ИНТЕЛЛЕКТУАЛЬНЫЕ МЕТОДЫ В ЗАДАЧАХ ЗАЩИТЫ ОТ КИБЕРУГРОЗ

ЛАБОРАТОРНАЯ РАБОТА № 5
Сверточные нейронные сети.

Выполнил:
Мосолков Е.Н.
Преподаватель:
Петров А.А.

Москва 2021 г.

ЦЕЛЬ РАБОТЫ

Сформировать CNN с опорой на некоторые моменты из глубокой нейронной сети ЛР№4. Оценить и сравнить точность относительно НС из ЛР№4. Проварьировать параметры (число слоёв, фильтров, кол-во эпох, размер партии и т.д.).

ОПИСАНИЕ ЗАДАЧИ ПРАКТИЧЕСКОЙ РАБОТЫ

1. Сформировать CNN с опорой на некоторые моменты из глубокой нейронной сети ЛР№4. Оценить и сравнить точность относительно НС из ЛР№4. Поварьировать параметры (число слоёв, фильтров, кол-во эпох, размер партии и т. д.).

2. Сделать фото какого-то предмета/объекта с соотношением сторон 1:1 и импортировать в нашу нейросеть. Запустить нашу нейросеть и идентифицировать объект.

КОД

```
import numpy as np
import matplotlib.pyplot as plt
from tensorflow import keras
from io import BytesIO
from PIL import Image
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.datasets import mnist # библиотека базы выборок Mnis
t
from google.colab import files
from tensorflow.keras.layers import Dense, Flatten, Dropout, Conv2D, MaxPo
oling2D

model = keras.Sequential([
    Conv2D(32, (3,3), padding='same', activation='relu', input_shape=(28,
28, 1)),
    MaxPooling2D((2, 2), strides=2),
    Conv2D(64, (3,3), padding='same', activation='relu'),
    MaxPooling2D((2, 2), strides=2),
    Flatten(),
    Dense(128, activation='relu'),
    Dense(10, activation='softmax')
])

(x_train, y_train), (x_test, y_test) = mnist.load_data()
# стандартизация входных данных
x_train = x_train / 255
x_test = x_test / 255
y_train_cat = keras.utils.to_categorical(y_train, 10)
y_test_cat = keras.utils.to_categorical(y_test, 10)

x_train = np.expand_dims(x_train, axis=3)
x_test = np.expand_dims(x_test, axis=3)
print( x_train.shape )

model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
his = model.fit(x_train, y_train_cat, batch_size=32, epochs=5,
validation_split=0.2)

x_train = np.expand_dims(x_train, axis=3)
x_test = np.expand_dims(x_test, axis=3)
print( x_train.shape )

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

```
his = model.fit(x_train, y_train_cat, batch_size=32, epochs=5,
validation_split=0.2)

model.evaluate(x_test, y_test_cat)

model = keras.applications.VGG16()
uploaded = files.upload()
img = Image.open(BytesIO(uploaded['ex224.jpg']))
plt.imshow( img )
# приводим к входному формату VGG-сети
img = np.array(img)
x = keras.applications.vgg16.preprocess_input(img)
print(x.shape)
x = np.expand_dims(x, axis=0)
# прогоняем через сеть
res = model.predict( x )
print(np.argmax(res))
```

ТЕСТИРОВАНИЕ НЕЙРОСЕТИ

Часть 1:

Без модификаций НС показывает хорошие результаты: на 10 эпохе значение точности близко к 1 (0.9857000112533569), потеря близка к 0 (0.04251173138618469), однако значения заметно отличаются от моих результатов НС в ЛР №4, а именно точность 0.9959, потеря: 0.0148

Часть 2:

Загрузил квадратную фотографию автомобиля со стороной 224 пикселя, в результате классификации получил, что 436-й нейрон из 1000 принял максимальное значение. Расшифровка: `'beach wagon, station wagon, wagon, estate car, beach waggon, station waggon, waggon'`.

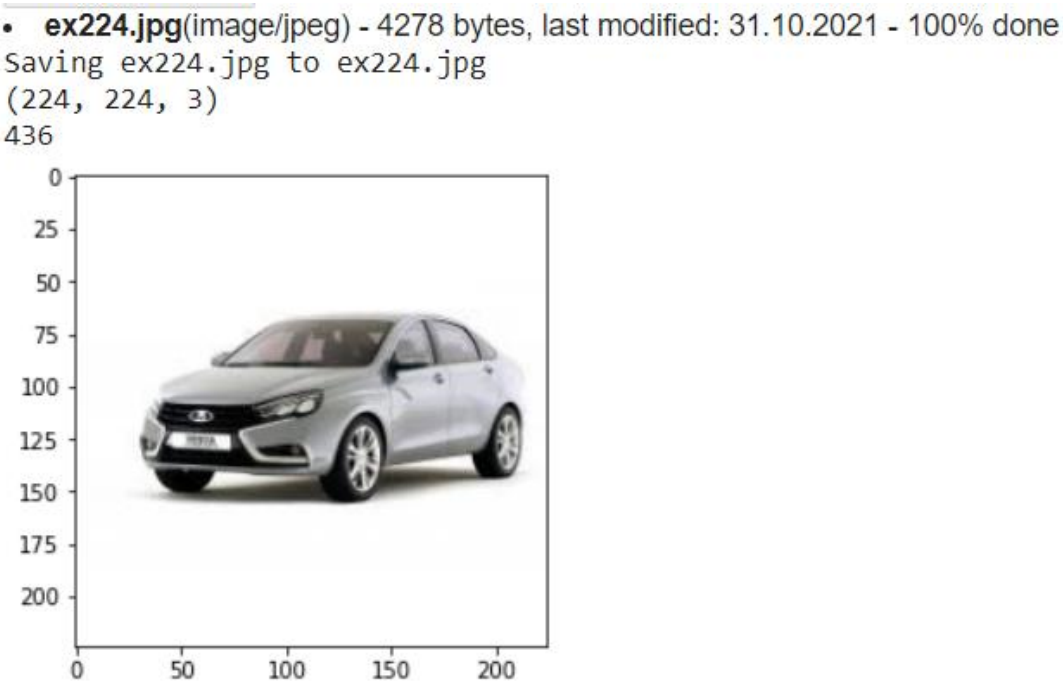

Рисунок 1

ВЫВОД

В первой части выполнения лабораторной работы я поработал со сверточной НС на основе библиотеки keras для распознавания рукописных цифр из датасета MNIST. Во второй части ЛР я поработал с нейросетью VGG-16 на основе той же библиотеки, с помощью нее классифицировал объект на изображении, загруженной мной.