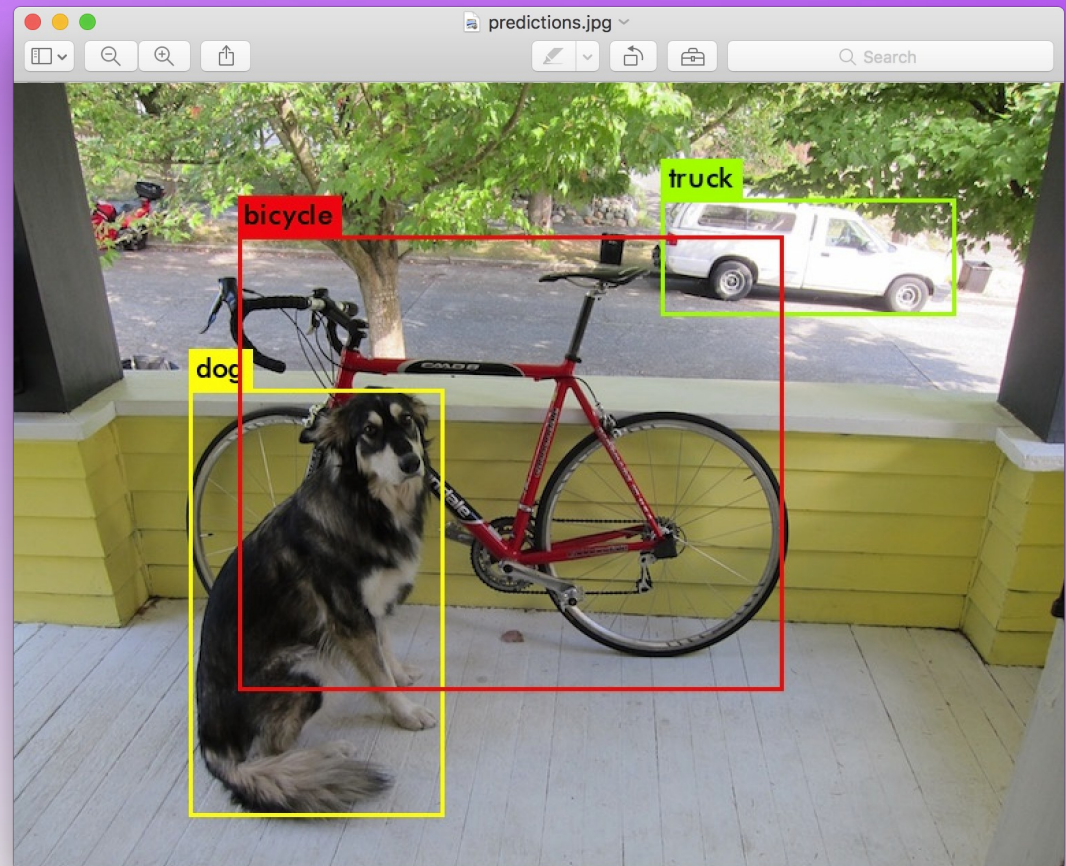
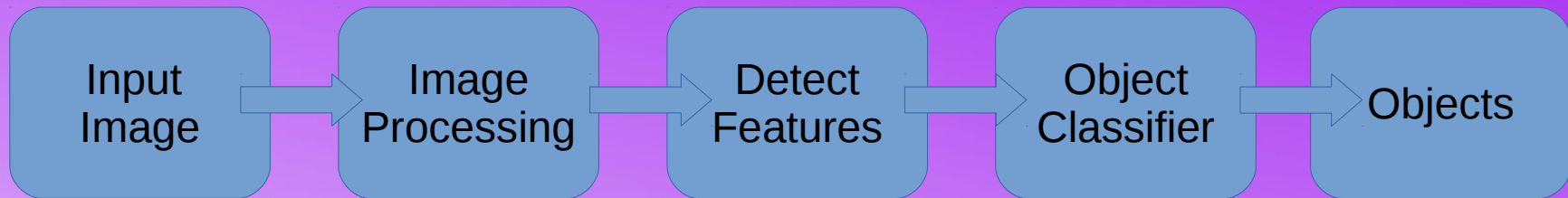


Computer Vision with YOLO (You Only Look Once)



Computer Vision



- Classic Computer Vision Model
 - Read/Decode Frame
 - Sequence of Image Processing
 - Convert colorspace
 - Reduce/Remove Noise (e.g. blur)
 - Thresholding/Edge Detection
 - Convolution (grow/shrink)
 - Pixels to Features
 - Bounding boxes to group features
 - Features to Object(s)



YOLO



- ~~Detector~~ \Rightarrow Classifier
- Classifier as a Detector
 - Subsample image into regions
 - Assume each region has an object, run classifier on each
 - Merge classifier groups into final object(s)

Detectable Objects

- This classifier can be custom trained
- Using a default configuration as part of setup

aeroplane	bird	cat	fire hydrant	knife	pottedplant	sofa	toilet
apple	boat	cell phone	fork	laptop	refrigerator	spoon	toothbrush
backpack	book	chair	frisbee	microwave	remote	sports ball	traffic light
banana	bottle	clock	giraffe	motorbike	sandwich	stop sign	train
baseball bat	bowl	cow	hair drier	mouse	scissors	suitcase	truck
baseball glove	broccoli	cup	handbag	orange	sheep	surfboard	tvmonitor
bear	bus	diningtable	horse	oven	sink	teddy bear	umbrella
bed	cake	dog	hot dog	parking meter	skateboard	tennis racket	vase
bench	car	donut	keyboard	person	skis	tie	wine glass
bicycle	carrot	elephant	kite	pizza	snowboard	toaster	zebra

Setup/Install

- Ubuntu 16.04
 - `sudo apt-get install -y libopencv-dev`
 - `sudo apt-get install -y python3`
 - `sudo apt-get install -y python3-dev`
 - `sudo apt-get install -y python3-pip`
 - `sudo pip3 install pip --upgrade`
 - `sudo pip3 install opencv-python`
 - `sudo pip3 install numpy`
 - `sudo pip3 install cython`
 - `sudo pip3 install yolo34py`
 - `wget https://raw.githubusercontent.com/pjreddie/darknet/master/cfg/yolov3.cfg`
 - `wget https://pjreddie.com/media/files/yolov3.weights`
 - `wget https://raw.githubusercontent.com/pjreddie/darknet/master/cfg/coco.data -O data/coco.data`
 - `wget https://raw.githubusercontent.com/pjreddie/darknet/master/data/coco.names -O data/coco.names`

Setup Directory Structure

```
/var/tmp/YOLO-python$ tree .
```

```
.
├── data
│   ├── coco.data
│   └── coco.names
├── example
├── sample.jpg
├── video.mp4
├── yolov3.cfg
└── yolov3.weights
```

1 directory, 7 files

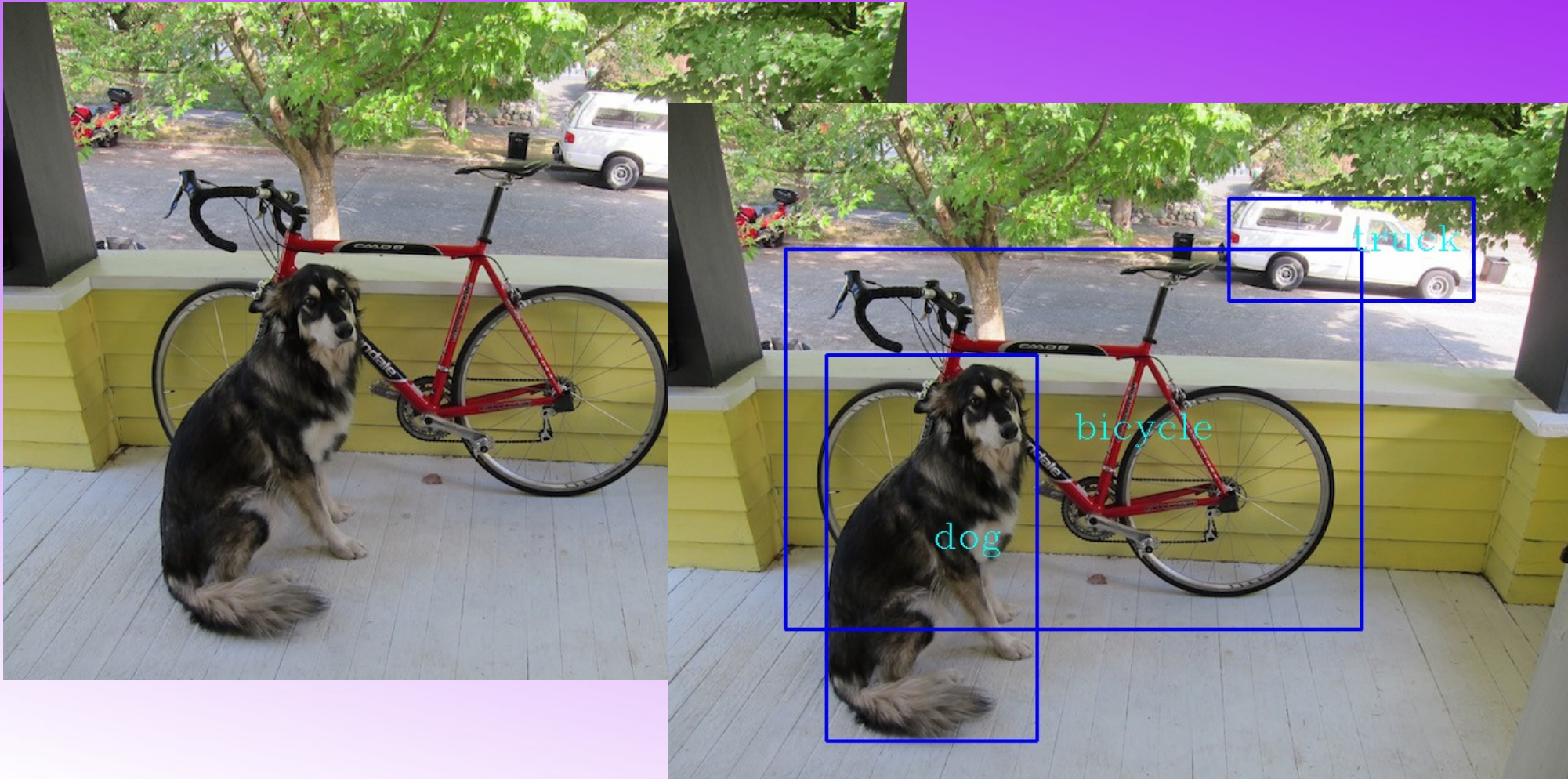
- Python example: *example*
- Test image: *sample.jpg*
- Test video: *video.mp4*

Python Example – Image

```
$ cat -n example
1      #!/usr/bin/python3
2
3      from pydarknet import Detector, Image
4      import cv2
5
6      net = Detector(bytes('yolov3.cfg', encoding='utf-8'), bytes('./yolov3.weights', encoding='utf-8'), 0,
bytes('./data/coco.data', encoding='utf-8'))
7
8      img = cv2.imread('./sample.jpg')
9      img_darknet = Image(img)
10
11     results = net.detect(img_darknet)
12
13     for cat, score, bounds in results:
14         x, y, w, h = bounds
15         cv2.rectangle(img, (int(x - w / 2), int(y - h / 2)), (int(x + w / 2), int(y + h / 2)), (255, 0, 0),
thickness=2)
16         cv2.putText(img, str(cat.decode('utf-8')), (int(x), int(y)), cv2.FONT_HERSHEY_COMPLEX, 1, (255, 255, 0))
17
18     cv2.imshow('output', img)
19     cv2.waitKey(0)
```

- Initialize Detector
- Read Image
- Process Image w/Detector
- Draw Bounding Box & Label Around Detections
- Display

Results - Image



Python Example – Video

```
$ cat -n example
1      #!/usr/bin/python3
2
3      from pydarknet import Detector, Image
4      import cv2
5
6      net = Detector(bytes('yolov3.cfg', encoding='utf-8'), bytes('./yolov3.weights', encoding='utf-8'), 0,
bytes('./data/coco.data', encoding='utf-8'))
7      vidCapture = cv2.VideoCapture('video.mp4')
8      K=0;
9      gotFrame=True;
10     while(vidCapture.isOpened() and gotFrame):
11         gotFrame, frame = vidCapture.read()
12         if gotFrame:
13             print('frame',K);
14             img_darknet = Image(frame);
15             results = net.detect(img_darknet);
16             for cat, score, bounds in results:
17                 x, y, w, h = bounds
18                 cv2.rectangle(frame, (int(x - w / 2), int(y - h / 2)), (int(x + w / 2), int(y + h / 2)), (255, 0,
0), thickness=2)
19                 cv2.putText(frame, str(cat.decode('utf-8')), (int(x), int(y)), cv2.FONT_HERSHEY_COMPLEX, 1, (255, 255, 0))
20                 cv2.imwrite('frame%03d.jpg'%(K), frame);
21             K+=1;
22     vidCapture.release();
23     cv2.destroyAllWindows();
```

- Initialize Detector
- Foreach Image
- Process Image w/Detector
- Draw Bounding Box & Label Around Detections
- Save Image

Results - Video



Performance

- Intel(R) Core(TM) i5-3210M CPU @ 2.50GHz
 - 8 year old dinosaur
 - ~30 sec / frame detection latency
- 1920x1080, 960x540, 480x270, 240x130
 - Approximately same detection latency
 - Contradicts typical pixel-wide image processing performance
- Actual Mileage will Vary
- GPU Acceleration is possible
- Real-Time Performance on COTS Systems (30 fps)

Contact Info

- Slides:
 - <https://github.com/fsk-software/pub/>
 - YOLO-pymntos
- FSK Consulting Inc.
 - <http://fsksoftware.com>
- Blog: <http://dragonquest64.blogspot.com>
- Slack: pymntos.slack.com [lipeltgm](https://lipeltgm.slack.com)

