Table of Content:

* Abstract
* Phase 1
* Phase 2

Abstract

As the last challenge we distributed the work on the same manner but with some different steps (1.Data collection and preparation, 2.Extracting number of Black Sea Bass fish on every frame and plotting the result).However we faced many challenges on getting the job done, that we will discuss in the following.

Phase 1 : Data Collection

We collected crab data from many sources:

* Images from google images
* Images from free-to-use image websites like creative commons
* Some frames from the given video

We used labelImg to annotate our images in a pascal voc format. After we finished labeling we uploaded the images to google drive so we can import it to google colab easily.

Phase 2 : Making the Model

We made an object detection model to detect the black sea bass using pytorch and detecto. We used google Colab to train our model as it’s much more powerful than training the model locally. And here is some snippets from the code and its description :

Installing libraries

!pip install detecto

!git clone https://github.com/alankbi/detecto.git

Copying data from drive to Colab

cp -v /content/drive/MyDrive/azure\_round\_2/voc/\*.xml /content/dataset

cp -v /content/drive/MyDrive/azure\_round\_2/obj/\*.jpg /content/dataset

Setting the dataset

import matplotlib.pyplot as plt

from detecto.utils import read\_image

from detecto.core import Model, DataLoader

from detecto.core import Dataset

labels = ['black\_sea\_bass']

model = Model(labels)

dataset = Dataset("/content/dataset")

Training the model

model.fit(dataset)

Phase 3 : plotting the data

We receive a csv file from the model that contains the number of fish in every frame, then we plot the data using matplotlib using the following code:

Importing libraries

import random

import collections

import matplotlib.pyplot as plt

import numpy as np

import pandas as pd

Functions:

Averaging function to average a given list

def Average(lst):

Average = int(sum(lst) / len(lst))

return Average

Simplifying function to

def Average(lst):

Average = int(sum(lst) / len(lst))

return Average

def simpler(Fish\_Number\_List):

simp = []

x = 16

simp.append(Average(Fish\_Number\_List[:x]))

while 1:

try:

simp.append(Average(Fish\_Number\_List[x:x+30]))

x += 30

except:

if len(Fish\_Number\_List[x:]) > 0:

simp.append(Average(Fish\_Number\_List[x:]))

print(x)

break

return simp

def plot(x, y, color, label, order, max\_x, max\_y, linewidth=1, markersize=12):

markerfacecolor = color

plt.subplot(3, 1, order)

plt.plot(x, y, color=color, linewidth=linewidth, markerfacecolor=markerfacecolor, markersize=markersize,

label=label)

plt.grid()

plt.legend([label])

plt.ylim(0, max\_y + 10)

plt.xlim(0, (max\_x + 2) / 30)

plt.yticks(np.arange(0, max\_y + 10, 5))

plt.xticks(np.arange(0, max\_x + 2, 200) / 30)

plt.xlabel('Seconds')

plt.ylabel('Fish')