OBJECT DETECTION AND RECOGNITION

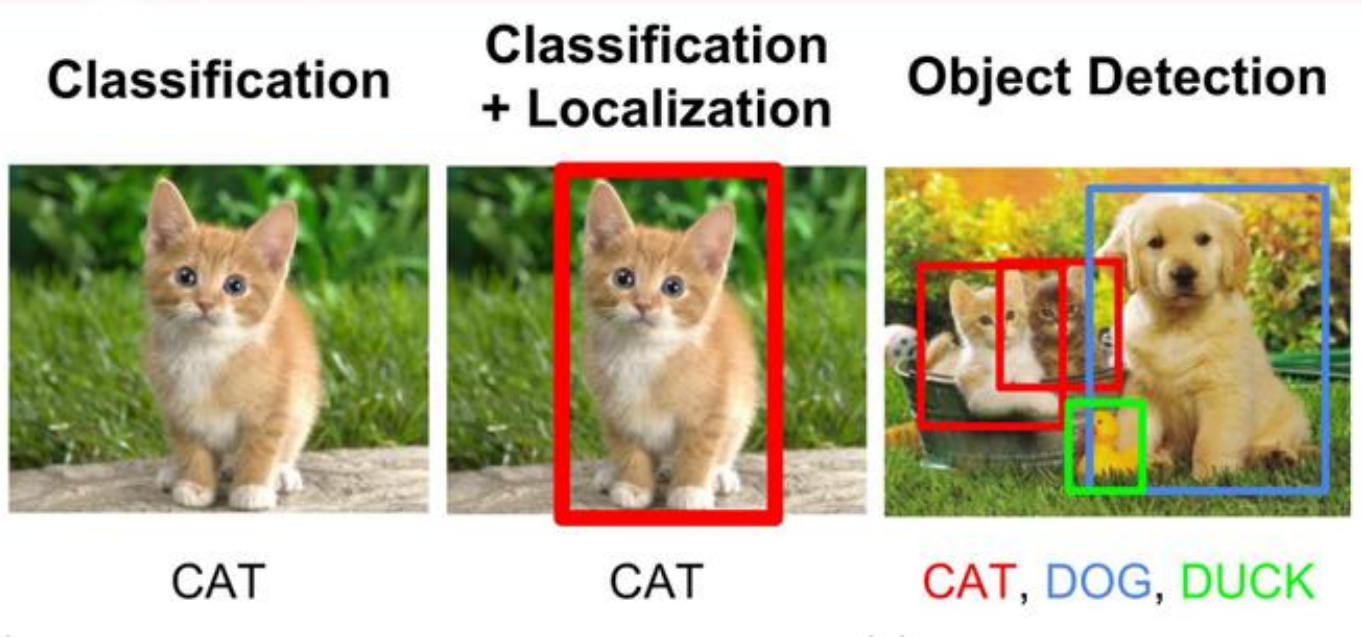
Christine Dewi Ph.D.

Department of Information Management,
Chaoyang University of Technology Taichung, Taiwan, R.O.C.
Department of Information Technology,
Satya Wacana Christian University Salatiga, Indonesia.

Website : https://sites.google.com/view/christinedewi/home

WHAT IS OBJECT DETECTION?

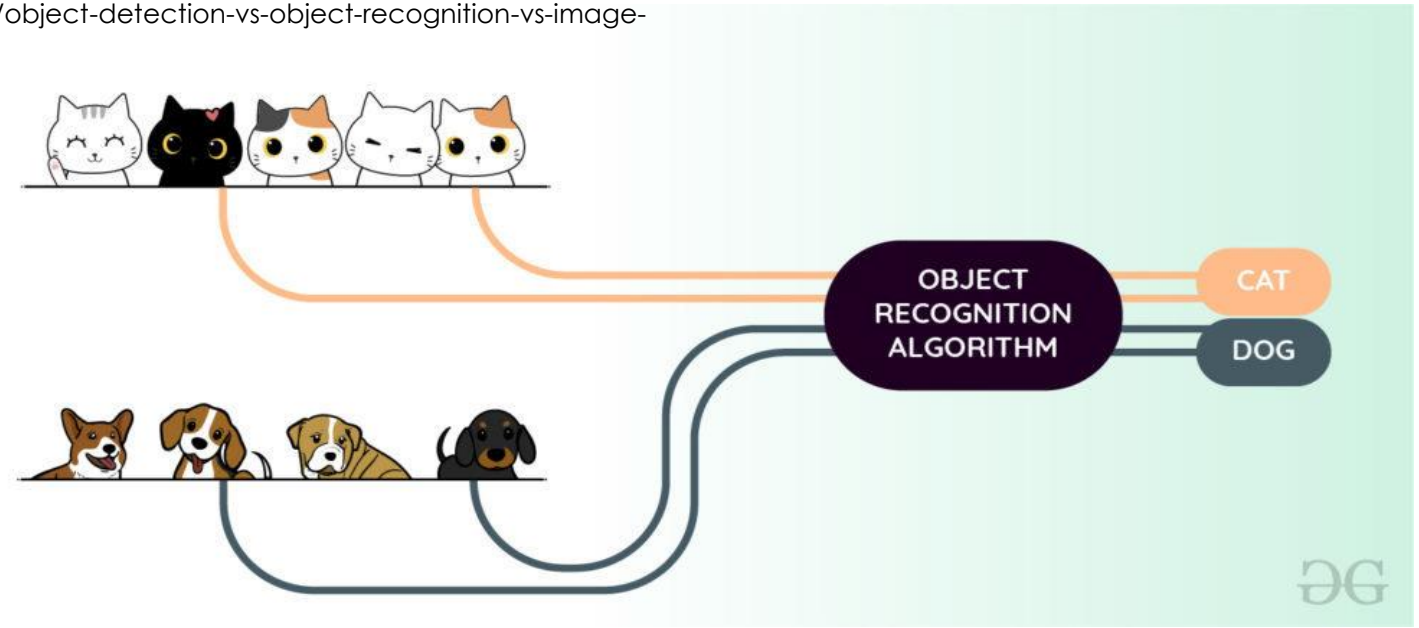
- Object Detection is a computer technology related to computer vision and image processing that deal with detecting instances of semantic object of certain class (such as human, building, car, table, dog, cat etc.).



WHAT IS OBJECT RECOGNITION?

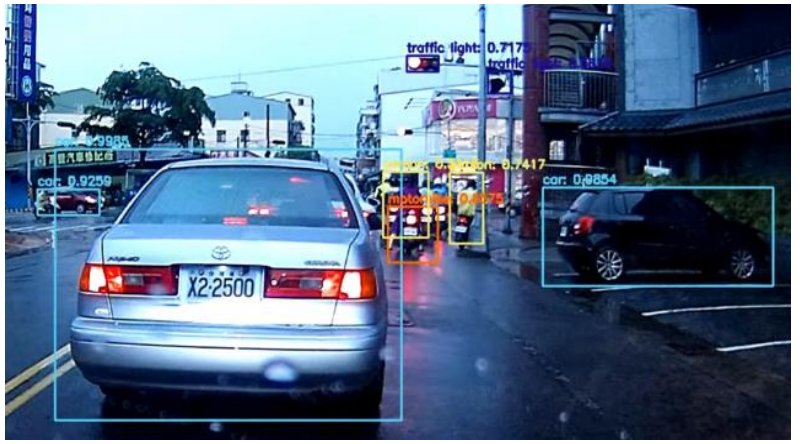
- Object recognition is the technique of identifying the object present in images and videos. It is one of the most important applications of machine learning and deep learning. The goal of this field is to teach machines to understand (recognize) the content of an image just like humans do.

https://www.geeksforgeeks.org/object-detection-vs-object-recognition-vs-image-segmentation/



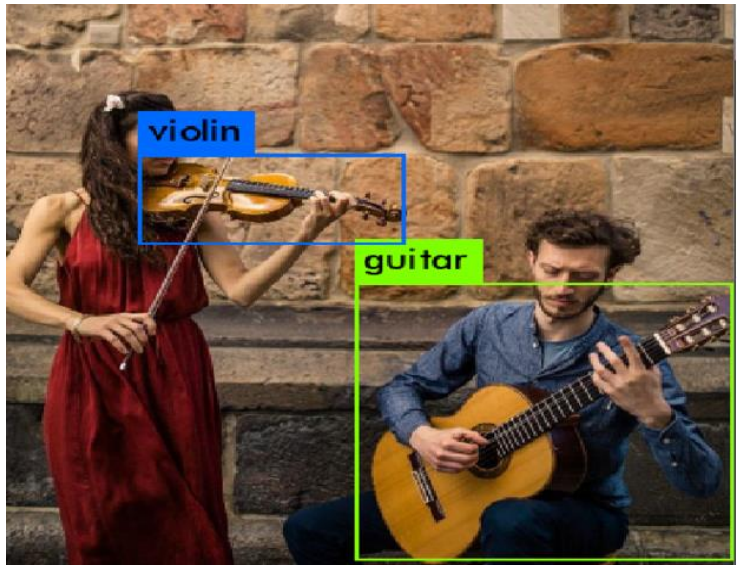
APPLICATIONS:

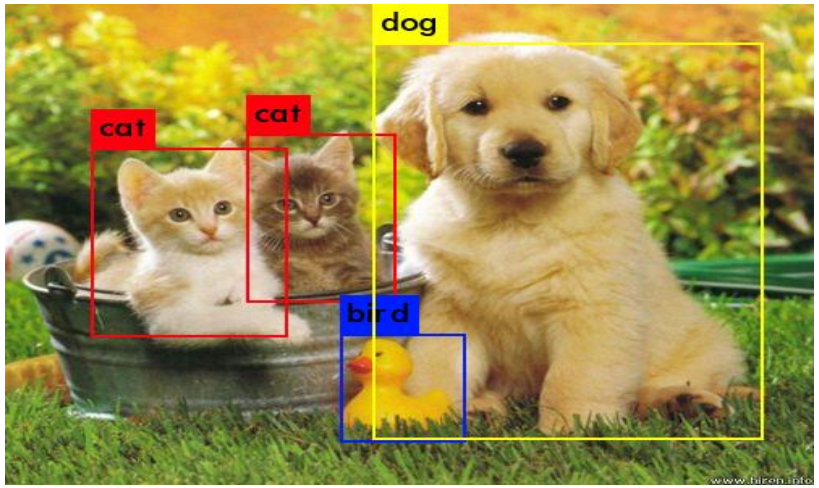
- **Driver-less Cars:** Object Recognition is used for detecting road signs, other vehicles, etc.
- **Medical Image Processing:** Object Recognition and Image Processing techniques can help detect disease more accurately. For Example, Google AI for breast cancer detection detects more accurately than doctors.
- **Surveillance and Security:** such as Face Recognition, Object Tracking, Activity Recognition (WALKING, WALKING_UPSTAIRS, WALKING_DOWNSTAIRS, SITTING, STANDING, LAYING), etc.

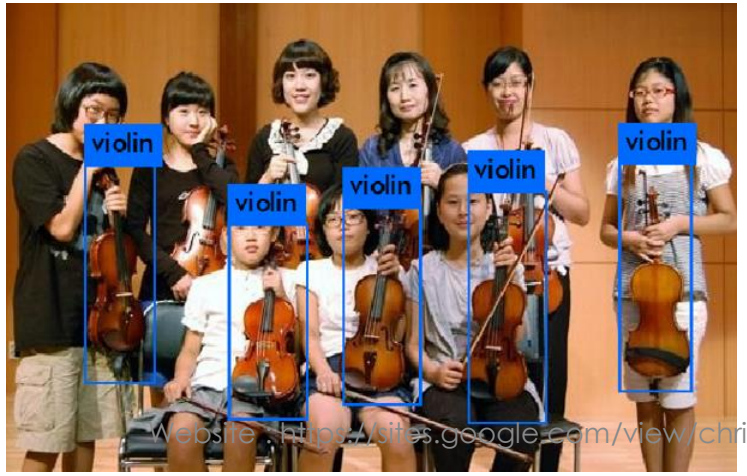


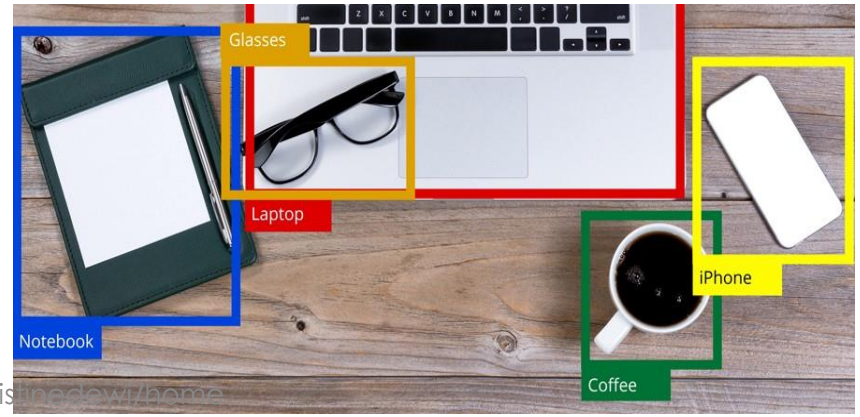
Website : https://sites.google.com/view/christinedewi/home

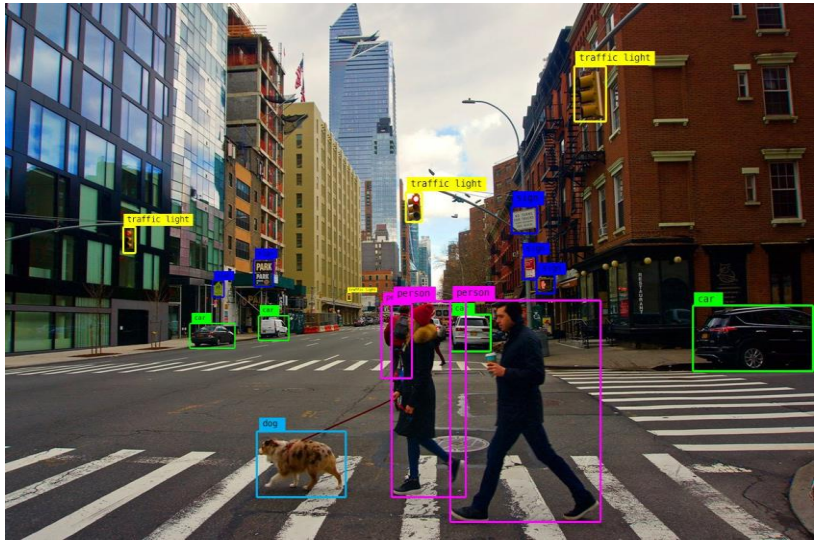
OBJECT EXAMPLE





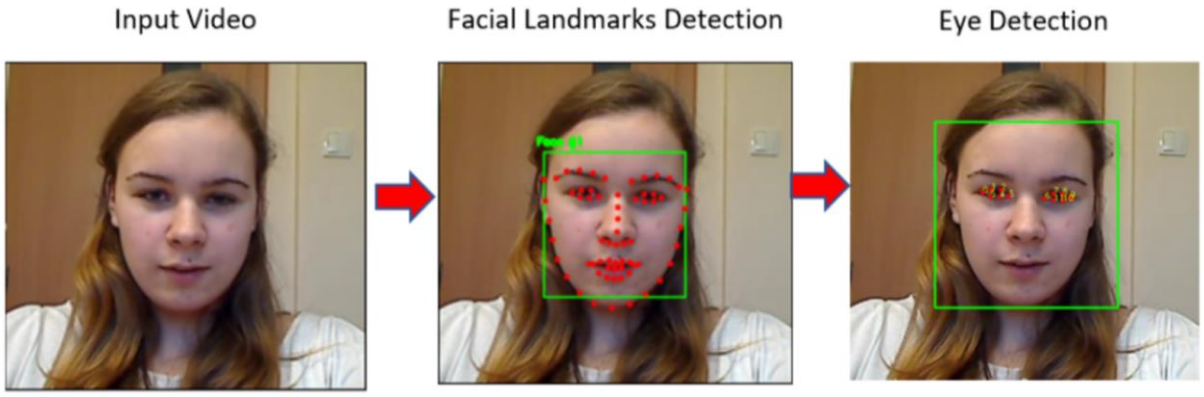


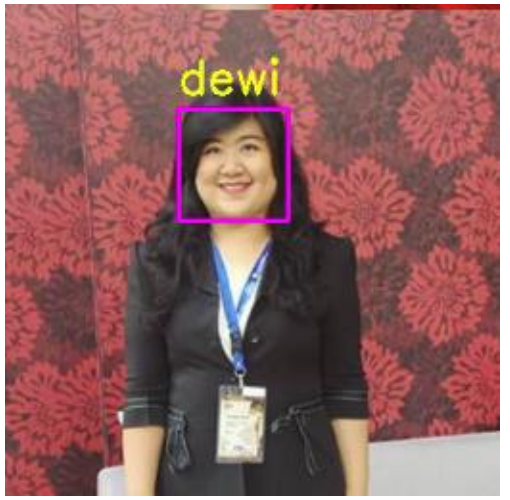


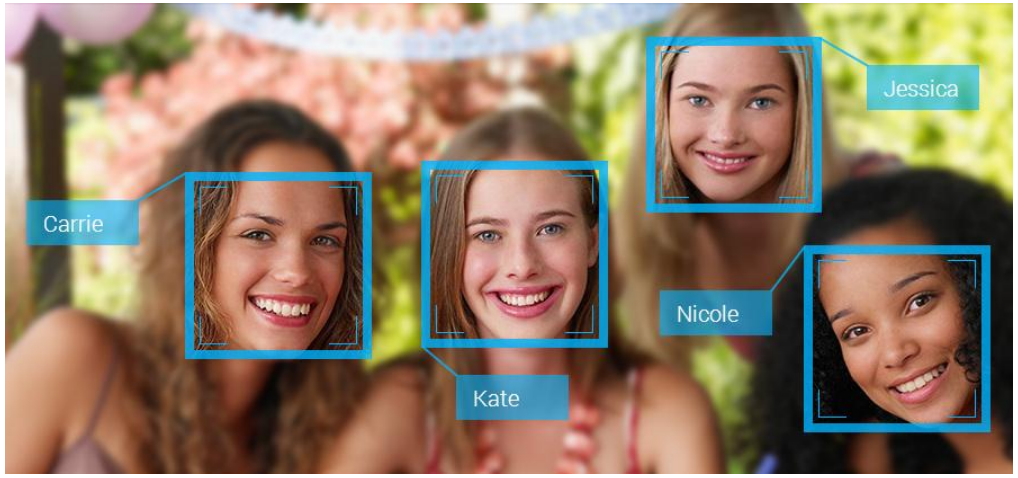


Website : https://sites.google.com/view/christinedewi/home

FACE RECOGNITION



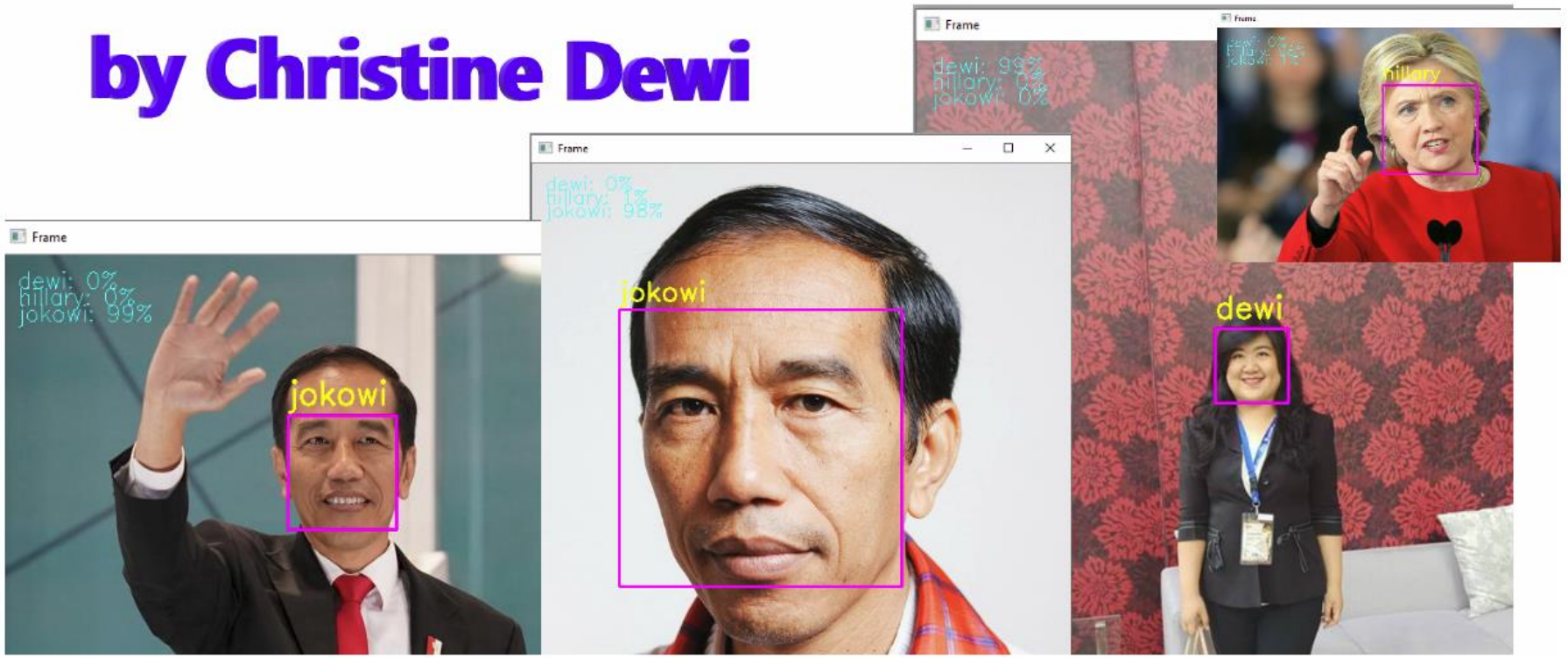




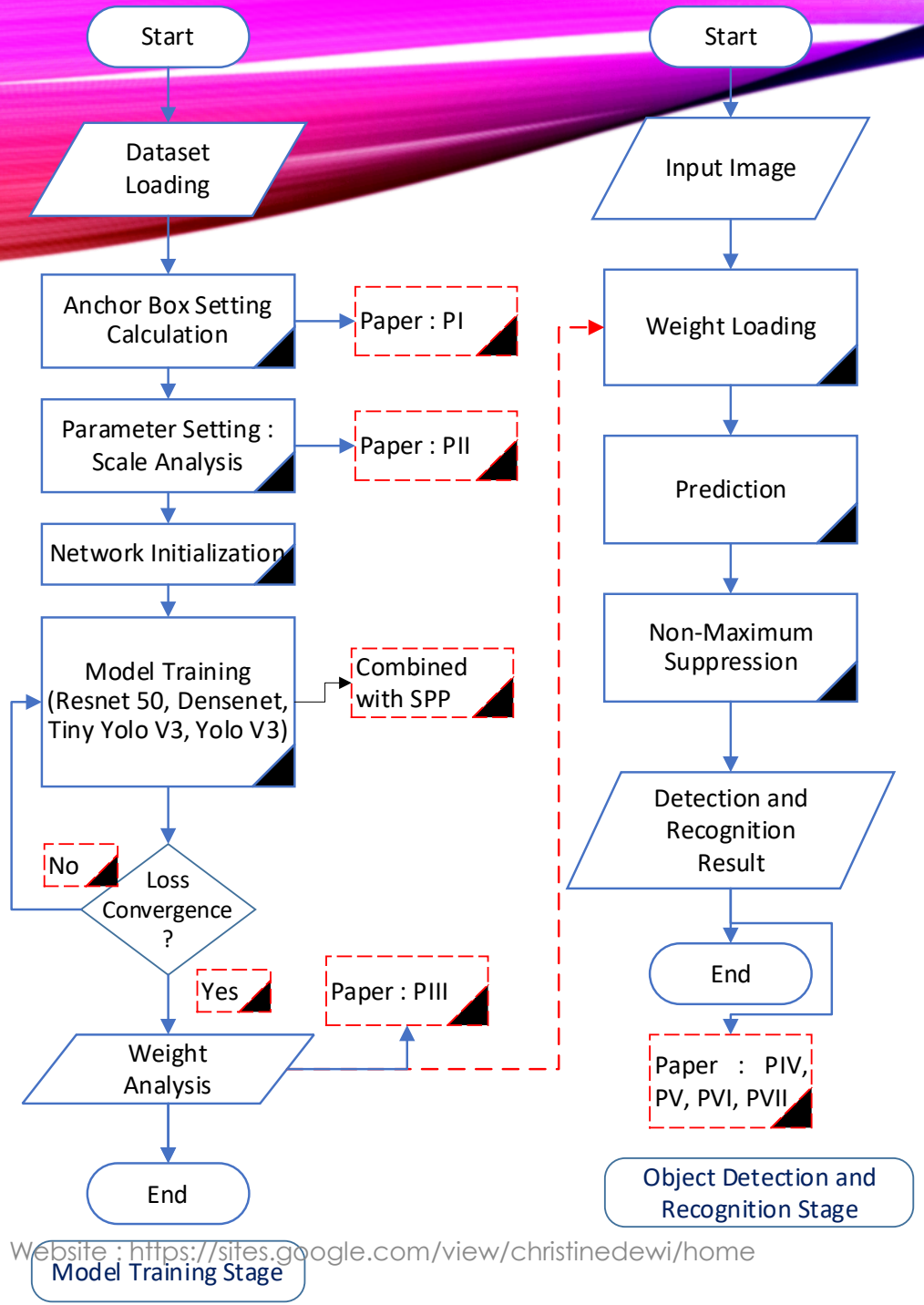
Website : https://sites.google.com/view/christinedewi/home

FACE RECOGNITION

by Christine Dewi



Link: https://www.youtube.com/watch?v=P6YJel0vgoE&list=PLkfg2Q8T49gn45o8_csDnsYK2c26rllzM



OBJECT DETECTION AND RECOGNITION FLOWCHART

1. MODEL TRAINING STAGE
2. OBJECT DETECTION AND RECOGNITION STAGE

MODEL TRAINING STAGE

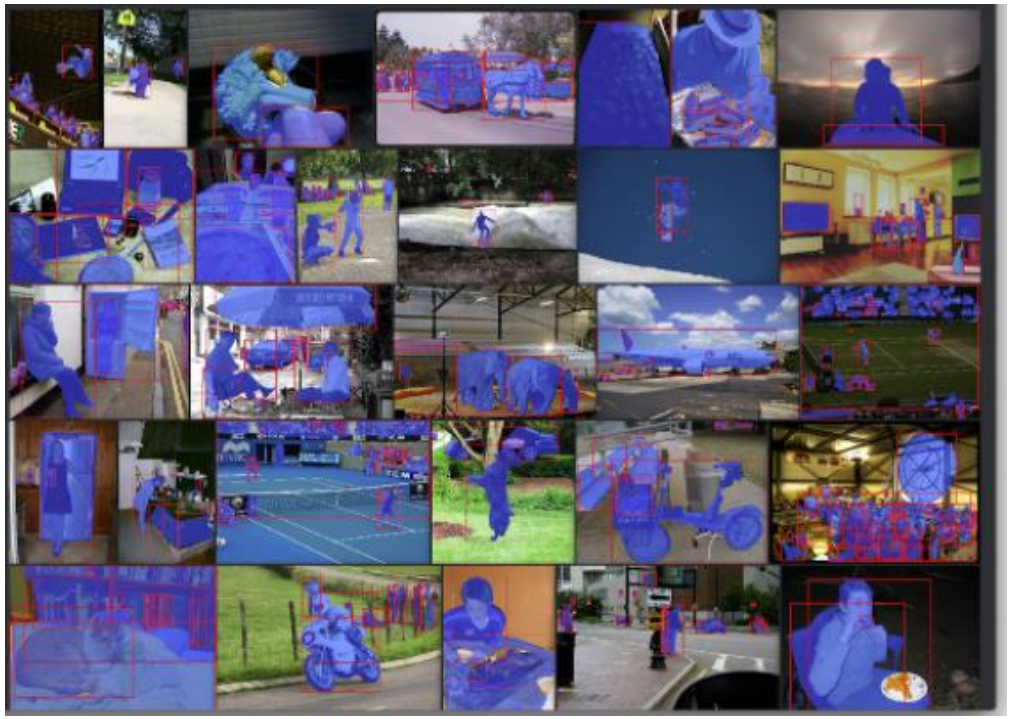
COCO Dataset

COCO is a large-scale object detection, segmentation, and captioning dataset.

COCO has several features:

1. Object segmentation
2. Recognition in context
3. Super pixel stuff segmentation
4. 330K images (>200K labeled)
5. 1.5 million object instances
6. 80 object categories
7. 91 stuff categories
8. 5 captions per image
9. 250,000 people with key points

https://cocodataset.org/#home



COCO DATASET

- coco.names

coco.names

```
1  person
2  bicycle
3  car
4  motorbike
5  aeroplane
6  bus
7  train
8  truck
9  boat
10 traffic light
11 fire hydrant
12 stop sign
13 parking meter
14 bench
15 bird
16 cat
17 dog
18 horse
19 sheep
20 cow
```

coco.names

```
21 elephant
22 bear
23 zebra
24 giraffe
25 backpack
26 umbrella
27 handbag
28 tie
29 suitcase
30 frisbee
31 skis
32 snowboard
33 sports ball
34 kite
35 baseball bat
36 baseball glove
37 skateboard
38 surfboard
39 tennis racket
40 bottle
```

coco.names

```
41 wine glass
42 cup
43 fork
44 knife
45 spoon
46 bowl
47 banana
48 apple
49 sandwich
50 orange
51 broccoli
52 carrot
53 hot dog
54 pizza
55 donut
56 cake
57 chair
58 sofa
59 pottedplant
60 bed
```

coco.names

```
61 diningtable
62 toilet
63 tvmonitor
64 laptop
65 mouse
66 remote
67 keyboard
68 cell phone
69 microwave
70 oven
71 toaster
72 sink
73 refrigerator
74 book
75 clock
76 vase
77 scissors
78 teddy bear
79 hair drier
80 toothbrush
```

COCO DATASET FORMAT

1. Object Detection

The annotations are stored using JSON.

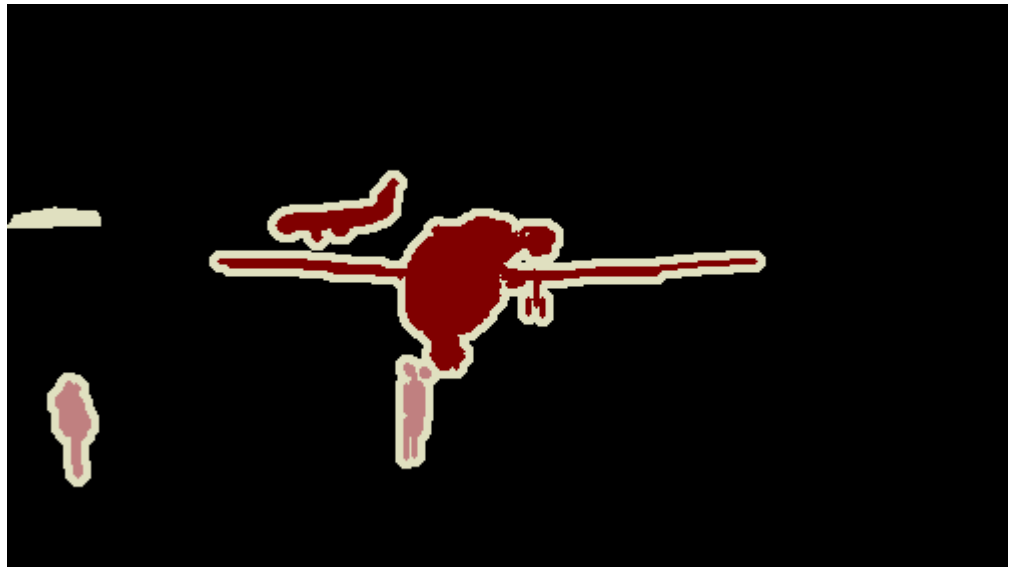
```
annotation{
   "id"                : int,
   "image_id"          : int,
   "category_id"       : int,
   "segmentation"      : RLE or [polygon],
   "area"              : float,
   "bbox"              : [x,y,width,height],
   "iscrowd"           : 0 or 1,
}

categories[{
   "id"                : int,
   "name"              : str,
   "supercategory"     : str,
}]
```

COCO DATASET IMAGES VOC 2007

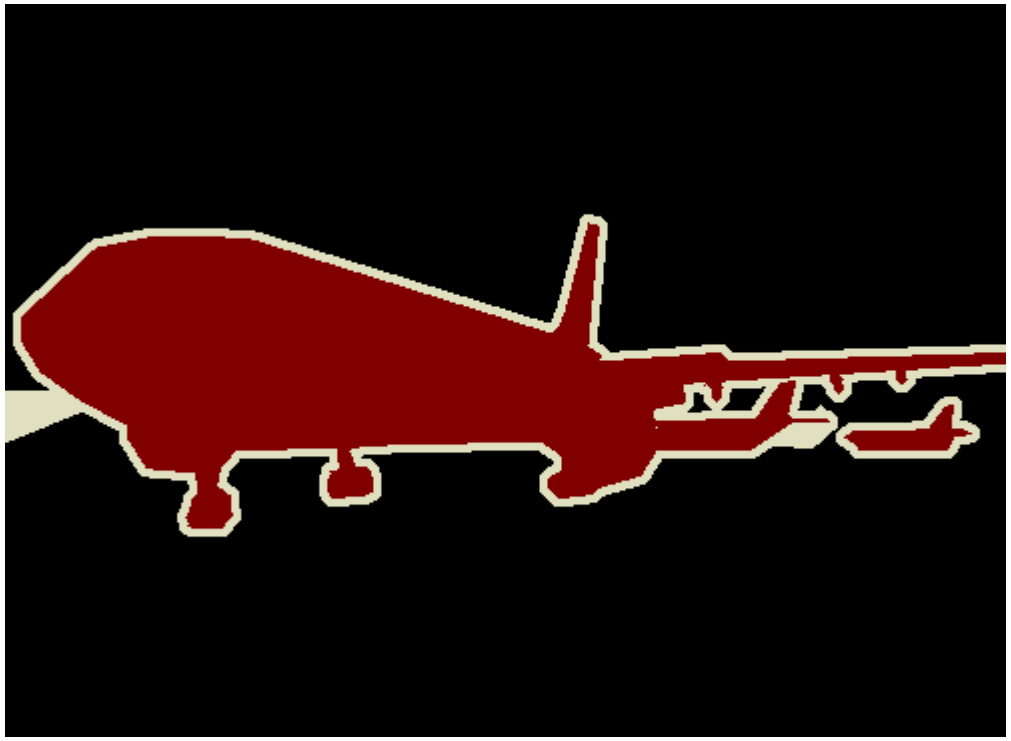
000032.jpg

000033.jpg

SEGMENTATION CLASS

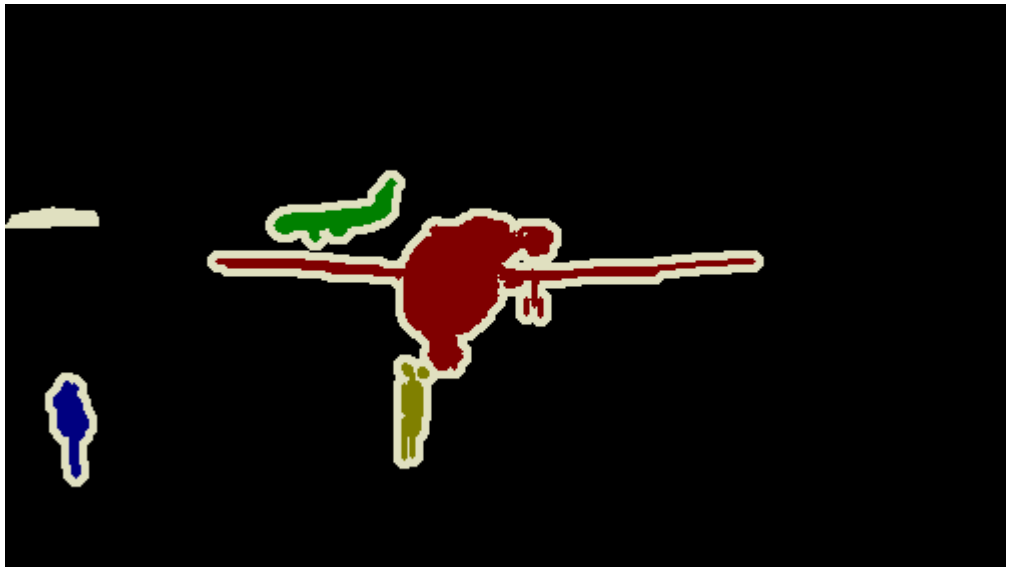


000032.jpg

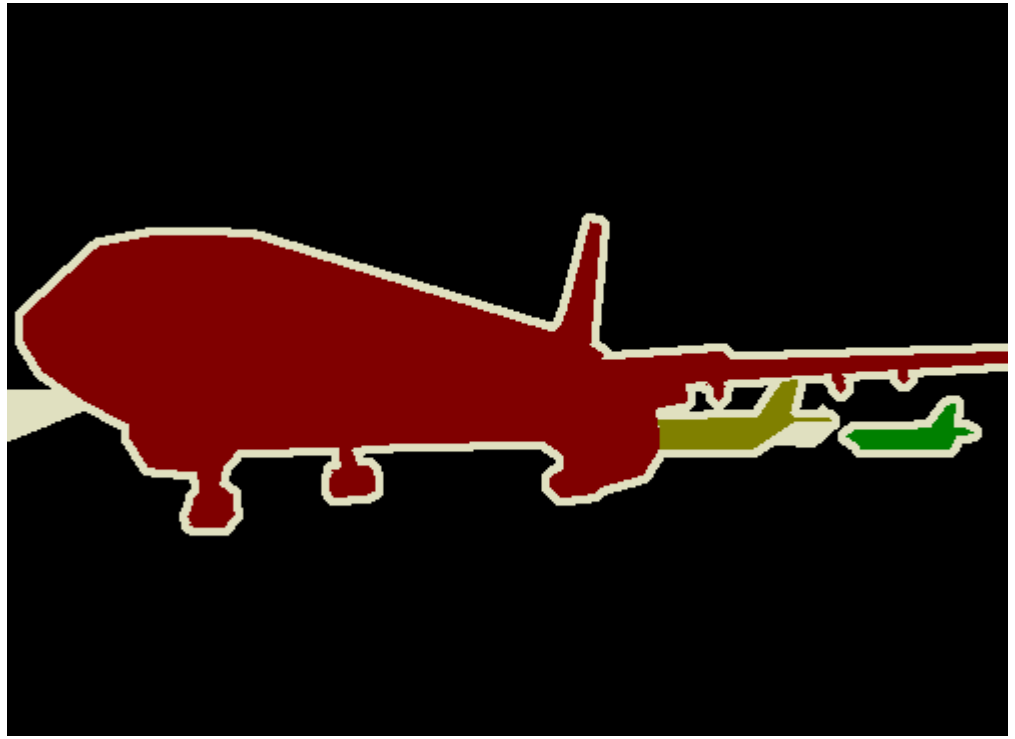


000033.jpg

SEGMENTATION OBJECT



000032.jpg



000033.jpg

朝陽科技大學 CYUT CHAOYANG UNIVERSITY OF TECHNOLOGY

LABELS

000032.txt - Notepad

File Edit Format View Help

```
0 0.4790000000000004 0.46441281138790036 0.542 0.3736654804270462
0 0.33 0.3754448398576512 0.128 0.1245551601423487
14 0.4080000000000003 0.727758007117437 0.03600000000000004
0.17437722419928825
14 0.07 0.759786476683273 0.03600000000000004 0.17437722419928825
```

000033.txt - Notepad

File Edit Format View Help

```
0 0.508 0.505464480874317 0.98 0.426229508196721
0 0.903 0.5819672131147541 0.122 0.07103825136612021
0 0.736 0.561475409836065 0.17200000000000001 0.09562841530054644
```

ANNOTATION

000032.jpg

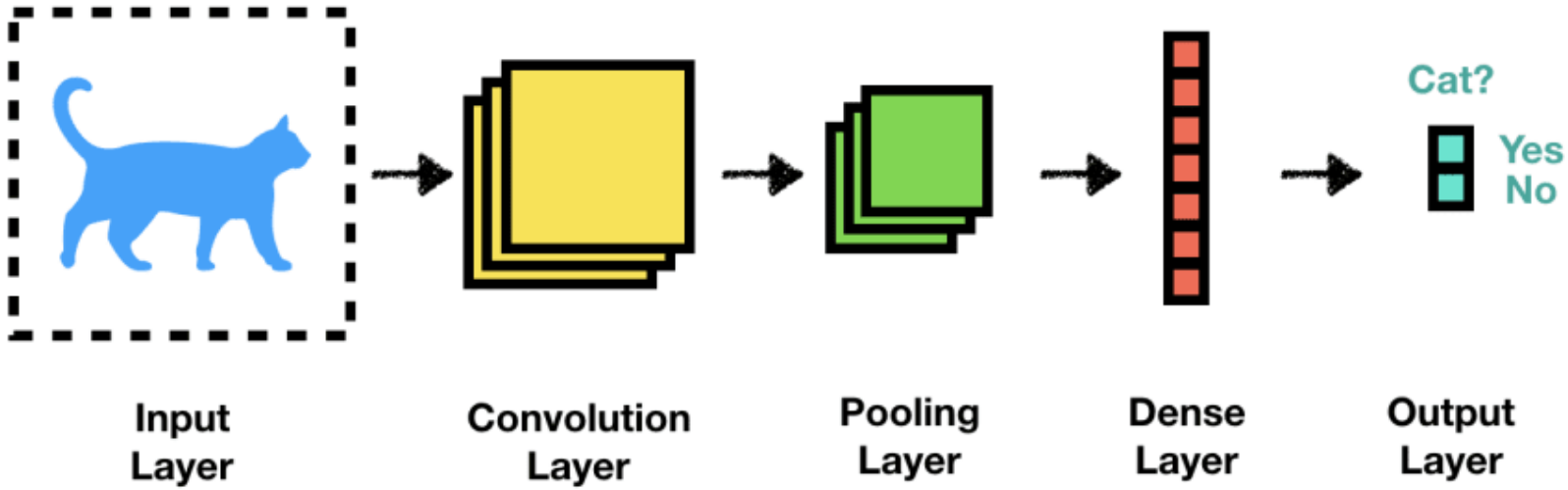
```
<?xml version="1.0"?>
- <annotation>
    <folder>VOC2007</folder>
    <filename>000032.jpg</filename>
  - <source>
      <database>The VOC2007 Database</database>
      <annotation>PASCAL VOC2007</annotation>
      <image>flickr</image>
      <flickrid>311023000</flickrid>
    </source>
  - <owner>
      <flickrid>-hi-no-to-ri-mo-rt-al-</flickrid>
      <name>?</name>
    </owner>
  - <size>
      <width>500</width>
      <height>281</height>
      <depth>3</depth>
    </size>
    <segmented>1</segmented>
  - <object>
      <name>aeroplane</name>
      <pose>Frontal</pose>
      <truncated>0</truncated>
      <difficult>0</difficult>
    - <bndbox>
        <xmin>104</xmin>
        <ymin>78</ymin>
        <xmax>375</xmax>
        <ymax>183</ymax>
      </bndbox>
    </object>
  - <object>
      <name>aeroplane</name>
      <pose>Left</pose>
      <truncated>0</truncated>
      <difficult>0</difficult>
    - <bndbox>
        <xmin>133</xmin>
        <ymin>88</ymin>
        <xmax>197</xmax>
        <ymax>123</ymax>
      </bndbox>
    </object>
  - <object>
      <name>person</name>
      <pose>Rear</pose>
      <truncated>0</truncated>
      <difficult>0</difficult>
    - <bndbox>
        <xmin>195</xmin>
        <ymin>180</ymin>
        <xmax>213</xmax>
        <ymax>229</ymax>
      </bndbox>
    </object>
  - <object>
      <name>person</name>
      <pose>Rear</pose>
      <truncated>0</truncated>
      <difficult>0</difficult>
    - <bndbox>
        <xmin>26</xmin>
        <ymin>189</ymin>
        <xmax>44</xmax>
        <ymax>238</ymax>
      </bndbox>
    </object>
</annotation>
```

000033.jpg

```
<?xml version="1.0"?>
- <annotation>
    <folder>VOC2007</folder>
    <filename>000033.jpg</filename>
  - <source>
      <database>The VOC2007 Database</database>
      <annotation>PASCAL VOC2007</annotation>
      <image>flickr</image>
      <flickrid>336881892</flickrid>
    </source>
  - <owner>
      <flickrid>temp13rec.</flickrid>
      <name>?</name>
    </owner>
  - <size>
      <width>500</width>
      <height>366</height>
      <depth>3</depth>
    </size>
    <segmented>1</segmented>
  - <object>
      <name>aeroplane</name>
      <pose>Unspecified</pose>
      <truncated>0</truncated>
      <difficult>0</difficult>
    - <bndbox>
        <xmin>9</xmin>
        <ymin>107</ymin>
        <xmax>499</xmax>
        <ymax>263</ymax>
      </bndbox>
    </object>
  - <object>
      <name>aeroplane</name>
      <pose>Left</pose>
      <truncated>0</truncated>
      <difficult>0</difficult>
    - <bndbox>
        <xmin>421</xmin>
        <ymin>200</ymin>
        <xmax>482</xmax>
        <ymax>226</ymax>
      </bndbox>
    </object>
  - <object>
      <name>aeroplane</name>
      <pose>Left</pose>
      <truncated>1</truncated>
      <difficult>0</difficult>
    - <bndbox>
        <xmin>325</xmin>
        <ymin>188</ymin>
        <xmax>411</xmax>
        <ymax>223</ymax>
      </bndbox>
    </object>
</annotation>
```

CONVOLUTIONAL NEURAL NETWORK (CNN) MODEL

- A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other.



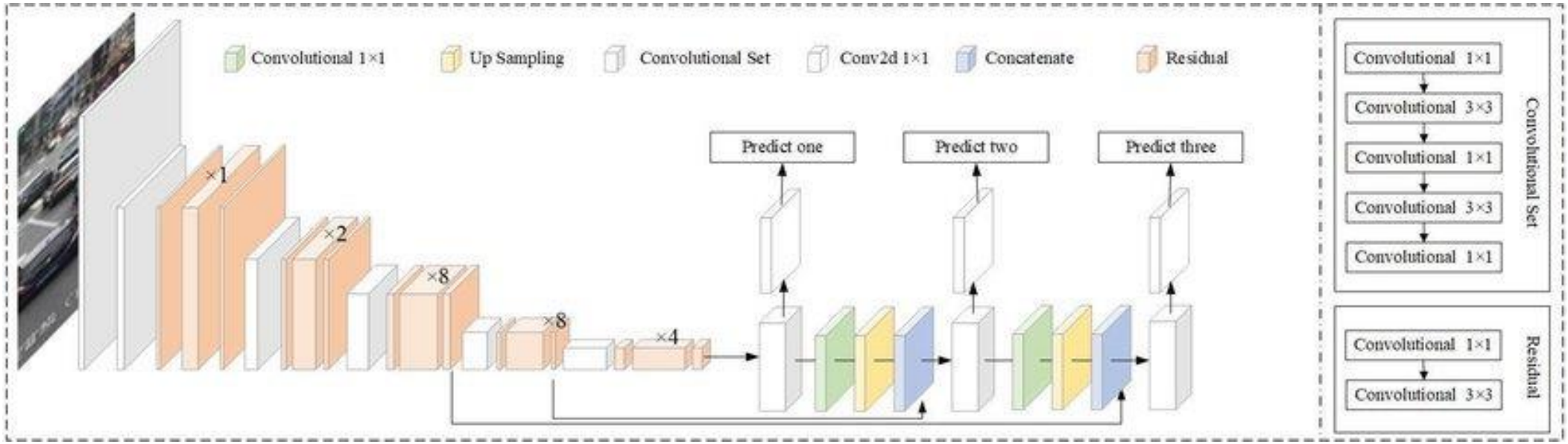
https://towardsdatascience.com/convolutional-neural-network-a-step-by-step-guide-a8b4c88d6943

MODEL CONFIGURATION YOLO

- You only look once (YOLO) is a state-of-the-art, real-time object detection system. On a Pascal Titan X it processes images at 30 FPS and has a mAP of 57.9% on COCO test-dev.
- YOLOv3 is extremely fast and accurate. In mAP measured at .5 IOU YOLOv3 is on par with Focal Loss but about 4x faster. Moreover, you can easily tradeoff between speed and accuracy simply by changing the size of the model, no retraining required!
- YOLOv3 (You Only Look Once, Version 3) is a real-time object detection algorithm that identifies specific objects in videos, live feeds, or images. YOLO uses features learned by a deep convolutional neural network to detect an object. Versions 1-3 of YOLO were created by Joseph Redmon and Ali Farhadi.

Link: https://pjreddie.com/darknet/yolo/

YOLO V3 ARCHITECTURE



YOLO V3 ARCHITECTURE

Layer	Filters		Size				Input				Output			
0	conv	32	3	x	3	/ 1	416	x	416	x 3	->	416	x	416	x 32
1	conv	64	3	x	3	/	416	x	416	x 32	->	208	x	208	x 64
2	conv	32	1	x	1	/ 1	208	x	208	x 64	->	208	x	208	x 32
3	conv	64	3	x	3	/ 1	208	x	208	x 32	->	208	x	208	x 64
4	Shortcut	Layer:	1												
5	conv	128	3	x	3	/	208	x	208	x 64	->	104	x	104	x 128
6	conv	64	1	x	1	/ 1	104	x	104	x 128	->	104	x	104	x 64
7	conv	128	3	x	3	/ 1	104	x	104	x 64	->	104	x	104	x 128
8	Shortcut	Layer:	5												
9	conv	64	1	x	1	/ 1	104	x	104	x 128	->	104	x	104	x 64
10	conv	128	3	x	3	/ 1	104	x	104	x 64	->	104	x	104	x 128
11	Shortcut	Layer:	8												
12	conv	256	3	x	3	/	104	x	104	x 128	->	52	x	52	x 256
13	conv	128	1	x	1	/ 1	52	x	52	x 256	->	52	x	52	x 128
14	conv	256	3	x	3	/ 1	52	x	52	x 128	->	52	x	52	x 256
15	Shortcut	Layer:	12												
16	conv	128	1	x	1	/ 1	52	x	52	x 256	->	52	x	52	x 128
17	conv	256	3	x	3	/ 1	52	x	52	x 128	->	52	x	52	x 256
18	Shortcut	Layer:	15												
19	conv	128	1	x	1	/ 1	52	x	52	x 256	->	52	x	52	x 128
20	conv	256	3	x	3	/ 1	52	x	52	x 128	->	52	x	52	x 256
21	Shortcut	Layer:	18												
22	conv	128	1	x	1	/ 1	52	x	52	x 256	->	52	x	52	x 128
23	conv	256	3	x	3	/ 1	52	x	52	x 128	->	52	x	52	x 256
24	Shortcut	Layer:	21												
25	conv	128	1	x	1	/ 1	52	x	52	x 256	->	52	x	52	x 128
26	conv	256	3	x	3	/ 1	52	x	52	x 128	->	52	x	52	x 256
27	Shortcut	Layer:	24												
28	conv	128	1	x	1	/ 1	52	x	52	x 256	->	52	x	52	x 128
29	conv	256	3	x	3	/ 1	52	x	52	x 128	->	52	x	52	x 256
30	Shortcut	Layer:	27												
31	conv	128	1	x	1	/ 1	52	x	52	x 256	->	52	x	52	x 128
32	conv	256	3	x	3	/ 1	52	x	52	x 128	->	52	x	52	x 256
33	Shortcut	Layer:	30												
34	conv	128	1	x	1	/ 1	52	x	52	x 256	->	52	x	52	x 128
35	conv	256	3	x	3	/ 1	52	x	52	x 128	->	52	x	52	x 256
36	Shortcut	Layer:	33												
37	conv	512	3	x	3	/	52	x	52	x 256	->	26	x	26	x 512
38	conv	256	1	x	1	/ 1	26	x	26	x 512	->	26	x	26	x 256
39	conv	512	3	x	3	/ 1	26	x	26	x 256	->	26	x	26	x 512
40	Shortcut	Layer:	37												
41	conv	256	1	x	1	/ 1	26	x	26	x 512	->	26	x	26	x 256
42	conv	512	3	x	3	/ 1	26	x	26	x 256	->	26	x	26	x 512
43	Shortcut	Layer:	40												
44	conv	256	1	x	1	/ 1	26	x	26	x 512	->	26	x	26	x 256
45	conv	512	3	x	3	/ 1	26	x	26	x 256	->	26	x	26	x 512
46	Shortcut	Layer:	43												
47	conv	256	1	x	1	/ 1	26	x	26	x 512	->	26	x	26	x 256
48	conv	512	3	x	3	/ 1	26	x	26	x 256	->	26	x	26	x 512
49	Shortcut	Layer:	46												
50	conv	256	1	x	1	/ 1	26	x	26	x 512	->	26	x	26	x 256
51	conv	512	3	x	3	/ 1	26	x	26	x 256	->	26	x	26	x 512
52	Shortcut	Layer:	49												
53	conv	256	1	x	1	/ 1	26	x	26	x 512	->	26	x	26	x 256
54	conv	512	3	x	3	/ 1	26	x	26	x 256	->	26	x	26	x 512
55	Shortcut	Layer:	52												
56	conv	256	1	x	1	/ 1	26	x	26	x 512	->	26	x	26	x 256
57	conv	512	3	x	3	/ 1	26	x	26	x 256	->	26	x	26	x 512
58	Shortcut	Layer:	55												
59	conv	256	1	x	1	/ 1	26	x	26	x 512	->	26	x	26	x 256
60	conv	512	3	x	3	/ 1	26	x	26	x 256	->	26	x	26	x 512
61	Shortcut	Layer:	58												
62	conv	1024	3	x	3	/ 2	26	x	26	x 512	->	13	x	13	x 1024
63	conv	512	1	x	1	/ 1	13	x	13	x 1024	->	13	x	13	x 512
64	conv	1024	3	x	3	/ 1	13	x	13	x 512	->	13	x	13	x 1024
65	Shortcut	Layer:	62												
66	conv	512	1	x	1	/ 1	13	x	13	x 1024	->	13	x	13	x 512
67	conv	1024	3	x	3	/ 1	13	x	13	x 512	->	13	x	13	x 1024
68	Shortcut	Layer:	65												
69	conv	512	1	x	1	/ 1	13	x	13	x 1024	->	13	x	13	x 512
70	conv	1024	3	x	3	/ 1	13	x	13	x 512	->	13	x	13	x 1024
71	Shortcut	Layer:	68												
72	conv	512	1	x	1	/ 1	13	x	13	x 1024	->	13	x	13	x 512
73	conv	1024	3	x	3	/ 1	13	x	13	x 512	->	13	x	13	x 1024
74	Shortcut	Layer:	71												
75	conv	512	1	x	1	/ 1	13	x	13	x 1024	->	13	x	13	x 512
76	conv	1024	3	x	3	/ 1	13	x	13	x 512	->	13	x	13	x 1024
77	conv	512	1	x	1	/ 1	13	x	13	x 1024	->	13	x	13	x 512
78	conv	1024	3	x	3	/ 1	13	x	13	x 512	->	13	x	13	x 1024
79	conv	512	1	x	1	/ 1	13	x	13	x 1024	->	13	x	13	x 512
80	conv	1024	3	x	3	/ 1	13	x	13	x 512	->	13	x	13	x 1024
81	conv	18	1	x	1	/ 1	13	x	13	x 1024	->	13	x	13	x 18
82	detection														
83	route	79													
84	conv	256	1	x	1	/ 1	13	x	13	x 512	->	13	x	13	x 256
85	upsample	2x	13	x	13	x 256	->	26	x	26	x 256				
86	route	85	61												
87	conv	256	1	x	1	/ 1	26	x	26	x 768	->	26	x	26	x 256
88	conv	512	3	x	3	/ 1	26	x	26	x 256	->	26	x	26	x 512
89	conv	256	1	x	1	/ 1	26	x	26	x 512	->	26	x	26	x 256
90	conv	512	3	x	3	/ 1	26	x	26	x 256	->	26	x	26	x 512
91	conv	256	1	x	1	/ 1	26	x	26	x 512	->	26	x	26	x 256
92	conv	512	3	x	3	/ 1	26	x	26	x 256	->	26	x	26	x 512
93	conv	18	1	x	1	/ 1	26	x	26	x 512	->	26	x	26	x 18
94	detection														
95	route	91													
96	conv	128	1	x	1	/ 1	26	x	26	x 256	->	26	x	26	x 128
97	upsample	2x	26	x	26	x 128	->	52	x	52	x 128				
98	route	97	36												
99	conv	128	1	x	1	/ 1	52	x	52	x 384	->	52	x	52	x 128
100	conv	256	3	x	3	/ 1	52	x	52	x 128	->	52	x	52	x 256
101	conv	128	1	x	1	/ 1	52	x	52	x 256	->	52	x	52	x 128
102	conv	256	3	x	3	/ 1	52	x	52	x 128	->	52	x	52	x 256
103	conv	128	1	x	1	/ 1	52	x	52	x 256	->	52	x	52	x 128
104	conv	256	3	x	3	/ 1	52	x	52	x 128	->	52	x	52	x 256
105	conv	18	1	x	1	/ 1	52	x	52	x 256	->	52	x	52	x 18
106	detection														

YOLO CONFIG FILE

yolov3.cfg

```
[net]
# Testing
# batch=1
# subdivisions=1
# Training
batch=64
subdivisions=16
width=608
height=608
channels=3
momentum=0.9
decay=0.0005
angle=0
saturation = 1.5
exposure = 1.5
hue=.1

learning_rate=0.001
burn_in=1000
max_batches = 500200
policy=steps
steps=400000,450000
scales=.1,.1

[convolutional]
batch_normalize=1
filters=32
size=3
stride=1
pad=1
activation=leaky

# Downsample

[convolutional]
batch_normalize=1
filters=64
size=3
stride=2
pad=1
activation=leaky
```

yolov3.cfg

```

[convolutional]
batch_normalize=1
filters=32
size=1
stride=1
pad=1
activation=leaky

[convolutional]
batch_normalize=1
filters=64
size=3
stride=1
pad=1
activation=leaky

[shortcut]
from=-3
activation=linear

# Downsample

[convolutional]
batch_normalize=1
filters=128
size=3
stride=2
pad=1
activation=leaky

[convolutional]
batch_normalize=1
filters=64
size=1
stride=1
pad=1
activation=leaky
```

yolov3.cfg

```
filters=128
size=1
stride=1
pad=1
activation=leaky

[convolutional]
batch_normalize=1
size=3
stride=1
pad=1
filters=256
activation=leaky

[convolutional]
batch_normalize=1
filters=128
size=1
stride=1
pad=1
activation=leaky

[convolutional]
batch_normalize=1
size=3
stride=1
pad=1
filters=256
activation=leaky

[convolutional]
size=1
stride=1
pad=1
filters=255
activation=linear


[yolo]
mask = 0,1,2
anchors = 10,13,  16,30,  33,23,  30,61,  62,45,  59,119,  116,90,  156,198,  373,326
classes=80
num=9
jitter=.3
ignore_thresh = .7
truth_thresh = 1
random=1
```

Filters = (Classes + 5)*3

Filters = (80 + 5)*3 = 255

Class = 80

YOLO WEIGHT FILE

https://pjreddie.com/darknet/yolo/

Performance on the COCO Dataset

Model	Train	Test	mAP	FLOPS	FPS	Cfg	Weights
SSD300	COCO trainval	test-dev	41.2	-	46		link
SSD500	COCO trainval	test-dev	46.5	-	19		link
YOLOv2 608x608	COCO trainval	test-dev	48.1	62.94 Bn	40	cfg	weights
Tiny YOLO	COCO trainval	test-dev	23.7	5.41 Bn	244	cfg	weights
SSD321	COCO trainval	test-dev	45.4	-	16		link
DSSD321	COCO trainval	test-dev	46.1	-	12		link
R-FCN	COCO trainval	test-dev	51.9	-	12		link
SSD513	COCO trainval	test-dev	50.4	-	8		link
DSSD513	COCO trainval	test-dev	53.3	-	6		link
FPN FRCN	COCO trainval	test-dev	59.1	-	6		link
Retinanet-50-500	COCO trainval	test-dev	50.9	-	14		link
Retinanet-101-500	COCO trainval	test-dev	53.1	-	11		link
Retinanet-101-800	COCO trainval	test-dev	57.5	-	5		link
YOLOv3-320	COCO trainval	test-dev	51.5	38.97 Bn	45	cfg	weights
YOLOv3-416	COCO trainval	test-dev	55.3	65.86 Bn	35	cfg	weights
YOLOv3-608	COCO trainval	test-dev	57.9	140.69 Bn	20	cfg	weights
YOLOv3-tiny	COCO trainval	test-dev	33.1	5.56 Bn	220	cfg	weights
YOLOv3-spp	COCO trainval	test-dev	60.6	141.45 Bn	20	cfg	weights

EXPERIMENT ENVIRONMENT

Spyder

↗ 3.3.6

Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

Notebook

↗ 6.0.1

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

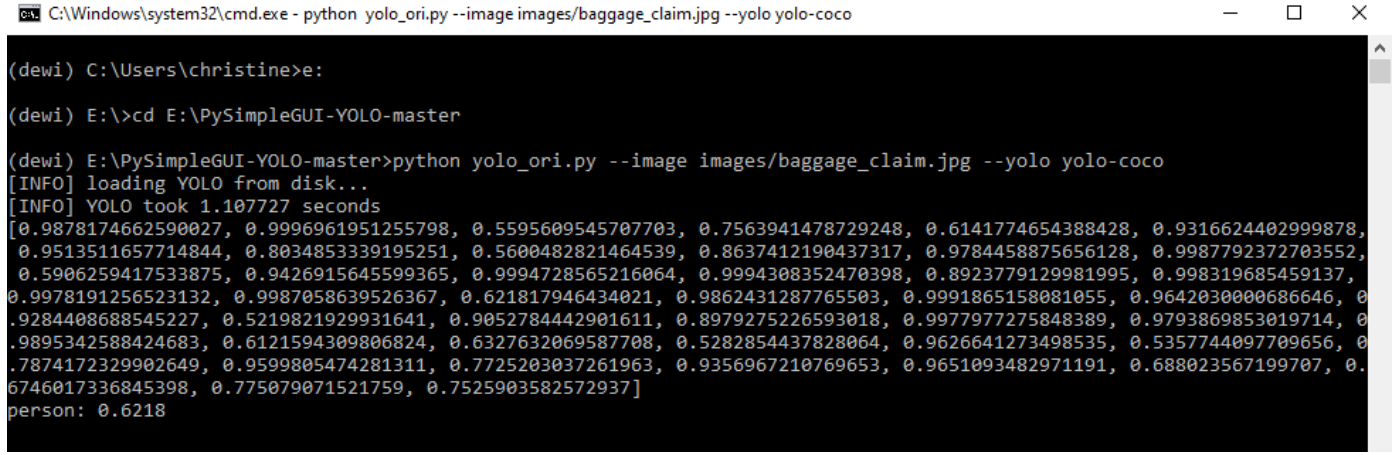
Website : https://sites.google.com/view/christinedewi/home

YOLO.PY

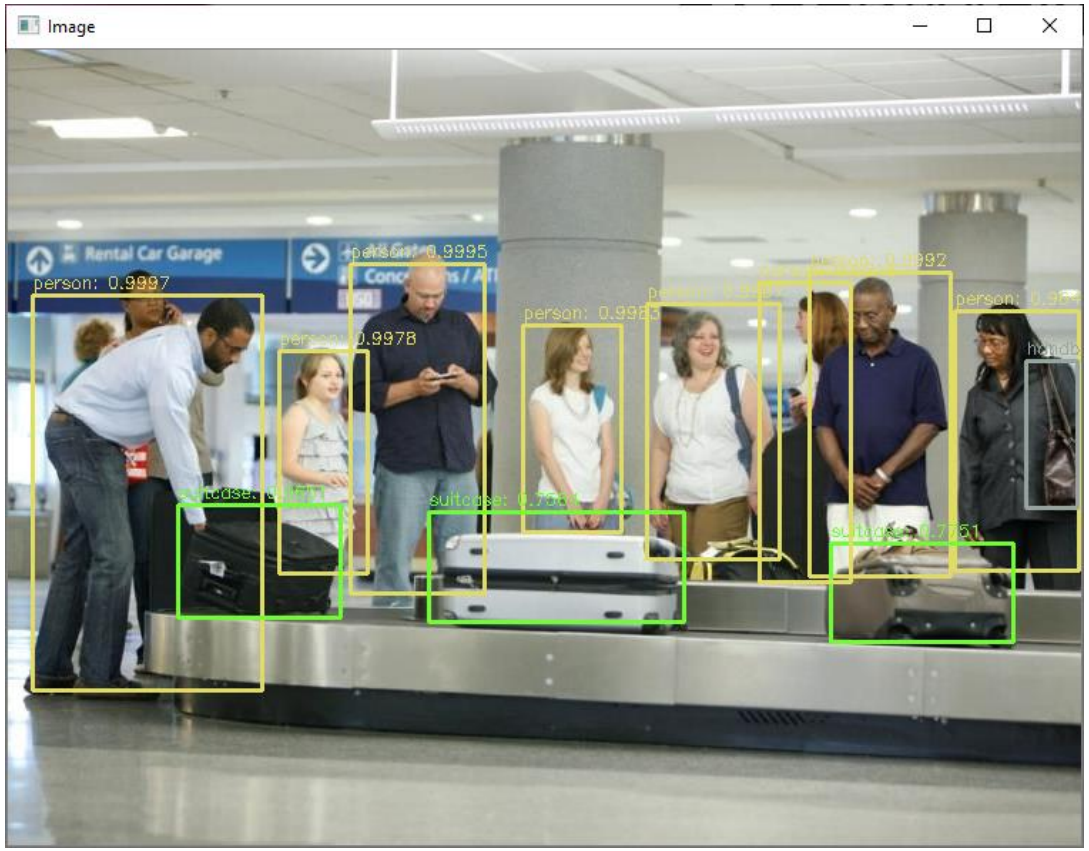
yolo.py

```
# USAGE
# python yolo.py --image images/baggage_claim.jpg --yolo yolo-coco

# import the necessary packages
import numpy as np
import argparse
import time
import cv2
import os

# construct the argument parse and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-i", "--image", required=True,
    help="path to input image")
ap.add_argument("-y", "--yolo", required=True,
    help="base path to YOLO directory")
ap.add_argument("-c", "--confidence", type=float, default=0.5,
    help="minimum probability to filter weak detections")
ap.add_argument("-t", "--threshold", type=float, default=0.3,
    help="threshold when applyong non-maxima suppression")
args = vars(ap.parse_args())

# load the COCO class labels our YOLO model was trained on
labelsPath = os.path.sep.join([args["yolo"], "coco.names"])
LABELS = open(labelsPath).read().strip().split("\n")

# initialize a list of colors to represent each possible class label
np.random.seed(42)
COLORS = np.random.randint(0, 255, size=(len(LABELS), 3),
    dtype="uint8")
```

EXPERIMENT IMAGE 1

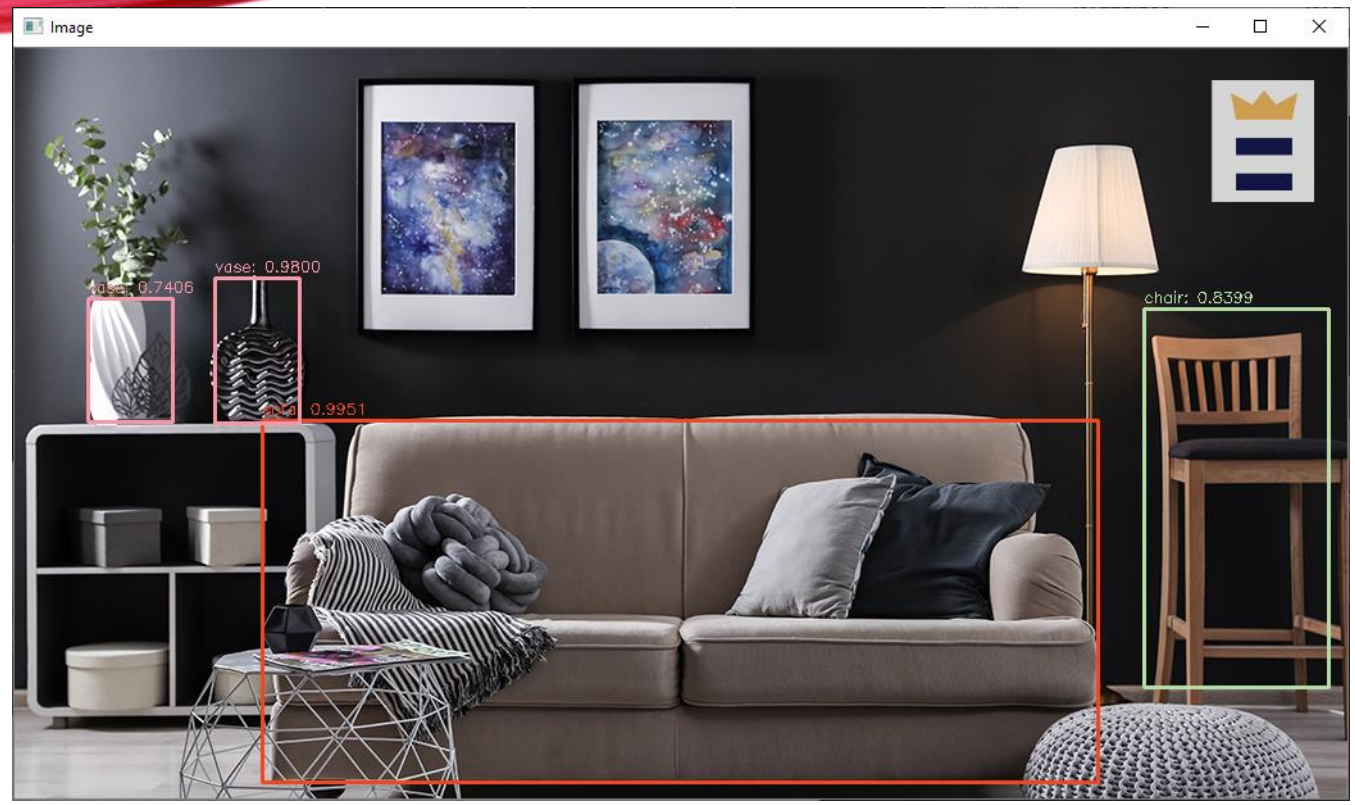


(dewi) E:\PySimpleGUI-YOLO-master>python yolo_ori.py --image images/baggage_claim.jpg --yolo yolo-coco
[INFO] loading YOLO from disk...
[INFO] YOLO took 1.107727 seconds



朝陽科技大學 CYUT CHAOYANG UNIVERSITY OF TECHNOLOGY

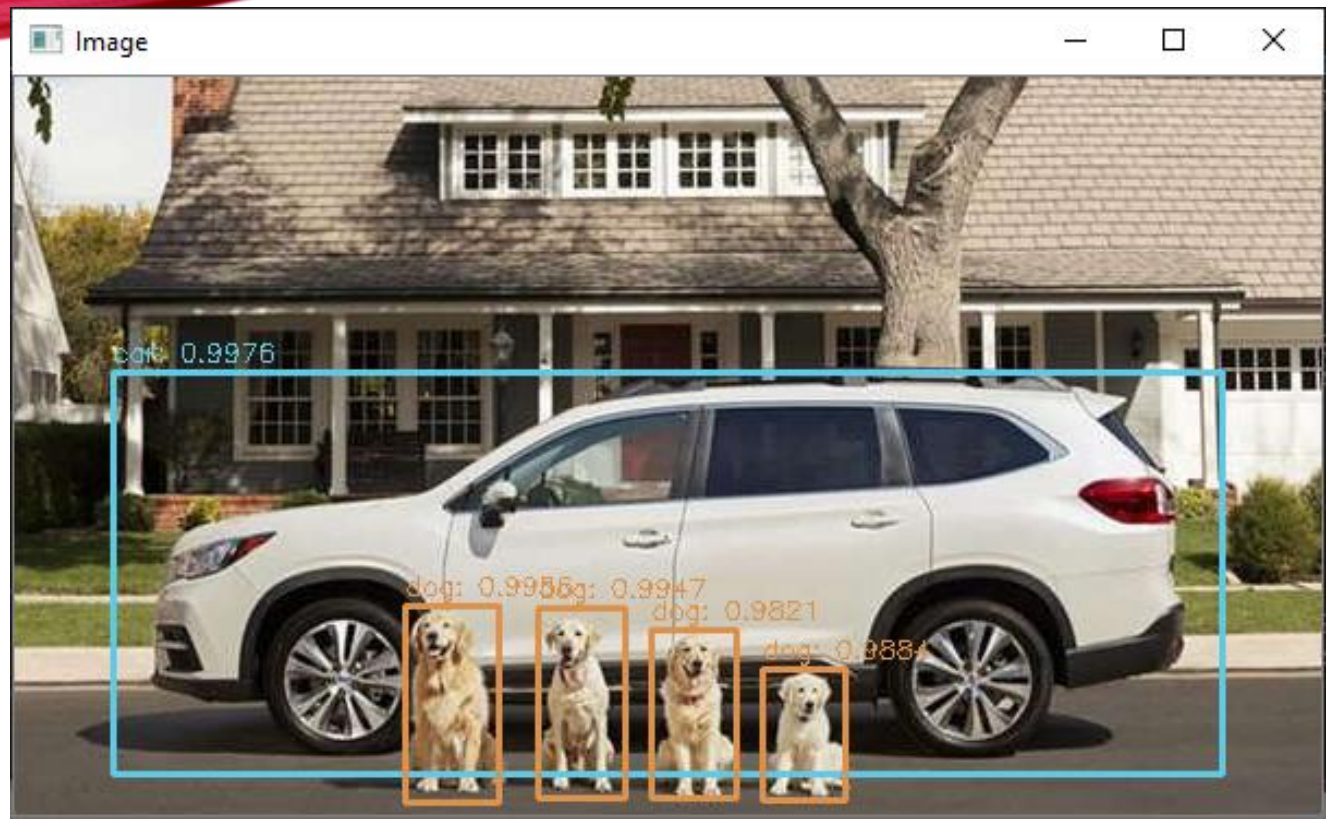
EXPERIMENT IMAGE 2



(dewi) E:\PySimpleGUI-YOLO-master>python yolo_ori.py --image images/living.jpg --yolo yolo-coco
[INFO] loading YOLO from disk...
[INFO] YOLO took 0.873198 seconds
[0.8048667311668396, 0.6233654022216797, 0.6938741207122803, 0.9951150417327881, 0.9745580554008484, 0.9139105081558228, 0.7007461786270142, 0.7405785918235779, 0.979971706867218, 0.5460813641548157, 0.8398553133010864, 0.6024660468101501, 0.658711314201355]
vase: 0.7406

```
(dewi) E:\PySimpleGUI-YOLO-master>python yolo_ori.py --image images/living.jpg --yolo yolo-coco
[INFO] loading YOLO from disk...
[INFO] YOLO took 0.873198 seconds
[0.8048667311668396, 0.6233654022216797, 0.6938741207122803, 0.9951150417327881, 0.9745580554008484, 0.9139105081558228,
 0.7007461786270142, 0.7405785918235779, 0.979971706867218, 0.5460813641548157, 0.8398553133010864, 0.6024660468101501,
0.658711314201355]
vase: 0.7406
```

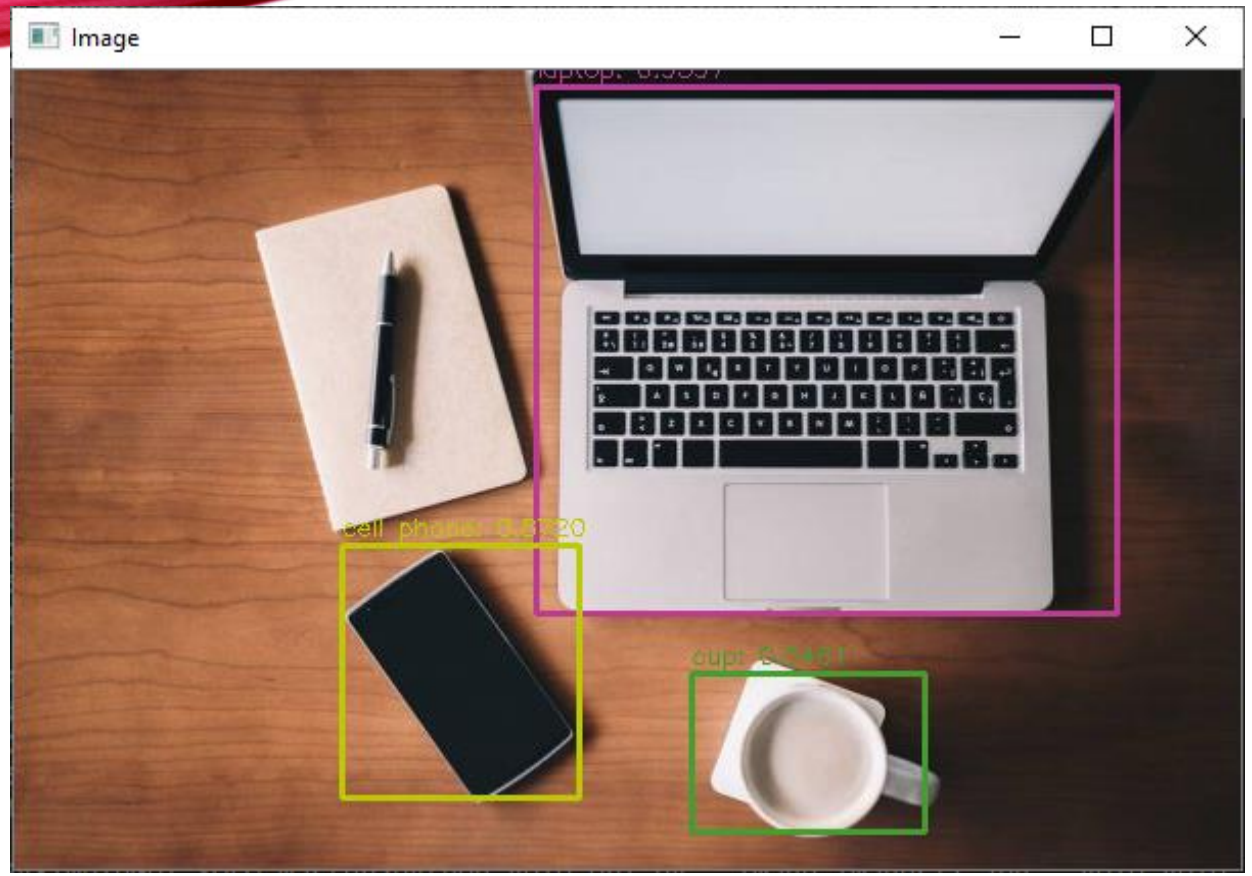
EXPERIMENT IMAGE 3




```
(dewi) E:\PySimpleGUI-YOLO-master>python yolo_ori.py --image images/dog.jpg --yolo yolo-coco
[INFO] loading YOLO from disk...
[INFO] YOLO took 0.809351 seconds
[0.9336661696434021, 0.752363383769989, 0.9975638389587402, 0.8648309707641602, 0.9368127584457397, 0.9505293369293213,
0.9734814167022705, 0.9436767101287842, 0.995621383190155, 0.9939004182815552, 0.994735062122345, 0.7475429773330688, 0.
9820849299430847, 0.749684751033783, 0.9884141683578491]
dog: 0.9821
```

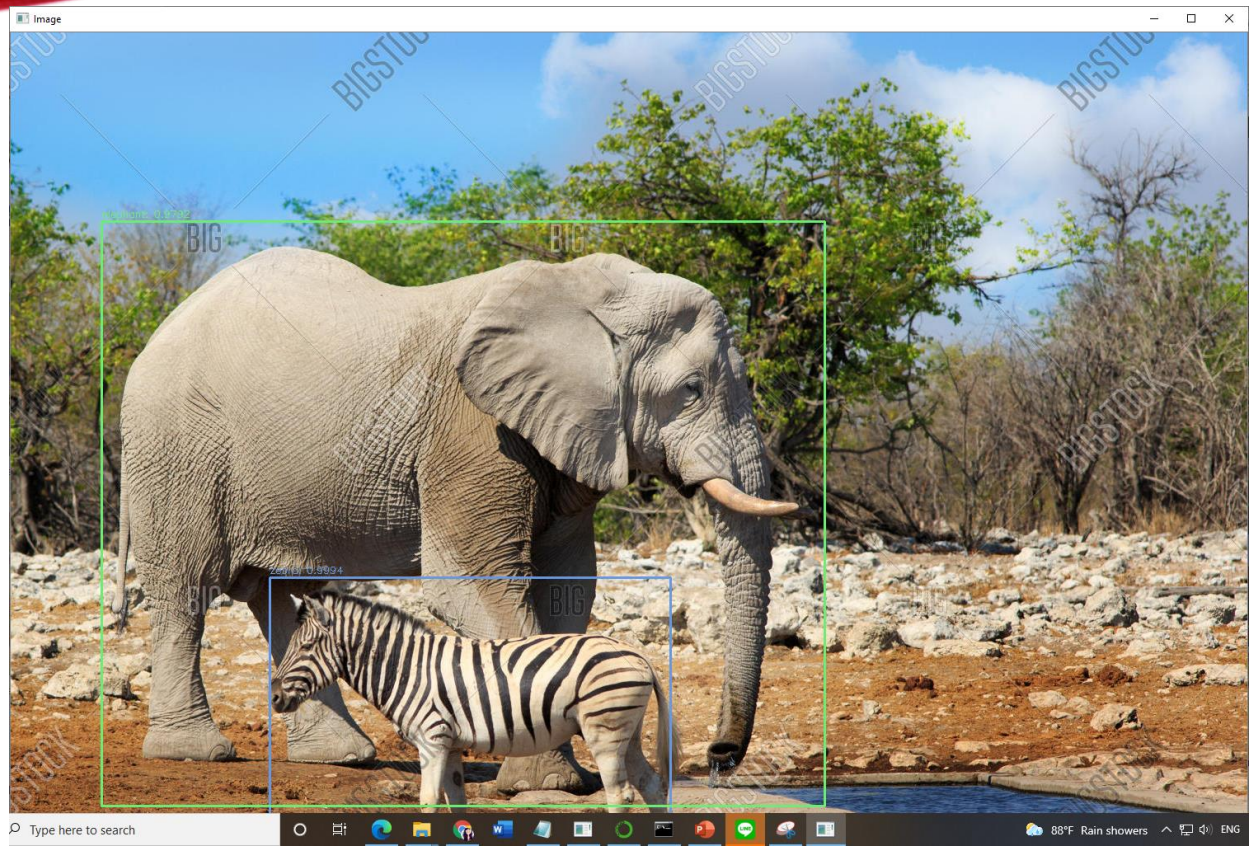

朝陽科技大學 CYUT CHAOYANG UNIVERSITY OF TECHNOLOGY

EXPERIMENT IMAGE 4



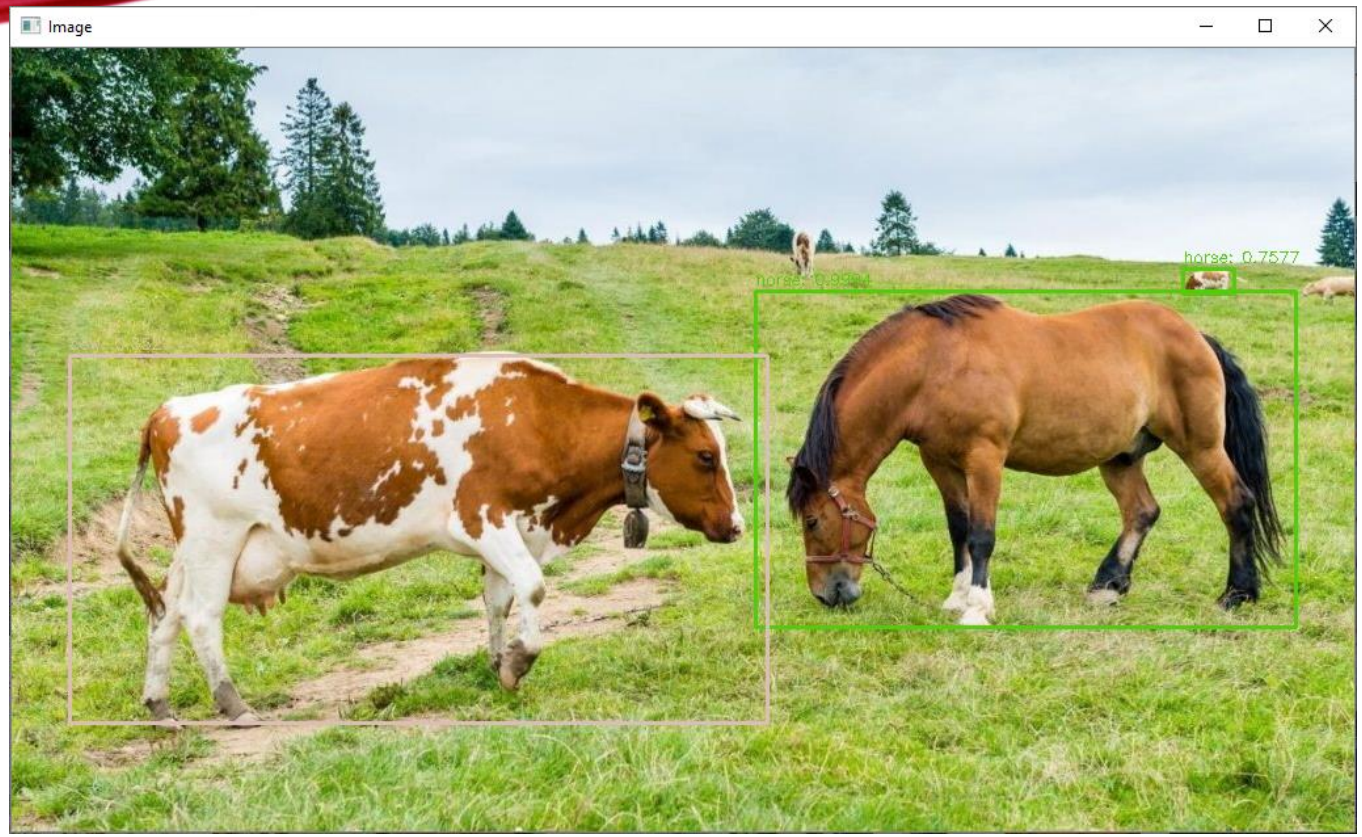

```
(dewi) E:\PySimpleGUI-YOLO-master>python yolo_ori.py --image images/1.jpg --yolo yolo-coco
[INFO] loading YOLO from disk...
[INFO] YOLO took 0.496559 seconds
[0.9597240090370178, 0.8608099818229675, 0.6954886317253113, 0.9011364579200745, 0.6049672365188599, 0.5460721254348755,
 0.8720163106918335, 0.5228443145751953]
cup: 0.5461
```

EXPERIMENT IMAGE 5




```
(dewi) E:\PySimpleGUI-YOLO-master>python yolo_ori.py --image images/zebra.jpg --yolo yolo-coco
[INFO] loading YOLO from disk...
[INFO] YOLO took 0.488465 seconds
[0.7991313338279724, 0.5297073721885681, 0.9792222380638123, 0.949263334274292, 0.9315980076789856, 0.8338237404823303,
0.9975195527076721, 0.9993630051612854, 0.9408066272735596, 0.9910464286804199]
elephant: 0.9792
```

EXPERIMENT IMAGE 6




```
(dewi) E:\PySimpleGUI-YOLO-master>python yolo_ori.py --image images/cow.jpg --yolo yolo-coco
[INFO] loading YOLO from disk...
[INFO] YOLO took 0.519670 seconds
[0.9993814826011658, 0.980201244354248, 0.8113096356391907, 0.6342653036117554, 0.9971794486045837, 0.8210050463676453,
0.8526701331138611, 0.5985972881317139, 0.7576660513877869, 0.7146040797233582]
horse: 0.7577
```

YOLO_VIDEO.PY

yolo.py | yolo_video.py

```python
# USAGE
# python yolo_video.py --input videos/airport.mp4 --output output/airport_outp

# import the necessary packages
import numpy as np
import argparse
import imutils
import time
import cv2
import os

# construct the argument parse and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-i", "--input", required=True,
    help="path to input video")
ap.add_argument("-o", "--output", required=True,
    help="path to output video")
ap.add_argument("-y", "--yolo", required=True,
    help="base path to YOLO directory")
ap.add_argument("-c", "--confidence", type=float, default=0.5,
    help="minimum probability to filter weak detections")
ap.add_argument("-t", "--threshold", type=float, default=0.3,
    help="threshold when applyong non-maxima suppression")
args = vars(ap.parse_args())

# load the COCO class labels our YOLO model was trained on
labelsPath = os.path.sep.join([args["yolo"], "coco.names"])
LABELS = open(labelsPath).read().strip().split("\n")

# initialize a list of colors to represent each possible class label
np.random.seed(42)
COLORS = np.random.randint(0, 255, size=(len(LABELS), 3),
    dtype="uint8")

# derive the paths to the YOLO weights and model configuration
weightsPath = os.path.sep.join([args["yolo"], "yolov3.weights"])
configPath = os.path.sep.join([args["yolo"], "yolov3.cfg"])
```

EXPERIMENT VIDEO 1

```
(dewi) E:\PySimpleGUI-YOLO-master>python yolo_video_save.py --input input_avi1.mp4 --output outputavi1.avi --yolo yolo-coco
[INFO] loading YOLO from disk...
[INFO] 5406 total frames in video
[INFO] single frame took 0.6460 seconds
[INFO] estimated total time to finish: 3492.1801
[INFO] cleaning up...

(dewi) E:\PySimpleGUI-YOLO-master>
```

(dewi) E:\PySimpleGUI-YOLO-master>python yolo_video_save.py --input input_avi1.mp4 --output outputavi1.avi --yolo yolo-coco
[INFO] loading YOLO from disk...
[INFO] 5406 total frames in video
[INFO] single frame took 0.6460 seconds
[INFO] estimated total time to finish: 3492.1801
[INFO] cleaning up...
(dewi) E:\PySimpleGUI-YOLO-master>

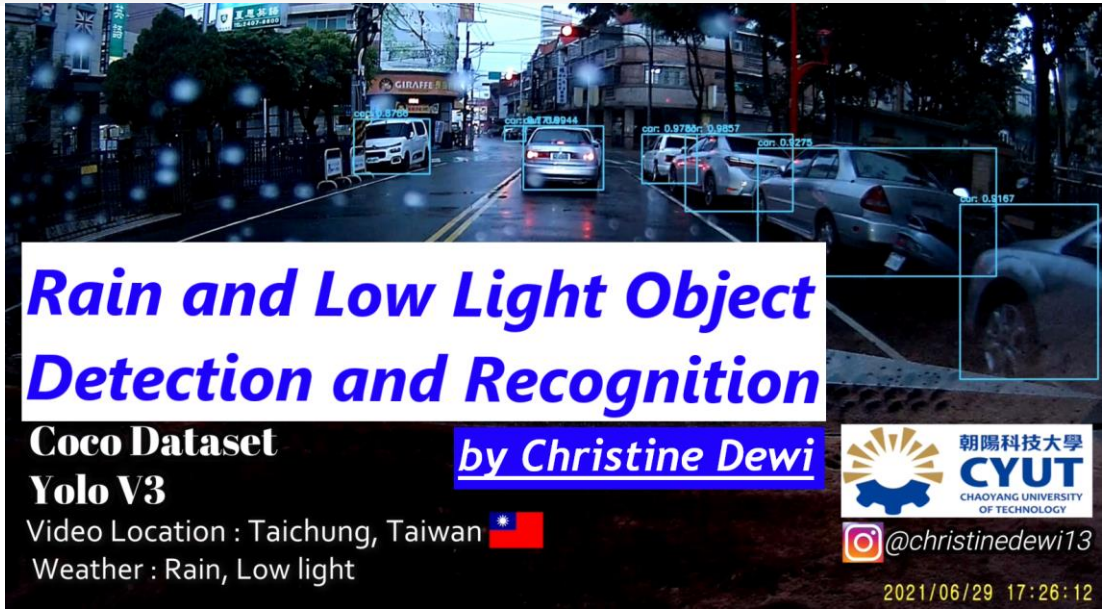
Link:
https://www.youtube.com/watch?v=D2iFhLWfH6A&list=PLkfg2Q8T49glvToafqoBWeZ4UVNkPny9V&index=4

EXPERIMENT VIDEO 2

```
(dewi) E:\PySimpleGUI-YOLO-master>python yolo_video_save.py --input input_avi2.AVI --output outputavi2.avi --yolo yolo-coco
[INFO] loading YOLO from disk...
[INFO] 5406 total frames in video
[INFO] single frame took 0.4772 seconds
[INFO] estimated total time to finish: 2579.9870
[INFO] cleaning up...

(dewi) E:\PySimpleGUI-YOLO-master>
```

(dewi) E:\PySimpleGUI-YOLO-master>python yolo_video_save.py --input input_avi2.AVI --output outputavi2.avi --yolo yolo-coco
[INFO] loading YOLO from disk...
[INFO] 5406 total frames in video
[INFO] single frame took 0.4772 seconds
[INFO] estimated total time to finish: 2579.9870
[INFO] cleaning up...
(dewi) E:\PySimpleGUI-YOLO-master>

Link:
https://www.youtube.com/watch?v=QeKRU4DAyRo&list=PLkfg2Q8T49glvToafqoBWeZ4UVNkPny9V&index=8

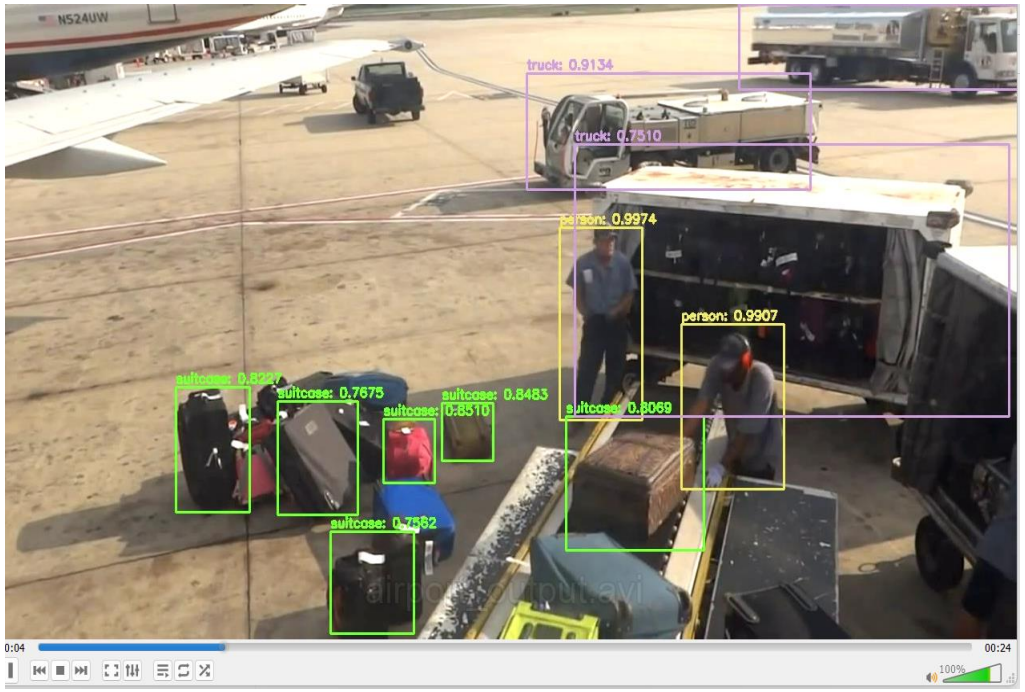


EXPERIMENT VIDEO 3

```
(yolov5a) D:\PROJECT\yolo-object-detection>python yolo_video.py --input videos/airport.mp4 --output output/airport_outpu
t.avi --yolo yolo-coco
[INFO] loading YOLO from disk...
[INFO] 749 total frames in video
[INFO] single frame took 0.2263 seconds
[INFO] estimated total time to finish: 169.5091
[INFO] cleaning up...

(yolov5a) D:\PROJECT\yolo-object-detection>
```

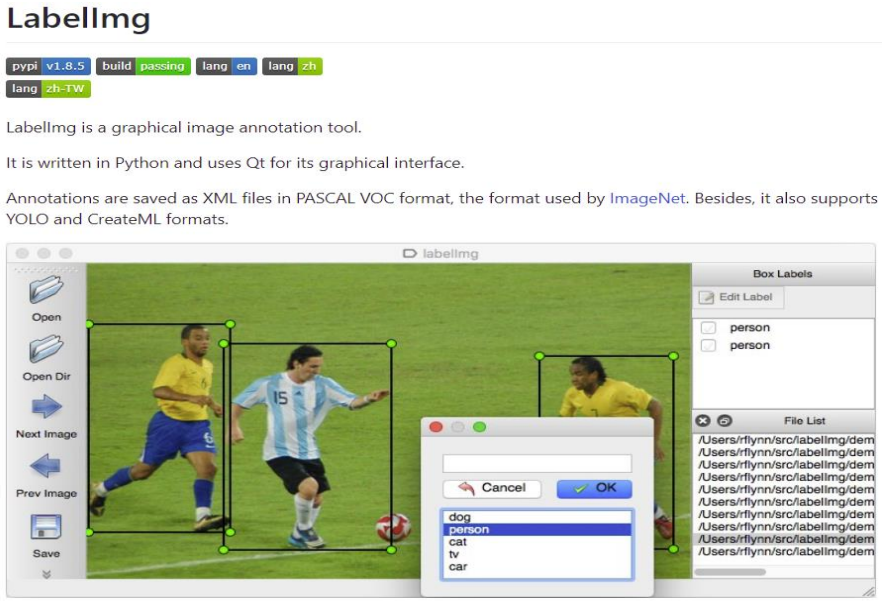
(yolov5a) D:\PROJECT\yolo-object-detection>python yolo_video.py --input videos/airport.mp4 --output output/airport_output.avi --yolo yolo-coco
[INFO] loading YOLO from disk...
[INFO] 749 total frames in video
[INFO] single frame took 0.2263 seconds
[INFO] estimated total time to finish: 169.5091
[INFO] cleaning up...

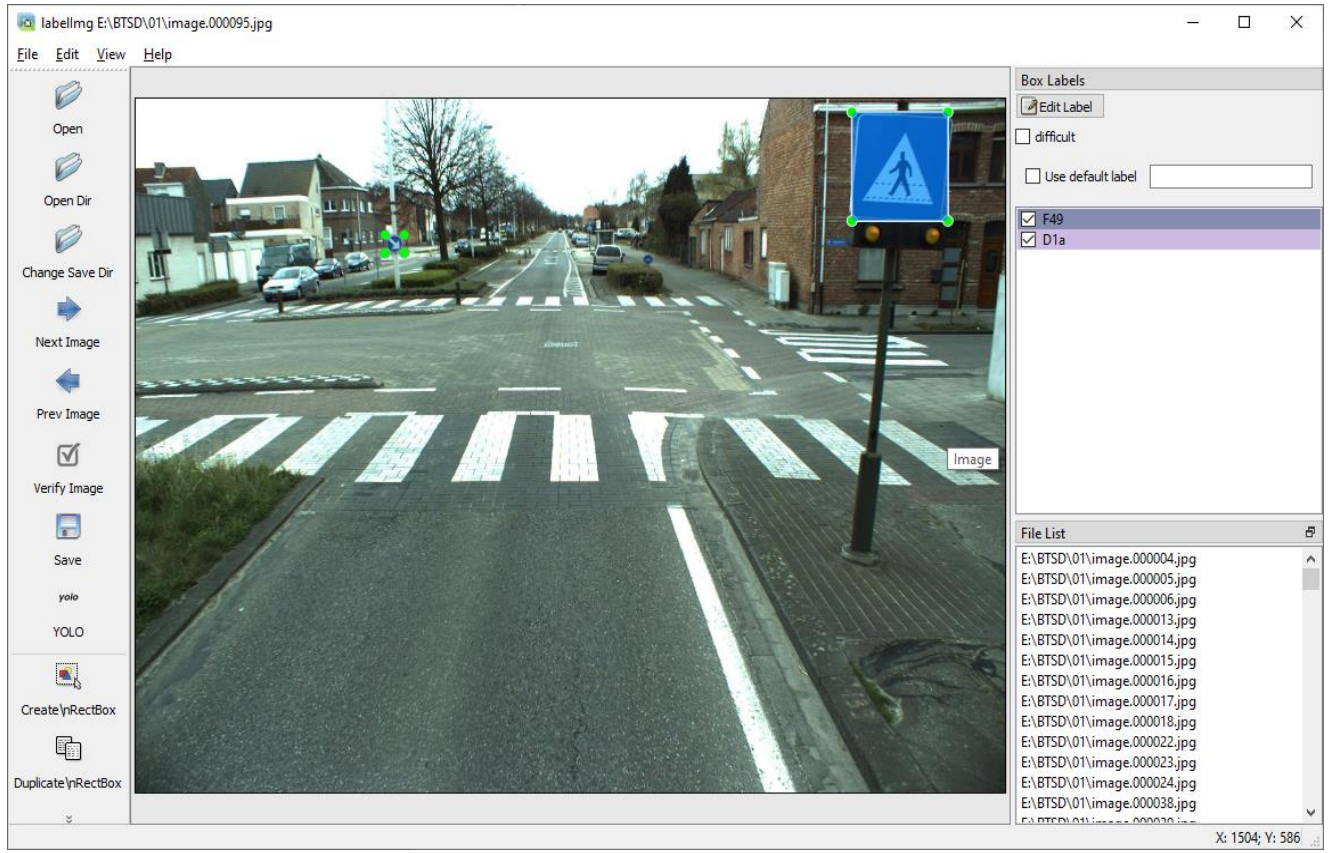
(yolov5a) D:\PROJECT\yolo-object-detection>



BUILD DATASET BY OURSELF

1. Prepare the image
   pip install labelImg
   python labelImg.py





(https://github.com/tzutalin/labelImg#labelimg)

朝陽科技大學 CYUT CHAOYANG UNIVERSITY OF TECHNOLOGY

BUILD DATASET BY OURSELF

image.000049.jpg

image.000049.txt

image.000055.jpg

image.000055.txt

image.000056.txt

image.000073.jpg

image.000073.txt

image.000074.jpg

image.000049.txt - Notepad

File Edit Format View Help

```
6 0.949631 0.304207 0.036855 0.048544
```

image.000055.txt - Notepad

File Edit Format View Help

```
6 0.944717 0.300971 0.039312 0.051780
```

image.000073.txt - Notepad

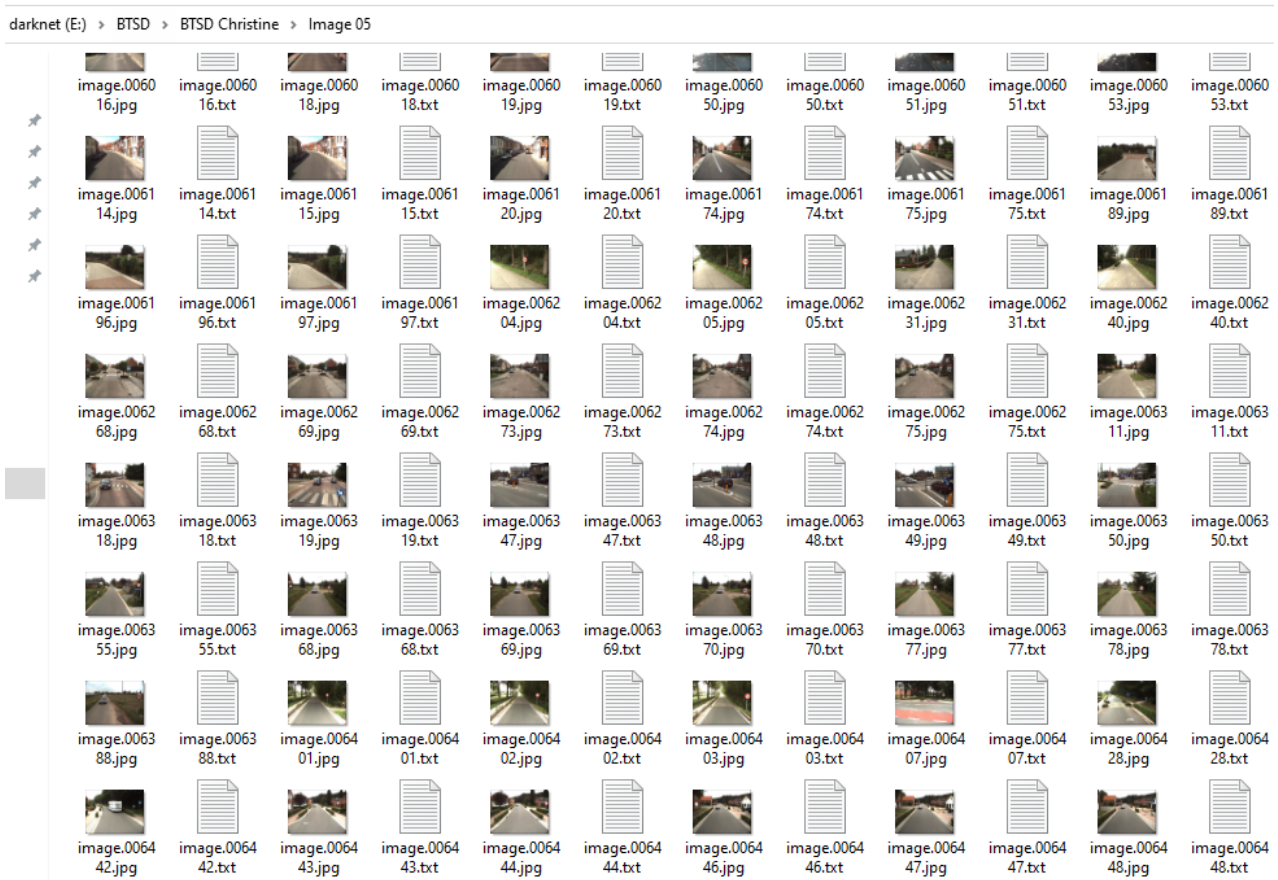
File Edit Format View Help

```
15 0.857494 0.373786 0.047912 0.063107
```

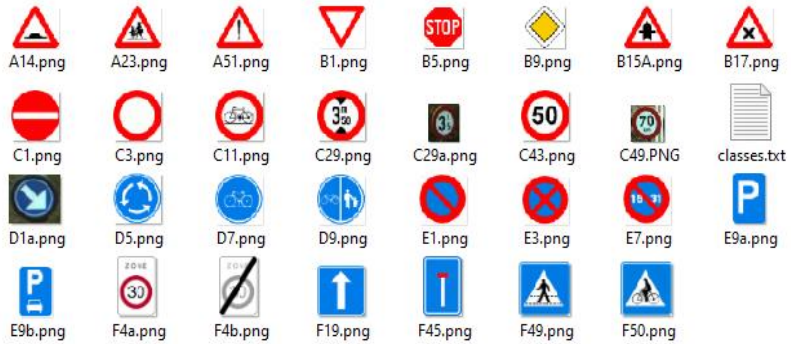
Website : https://sites.google.com/view/christinedewi/home

朝陽科技大學 CYUT CHAOYANG UNIVERSITY OF TECHNOLOGY

BUILD DATASET BY OURSELF



	A	B
1	0	A14
2	1	A23
3	2	A51
4	3	B1
5	4	B5
6	5	B9
7	6	B15A
8	7	B17
9	8	C1
10	9	C3
11	10	C11
12	11	C29
13	12	C29a
14	13	C43
15	14	C49
16	15	D1a
17	16	D5
18	17	D7
19	18	D9
20	19	E1
21	20	E3
22	21	E7
23	22	E9a
24	23	E9b
25	24	F4a
26	25	F4b
27	26	F19
28	27	F45
29	28	F49
30	29	F50



Website : https://sites.google.com/view/christinedewi/home

朝陽科技大學 CYUT CHAOYANG UNIVERSITY OF TECHNOLOGY

TRAINING WEIGHT BY OURSELF USING DARKNET

Yolo v4, v3 and v2 for Windows and Linux

(neural networks for object detection)

Paper YOLO v4: https://arxiv.org/abs/2004.10934

Paper Scaled YOLO v4: https://arxiv.org/abs/2011.08036 use to reproduce results: ScaledYOLOv4

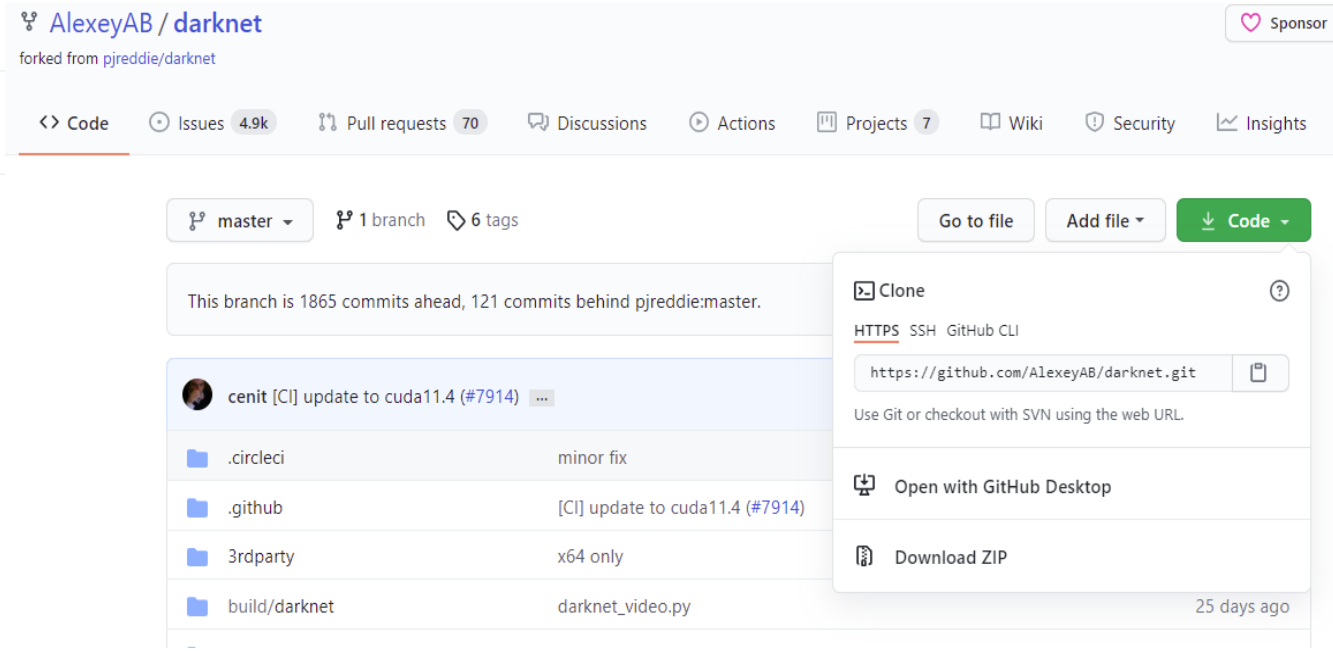
More details in articles on medium:

- Scaled_YOLOv4
- YOLOv4

Manual: https://github.com/AlexeyAB/darknet/wiki

Discussion:

- Reddit
- Google-groups
- Discord



https://github.com/AlexeyAB/darknet

Website : https://sites.google.com/view/christinedewi/home

朝陽科技大學 CYUT CHAOYANG UNIVERSITY OF TECHNOLOGY

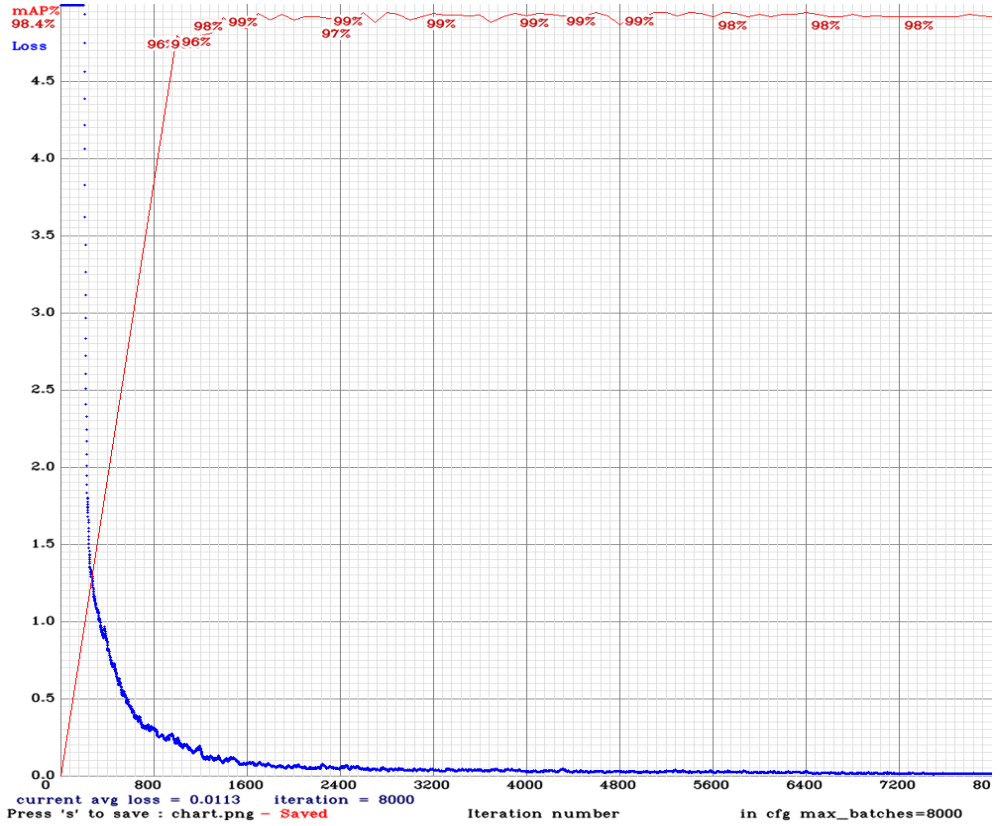
TRAINING WEIGHT BY OURSELF

How to train:

darknet.exe detector train cfg/coco.data yolov4.cfg yolov4.conv.137 -dont_show -mjpeg_port 8090 –map
darknet.exe detector train cfg/TT100K.data cfg/yolov3TT100K.cfg darknet53.conv.74 -map

- Train on **Amazon EC2**, to see mAP & Loss-chart using URL like: `http://ec2-35-160-228-91.us-west-2.compute.amazonaws.com:8090` in the Chrome/Firefox (**Darknet should be compiled with OpenCV**): `./darknet detector train cfg/coco.data yolov4.cfg yolov4.conv.137 -dont_show -mjpeg_port 8090 -map`

https://github.com/AlexeyAB/darknet



朝陽科技大學 CYUT CHAOYANG UNIVERSITY OF TECHNOLOGY

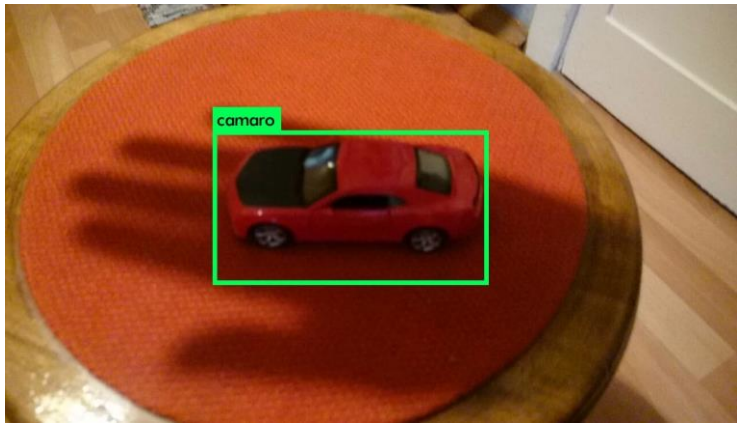
TESTING

How to test:

darknet.exe detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights -ext_output -dont_show -out result.json < data/train.txt

darknet.exe detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights -thresh 0.25 -dont_show -save_labels < data/new_train.txt

- To process a list of images `data/train.txt` and save results of detection to `result.json` file use: `darknet.exe detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights -ext_output -dont_show -out result.json < data/train.txt`
- To process a list of images `data/train.txt` and save results of detection to `result.txt` use: `darknet.exe detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights -dont_show -ext_output < data/train.txt > result.txt`
- Pseudo-labelling - to process a list of images `data/new_train.txt` and save results of detection in Yolo training format for each image as label `<image_name>.txt` (in this way you can increase the amount of training data) use: `darknet.exe detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights -thresh 0.25 -dont_show -save_labels < data/new_train.txt`



https://github.com/AlexeyAB/darknet

RESEARCH TOPIC

The Study of Robust Detection Method for Improving Traffic Sign Recognition Based on Spatial Pyramid Pooling

- Small Traffic Sign Recognition Video 1

https://www.youtube.com/watch?v=if2eqVVxuF0

- Small Traffic Sign Recognition Video 2

https://www.youtube.com/watch?v=TLxZY875d0M

Website : https://sites.google.com/view/christinedewi/home

RESEARCH TOPIC

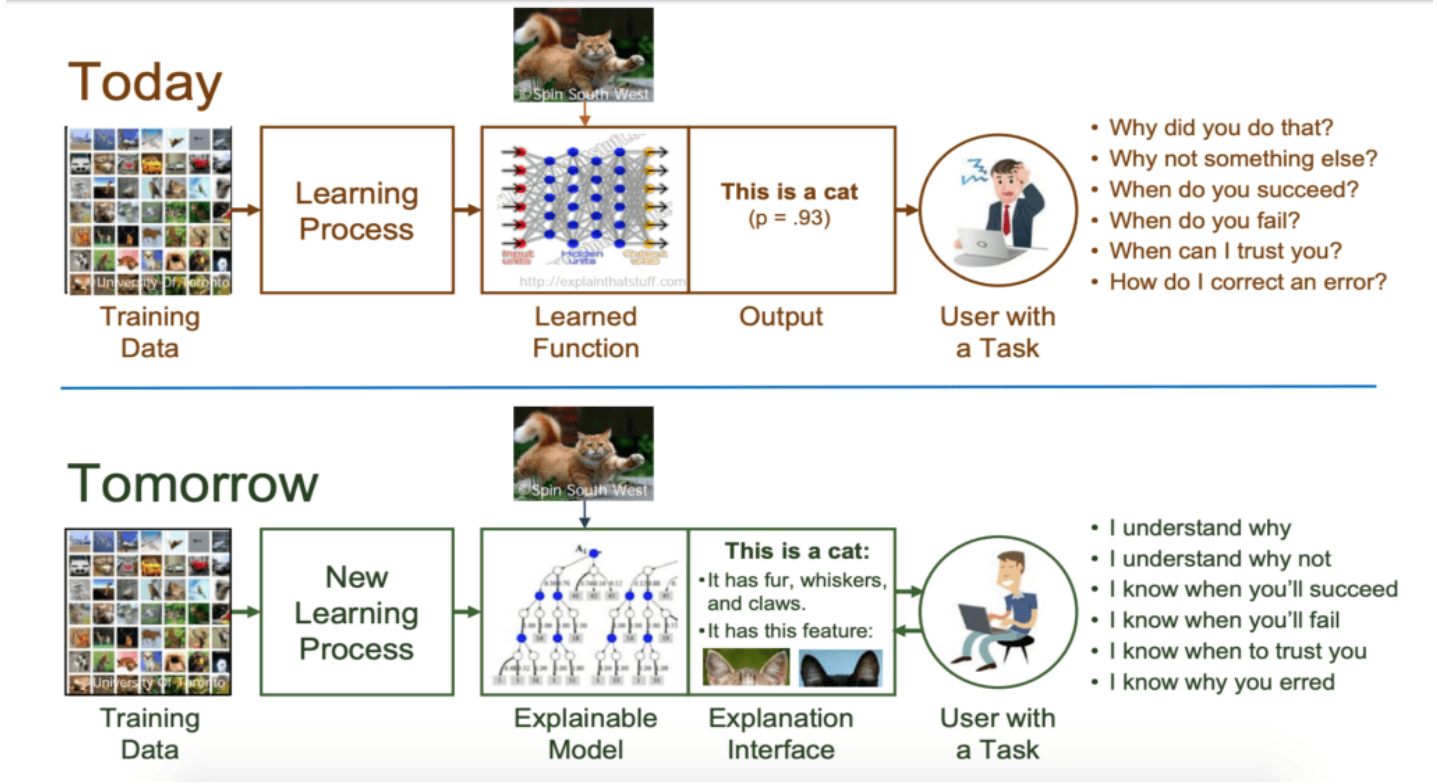
- Face Recognition

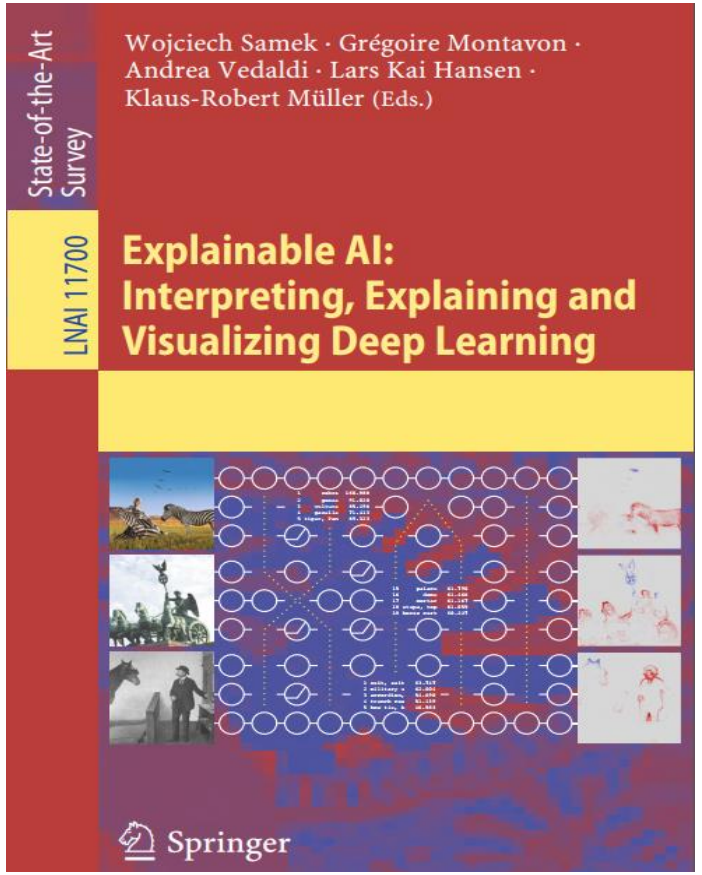
(Eyes, Ear, Nose, Mouth)

- Face Emotion Recognition

RESEARCH TOPIC

- Explainable AI (XAI)





Website : https://sites.google.com/view/christinedewi/home

朝陽科技大學 CYUT CHAOYANG UNIVERSITY OF TECHNOLOGY

RESEARCH TOPIC

- Explainable AI (XAI)
- **Interpretable Text-to-Image Synthesis with Hierarchical Semantic Layout Generation**

Fig. 5.6. Multiple samples generated from a text description.

Explainable AI:
Interpreting, Explaining and Visualizing Deep Learning

朝陽科技大學 CYUT CHAOYANG UNIVERSITY OF TECHNOLOGY

Website : https://sites.google.com/view/christinedewi/home

RESEARCH TOPIC

- Explainable AI (XAI)
- **Visual Scene Understanding for Autonomous Driving Using Semantic Segmentation**

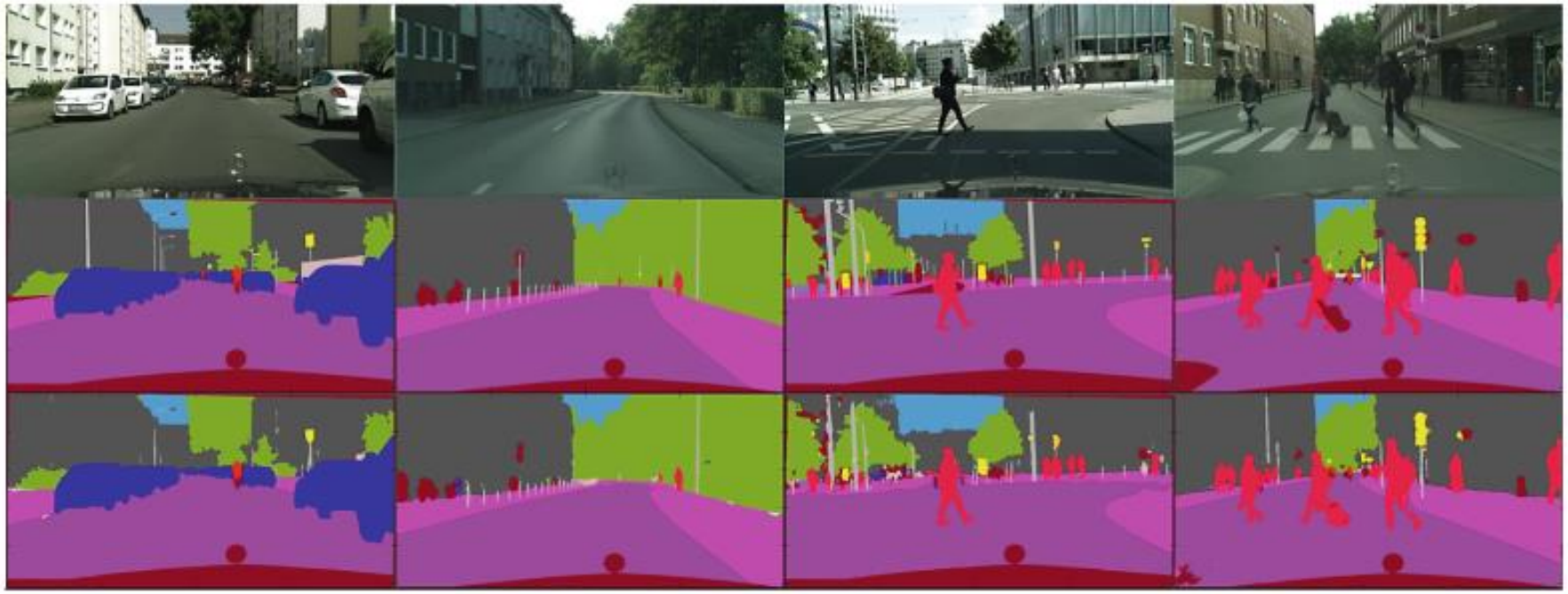


Fig. 15.1. Example output (bottom) of the network with ground truth (center) on images from the Cityscapes validation set.

https://www.cityscapes-dataset.com/

Website : https://sites.google.com/view/chihtinedewi/home

朝陽科技大學 CYUT CHAOYANG UNIVERSITY OF TECHNOLOGY

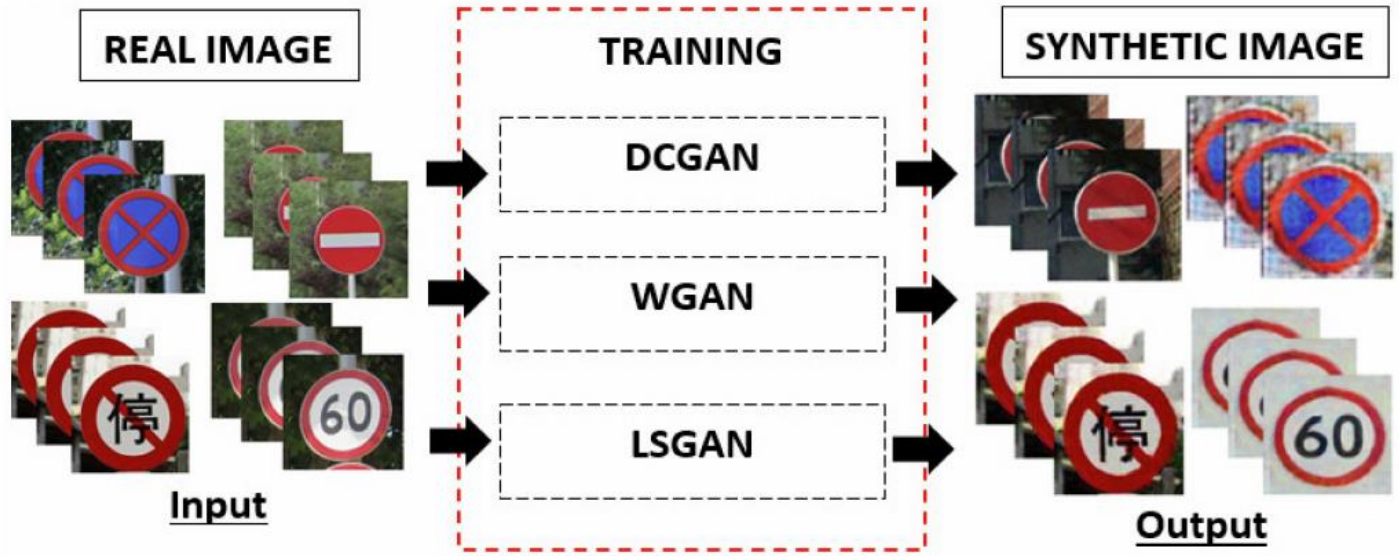
RESEARCH TOPIC

- Drowsiness detection for safety drivers

https://www.youtube.com/watch?v=lO-yAWMtDAk
https://www.youtube.com/watch?v=V4gY8upL4tA

RESEARCH TOPIC

- Synthetic data generation using Generative Adversarial Network (GAN)



https://www.mdpi.com/2076-3417/11/7/2913

RESEARCH TOPIC

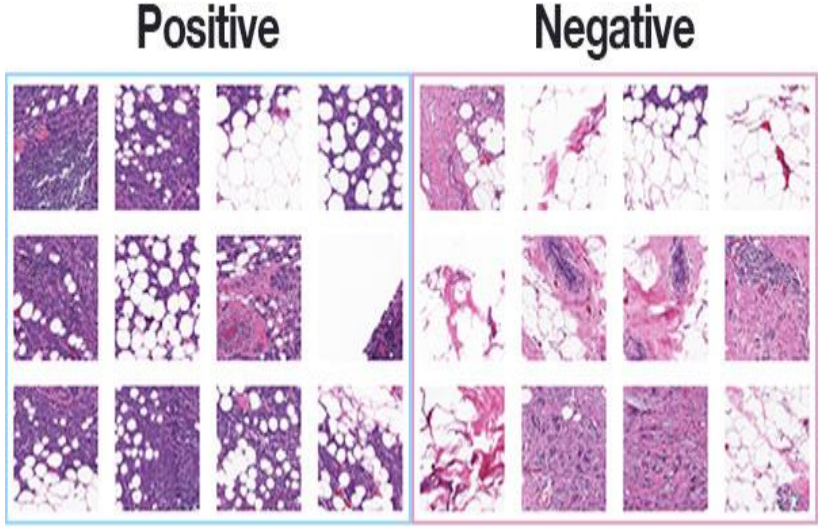
- Synthetic data generation using Generative Adversarial Network (GAN)
- DCGAN, StyleGAN, BiGAN, MGAN, LSGAN, WGAN

Website : https://sites.google.com/view/christinedewi/home

RESEARCH TOPIC

Artificial Intelligence for Biomedicine

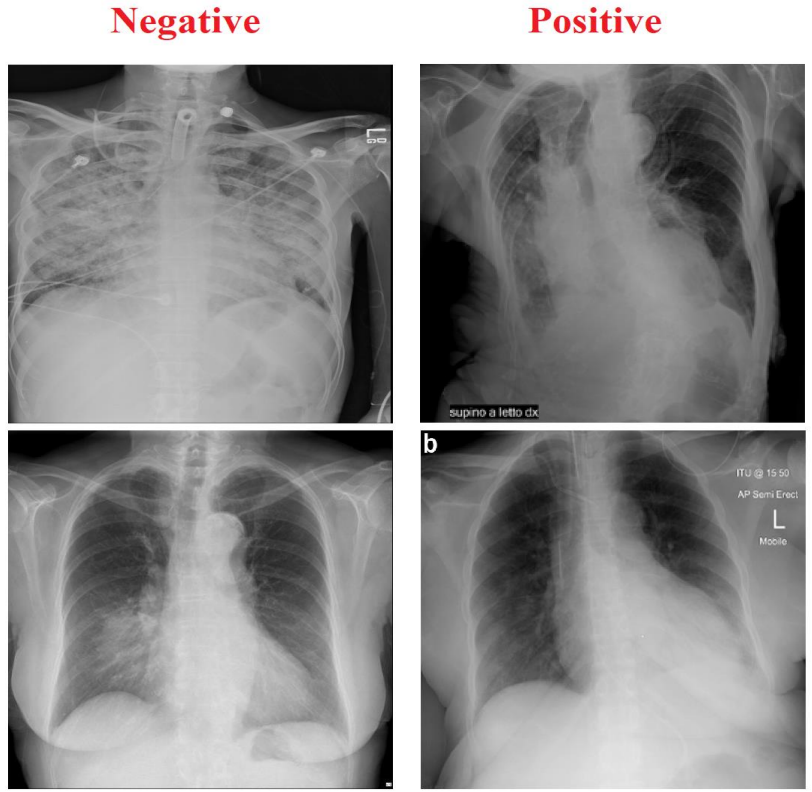
- machine learning-enhanced optical imaging and sensing
- image segmentation
- virtual tissue staining
- artificial intelligence as a tool to enhance decision-making in personalized medicine and drug screening
- multiple-sources data structuring and combination in complex biomedical decision-making
- legal and ethical aspects of the use of artificial intelligence as a tool for decision-making in medicine



Website : https://sites.google.com/view/christinedewi/home

RESEARCH TOPIC

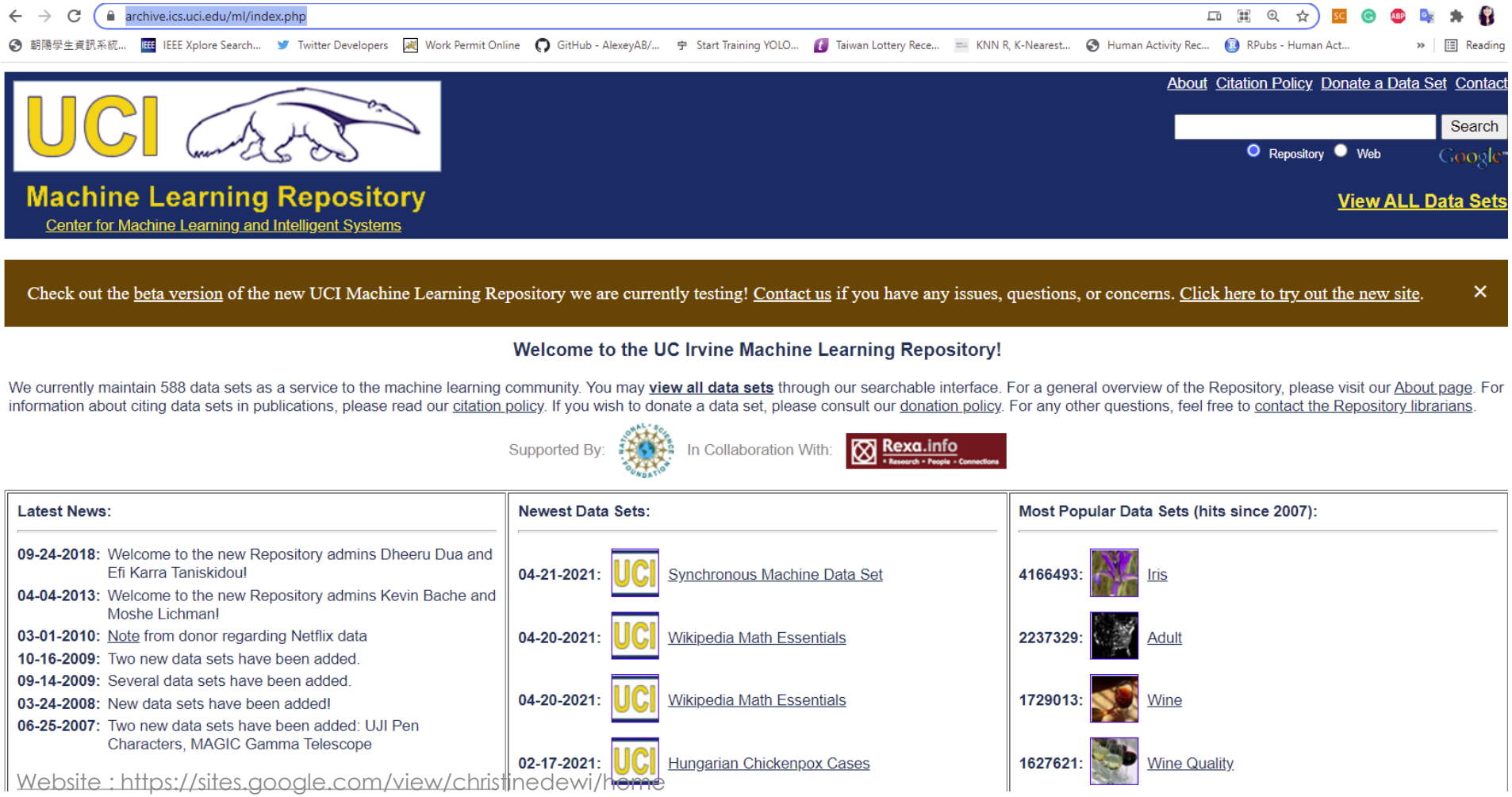
- Voice recognition
- Natural Language Processing (NLP)
- Sympathetic the temporal evolution of COVID-19 Research Through machine learning and natural language processing
- AIoT-Artificial Intelligence of Things



朝陽科技大學
CYUT
CHAOYANG UNIVERSITY OF TECHNOLOGY

DATASET

- https://archive.ics.uci.edu/ml/index.php



Website : https://sites.google.com/view/christinedewi/home

DATASET

- https://archive.ics.uci.edu/ml/index.php

Browse Through: **588** Data Sets

Table View List View

Default Task
Classification (442)
Regression (137)
Clustering (117)
Other (56)

Attribute Type
Categorical (38)
Numerical (396)
Mixed (55)

Data Type
Multivariate (456)
Univariate (27)
Sequential (57)
Time-Series (121)
Text (66)
Domain-Theory (23)
Other (21)

Area
Life Sciences (138)
Physical Sciences (57)
CS / Engineering (215)
Social Sciences (38)
Business (44)
Game (11)
Other (80)

Attributes
Less than 10 (151)
10 to 100 (266)
Greater than 100 (106)

Instances
Less than 100 (36)
100 to 1000 (199)
Greater than 1000 (318)

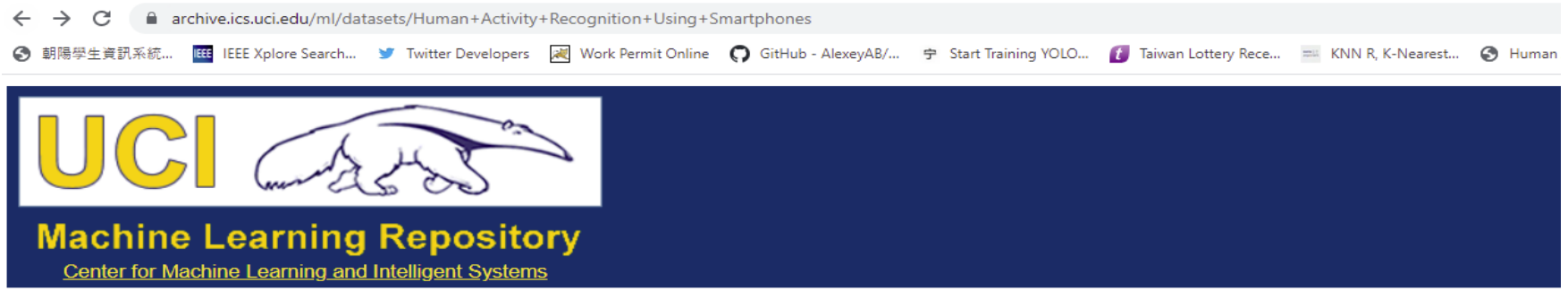
Name	Data Types	Default Task	Attribute Types	# Instances	# Attributes	Year
Abalone	Multivariate	Classification	Categorical, Integer, Real	4177	8	1995
Adult	Multivariate	Classification	Categorical, Integer	48842	14	1996
Annealing	Multivariate	Classification	Categorical, Integer, Real	798	38	
Anonymous Microsoft Web Data		Recommender-Systems	Categorical	37711	294	1998
Arrhythmia	Multivariate	Classification	Categorical, Integer, Real	452	279	1998
Artificial Characters	Multivariate	Classification	Categorical, Integer, Real	6000	7	1992
Audiology (Original)	Multivariate	Classification	Categorical	226		1987
Audiology (Standardized)	Multivariate	Classification	Categorical	226	69	1992
Auto MPG	Multivariate	Regression	Categorical, Real	398	8	1993

Website : https://sites.google.com/view/christinedewi/home

朝陽科技大學 CYUT CHAOYANG UNIVERSITY OF TECHNOLOGY

DATASET

- https://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones



Human Activity Recognition Using Smartphones Data Set

Download: Data Folder, Data Set Description

Abstract: Human Activity Recognition database built from the recordings of 30 subjects performing activities of daily living (ADL) while carrying a waist-mounted smartpl

Data Set Characteristics:	Multivariate, Time-Series	**Number of Instances:**	10299	**Area:**	Computer
Attribute Characteristics:	N/A	**Number of Attributes:**	561	**Date Donated**	2012-12-10
Associated Tasks:	Classification, Clustering	**Missing Values?**	N/A	**Number of Web Hits:**	1152801

Source:

Jorge L. Reyes-Ortiz(1,2), Davide Anguita(1), Alessandro Ghio(1), Luca Oneto(1) and Xavier Parra(2)
1 - Smartlab - Non-Linear Complex Systems Laboratory
DITEN - Università degli Studi di Genova, Genoa (I-16145), Italy.
2 - CETpD - Technical Research Centre for Dependency Care and Autonomous Living
Universitat Politècnica de Catalunya (BarcelonaTech). Vilanova i la Geltrú (08800), Spain
activityrecognition '@' smartlab.ws

Website: https://sites.google.com/view/christinadewi/home

ARTIFICIAL INTELLIGENT AND MACHINE LEARNING

Lecturer: Prof. Rung-Ching Chen

Teaching Assistant: Christine Dewi

#1 Machine Learning (AI Class 2020)
https://www.youtube.com/watch?v=ljPydoiA-qk&t=636s
#2 Gradient Descent (AI Class 2020)
https://www.youtube.com/watch?v=9JxlN7MSZS4
#3 Rule Based Expert System (AI Class 2020)
https://www.youtube.com/watch?v=VadjsY6HbUU&t=7s
#4 Rule Based Expert System (AI Class 2020)
https://www.youtube.com/watch?v=qJ-1aR2rqRI
#5 Rule Based Expert System (AI Class 2020)
https://www.youtube.com/watch?v=Q9mKsbrvgEQ&t=1s
#6 Certainty Factors (AI Class 2020)
https://www.youtube.com/watch?v=K2XfAqPPUWc&t=23s
#7 Fuzzy Systems (AI Class 2020)
https://www.youtube.com/watch?v=p3qx52Sw3Hs
#8 Fuzzy Inference (AI Class 2020)
https://www.youtube.com/watch?v=Ux_PORZQDVU&t=4s

Website : https://sites.google.com/view/christinedewi/home

朝陽科技大學
CYUT
CHAOYANG UNIVERSITY OF TECHNOLOGY

INTERNET OF THINGS

Lecturer: Prof. Rung-Ching Chen

Teaching Assistant: Christine Dewi

#1 Smart Application Based on IoT and Big Data (IoT Class 2020)

https://www.youtube.com/watch?v=8rVJA-XGbyo&list=PLktg2Q8T49gkHGog1ZYMUsoW0Z7anuJPN

#2 Smart Application Based on IoT (IoT Class 2020)

https://www.youtube.com/watch?v=1SJwjeKLUOg&list=PLkfg2Q8T49gkHGog1ZYMUsoW0Z7anuJPN&index=2

#3 Smart Application Based on IoT Part 3 (IoT Class 2020)

https://www.youtube.com/watch?v=rSzItnZaYdg&list=PLkfg2Q8T49gkHGog1ZYMUsoW0Z7anuJPN&index=4

#4 Fog Computing and Services (IoT Class 2020)

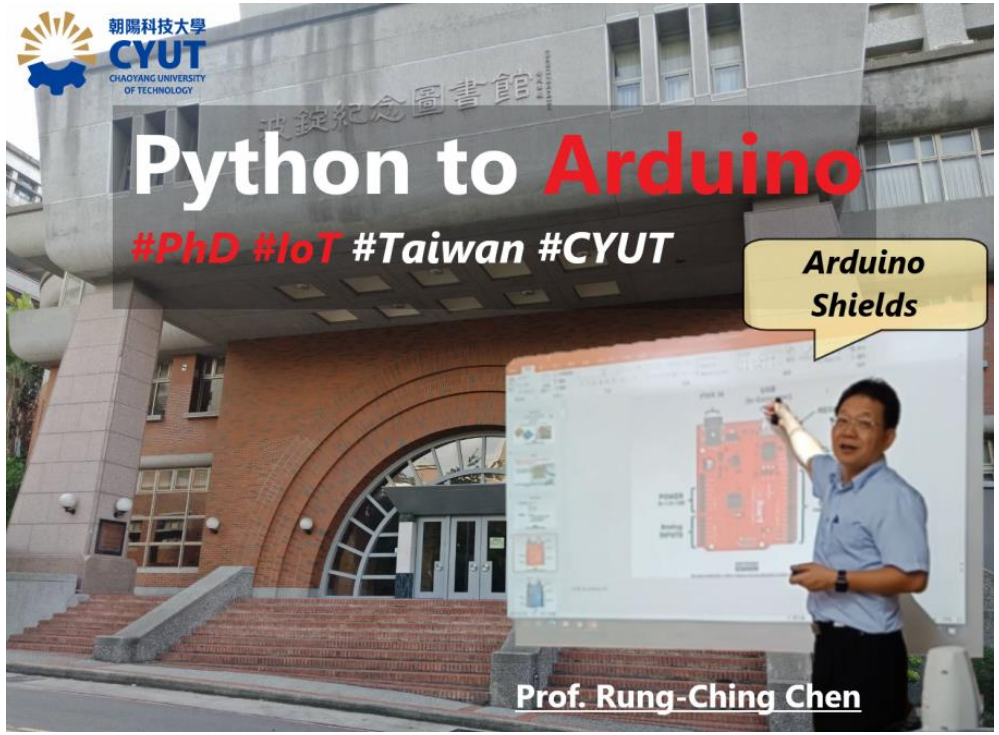
https://www.youtube.com/watch?v=yJsw7tJFT8g&list=PLkfg2Q8T49gkHGog1ZYMUsoW0Z7anuJPN&index=3

#5 Smart Applications (IoT Class 2020)

https://www.youtube.com/watch?v=sjA8CER1sH4&list=PLkfg2Q8T49gkHGog1ZYMUsoW0Z7anuJPN&index=5

#6 Python to Arduino (IoT Class 2020)

https://www.youtube.com/watch?v=4ZY8s3pbk3A&list=PLkfg2Q8T49gkHGog1ZYMUsoW0Z7anuJPN&index=6



Website : https://sites.google.com/view/christinedewi/home

PUBLICATIONS 2018-2021

JOURNAL

[1] R. C. Chen, **C. Dewi**, S. W. Huang, and R. E. Caraka, "Selecting critical features for data classification based on machine learning methods," *Journal of Big Data*, vol. 7, no. 52, pp. 1–26, 2020. (Springer, Q1 SCIE)

[2] **C. Dewi** and R.-C. Chen, "Random Forest and Support Vector Machine on Features Selection for Regression Analysis," *International Journal of Innovative Computing, Information and Control (IJICIC)*, vol. 15, no. 6, pp. 2027–2038, 2019. (Q2 SCIE)

[3] S. Tai, **C. Dewi**, R. Chen, Y. Liu, and X. Jiang, "Deep Learning for Traffic Sign Recognition Based on Spatial Pyramid Pooling with Scale Analysis," *Applied Sciences (Switzerland)*, vol. 10, no. 19, p. 6997, 2020. (MDPI, Q1 SCIE)

[4] **C. Dewi**, R. C. Chen, and H. Yu, "Weight analysis for various prohibitory sign detection and recognition using deep learning," *Multimedia Tools and Applications*, pp. 1–21, 2020. (Springer, Q1 SCIE)

[5] **C. Dewi**, R.-C. Chen, and S.-K. Tai, "Evaluation of Robust Spatial Pyramid Pooling Based on Convolutional Neural Network for Traffic Sign Recognition System," *Electronics*, vol. 9, no. 6, p. 889, 2020. (MDPI, Q2 SCIE)

[6] **C. Dewi** and R.-C. Chen, "Integrating Real-Time Weather Forecasts Data Using OpenWeatherMap and Twitter," *International Journal of Information Technology and Bussiness*, vol. 1, no. 2, pp. 48–52, 2019. (Scopus)

[7] R.-C. Chen, **C. Dewi**, W.-W. Zhang, and J.-M. Liu, "Integrating Gesture Control Board and Image Recognition for Gesture Recognition Based on Deep Learning," *International Journal of Applied Science and Engineering (IJASE)*, pp. 1–10, 2020. (Q2 SCIE)

[8] V. Suthamathi, R.-C. Chen, **C. Dewi**, and L.-S. Chen, "Solving Unbounded Knapsack Problem Using Evolutionary Algorithms with Bound Constrained Strategy," *International Journal of Applied Science and Engineering (IJASE*, pp. 1–12, 2020. (Q2 SCIE)

[9] **C. Dewi**, R. Chen, Y. Liu, and H. Yu, "Various Generative Adversarial Networks Model for Synthetic Prohibitory Sign Image Generation," Applied Sciences, vol. 11, p. 2913, 2021. (Q1 SCIE)

[10] **C. Dewi**, R.-C. Chen, Y.-T. Liu, and S.-K. Tai, "Synthetic Data generation using DCGAN for improved traffic sign recognition," Neural Computing and Applications, 2021. (Q1 SCIE)

[11] **Dewi, C.,** Chen, R.-C., Liu, Y.-T., Jiang, X., & Hartomo, K. D. (2021). Yolo V4 for advanced Traffic Sign Recognition with synthetic training data generated by various GAN. *IEEE Access*. https://doi.org/10.1109/ACCESS.2021.3094201. (Q1 SCIE)

PUBLICATIONS 2018-2021

CONFERENCE

[1] **C. Dewi**, R. C. Chen, and Y.-T. Liu, "Taiwan Stop Sign Recognition with Customize Anchor," in *ICCMS '20, February 26–28, 2020, Brisbane, QLD, Australia*, 2020.

[2] **C. Dewi** and R.-C. Chen, "Human Activity Recognition Based on Evolution of Features Selection and Random Forest," *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pp. 2496–2501, Oct. 2019.

[3] **C. Dewi**, R.-C. Chen, Hendry, and Y.-T. Liu, "Similar Music Instrument Detection via Deep Convolution YOLO-Generative Adversarial Network," *2019 IEEE 10th International Conference on Awareness Science and Technology (iCAST)*, pp. 1–6, Oct. 2019.

[4] **C. Dewi** and R.-C. Chen, *Decision making based on IoT data collection for precision agriculture*, vol. 830. 2020.

[5] **C. Dewi**, R. C. Chen, Hendry, and H. Te Hung, "Comparative Analysis of Restricted Boltzmann Machine Models for Image Classification," in *Asian Conference on Intelligent Information and Database Systems ACIIDS 2020*, 2020, vol. 12034 LNAI, pp. 285–296.

[6] V. Suthamathi, R.-C. Chen, **C. Dewi**, and L.-S. Chen, "Montecarlo Approach for Solving Unbound Knapsack Problem," in The 7th Multidisciplinary International Social Networks Conference (MISNC 2020), Taiwan, 2020, pp. 1–10.

[7] Y. T. Liu, R. C. Chen, and **C. Dewi**, "Generate Realistic Traffic Sign Image using Deep Convolutional Generative Adversarial Networks," in 2021 IEEE Conference on Dependable and Secure Computing, DSC 2021, 2021.

[8] **C. Dewi**, R.-C. Chen, and Y.-T. Liu, "Wasserstein Generative Adversarial Networks for Realistic Traffic Sign Image Generation," 2021.

[9] R. C. Chen, Y. C. Shiao, **C. Dewi**, and Q. E. Liu, "Predicting the Japanese Yen to US Dollar Exchange Rate Based on Machine Learning Models," in Proceedings - 2020 International Computer Symposium, ICS 2020, 2020.

☺ THANK YOU ☺

Christine Dewi
E-mail :
christine.dewi13@gmail.com
christine.dewi@uksw.edu,

Department of Information Management,
Chaoyang University of Technology
168, Jifong East Road, Wufong Dist., Taichung City 41349,
Taiwan (R.O.C.)

Department of Information Technology
Satya Wacana Christian University
JL. Diponegoro 52 – 60, Salatiga 50711Central Java, Indonesia.

Website :
https://sites.google.com/view/christinedewi/home