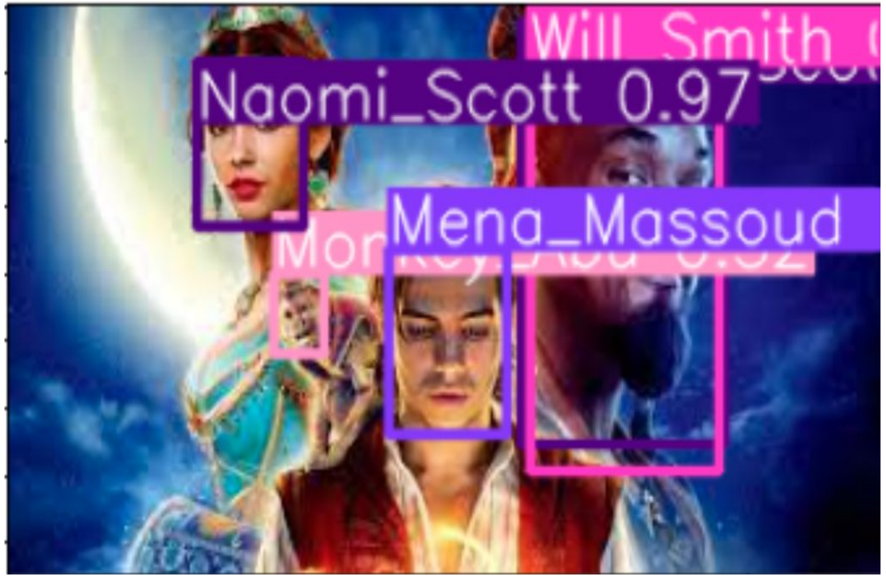
Object Detection using YOLO

Archana G S

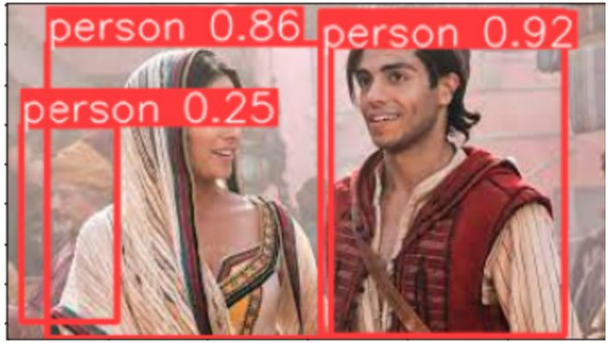
Identify person/object using YOLO

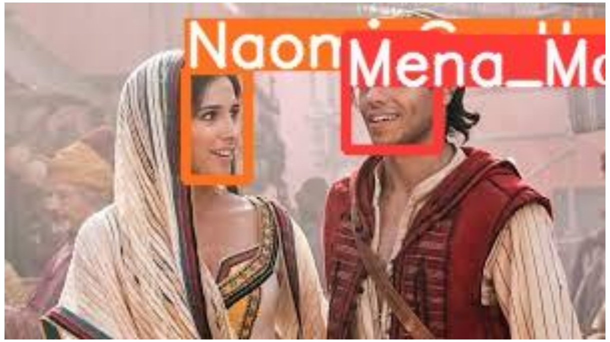


Object detection

Involves both localizing one or more objects within an image and classifying each object in the image.

- Input: An image with one or more objects (e.g. photograph)
- Output: One or more bounding boxes (e.g. defined by a point, width, and height), and a class label for each bounding box.





YOLO (*You Only Look Once*)

Our unified architecture is extremely fast. Our base YOLO model processes images in real-time at 45 frames per second. A smaller version of the network, Fast YOLO, processes an astounding 155 frames per second …

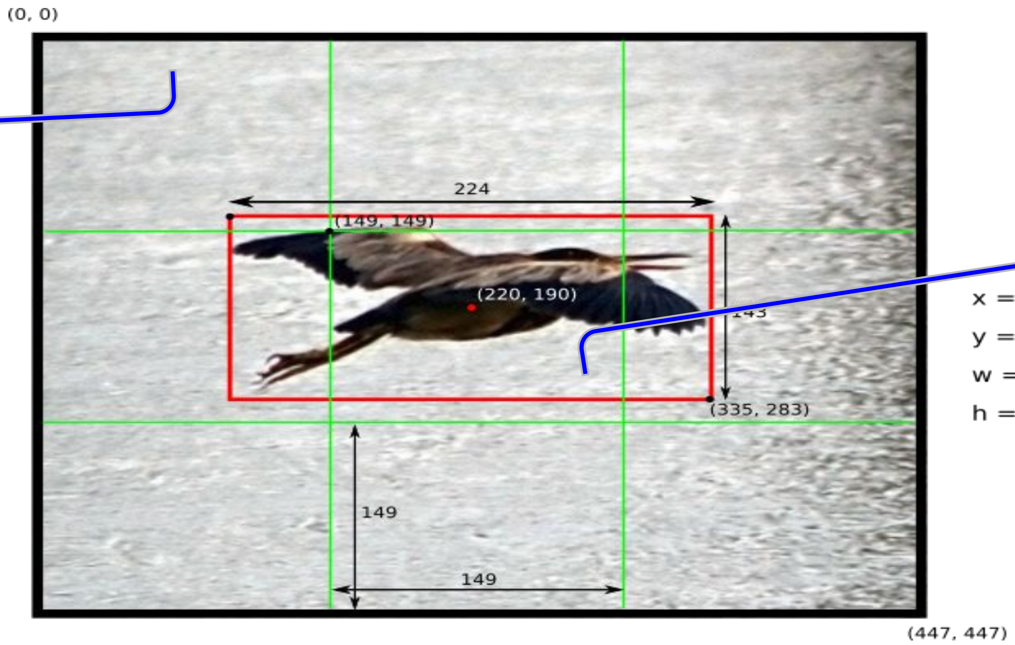
— You Only Look Once: Unified, Real-Time Object Detection, 2015.

We reframe object detection as a single regression problem, straight from image pixels to bounding box coordinates and class probabilities.

— You Only Look Once: Unified, Real-Time Object Detection, 2015

YOLO – Prediction Vector

Pc	0
x	
y	
w	
h	
C0 (Bird)	
C1 (Cat)	



x = (220-149) / 149 = 0.48

y = (190-149) / 149 = 0.28

w = 224 / 448 = 0.50

h = 143 / 448 = 0.32

Pc	1
x	0.48
y	0.28
w	0.50
h	0.32
C0 (Bird)	1
C1 (Cat)	0

Figure 1. Calculate box coordinates in a 448x448 image with S=3 (Menegaz, 2018)

Architecture: Object Detection

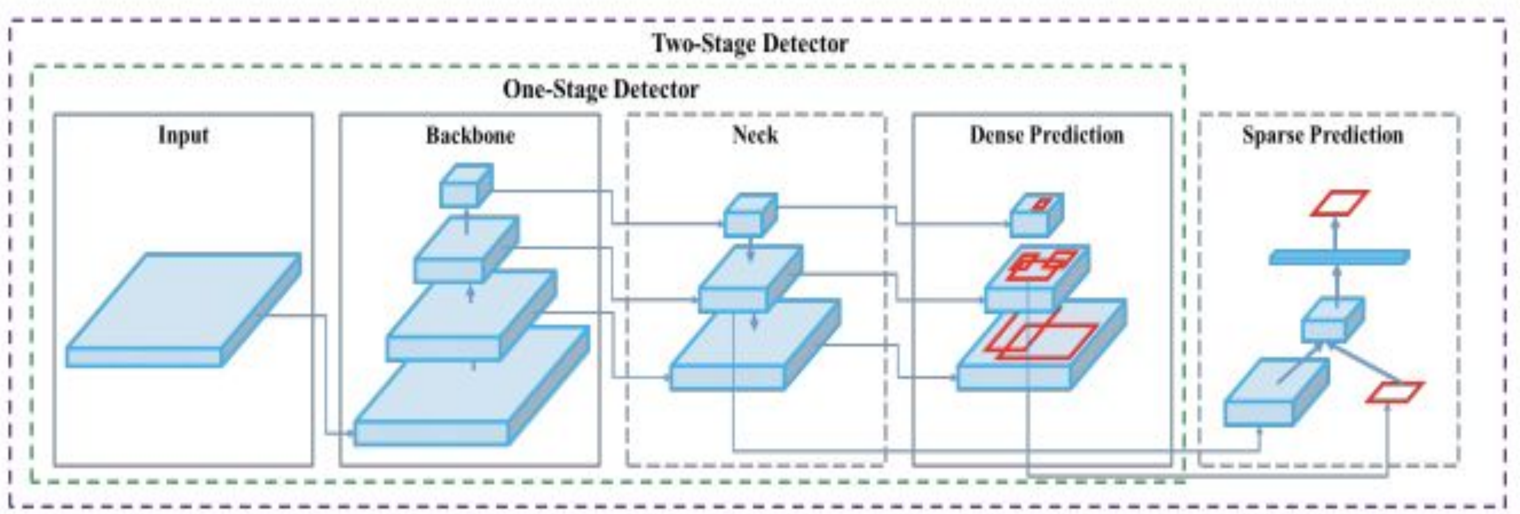


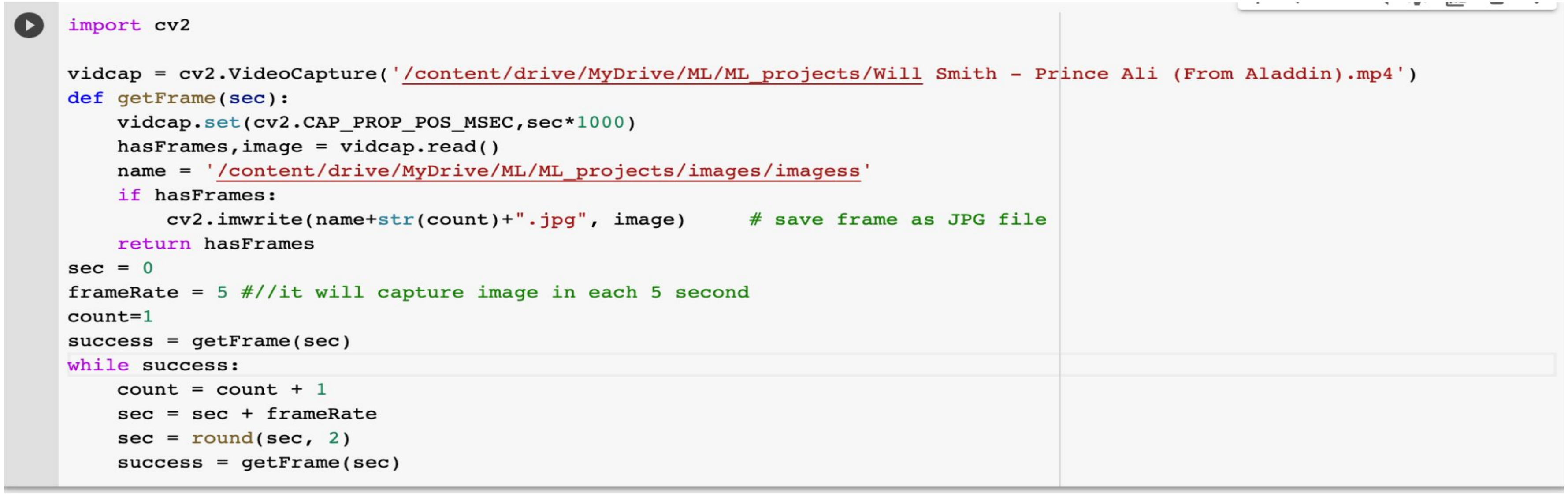
Figure 2. Two concepts of architectural object detection. (Solawetz, 2020)

Data Source

- Google Images
- YouTube Video

Data collection and preprocessing

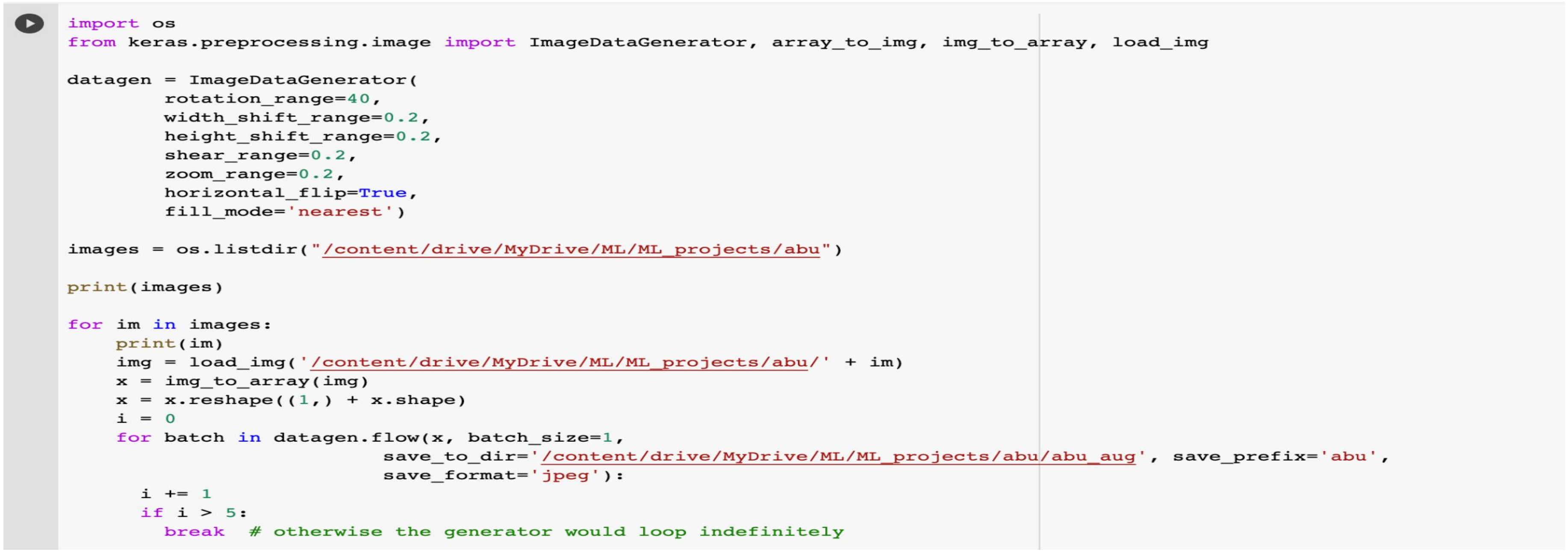
1. Video to image



```
import cv2

vidcap = cv2.VideoCapture('/content/drive/MyDrive/ML/ML_projects/Will Smith - Prince Ali (From Aladdin).mp4')
def getFrame(sec):
    vidcap.set(cv2.CAP_PROP_POS_MSEC,sec*1000)
    hasFrames,image = vidcap.read()
    name = '/content/drive/MyDrive/ML/ML_projects/images/imagess'
    if hasFrames:
        cv2.imwrite(name+str(count)+".jpg", image)     # save frame as JPG file
    return hasFrames
sec = 0
frameRate = 5 #//it will capture image in each 5 second
count=1
success = getFrame(sec)
while success:
    count = count + 1
    sec = sec + frameRate
    sec = round(sec, 2)
    success = getFrame(sec)
```

Data collection and preprocessing contd..

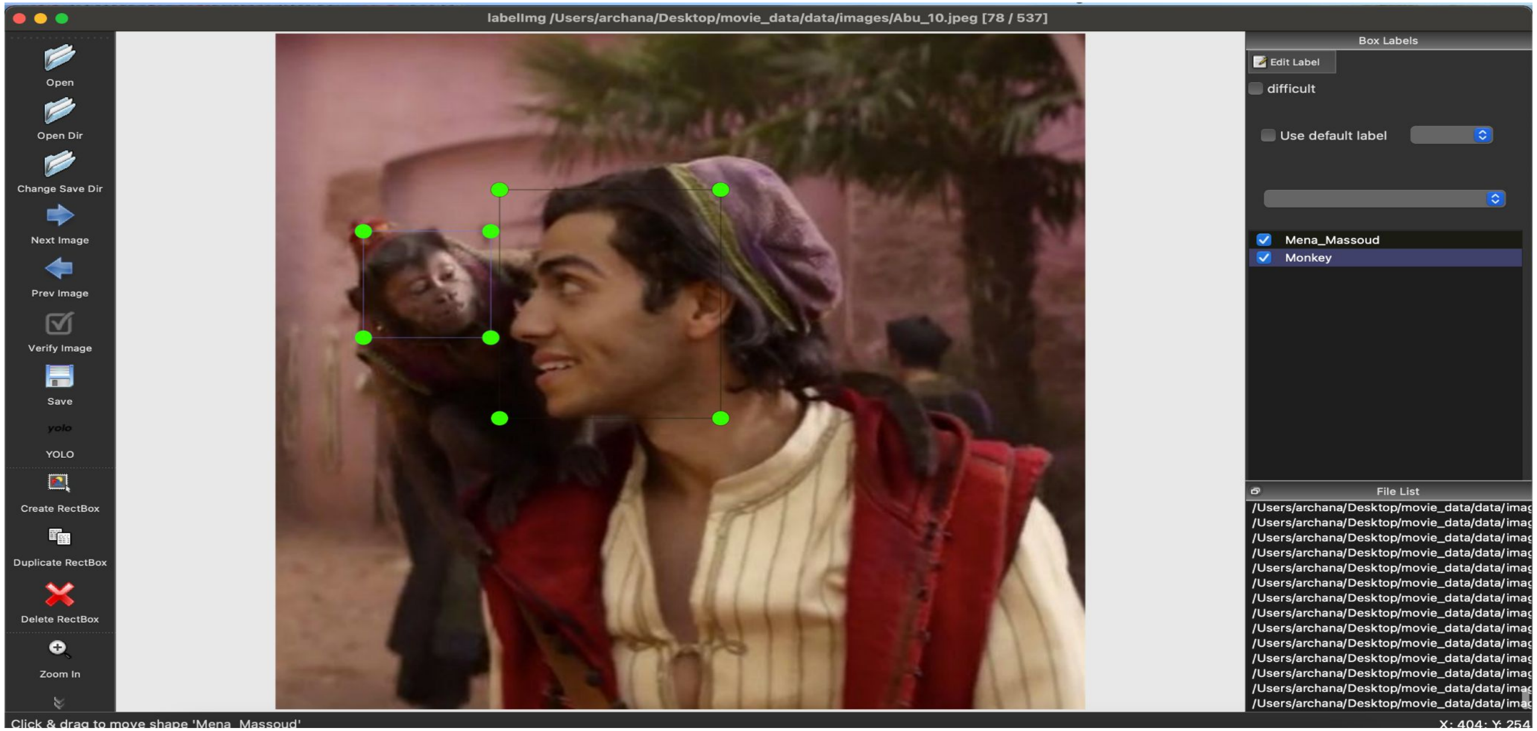
2. Data augmentation



```
import os
from keras.preprocessing.image import ImageDataGenerator, array_to_img, img_to_array, load_img

datagen = ImageDataGenerator(
        rotation_range=40,
        width_shift_range=0.2,
        height_shift_range=0.2,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True,
        fill_mode='nearest')

images = os.listdir("/content/drive/MyDrive/ML/ML_projects/abu")

print(images)

for im in images:
    print(im)
    img = load_img('/content/drive/MyDrive/ML/ML_projects/abu/' + im)
    x = img_to_array(img)
    x = x.reshape((1,) + x.shape)
    i = 0
    for batch in datagen.flow(x, batch_size=1,
                              save_to_dir='/content/drive/MyDrive/ML/ML_projects/abu/abu_aug', save_prefix='abu',
                              save_format='jpeg'):
      i += 1
      if i > 5:
        break  # otherwise the generator would loop indefinitely
```

Data collection and preprocessing contd..

3. Data annotation



labelImg /Users/archana/Desktop/movie_data/data/images/Abu_10.jpeg [78 / 537]

- ✓ Mena_Massoud
- ✓ Monkey

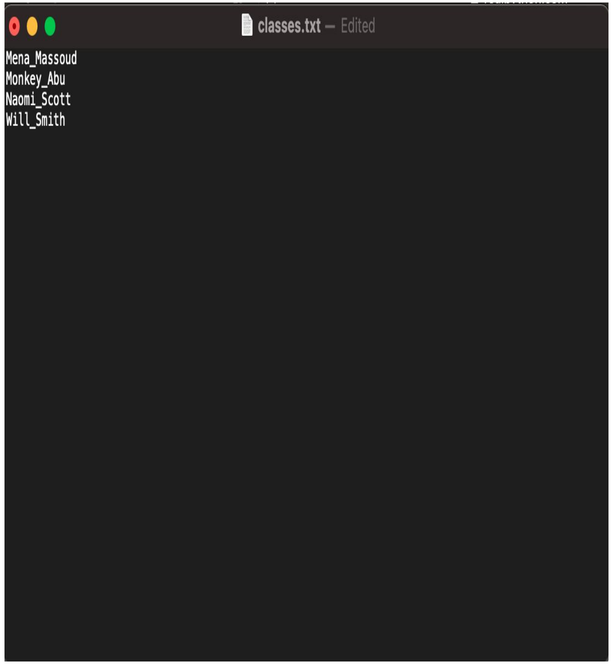
```
/Users/archana/Desktop/movie_data/data/imag
/Users/archana/Desktop/movie_data/data/imag
/Users/archana/Desktop/movie_data/data/imag
/Users/archana/Desktop/movie_data/data/imag
/Users/archana/Desktop/movie_data/data/imag
/Users/archana/Desktop/movie_data/data/imag
/Users/archana/Desktop/movie_data/data/imag
/Users/archana/Desktop/movie_data/data/imag
/Users/archana/Desktop/movie_data/data/imag
/Users/archana/Desktop/movie_data/data/imag
/Users/archana/Desktop/movie_data/data/imag
/Users/archana/Desktop/movie_data/data/imag
/Users/archana/Desktop/movie_data/data/imag
/Users/archana/Desktop/movie_data/data/imag
```

Click & drag to move shape 'Mena_Massoud'

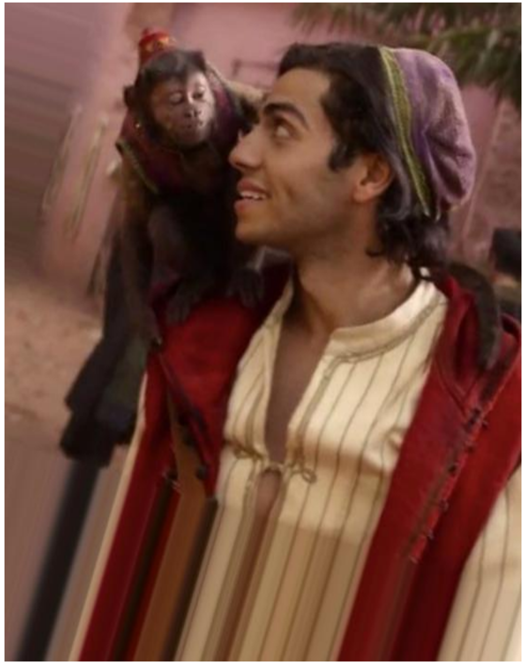
X: 404; Y: 254

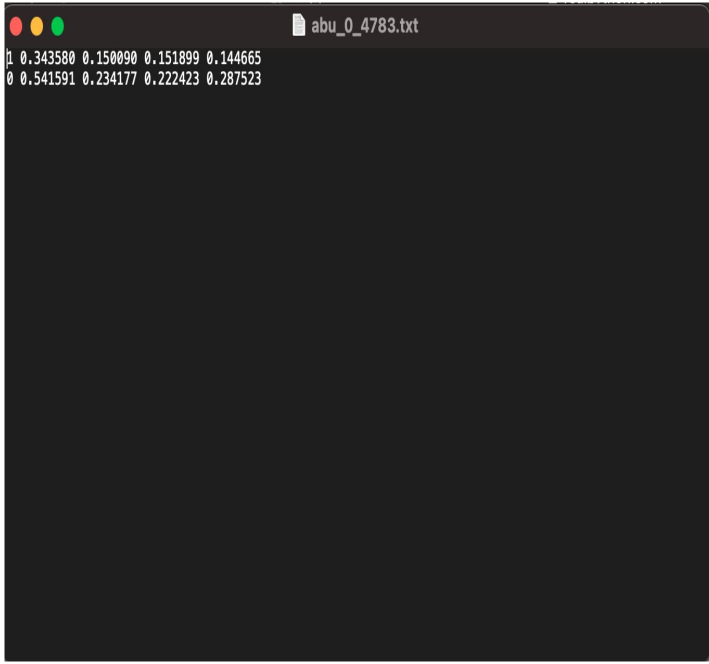
Data collection and preprocessing contd..

3. Data annotation contd..



```
Mena_Massoud
Monkey_Abu
Naomi_Scott
Will_Smith
```



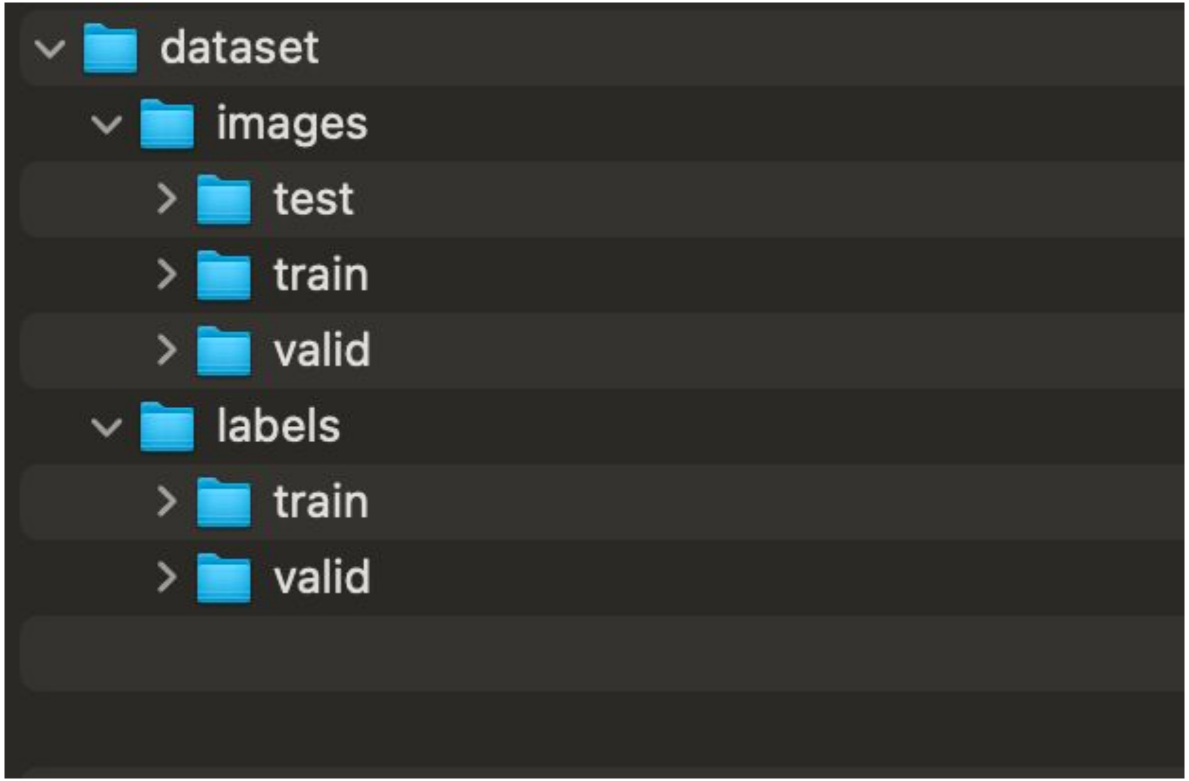


```
1 0.343580 0.150090 0.151899 0.144665
0 0.541591 0.234177 0.222423 0.287523
```

Dataset creation

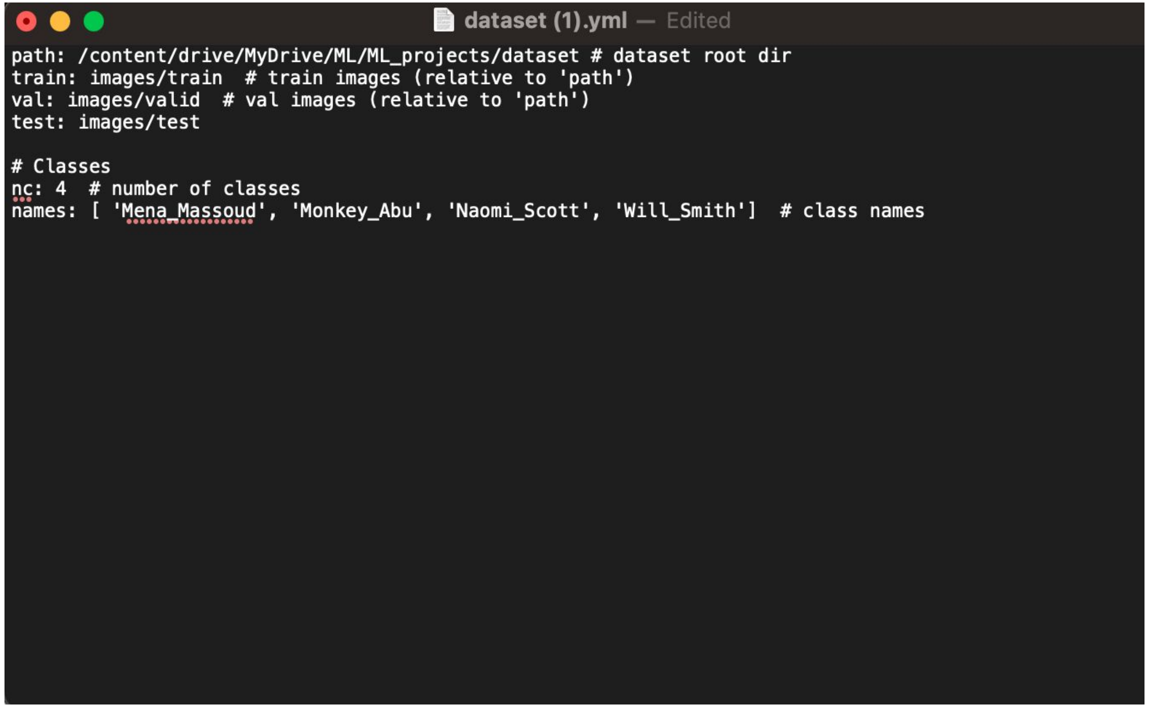
- Create a balance dataset
- Label every instance of every class
- Dataset should have a lot of variety
- Annotation should be consistent
- Dataset is divided in training, validation and test set

Folder structure



```
dataset
  images
    test
    train
    valid
  labels
    train
    valid
```

YAML file for training

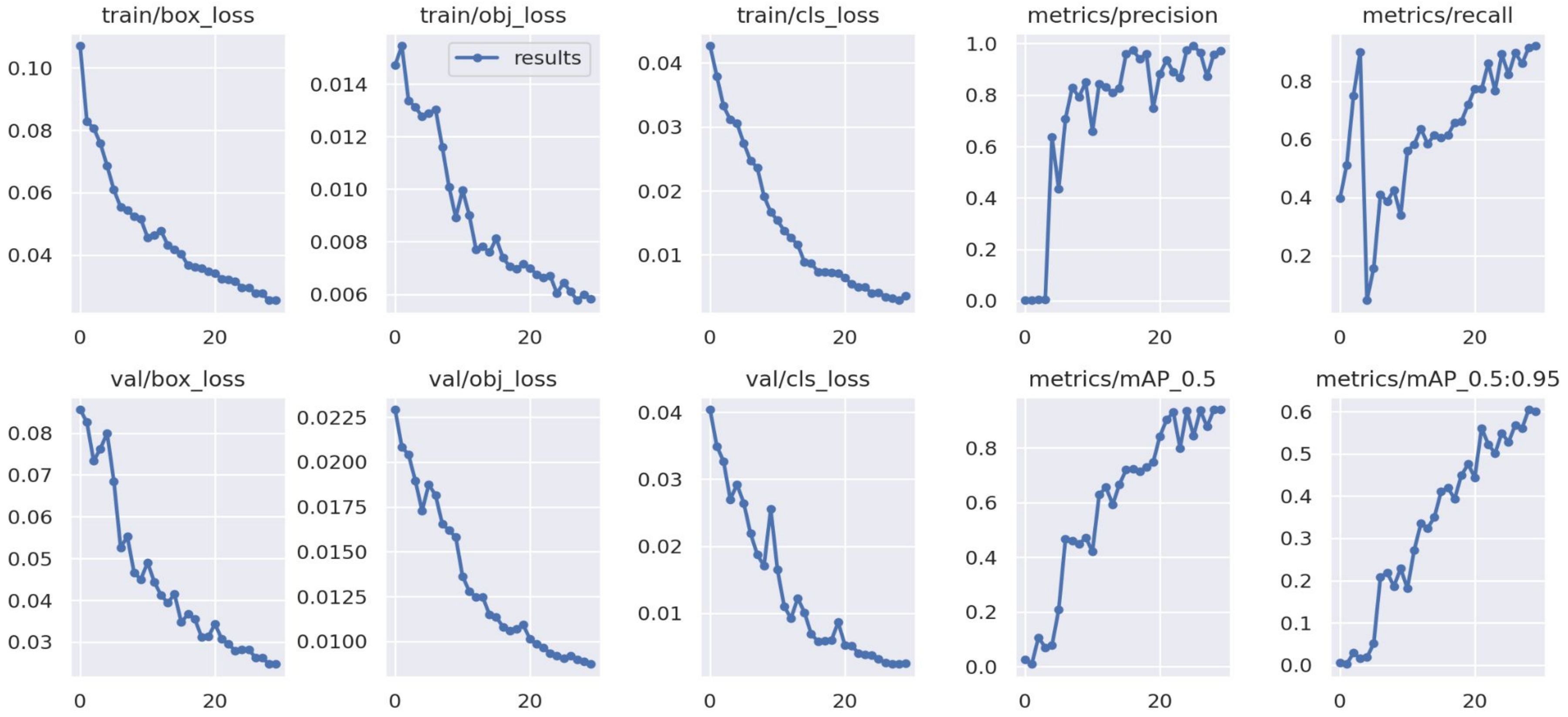

dataset (1).yml — Edited

```
path: /content/drive/MyDrive/ML/ML_projects/dataset # dataset root dir
train: images/train  # train images (relative to 'path')
val: images/valid  # val images (relative to 'path')
test: images/test

# Classes
nc: 4  # number of classes
names: [ 'Mena_Massoud', 'Monkey_Abu', 'Naomi_Scott', 'Will_Smith']  # class names
```

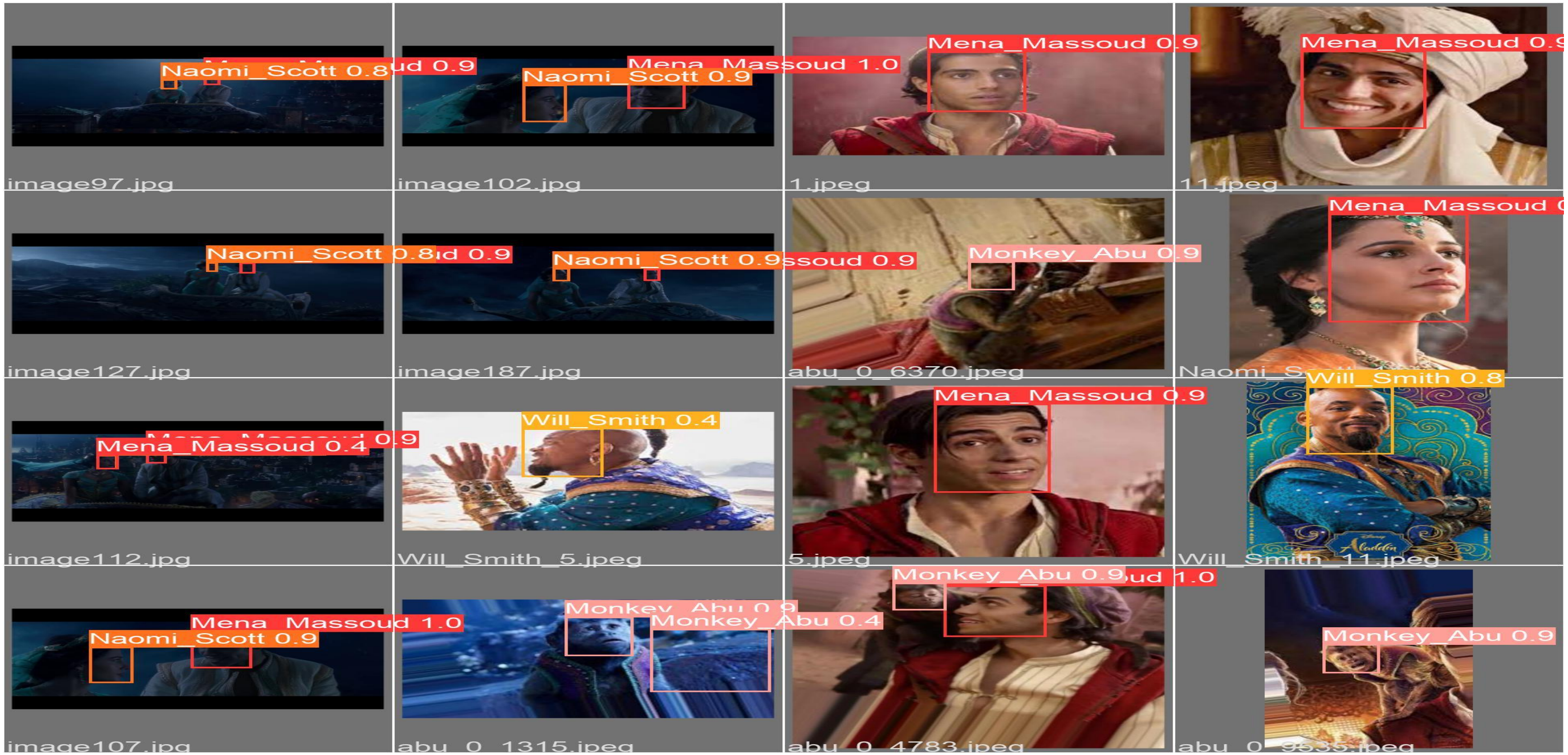
Training

```
!cd yolov5 && python train.py --img 416 --batch 16 --epochs 30
--data '/content/drive/dataset/dataset.yml' --cfg
'/content/drive/dataset/custom_yolov5s.yaml' --weights
yolov5s.pt  --cache
```

Result


train/box_loss
train/obj_loss
results
train/cls_loss
metrics/precision
metrics/recall
val/box_loss
val/obj_loss
val/cls_loss
metrics/mAP_0.5
metrics/mAP_0.5:0.95

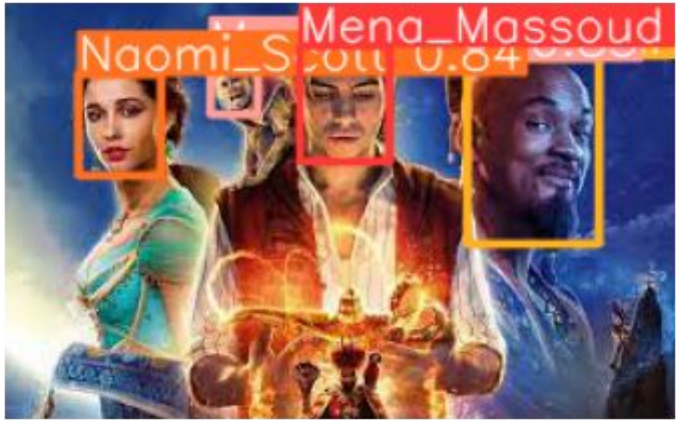
Validation Results

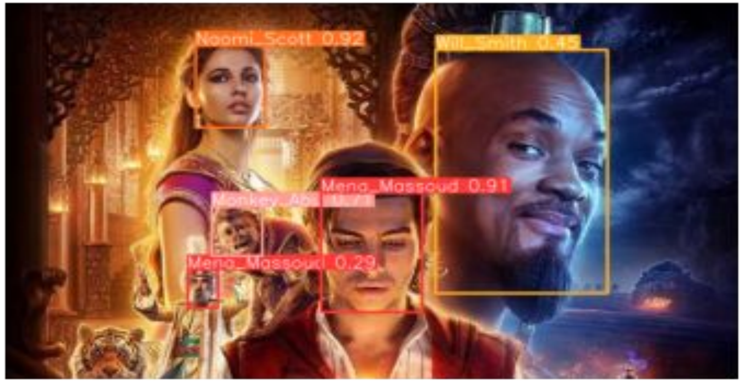


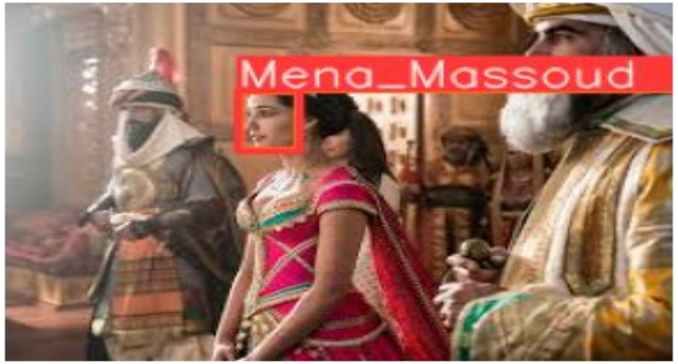
Detect

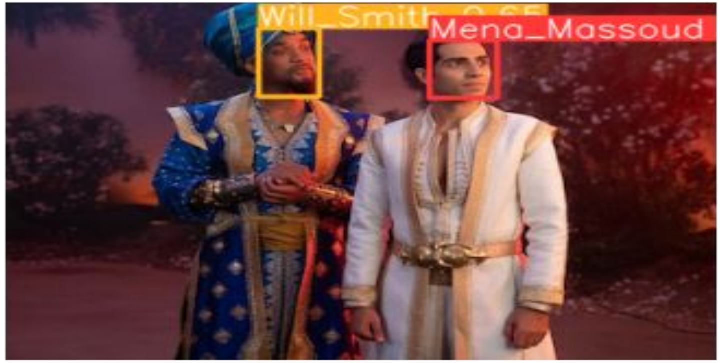
```
!cd yolov5 &&python detect.py --weights
'/content/yolov5/runs/train/exp2/weights/best.pt' --source
'/content/drive/dataset/images/test' --name 'detect_test'
```

Result


Mena_Massoud
Naomi_Scott 0.84


Naomi_Scott 0.92
Will_Smith 0.45
Mena_Massoud 0.91
Monkey_Abu
Mena_Massoud 0.29


Mena_Massoud


Will_Smith
Mena_Massoud

Use Cases

- Identify the artist who is playing a character in a movie
- Identify/track a player/object in a sports event
- Identify/track a person/object in a crowd

Limitation

- Small dataset
- Limited to one movie

Improvement

- Collect more images
- Try with different versions of YOLOV5

Questions ?

Thank You