МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Кафедра «ПСАТП»

Практическая работа № 5

по дисциплине: «АОБМИ»

По теме: «Разработка программы автоматического распознавания рукописных символов»

Вариант - 5

Выполнил: ст. гр. БС/б-19-1-о

Дзюба И. А.

Проверил: доцент

Осипов К. Н.

Севастополь

2022

**Цель работы**: Разработать простейший интерфейс обработки информации, исходя из тематики курсовой работы.

**Ход работы:**

Код программы на языке Python:

## Подключим необходимые библиотеки

import numpy as np

import cv2

from keras.datasets import mnist

from keras.models import Sequential

from keras.layers.core import Dense, Dropout, Activation

from tensorflow.keras.optimizers import SGD, Adam, RMSprop

from keras.utils import np\_utils;

import pandas as pd;

import matplotlib.pyplot as plt;

batch\_size = 128; #размер партии данных, которая поступает на обучение, т.е.

nr\_classes = 10; # Количество классов

nr\_iterations = 20; #Количество эпох (количество итераций)

#Загрузка датасета

with np.load('mnist.npz') as data:

x\_train, y\_train, x\_test, y\_test = data['x\_train'], data['y\_train'], data['x\_test'], data['y\_test'];

x\_train = x\_train.reshape(60000, 784); # 60000 - это количество картинок, 784 - размер 1-й картинки (28х28=784)

x\_test = x\_test.reshape(10000, 784);

# Уточняем тип данных

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

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

# Нормируем входные значения, чтобы получить числа от 0 до 1!!!

x\_train /= 255

x\_test /= 255

# Делаем 10 бинарных столбцов (так как 10 цифр), формируем зависимые переменные, ответы.

y\_train = np\_utils.to\_categorical(y\_train, nr\_classes)

y\_test = np\_utils.to\_categorical(y\_test, nr\_classes)

a=pd.DataFrame(y\_train)# Это не обязательный параметр.

a.head() # Посмотрим, что содержится в данных Y\_train

# Описываем сеть. Входной слой, один внутренний слой, выходной слой. Всего три слоя!!!

model = Sequential()# Начало создания сети.

model.add(Dense(196, input\_shape=(784,))) # входной (784 = 28х28 нейрона) и второй слой (196 нейронов).

model.add(Activation('relu'));

model.add(Dropout(0.5));

model.add(Dense(10))# количество нейронов в выходном слое;

model.add(Activation('softmax'));

model.summary() #Не обязательный параметр.

# Параметры целевой функции потерь.

model.compile(loss='categorical\_crossentropy',

optimizer=Adam(),

metrics=['accuracy'])

np.random.seed(1337) # для воспроизводимости сети

# Обучаем сеть.

net\_res\_1 = model.fit(x\_train, y\_train,epochs=nr\_iterations,

batch\_size = batch\_size,

verbose = 1, validation\_data = (x\_test, y\_test));

score = model.evaluate(x\_test, y\_test, verbose = 0)# Вычислим ошибки обучения.

print(score)

# Визуализация работы программы.

plt.imshow(x\_test[9].reshape(28,28)) # Посмотрим на картинку №28.

df=x\_test[9].reshape(1,784); #Растянем картинку в вектор.

ans=model.predict(df); # Воспользуемся моделью для решения задачи классификации.

ans=pd.DataFrame(ans); # Представим ответ в формате DataFrame для удобства

round(ans,2) # Посмотрим на результат.

# Визуализация работы программы.

img=cv2.imread("SYS.png",0) #Загружаем изображение.

newimg=cv2.resize(img, (28,28)) #Меняем размер изображения.

dZ=newimg.reshape(-1,784); #Растянем картинку в вектор.

ans=model.predict(dZ); # Воспользуемся моделью для решения задачи классификации.

ans=pd.DataFrame(ans); # Представим ответ в формате DataFrame для удобства просмотра.

round(ans,2) # Посмотримd на результат.

plt.imshow(newimg) #Смотрим на картинку.

Вывод: научились использовать простейшие нейросети.