

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ»



Инженерная школа информационных технологий и робототехники

Отделение информационных технологий

09.04.01 «Информатика и вычислительная техника»

Отчет по практической работе №4
«Распознавание изображений с использованием архитектуры ResNet»

МАШИННОЕ ОБУЧЕНИЕ

Исполнитель:

студент группы

8BM22

Ямкин Н.Н.

11.09.2023

Руководитель:

Доцент (ОИТ, ИШИТР)

Друки А.А.

Содержание

1. Цель работы	3
2. Задачи	3
3. Ход работы.....	3
4. Заключение	7

1. Цель работы

Получить навыки распознавания изображений с помощью модели нейронной сети ResNet.

2. Задачи

1. Ознакомиться с работой сверточных нейронных сетей в библиотеке Keras;
2. Научиться применять модель сверточной нейронной сети ResNet для распознавания изображений.

3. Ход работы

Подключаем необходимые библиотеки:

```
import numpy as np
import matplotlib.pyplot as plt
import keras as N1
import keras
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense, Flatten
from tensorflow import keras
from keras.optimizers import Adam
from tensorflow.keras.datasets import cifar10
```

Загружаем исходный набор данных:

```
(x_train, y_train), (x_test, y_test) = cifar10.load_data()
```

Отрисуем первые 16 изображений из датасета:

```
plt.figure(figsize=(6,6))
for i in range(16):
    plt.subplot(4,4,i+1)
    plt.xticks([])
    plt.yticks([])
    plt.imshow(x_train[i], cmap=plt.cm.binary)
    plt.xlabel(y_train[i])
```

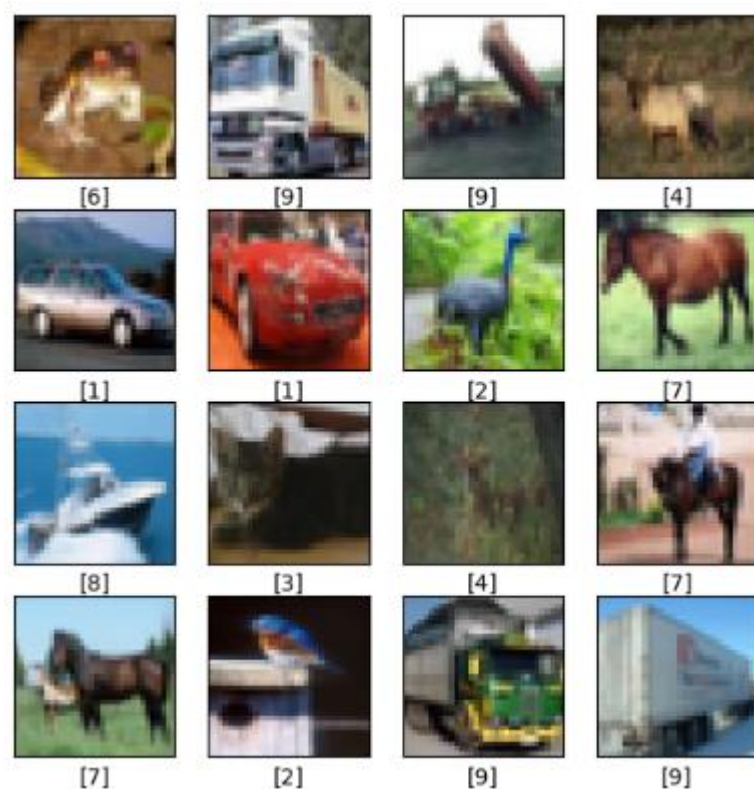


Рисунок 1 – Изображения из датасета CIFAR-10

Нормализуем изображения из обучающей и тестовой выборки:

```
# Нормализация данных
x_train = x_train / 255
x_test = x_test / 255
```

Подготовим желаемые выходные данные. Разрабатываемая нейронная сеть будет иметь 10 выходов, согласно количеству классов изображений. Таким образом число, соответствующее изображению, нужно привести к вектору, состоящему из 10 элементов. В данном векторе все элементы, кроме одного, соответствующего числу на изображении, должны быть равны нулю. Элемент, соответствующий числу на изображении, должен быть равен единице.

```
num_classes = 10

# преобразование выходных целевых данных нейронной сети к категориальному
виду
# y_train - вектор из перекодированных леблов
# число 5 будет выглядеть вот так [0,0,0,0,0,1,0,0,0,0] и т.д.
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)
```

Создаём нейронную сеть заданной архитектуры (рисунок 2).

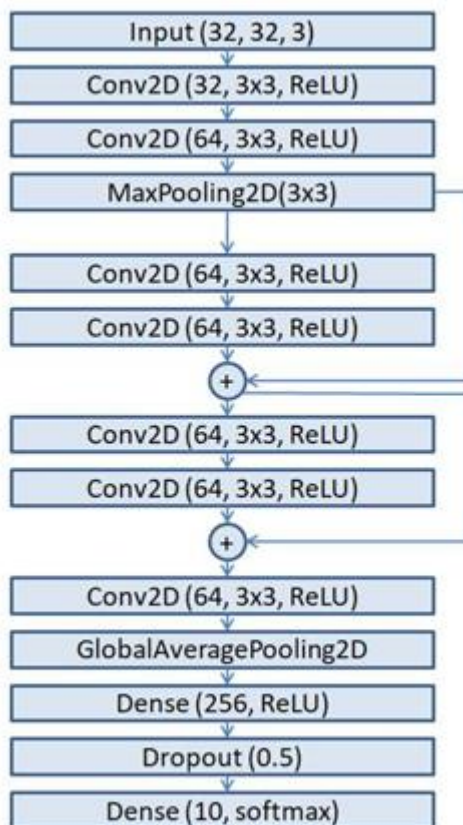


Рисунок 2 – Архитектура нейронной сети ResNet

```

input_shape = (32, 32, 3)
inputs = keras.Input(shape=input_shape, name='img')
x = layers.Conv2D(32, kernel_size=(3, 3), activation="relu")(inputs)
x = layers.Conv2D(64, kernel_size=(3, 3), activation="relu")(x)
block_1_output = layers.MaxPooling2D(pool_size=(3, 3))(x)

# Residual Block 1
x = layers.Conv2D(64, kernel_size = (3, 3), activation="relu", padding =
"same")(block_1_output)
x = layers.Conv2D(64, kernel_size = (3, 3), activation="relu", padding =
"same")(x)
block_2_output = layers.add([x, block_1_output])

# Residual Block 2
x = layers.Conv2D(64, kernel_size=(3, 3), activation="relu",
padding="same")(block_2_output)
x = layers.Conv2D(64, kernel_size=(3, 3), activation="relu",
padding="same")(x)
block_3_output = layers.add([x, block_2_output])

x = layers.Conv2D(64, kernel_size=(3, 3), activation="relu",
padding="same")(block_3_output)
x = layers.GlobalAveragePooling2D()(x)
x = layers.Dense(256, activation="relu")(x)
x = layers.Dropout(0.5)(x) # Дропаут
outputs = layers.Dense(num_classes, activation="softmax")(x)

model = keras.Model(inputs=inputs, outputs=outputs)
model.summary()

```

Обучаем нейронную сеть.

```

batch_size = 100
epochs = 10
model.compile(loss="MSE", optimizer="adam", metrics=["accuracy"])
model.fit(x_train, y_train, batch_size=batch_size, epochs=epochs,
validation_split=0.1)

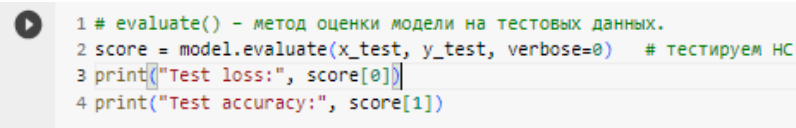
```

Выполним оценку качества обучения.

```

score = model.evaluate(x_test, y_test, verbose=0) # тестируем НС
print("Test loss:", score[0])
print("Test accuracy:", score[1])

```



```

1 # evaluate() - метод оценки модели на тестовых данных.
2 score = model.evaluate(x_test, y_test, verbose=0) # тестируем НС
3 print("Test loss:", score[0])
4 print("Test accuracy:", score[1])

```

Test loss: 0.03232434764504433
Test accuracy: 0.7750999927520752

Рисунок 3 – Оценка качества обучения нейросети ResNet

Проверим работу нейронной сети на тестовой выборке.

```
y_pred = model.predict(x_test)
print(y_pred.shape)
```

Выведем результат:

```
I = 7 # номер изображения для вывода
plt.imshow(x_test[I].reshape([32, 32, 3]), cmap='gray') # отрисовка
изображения
print("Мнение нейронной сети: ",
      np.argmax(model.predict(x_test[I].reshape([1, 32, 32, 3])))) # отрисовка
изображения
print("Верный ответ: ", np.argmax(y_test[I]))
```



Рисунок 4 – Результат работы ResNet

4. Заключение

В результате выполнения данной лабораторной работы получил навыки распознавания объектов на изображениях из выборки CIFAR-10 с использованием библиотеки машинного обучения Keras. Также научился реализовывать топологию сверточной нейронной сети ResNet, обучать её и тестировать.