**Цель работы**

Реализация алгоритмов распознавания образов с помощью нейронных сетей на языке программирования и изучение их свойств.

**Практическая часть**

1. Изучить устройство и работу однослойной, двухслойной нейронных сетей, сети встречного распространения. 2. Выбрать исходный алфавит классов (цифры, буквы русского алфавита, буквы английского алфавита, буквы греческого алфавита, знаки препинания и арифметические символы, радиоэлементы, геометрические фигуры, узоры, дорожные знаки, топологические элементы, картографические обозначения) по согласованию с преподавателем. 3. Составить программу, реализующую однослойную, двухслойную нейронные сети и сеть встречного распространения для задачи распознавания. Обучить нейронные сети. 4. Программное обеспечение должно позволять просматривать эталоны (классы) изображений, а также распознаваемые изображения; записывать эталоны изображений в библиотеку на диск, записывать входной образ на диск; позволять редактировать входной образ, сохранять веса нейронов при обучении сети. 5. Испытать программное обеспечение для различных входных данных. 6. Произвести оценку качества распознавания для различных случаев, систематизировав полученные результаты в таблицы. Построить графики выявленных зависимостей, сделать выводы. 7. Результаты работы оформить в виде отчета в текстовом редакторе.

Код программы, реализующий алгоритм распознавания образов с помощью нейронной сети, на языке python представлен в приложении.

Приложение — Код на python

from tensorflow.keras.datasets import fashion\_mnist

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Dropout

from tensorflow.keras import utils

from tensorflow.keras.preprocessing import image

import numpy as np

import matplotlib.pyplot as plt

from PIL import Image

# В Keras встроены средства работы с популярными наборами данных

# (x\_train, y\_train) - набор данных для обучения

# (x\_test, y\_test) - набор данных для тестирования

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

#Список с названиями классов

classes = ['футболка', 'брюки', 'свитер', 'платье', 'пальто', 'туфли', 'рубашка', 'кроссовки', 'сумка', 'ботинки']

#Просматриваем примеры изображений

plt.figure(figsize=(10,10))

for i in range(100,150):

plt.subplot(5,10,i-100+1)

plt.xticks([])

plt.yticks([])

plt.grid(False)

plt.imshow(x\_train[i], cmap=plt.cm.binary)

plt.xlabel(classes[y\_train[i]])

#Преобразование размерности данных в наборе

x\_train = x\_train.reshape(60000, 784)

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

# Векторизованные операции

# Применяются к каждому элементу массива отдельно

x\_train = x\_train / 255

x\_test = x\_test / 255

#Работа с правильными ответами

n = 0

print(y\_train[n])

#Преобразуем метки в формат one hot encoding

y\_train = utils.to\_categorical(y\_train, 10)

y\_test = utils.to\_categorical(y\_test, 10)

#Правильный ответ в формате one hot encoding

print(y\_train[n])

#Создаем нейронную сеть

# Создаем последовательную модель

model = Sequential()

# Входной полносвязный слой, 800 нейронов, 784 входа в каждый нейрон

model.add(Dense(800, input\_dim=784, activation="relu"))

# Выходной полносвязный слой, 10 нейронов (по количеству рукописных цифр)

model.add(Dense(10, activation="softmax"))

#Компилируем сеть

model.compile(loss="categorical\_crossentropy", optimizer="SGD", metrics=["accuracy"])

print(model.summary())

#Обучаем нейронную сеть

history = model.fit(x\_train, y\_train,

batch\_size=200,

epochs=40,

validation\_split=0.2,

verbose=1)

#Оценка качества обучения

#Проверка качества работы на наборе данных для тестирования

scores = model.evaluate(x\_test, y\_test, verbose=1)

print("Доля верных ответов на тестовых данных, в процентах:", round(scores[1] \* 100, 4))

#Используем сеть для распознавания предметов одежды

n\_rec = 250

plt.imshow(x\_test[n\_rec].reshape(28, 28), cmap=plt.cm.binary)

plt.show()

#Меняем размерность изображения и нормализуем его

x = x\_test[n\_rec]

x = np.expand\_dims(x, axis=0)

#Запускаем распознавание

prediction = model.predict(x)

#Печатаем результаты распознавания

print(prediction)

#Преобразуем результаты из формата one hot encoding

prediction = np.argmax(prediction[0])

print("Номер класса:", prediction)

print("Название класса:", classes[prediction])

#Печатаем правильный ответ

label = np.argmax(y\_test[n\_rec])

print("Номер класса:", label)

print("Название класса:", classes[label])

**Результат**

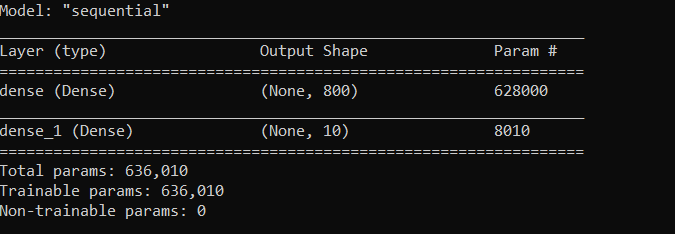
****

Рис.1 Компиляция сети

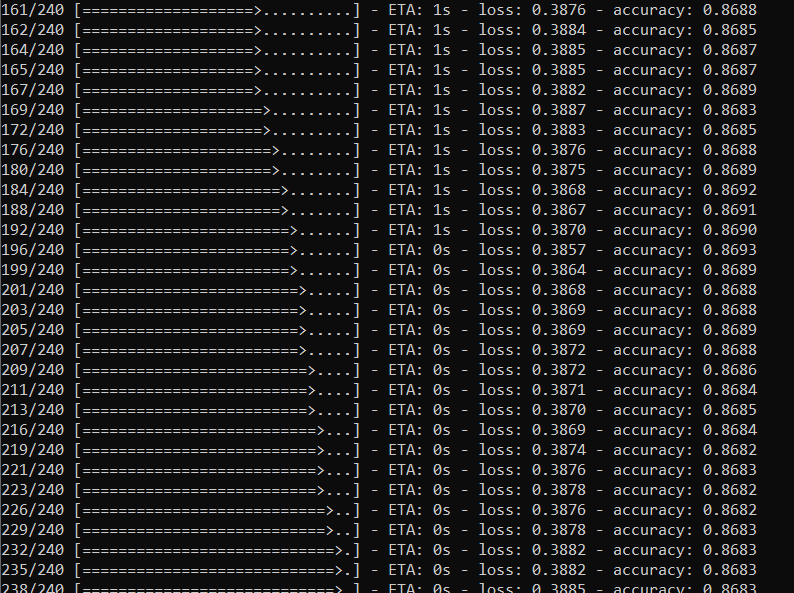


Рис.2 Обучение сети



Рис.3 Доля верных ответов на тестовых данных



Рис.4 Данные на выходе из сети в формате one-hot-encoding



Рис.5 Dataset(сборник фотографий вещей)

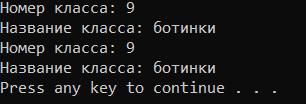


Рис.6 Результат работы

**Заключение**

В результате выполнения лабораторной работы мы изучили способ распознавания образов с помощью нейронных сетей на языке Python программирования и изучение их свойств.