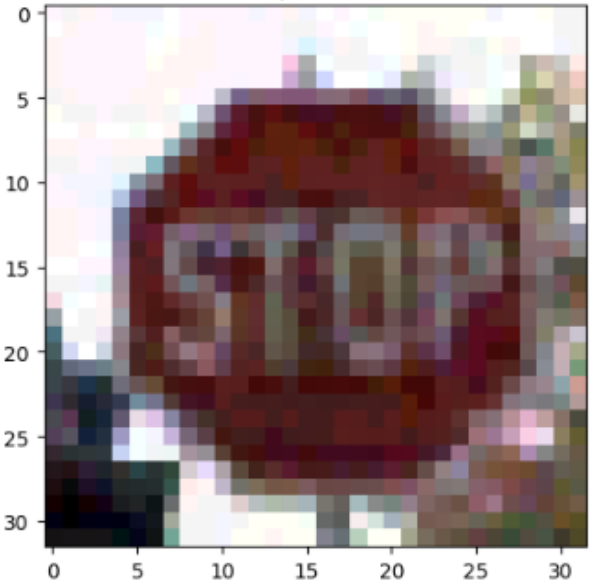
Изображение с eps: 0.0392156862745098 , предсказанный класс: 1, действительный класс 14



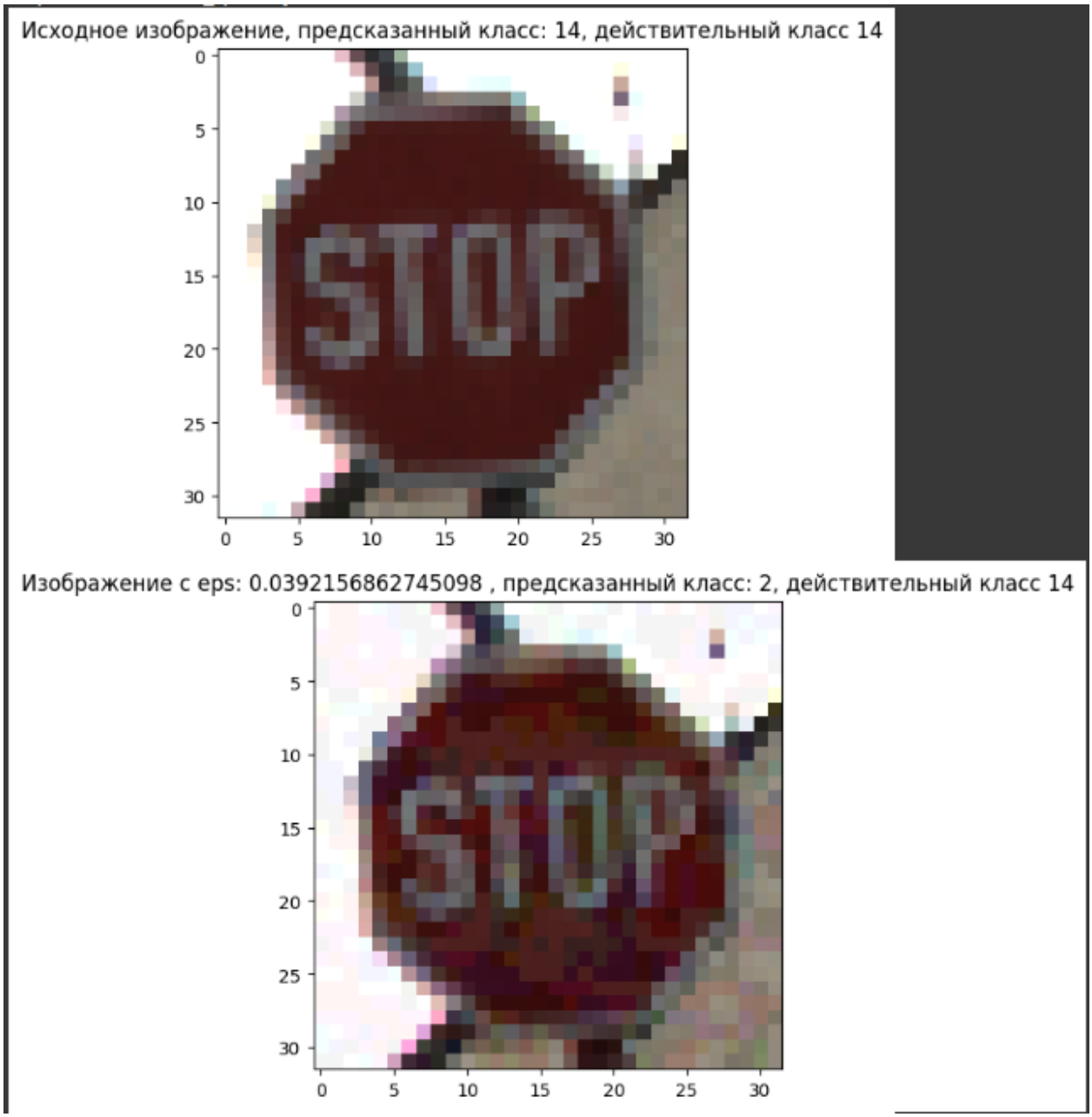
Исходное изображение, предсказанный класс: 14, действительный класс 14



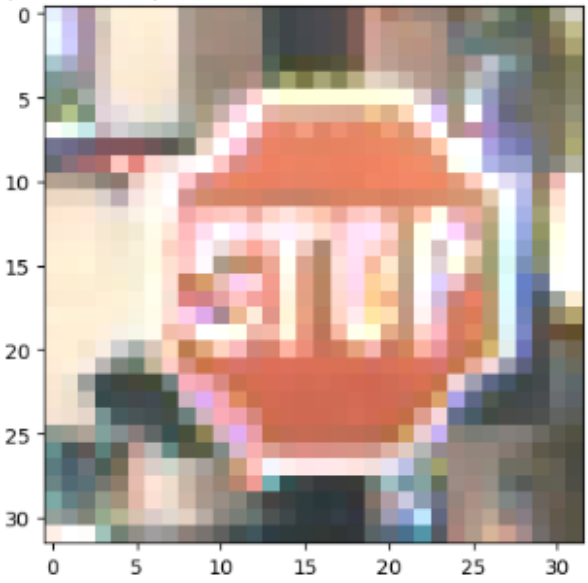
Изображение с eps: 0.0392156862745098 , предсказанный класс: 2, действительный класс 14



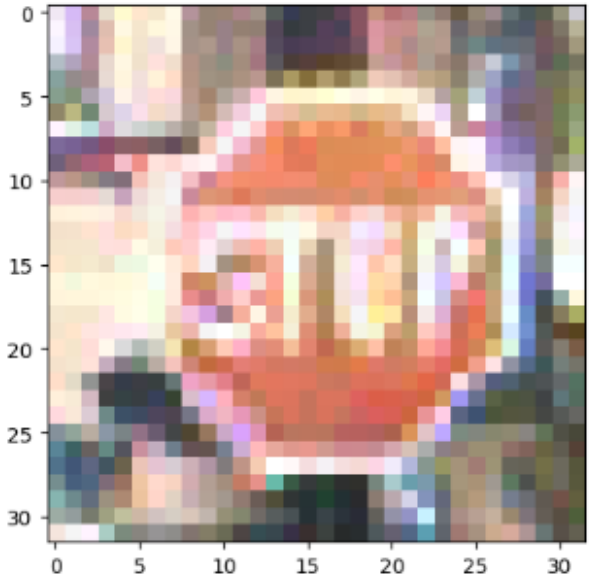
Повторение атаки методом FGSM



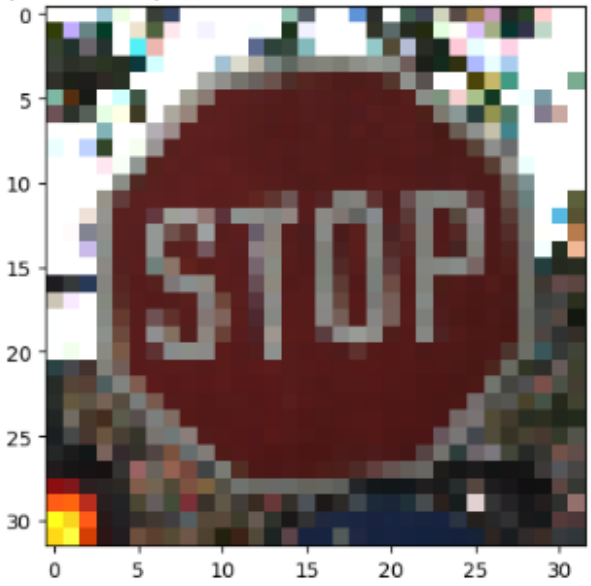
Исходное изображение, предсказанный класс: 14, действительный класс 14



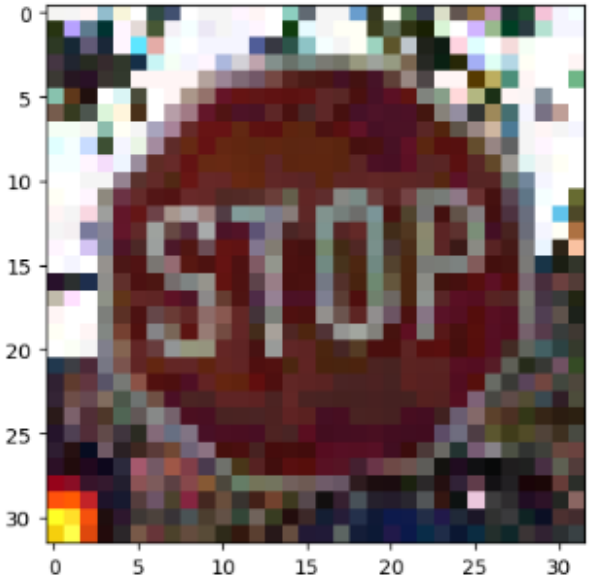
Изображение с eps: 0.0392156862745098 , предсказанный класс: 1, действительный класс 14



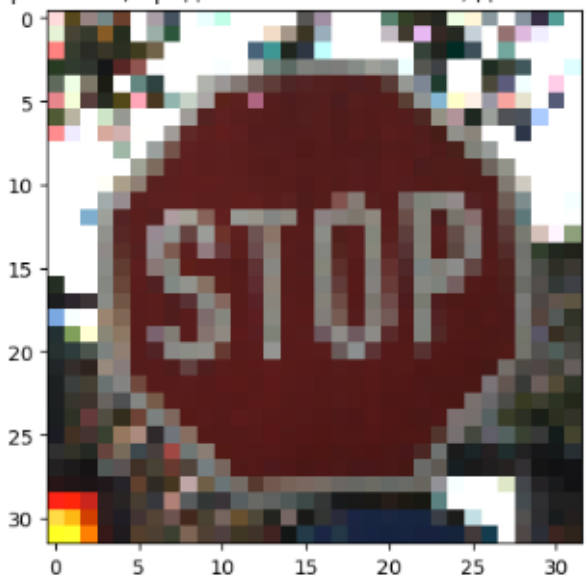
Исходное изображение, предсказанный класс: 14, действительный класс 14



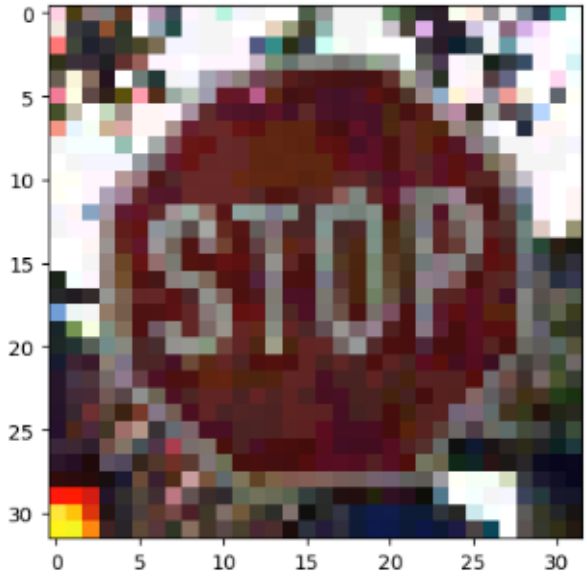
Изображение с eps: 0.0392156862745098 , предсказанный класс: 2, действительный класс 14



Исходное изображение, предсказанный класс: 14, действительный класс 14



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



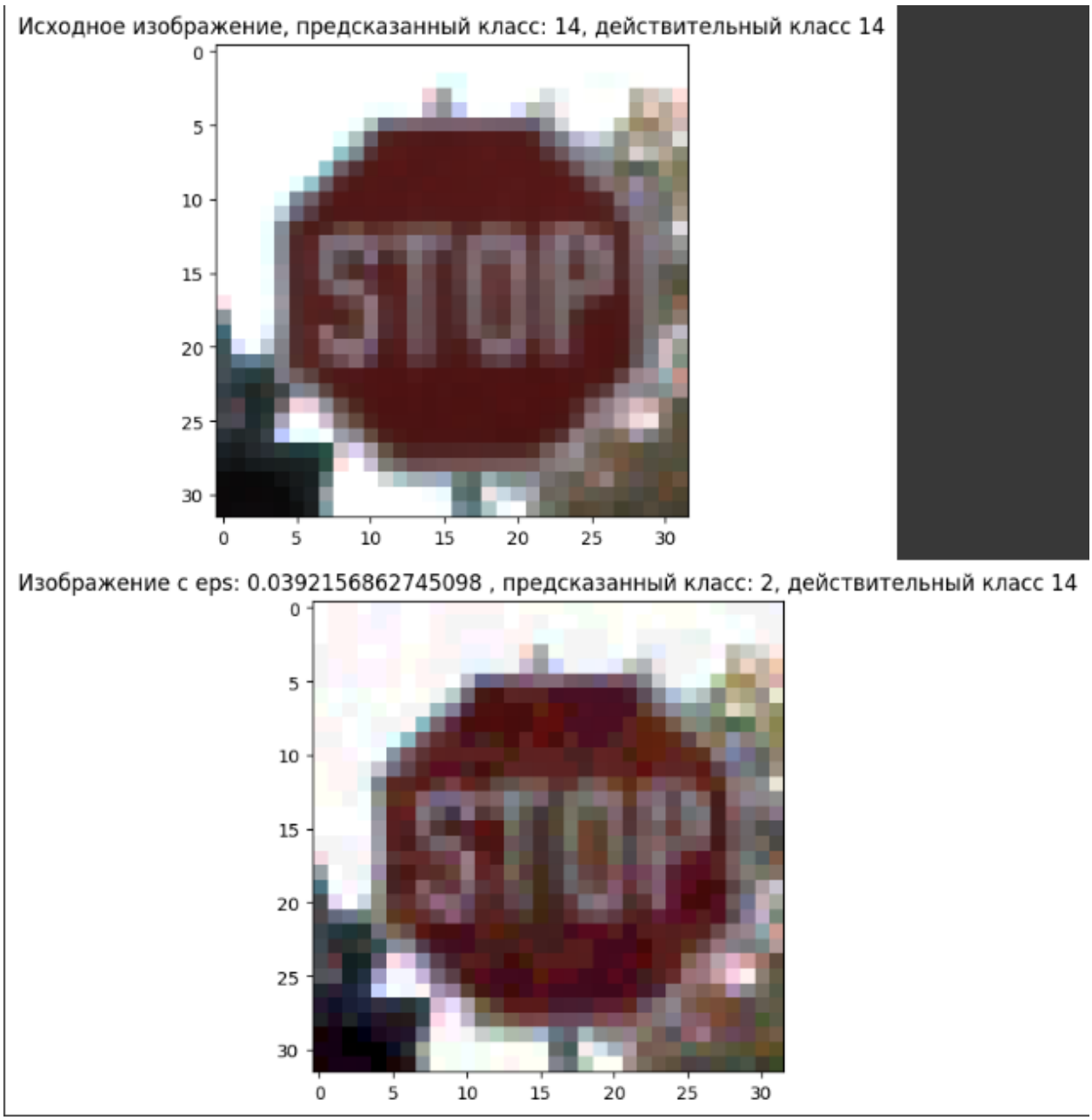


Таблица 3. Задание 3

Искажение	**PGD attack – Stop sign images, %**	**FGSM attack – Stop sign images, %**
ϵ=1/255	97.4	88.1
ϵ=3/255	90	71.1
ϵ=5/255	79.6	47.8
ϵ=10/255	70.4	11.5
ϵ=20/255	37.8	0
ϵ=50/255	1.9	0
ϵ=80/255	0.3	0

Вывод

Метод FGSM не подходит для целевых атак, так как с ростом eps и шума, классификация будет ошибочной. Наилучшим значением искажения будет являться 10/255, после этого модель будет ошибаться.

Метод PGD подходит для целевых атак, при больших значениях eps модель всегда будет определять заданный нами класс, однако минусом здесь является то, что изображение будет довольно сильно искажаться. Наилучшее значение искажения для данной атаки является 50/255