Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования «Пермский национальный исследовательский

политехнический университет» (ПНИПУ)

Кафедра вычислительной математики и механики

**Лабораторная работа № 5**

**по дисциплине: «Интеллектуальные ИСИТ»**

Выполнила

студент группы ИСТ-19-1б

Лопина М.С.

Проверил

ассистент кафедры ВММБ

Нетбай Г.В.

Пермь, 2022

**Цель**: сформировать у студентов способность построения нейросетевой интеллектуальной системы для решения задач распознавания образов.

**Описание**: в рамках данной работы необходимо реализовать приложение, решающее задачу искусственных нейронных сетей.

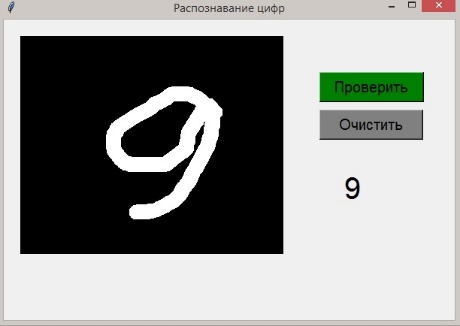
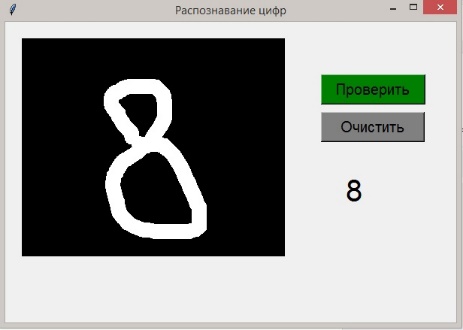
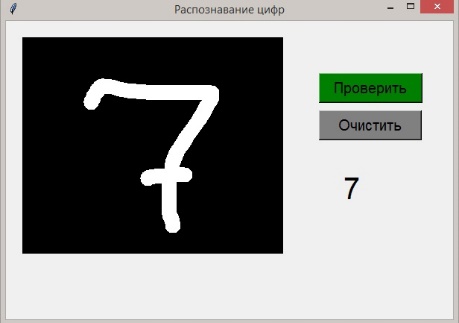
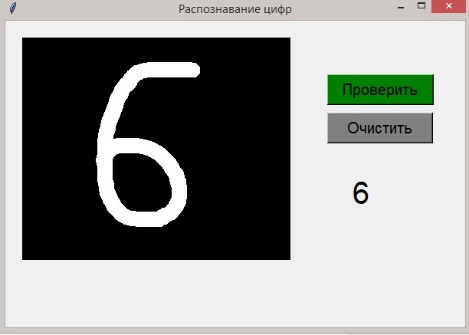
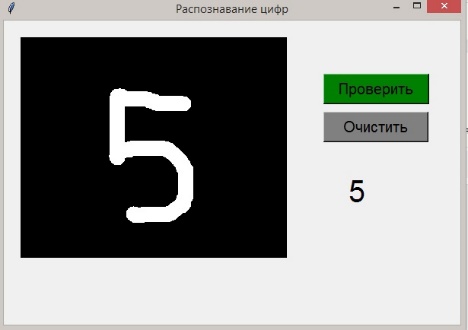
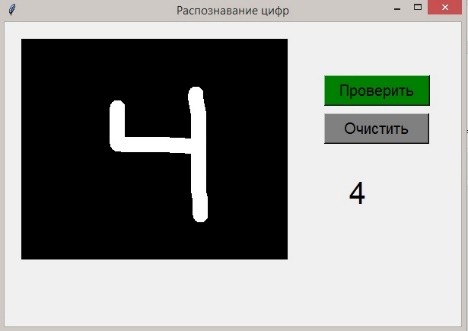
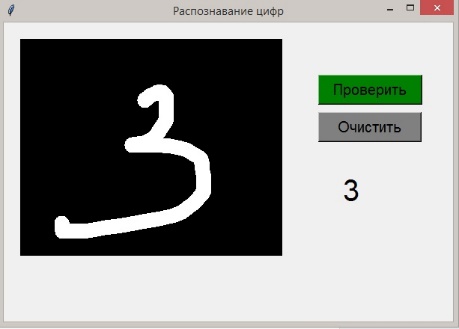
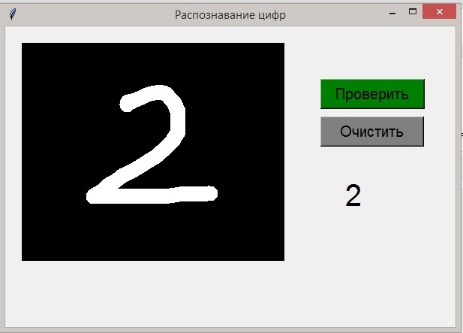
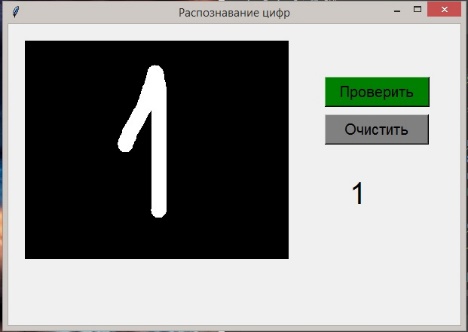
**Результат:**

Предметная область – распознавание цифр.

Для распознавания написанных от руки цифр используем набор данных файла MNIST. Это набор данных применяемый в глубоком обучении, содержит в себе тренировочные изображения написанных от руки цифр.

Программа состоит из двух файлов, реализованных на языке python. Запуск main проводит чтение данных из файла MNIST, создание модели и тренировка модели. Запуск ModelGui запускает графический интерфейс для записи цифр.

Пример работы программы, результат распознавания цифр от 1 до 9:



Распознавание цифр от 1 до 9

Код программы:

**Main**:

import keras

import tensorflow as tf

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

(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data('mnist-original.mat')

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

# модель сети

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(Dropout(0.25))

model.add(Flatten())

model.add(Dense(256, activation='relu'))

model.add(Dropout(0.5))

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

# компиляция модели с помощью оптимизатора Adadelta

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

# Тестирование модели

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

model.save('mnist.h5')

**ModelGui:**

from tkinter import \*  
from keras.models import load\_model  
import numpy as np  
from PIL import ImageGrab  
window = Tk()  
window.title("Распознавание цифр")  
l1 = Label()  
model = load\_model('mnist.h5')  
def predict\_digit(img):  
# изменение рзмера изобржений на 28x28  
img = img.resize((28,28))  
# конвертируем rgb в grayscale  
img = img.convert('L')  
img = np.array(img)  
# изменение размерности для поддержки модели ввода и нормализации  
img = img.reshape(1,28,28,1)  
img = img/255.0  
# предстказание цифры  
res = model.predict([img])[0]  
return np.argmax(res), max(res)  
  
def MyProject():  
global l1  
widget = cv  
# Установка координат холста  
x = window.winfo\_rootx() + widget.winfo\_x()  
y = window.winfo\_rooty() + widget.winfo\_y()  
x1 = x + widget.winfo\_width()  
y1 = y + widget.winfo\_height()  
n=28  
m=n\*n  
# Изображение захватывается с холста и имеет размер (28 X 28) пикселей  
img = ImageGrab.grab().crop((x, y, x1, y1)).resize((n, n))  
digit, acc = predict\_digit(img)  
# Вывод результата на экран  
l1 = Label(window, text=str(digit),justify=CENTER, font=('Arial', 30))  
l1.place(x=450, y=200)  
lastx, lasty = None, None  
# Очистка поля  
def clear\_widget():  
global cv, l1  
cv.delete("all")  
l1.destroy()  
# Проверка введенного числа  
def event\_activation(event):  
global lastx, lasty  
cv.bind('<B1-Motion>', draw\_lines)  
lastx, lasty = event.x, event.y  
# Рисование цифры на поле  
def draw\_lines(event):  
global lastx, lasty  
x, y = event.x, event.y  
cv.create\_line((lastx, lasty, x, y), width=20, fill='white', capstyle=ROUND, smooth=TRUE, splinesteps=12)  
lastx, lasty = x, y  
# Кнопка очистки поля  
b1 = Button(window, text=" Очистить ", bg="grey", fg="black", font=('Arial', 15), command=clear\_widget)  
b1.place(x=420, y=120)  
# Кнопка для предсказания цифры, нарисованной на холсте  
b2 = Button(window, text=" Проверить ", bg="green", fg="black", font=('Arial', 15), command=MyProject)  
b2.place(x=420, y=70)  
# Экран рисования цифры  
cv = Canvas(window, width=350, height=290, bg='black')  
cv.place(x=20, y=20)  
cv.bind('<Button-1>', event\_activation)  
window.geometry("600x400")  
window.mainloop()