

**ББМО-02-22 Филиппов Л.А**

**«Анализ защищенности систем искусственного интеллекта»**

**Практическая работа 4**

## 1. Установка Adversarial Robustness Twoblox(ART) и matplotlib

```
1. Установка Adversarial Robustness Twoblox (ART) и matplotlib

!pip3 install adversarial-robustness-toolbox matplotlib

Requirement already satisfied: adversarial-robustness-toolbox in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (1.17.0)
Collecting matplotlib
  Downloading matplotlib-3.8.2-cp311-cp311-macosx_11_0_arm64.whl (7.5 MB)
    7.5/7.5 MB 156.2 KB/s eta 0:00:00
Requirement already satisfied: numpy>=1.18.0 in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (from adversarial-robustness-toolbox)
Requirement already satisfied: scipy>=1.4.1 in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (from adversarial-robustness-toolbox)
Requirement already satisfied: scikit-learn<1.2.0,>=0.22.2 in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (from adversarial-robustness-toolbox)
Requirement already satisfied: six in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (from adversarial-robustness-toolbox) (1.16.0)
Requirement already satisfied: setuptools in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (from adversarial-robustness-toolbox) (4.6.6)
Requirement already satisfied: tqdm in /Library/Frameworks/Python.framework/Versions/3.11/lib/python3.11/site-packages (from adversarial-robustness-toolbox) (4.66.0)
Collecting contourpy>=1.0.1 (from matplotlib)
  Downloading contourpy-1.2.0-cp311-cp311-macosx_11_0_arm64.whl (243 kB)
    243.4/243.4 kB 390.2 KB/s eta 0:00:00
Collecting cycler>=0.10 (from matplotlib)
  Downloading cycler-0.12.1-py3-none-any.whl (8.3 kB)
Collecting fonttools>=4.22.0 (from matplotlib)
  Downloading fonttools-4.47.2-cp311-cp311-macosx_10_9_universal2.whl (2.8 MB)
    2.8/2.8 MB 136.6 KB/s eta 0:00:00
Collecting kiwisolver>=1.3.1 (from matplotlib)
  Downloading kiwisolver-1.4.5-cp311-cp311-macosx_11_0_arm64.whl (66 kB)
    66.2/66.2 kB 88.5 KB/s eta 0:00:00
Requirement already satisfied: packaging>=20.0 in /Users/l.filippov/Library/Python/3.11/lib/python/site-packages (from matplotlib) (23.1)
Collecting pillow>=8 (from matplotlib)
  Downloading pillow-10.2.0-cp311-cp311-macosx_11_0_arm64.whl (3.3 MB)
    3.3/3.3 MB 131.4 KB/s eta 0:00:00
...
Successfully installed contourpy-1.2.0 cycler-0.12.1 fonttools-4.47.2 kiwisolver-1.4.5 matplotlib-3.8.2 pillow-10.2.0 pyparsing-3.1.1

[notice] A new release of pip is available: 23.1.2 -> 23.3.2
[notice] To update, run: pip3 install --upgrade pip
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

## 2. Импорт необходимых библиотек

```
from __future__ import absolute_import, division, print_function, unicode_literals

import os, sys
from os.path import abspath

module_path = os.path.abspath(os.path.join('.', ''))
if module_path not in sys.path:
    sys.path.append(module_path)

import warnings
warnings.filterwarnings('ignore')

import tensorflow as tf
tf.compat.v1.disable_eager_execution()
tf.get_logger().setLevel('ERROR')

import tensorflow.keras.backend as k
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Flatten, Conv2D, MaxPooling2D, Activation, Dropout
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

from art.estimators.classification import KerasClassifier
from art.attacks.poisoning import PoisoningAttackBackdoor, PoisoningAttackCleanLabelBackdoor
from art.attacks.poisoning.perturbations import add_pattern_bd
from art.utils import load_mnist, preprocess, to_categorical
from art.defences.trainer import AdversarialTrainerMadryPGD
```

## 3. Загрузка датасета MNIST

```
3. Загрузка датасета MNIST

# Набор данных MNIST
(x_raw, y_raw), (x_raw_test, y_raw_test), min_, max_ = load_mnist(raw=True)

# Случайная выборка
n_train = np.shape(x_raw)[0]
num_selection = 10000
random_selection_indices = np.random.choice(n_train, num_selection)
x_raw = x_raw[random_selection_indices]
y_raw = y_raw[random_selection_indices]
```

## 4. Предобработка данных

### 4. Предобработка данных

markdown

```
# Отравление обучающей выборки
percent_poison = .33
x_train, y_train = preprocess(x_raw, y_raw)
x_train = np.expand_dims(x_train, axis=3)

x_test, y_test = preprocess(x_raw_test, y_raw_test)
x_test = np.expand_dims(x_test, axis=3)

# Перемешивание данных
n_train = np.shape(y_train)[0]
shuffled_indices = np.arange(n_train)
np.random.shuffle(shuffled_indices)
x_train = x_train[shuffled_indices]
y_train = y_train[shuffled_indices]
```

[7] ✓ 0.0s

+ Code + Markdown

Python

## 5. Функция create\_model() для создания последовательности из 9 слоев

### 5. Функция create\_model() для создания последовательной модели из 9 слоев

markdown

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Flatten, Conv2D, MaxPooling2D, Dropout

def create_model():
    model = Sequential() # Установка архитектуры модели
    model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1))) # 1 свёрточный слой
    model.add(Conv2D(64, (3, 3), activation='relu')) # 2 свёрточный слой
    model.add(MaxPooling2D(pool_size=(2, 2))) # Слой пулинга
    model.add(Dropout(0.25)) # 1 дропаут слой
    model.add(Flatten()) # Слой выравнивания
    model.add(Dense(128, activation='relu')) # 1 полносвязный слой
    model.add(Dropout(0.25)) # 2 дропаут слой
    model.add(Dense(10, activation='softmax')) # 2 полносвязный слой
    # Компиляция
    model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
    # Возврат модели
    return model
```

[8] ✓ 0.0s

+ Code + Markdown

Python

## 6. Создание атаки

### 6. Создание атаки

markdown

```
backdoor = PoisoningAttackBackdoor(add_pattern_bd)
example_target = np.array([0, 0, 0, 0, 0, 0, 0, 0, 1])
pdata, plabels = backdoor.poison(x_test, y=example_target)

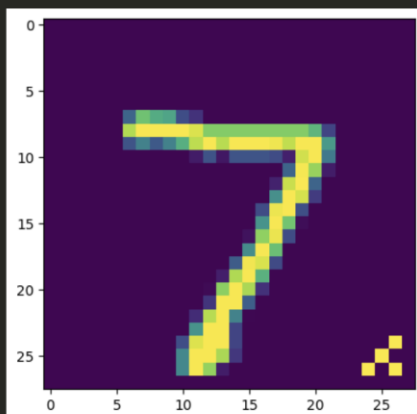
plt.imshow(pdata[0].squeeze())
```

[13] ✓ 0.0s

+ Code + Markdown

Python

<matplotlib.image.AxesImage at 0x29ea5ccd0>



📄 📊 📷

## 7. Определение целевого класса атаки

```
7. Определение целевого класса атаки
```

```
targets = to_categorical([9], 10)[0]
```

[14] ✓ 0.0s Python

## 8. Создание модели

```
8. Создание модели
```

```
model = KerasClassifier(create_model())
proxy = AdversarialTrainerMadryPGD(KerasClassifier(create_model()), nb_epochs=10, eps=0.15, eps_step=0.001)
proxy.fit(x_train, y_train)
```

[15] ✓ 4m 28.4s Python

```
... Precompute adv samples: 100%|██████████| 1/1 [00:00<00:00, 8542.37it/s]
Adversarial training epochs: 0%|██████████| 0/10 [00:00<?, ?it/s]2024-01-28 20:04:52.914723: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:388] MLIF
2024-01-28 20:04:52.948450: W tensorflow/c/c_api.cc:305] Operation '{name:'total_1/Assign' id:407 op device:{requested: '', assigned: ''} def:{{{node total_1/Assign
2024-01-28 20:04:53.344668: W tensorflow/c/c_api.cc:305] Operation '{name:'dense_3/Softmax' id:402 op device:{requested: '', assigned: ''} def:{{{node dense_3/Soft
2024-01-28 20:04:53.481095: W tensorflow/c/c_api.cc:305] Operation '{name:'loss_1/mul' id:478 op device:{requested: '', assigned: ''} def:{{{node loss_1/mul}} = Mu
2024-01-28 20:04:53.515022: W tensorflow/c/c_api.cc:305] Operation '{name:'training/Adam/dense_2/bias/v/Assign' id:770 op device:{requested: '', assigned: ''} def:
Adversarial training epochs: 100%|██████████| 10/10 [04:28<00:00, 26.81s/it]
```

## 9. Исполнение атаки

```
9. Исполнение атаки
```

```
attack = PoisoningAttackCleanLabelBackdoor(backdoor=backdoor,
                                           proxy_classifier=proxy.get_classifier(),
                                           target=targets,
                                           pp_poison=percent_poison,
                                           norm=2,
                                           eps=5,
                                           eps_step=0.1,
                                           max_iter=200)
pdata, plabels = attack.poison(x_train, y_train)
```

[16] ✓ 19.6s Python

```
... PGD - Random Initializations: 100%|██████████| 1/1 [00:01<00:00, 1.93s/it]
PGD - Random Initializations: 100%|██████████| 1/1 [00:01<00:00, 1.85s/it]
PGD - Random Initializations: 100%|██████████| 1/1 [00:01<00:00, 1.90s/it]
PGD - Random Initializations: 100%|██████████| 1/1 [00:01<00:00, 1.78s/it]
PGD - Random Initializations: 100%|██████████| 1/1 [00:01<00:00, 1.69s/it]
PGD - Random Initializations: 100%|██████████| 1/1 [00:01<00:00, 1.84s/it]
PGD - Random Initializations: 100%|██████████| 1/1 [00:01<00:00, 1.89s/it]
PGD - Random Initializations: 100%|██████████| 1/1 [00:01<00:00, 1.91s/it]
PGD - Random Initializations: 100%|██████████| 1/1 [00:01<00:00, 1.83s/it]
PGD - Random Initializations: 100%|██████████| 1/1 [00:01<00:00, 1.82s/it]
PGD - Random Initializations: 100%|██████████| 1/1 [00:00<00:00, 1.00it/s]
```

## 10. Создание отравленных примеров данных

### 10. Создание отравленных примеров данных

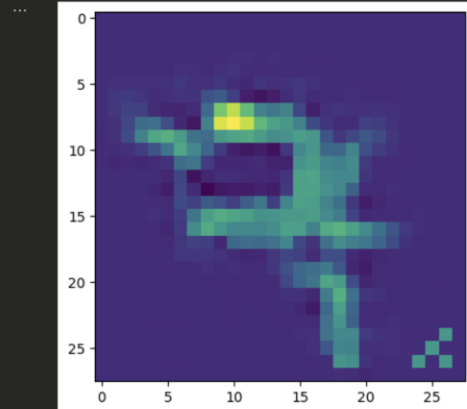
markdown

```
poisoned = pdata[np.all(labels == targets, axis=1)]
poisoned_labels = plabels[np.all(labels == targets, axis=1)]
print(len(poisoned))
idx = 0
plt.imshow(poisoned[idx].squeeze())
print(f"Label: {np.argmax(poisoned_labels[idx])}")
```

[17] ✓ 0.1s

Python

... 1007  
Label: 9



## 11. Обучение модели на отравленных данных

### 11. Обучение модели на отравленных данных

markdown

```
model.fit(pdata, plabels, nb_epochs=10)
```

[18] ✓ 40.6s

Python

... 2024-01-28 20:17:35.870551: W tensorflow/c/c\_api.cc:305] Operation '{name:'loss/mul' id:197 op device:{requested: '', assigned: ''} def:{{{node loss/mul}}} = Mul[T=2024-01-28 20:17:35.910139: W tensorflow/c/c\_api.cc:305] Operation '{name:'training\_2/Adam/dense/bias/m/Assign' id:993 op device:{requested: '', assigned: ''} def:

## 12. Тестирование на чистой модели

### 12. Тестирование на чистой модели

markdown

```
clean_preds = np.argmax(model.predict(x_test), axis=1)
clean_correct = np.sum(clean_preds == np.argmax(y_test, axis=1))
clean_total = y_test.shape[0]
clean_acc = clean_correct / clean_total

print("\nТочность чистого тестового набора: %.2f%%" % (clean_acc * 100))

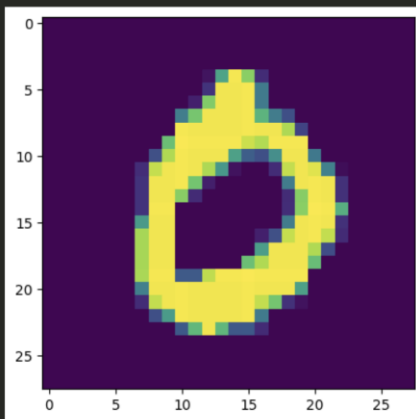
# Отображение картинки, класса, и предсказания для чистого примера
# чтобы показать как отравленная модель классифицирует чистый пример
c = 0 # Класс
i = 0 # Изображение
c_idx = np.where(np.argmax(y_test, 1) == c)[0][i] # индекс изображения в массиве чистых примеров
plt.imshow(x_test[c_idx].squeeze())
plt.show()
clean_label = c
print("Прогноз: " + str(clean_preds[c_idx]))
```

[19] ✓ 1.1s

Python

... 2024-01-28 20:19:06.709561: W tensorflow/c/c\_api.cc:305] Operation '(name:'dense\_1/Softmax' id:121 op device:{requested: '', assigned: ''} def:{{{node dense\_1/Soft

Точность чистого тестового набора: 98.05%



... Прогноз: 0

## 13.Получение результатов атаки на модель

13. Получение результатов атаки на модель

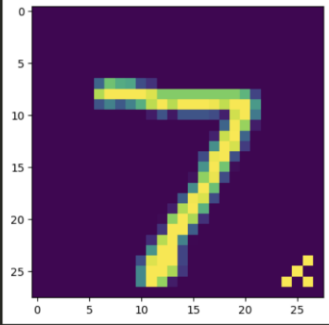
markdown

not\_target = np.logical\_not(np.all(y\_test == targets, axis=1))  
px\_test, py\_test = backdoor.poisson(x\_test[not\_target], y\_test[not\_target])  
poison\_preds = np.argmax(model.predict(px\_test), axis=1)  
poison\_correct = np.sum(poison\_preds == np.argmax(y\_test[not\_target], axis=1))  
poison\_total = poison\_preds.shape[0]  
poison\_acc = poison\_correct / poison\_total  
  
print("\nТочность отравленного набора: %.2f%%" % (poison\_acc \* 100))  
  
c = 0 # индекс для отображения  
plt.imshow(px\_test[c].squeeze())  
plt.show()  
clean\_label = c  
print("Прогноз: " + str(poison\_preds[c]))

[38] ✓ 1.0s

Python

Точность отравленного набора: 3.16%



Прогноз: 9