Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**ЛАБОРАТОРНАЯ РАБОТА №6**

**Дисциплина: Алгоритмы цифровой обработки мультимедиа**

Работу выполнил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Д.Е.Гиренко

Направление подготовки: 02.03.02 Фундаментальная информатика и информационные технологии

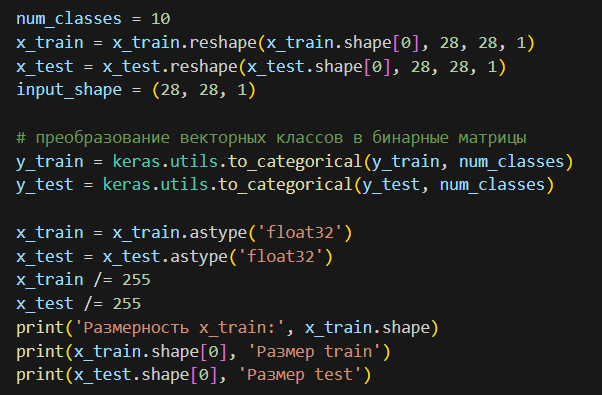
Преподаватель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.А. Крамаренко

**Цель работы:** Решение задач классификации при помощи нейронных сетей.

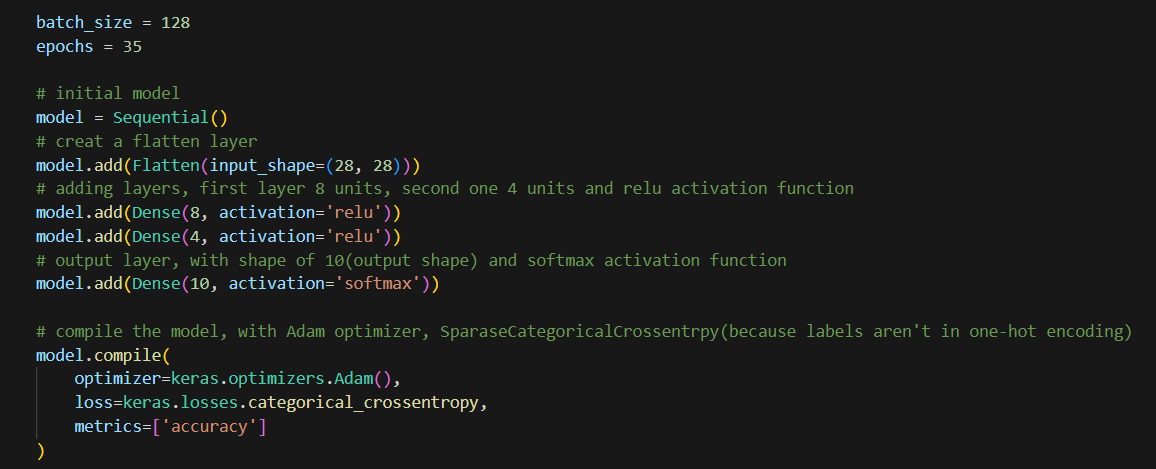
**Ход работы:**

Задание 1. Построить многослойный персептрон

С помощью библиотеки keros вначале был скачан и обработан датасет рукописных цифр mnist:



После чего создана модель простейшего персептрона, для которого было проведено обучение и последующая проверка результатов на тестовых данных:



Задание 2. Исследовать нейронную сеть.

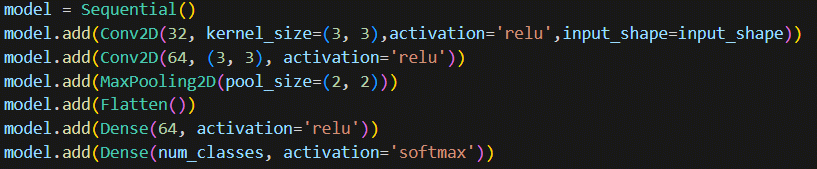
Необходимо было подобрать оптимальное количество эпох обучения для работы на данной тестовой базе, после чего сравнить работу алгоритма по нескольким параметрам. Результаты приведены в таблице:

|  |  |  |  |
| --- | --- | --- | --- |
| Количество эпох | Время обучения с. | Время работы с. | Точность |
| 5 | 3 | 0.2 | 0.87 |
| 15 | 8 | 0.2 | 0.90 |
| 35 | 17 | 0.2 | 0.91 |

По приведённым параметрам видно, что время обучения линейно зависит от количества эпох, время работы не зависит, а точность улучшается с количеством эпох, но с убывающей полезностью.

Задание 3. Построить свёрточную нейронную сеть

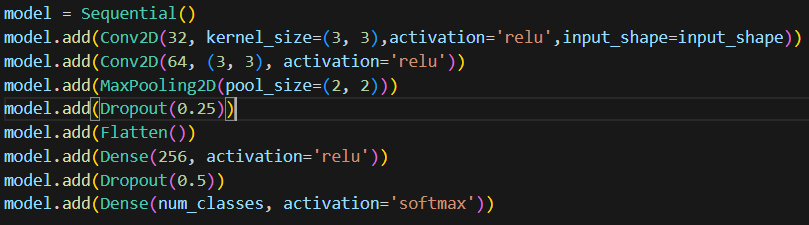
Для решения аналогичной задачи было решено построить свёрточную сеть и посмотреть результаты её работы при 10 эпохах при различных архитектурах. По результатам тестов были получены такие результаты:

1. Архитектура: 

Точность: 0.86

Время обучения: 6 мин. 8 с.

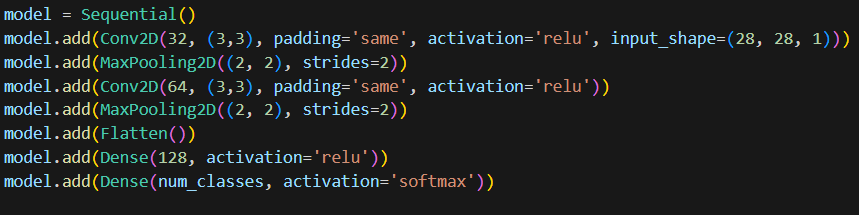
Время работы: 1.4 с.

1. Архитектура: 

Точность: 0.84

Время обучения: 13 мин.

Время работы: 3.4 с

1. Архитектура:

Точность: 0.75

Время обучения: 5 мин. 15 с.

Время работы: 1.1 c.

По результатам испытаний можно заметить, что время работы и обучения на прямую зависят от сложности модели, однако точность зависит именно от правильности построения архитектуры.

**Листинг программы**

import keras

from keras.datasets import mnist

from keras.models import Sequential

from keras.layers import Dense, Dropout, Flatten

from keras.layers import Conv2D, MaxPooling2D

from keras import backend as K

# скачиваем данные и разделяем на надор для обучения и тесовый

(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()

print(x\_train.shape, y\_train.shape)

num\_classes = 10

x\_train = x\_train.reshape(x\_train.shape[0], 28, 28, 1)

x\_test = x\_test.reshape(x\_test.shape[0], 28, 28, 1)

input\_shape = (28, 28, 1)

# преобразование векторных классов в бинарные матрицы

y\_train = keras.utils.to\_categorical(y\_train, num\_classes)

y\_test = keras.utils.to\_categorical(y\_test, num\_classes)

x\_train = x\_train.astype('float32')

x\_test = x\_test.astype('float32')

x\_train /= 255

x\_test /= 255

print('Размерность x\_train:', x\_train.shape)

print(x\_train.shape[0], 'Размер train')

print(x\_test.shape[0], 'Размер test')

batch\_size = 128

epochs = 10

model = Sequential()

model.add(Conv2D(32, kernel\_size=(3, 3),activation='relu',input\_shape=input\_shape))

model.add(Conv2D(64, (3, 3), activation='relu'))

model.add(MaxPooling2D(pool\_size=(2, 2)))

model.add(Flatten())

model.add(Dense(64, activation='relu'))

model.add(Dense(num\_classes, activation='softmax'))

model.compile(loss=keras.losses.categorical\_crossentropy,optimizer=keras.optimizers.Adadelta(),metrics=['accuracy'])

hist = model.fit(x\_train, y\_train, batch\_size = batch\_size, epochs=epochs, verbose=1, validation\_data=(x\_test, y\_test))

print("Модель успешно обучена")

model.save('svertka'+str(epochs)+ '.keras')

print("Модель сохранена как svertka.keras")

score = model.evaluate(x\_test, y\_test, verbose=0)

print('Потери на тесте:', score[0])

print('Точность на тесте:', score[1])