```
#### Задание ####
# По видео определить является ли движущиˇся объект человеком
# 1. Взять видео с движущимся человеком и не человеком
# 2. Движущиеся объекты выделить прямоугольником,
#     сохранить прямоугольники в виде фаиˇлов jpg (crop),
#     для каждого прямоугольника определить класс (человек, не человек) - сетка,
#     в лог фаиˇле написать что на таком - то кадре был человек с кординатами x1,y1,x2,y2

# 22.12.2020 - Промежуточные итоги
# VGG16 делает неправильный предикт

import cv2 as cv
import numpy as np
import pandas as pdfunctions.py
import functions # мои функции

import keras
from keras import layers, optimizers
from keras.models import Model, Sequential
from keras.applications.vgg16 import VGG16, preprocess_input, decode_predictions
from keras.layers import Flatten, Dense, GlobalAveragePooling2D
from keras.preprocessing import image
from keras.preprocessing.image import load_img, img_to_array

print('import success')

# Считывание видео
video = cv.VideoCapture('test_video.mp4')
length = int(video.get(cv.CAP_PROP_FRAME_COUNT))
width = int(video.get(cv.CAP_PROP_FRAME_WIDTH))
height = int(video.get(cv.CAP_PROP_FRAME_HEIGHT))

# VGG16
model = VGG16(weights='imagenet',
              include_top=True, # подключаем классификатор
              input_shape=(224, 224, 3))

# Для подсчета кадров видео в цикле
k = 0

if not video.isOpened():
    print('ERROR OPEN VIDEO')

cv.waitKey(1) # видео - это переключающиеся картинки -
```

```
44 задаем время переключения (1 мс)
45 
46 # для записи конечного видео
47 writer = cv.VideoWriter_fourcc(*'mp4v')
48 out = cv.VideoWriter('test_video_vgg16.mp4', writer, 24.0
   , (width, height))
49 
50 # считываем видео
51 _, frame1 = video.read()
52 _, frame2 = video.read()
53 
54 # Путь для записи в лог
55 log_file_path = "/Users/ardandorzhiev/Downloads/UCHEBA_FA/
   Машинное зрение 2020/zachetnoe_zadanie/log_data_vgg16.csv"
56 
57 while (video.isOpened() and k < length):
58 
59     # Находим разницу между кадрами
60     diff = cv.absdiff(frame1, frame2)
61 
62     # Чтобы легче было найти контуры переведем в серый и
   заблюрим
63     gray = cv.cvtColor(diff, cv.COLOR_BGR2GRAY)
64     blur = cv.GaussianBlur(gray, (5, 5), 0)
65 
66     # Определим порог 20
67     _, thresh = cv.threshold(blur, 20, 255, cv.
   THRESH_BINARY)
68 
69     # Выполним дилатацию, чтобы "залить" все "дырки" из
   изображния с порогом
70     dilated = cv.dilate(thresh, None, iterations=3)
71 
72     # Находим контуры
73     contours, _ = cv.findContours(dilated, cv.RETR_TREE, cv
   .CHAIN_APPROX_SIMPLE)
74 
75     # Обходим контуры
76     cnt_contours = 0
77 
78     for contour in contours:
79         (x, y, w, h) = cv.boundingRect(contour)
80 
81         ### Если контур маленький, то ничего  не делаем, и
   переходим к следующему контуру
82         if cv.contourArea(contour) < 500:
83             continue
84 
85         # Рисуем рамку
86         cv.rectangle(frame1, (x, y), (x + w, y + h), (0,
```

```
 86 255, 0), 2)
 87
 88         # Вырезаем объект из рамки
 89         obj = frame1.copy()[y:y+h, x:x+w]
 90
 91         # Определяем класс объекта. Используем нейросеть
    из коробки
 92         image = cv.resize(obj, (224,224))
 93         image = img_to_array(image)
 94         image = image.reshape((1, image.shape[0], image.
    shape[1], image.shape[2]))
 95         image = preprocess_input(image)
 96         predict = model.predict(image)
 97
 98         label = decode_predictions(predict)
 99         label = label[0][0]
100
101         class_name = str(label[1])
102         proba = label[2].item() # .item() - из np.float32
     -> float
103
104         ## Если уверенность в классе больше 70 процентов,
    то сохраняем и выводим класс на экран
105         if proba > 0.1:
106             proba = int(round(proba * 100))
107
108             # сохраняем объект, распознанный в кадре
109             path_name = 'objects_vgg16/' + 'frame_' + str(
    k) + '_obj_'+ class_name + '_' + str(cnt_contours) + '.jpg
    '
110             cv.imwrite(path_name, obj)
111
112             # Название объекта в рамке
113             class_name_prob = class_name + '_' + str(proba
    ) + '%'
114
115             cv.putText(frame1, class_name_prob, (x, y-5),
    cv.FONT_HERSHEY_COMPLEX_SMALL, 1, (0, 0, 255), 2)
116
117             #Записываем в лог
118             functions.save_log(id_frame = k, id_object =
    cnt_contours,
119                                x_coord = x, y_coord = y,
120                                width = w, height = h,
121                                obj_class = class_name,
    obj_proba = proba,
122                                log_file_path =
    log_file_path)
123
124             # Обработали один контур
```

```
125             cnt_contours += 1
126 
127     # Прочитали один кадр
128     k += 1
129 
130     # Записываем итог
131     out.write(frame1)
132 
133     # Выводим на экран итог
134     cv.imshow("Video", frame1)
135     frame1 = frame2
136     ret, frame2 = video.read()
137 
138     if k > length - 2:
139         video.release()
140         out.release()
141         cv.destroyAllWindows()
142 
143     if cv.waitKey(1) & 0xFF == ord('q'): # нажимаем q если
    надо остановить воспроизвдеение
144         video.release()
145         out.release()
146         cv.destroyAllWindows()
```