**Санкт-Петербургский политехнический университет Петра Великого**

**Институт кибербезопасности и защиты информации**

ЛАБОРАТОРНАЯ РАБОТА № 3

**Использование Python-библиотек для работы**

**с ИИ**

по дисциплине "Цифровая культура"

Выполнил

студент гр. 4851004/10001 Веремейчук Я. Ю.

<*подпись*>

Помощник преподавателя Данилов В. Д.

<*подпись*>

Санкт-Петербург

2023

**Цель работы**

Получение навыков работы с методами ИИ для решения задачи классификации данных с использованием языка программирования Python.

**Задание на работу**

1. Изучить исходный код выданного скрипта lab3.py. Изучить используемые в скрипте библиотеки. Изучить описание скрипта по ссылке: [https://colab.research.google.com/drive/1nseete5huZlWM7Ak0qL-T75Dbk0mdrZ?usp=sharing#scrollTo=0mLij6qde6Ox].
2. Установить необходимые для работы скрипта модули.
3. Скачать наборы данных для проверки работоспособности скрипта:
4. <https://github.com/btphan95/simplecnn/blob/master/train.zip?raw=true>
5. https://github.com/btphan95/simplecnn/blob/master/valid.zip?raw=true
6. Сформировать датасеты для обучения и тестирования (валидации). Набор данных должен представлять собой набор изображений. Первый набор изображений – люди, второй – животное в зависимости от варианта (вариант соответствует порядковому номеру в списке группы). Для формирования датасетов может быть использован сервис «Hugging Face», либо любые другие открытые источники.
7. Если в обучающей выборке недостаточно изображений – уменьшите значение batch\_size при вызове функции train\_datagen.flow\_from\_directory.
8. Модифицировать скрипт в соответствии с требованиями: − на вход скрипту подаются параметры: путь до тренировочного датасета, путь до датасета для тестирования (валидации), путь до файла, который необходимо классифицировать; − вывод скрипта – результат бинарной классификации (что изображено на картинке: человек или животное).
9. Протестировать работу программы на изображениях:

1) Из обучающего набора данных.

2) Из тестового набора данных.

3) Из произвольного источника.

1. Провести эксперименты по изменению параметров модели классификации для повышения эффективности.
2. Сделать соответствующие выводы о работе скрипта.
3. Ответить на контрольные вопросы

**Ход работы**

Перед началом работы был изучен код выданного скрипта и скачены необходимые библиотеки. Также были скачены датасеты для тестирования исходного кода.

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Рисунок 1 – Тестирование исходного кода

Изображение выглядит как снимок экрана, растение, текст, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 2 – Тестирование исходного кода

После этого в код были вставлены функции ввода информации, чтобы у пользователя была возможность выбирать директорию. Также для функций train\_datagen.flow\_from\_directory и validation\_datagen.flow\_from\_director были изменены параметры batch\_size в соответствии с количеством изображений для обучения по формуле:

steps\_per\_epoch \* batch\_size = total.

Теперь обучение занимает больше времени, так как обучающих картинок больше, чем в исходном варианте работы.

Затем были протестированы новые датасеты.

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, веб-страница

Автоматически созданное описание

Рисунок 3 – Тестирование программы с новыми датасетами

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, снимок экрана

Автоматически созданное описание

Рисунок 4 – Тестирование программы с новыми датасетами

**Ответы на контрольные вопросы**

1. **Какие классы задач могут быть решены с помощью методов искусственного интеллекта?**

Классификация, регрессия (числовые значения), кластеризация, обнаружение аномалий.

Экспертные системы. Они состоят из двух основных частей: механизма логического вывода и базы знаний. Если знания представлять в определённом виде, можно с небольшими настройками использовать один и тот же механизм логического вывода для разных областей знаний, экспертная система может объяснить всю цепочку рассуждений.

Нейронные сети. Распознавание и классификация объектов, совершенствование алгоритмов, машинное обучение.

Вывод на фреймах. Классификация сложных нечётких объектов. Вывод на фреймах использует принципы человеческой психики, чтобы делать логические заключения и предположения.

Генетические алгоритмы. Возникают ситуации, когда мы не знаем возможных решений. Мы просто ищем такое решение, которое будет оптимальным. То есть ищем минимум или максимум в нечёткой задаче. Генетические алгоритмы – это тоже метод искусственного интеллекта, однако подражает он не психике и не мозгу, а процессу эволюции.

Семантические сети. Представление знаний – отдельная задача в сфере искусственного интеллекта. Одно из её решений – семантические сети. Они представляют собой описание набора сущностей и связей между ними и изображаются в виде графа.

1. **Чем отличается обучающий набор данных от тестового?**

Обучающий набор данных должен отличаться разнообразием данных, а также количеством, чтобы сформировать наиболее точные модели представления предмета. Нет смысла обучать ИИ на малых данных, а тестировать на больших.

1. **Что такое признак в контексте методов искусственного интеллекта? Что такое метка в контексте методов искусственного интеллекта? В чем их разница?**

В системах машинного обучения или же системах нейросетей существуют входы и выходы. То, что подаётся на входы, принято называть признаками. Признаки по существу являются тем же, что и переменные в научном эксперименте — они характеризуют какой-либо наблюдаемый феномен и их можно как-то количественно измерить. Когда признаки подаются на входы системы машинного обучения, эта система пытается найти совпадения, заметить закономерность между признаками. На выходе генерируется результат этой работы.

Этот результат принято называть меткой, поскольку у выходов есть некая пометка, выданная им системой, т. е. предположение (прогноз) о том, в какую категорию попадает выход после классификации.

Процесс обучения модели — это подача данных для нейросети, которая в результате должна вывести определённые шаблоны для данных. В процессе обучения модели с учителем на вход подаются признаки и метки.

1. **Чем методы глубокого обучения отличаются от других методов искусственного интеллекта?**

Глубокое обучение имеет сходство с машинным обучением, но отличается тем, что машинное обучение нуждается в некотором наблюдении при выполнении своих задач обучения, в то время как модель глубокого обучения будет эффективно выполнять свою задачу даже без человеческого руководства (глубокое обучение может контролироваться, частично контролироваться или выполняться без участия человека).

1. **Из чего состоит слой в нейронной сети? Какие слои бывают? Что такое нейрон?**

Нейро́нная сеть (также искусственная нейронная сеть, ИНС) — математическая модель, а также её программное или аппаратное воплощение, построенная по принципу организации и функционирования биологических нейронных сетей.

Нейрон — это вычислительная единица, которая получает информацию, производит над ней простые вычисления и передает ее дальше. Они делятся на три основных типа: входной (синий), скрытый (красный) и выходной (зеленый). Также есть нейрон смещения и контекстный нейрон.

Нейронные сети обычно предстают перед программистом в виде конструктора, состоящего из более-менее простых блоков (слоев). Вот две простейшие их разновидности:

* Линейный слой (linear layer, dense layer) — линейное преобразование над входящими данными (его обучаемые параметры — это матрица и вектор).
* Функция активации (activation function) — нелинейное преобразование, поэлементно применяющееся к пришедшим на вход данным. Благодаря функциям активации нейронные сети способны порождать более информативные признаковые описания, преобразуя данные нелинейным образом.

Нейронные сети могут быть составлены из нескольких соединенных слоев, которые называются многослойными сетями. Обычные нейронные сети могут быть разделены на 3 класса.

Входной слой – это первый слой в нейронной сети, который принимает входящие сигналы и передает их на последующие уровни.

Скрытый (вычислительный) слой применяет различные преобразования ко входным данным. Все нейроны в скрытом слое связаны с каждым нейроном в следующем слое.

Выходной слой – последний слой в сети, который получает данные от последнего скрытого слоя. С его помощью мы сможем получить нужное количество значений в желаемом диапазоне.

**Вывод**

В ходе работы были получены навыки работы с методами ИИ для решения задачи классификации данных с использованием языка программирования Python.

**Приложение**

import os

import numpy as np

import tensorflow as tf

import matplotlib.pyplot as plt

import matplotlib.image as mpimg

import pickle

from scipy import interp

from itertools import cycle

from sklearn import svm, datasets

from sklearn.metrics import roc\_auc\_score

from sklearn.metrics import roc\_curve, auc

from sklearn.preprocessing import label\_binarize

from tensorflow.keras.preprocessing import image

from sklearn.multiclass import OneVsRestClassifier

from sklearn.model\_selection import train\_test\_split

from tensorflow.keras.preprocessing.image import ImageDataGenerator

#train\_dandelion\_dir = input("Введите расположение тренировочных картинок коровы: ")

#train\_grass\_dir = input("Введите расположение тренировочных картинок человека: ")

#valid\_dandelion\_dir = input("Введите расположение тестовых картинок коровы: ")

#valid\_grass\_dir = input("Введите расположение тестовых картинок человека: ")

train\_dandelion\_dir = os.path.join('train1/cow')

train\_grass\_dir = os.path.join('train1/human')

valid\_dandelion\_dir = os.path.join('valid1/cow')

valid\_grass\_dir = os.path.join('valid1/human')

train\_dandelion\_names = os.listdir(train\_dandelion\_dir)

train\_grass\_names = os.listdir(train\_grass\_dir)

validation\_grass\_names = os.listdir(valid\_grass\_dir)

print('total training cow images:', len(os.listdir(train\_dandelion\_dir)))

print('total training human images:', len(os.listdir(train\_grass\_dir)))

print('total validation cow images:', len(os.listdir(valid\_dandelion\_dir)))

print('total validation human images:', len(os.listdir(valid\_grass\_dir)))

"""nrows = 4

ncols = 4

pic\_index = 0

fig = plt.gcf()

fig.set\_size\_inches(ncols \* 4, nrows \* 4)

pic\_index += 8

next\_dandelion\_pic = [os.path.join(train\_dandelion\_dir, fname)

for fname in train\_dandelion\_names[pic\_index-8:pic\_index]]

next\_grass\_pic = [os.path.join(train\_grass\_dir, fname)

for fname in train\_grass\_names[pic\_index-8:pic\_index]]

for i, img\_path in enumerate(next\_dandelion\_pic + next\_grass\_pic):

sp = plt.subplot(nrows, ncols, i + 1)

sp.axis('Off')

img = mpimg.imread(img\_path)

plt.imshow(img)

#plt.show()"""

train\_datagen = ImageDataGenerator(rescale=1/255)

validation\_datagen = ImageDataGenerator(rescale=1/255)

train\_generator = train\_datagen.flow\_from\_directory(

'train1/',

classes = ['cow', 'human'],

target\_size=(200, 200),

batch\_size=657,

class\_mode='binary')

validation\_generator = validation\_datagen.flow\_from\_directory(

'valid1/',

classes = ['cow', 'human'],

target\_size=(200, 200),

batch\_size=10,

class\_mode='binary',

shuffle=False)

flag = 1

model = tf.keras.models.load\_model('model.h5')

if model:

flag = 0

if (flag):

model = tf.keras.models.Sequential([tf.keras.layers.Flatten(input\_shape = (200,200,3)),

tf.keras.layers.Dense(128, activation=tf.nn.relu),

tf.keras.layers.Dense(1, activation=tf.nn.sigmoid)])

model.summary()

model.compile(optimizer = tf.keras.optimizers.Adam(),

loss = 'binary\_crossentropy',

metrics=['accuracy'])

history = model.fit(train\_generator,

steps\_per\_epoch=8,

epochs=15,

verbose=1,

validation\_data = validation\_generator,

validation\_steps=8)

uploaded = ['test2.jpg']

if flag:

model.save('model.h5')

#content\_dir = input("Введите расположение файла для распознавания: ")

#uploaded = [input("Введите название файла: ")]

choice = int(input("Введите количество картинок: "))

if choice == 1:

way = input("Напишите путь к файлу: ")

path = fr"{way}"

img = image.load\_img(path, target\_size=(200, 200))

x = image.img\_to\_array(img)

plt.imshow(x / 255.)

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

images = np.vstack([x])

classes = model.predict(images, batch\_size=10)

print(classes[0])

if classes[0] < 0.5:

print("Это коровка")

else:

print("Это человек")

else:

path = r'test1'

count\_cow = 0

count\_human = 0

file\_names = os.listdir(path)

for file\_name in file\_names:

fullpath = os.path.join(path, file\_name)

img = image.load\_img(fullpath, target\_size=(200, 200))

x = image.img\_to\_array(img)

plt.imshow(x / 255.)

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

images = np.vstack([x])

classes = model.predict(images, batch\_size=10)

print(classes[0])

if classes[0] < 0.5:

count\_cow += 1

else:

count\_human += 1