Advanced Deep Learning in Computer Vision

Day 1

Transfer Learning and Object Detection

Materials:
http://bit.ly/ADLCV_Jan21

Warm up!

Step 1: Go to the following url

http://bit.ly/warmup

Step 2: facilitator will walk you through the following questions

10 mins

Introduction of Trainer

Mr Seow Khee Wei

Name
Seow Khee Wei

Telegram
@kwseow

Email
seow_khee_wei@rp.edu.sg

Dr Jimmy Goh

Name
Jimmy Goh

Telegram
@jimmygohRP

Email
Jimmy_goh@rp.edu.sg

Programme

Day 1	Transfer Learning Activity – Transfer Learning Fine Tuning Object Detection and Localization Activity – Localization using Haar Cascades	More Object Detection and Localization Activity – Using YOLOv3 and SSD Annotation Activity – Annotation Hands-on
Day 2	Image Segmentation Activity – - OpenCV Mask RCNN - Keras Mask RCNN - Training Customized Mask RCNN	Activity: Using Customized Mask RCNN Face Detection and Recognition Activity: - Create Face Database - Face Recognition
Day 3	Advanced Generative Adversarial Network Activity - DCGAN for small color photographs - Conditional GAN	Customised Dataset with Yolo Activity - Thermal Images - Aerial Images

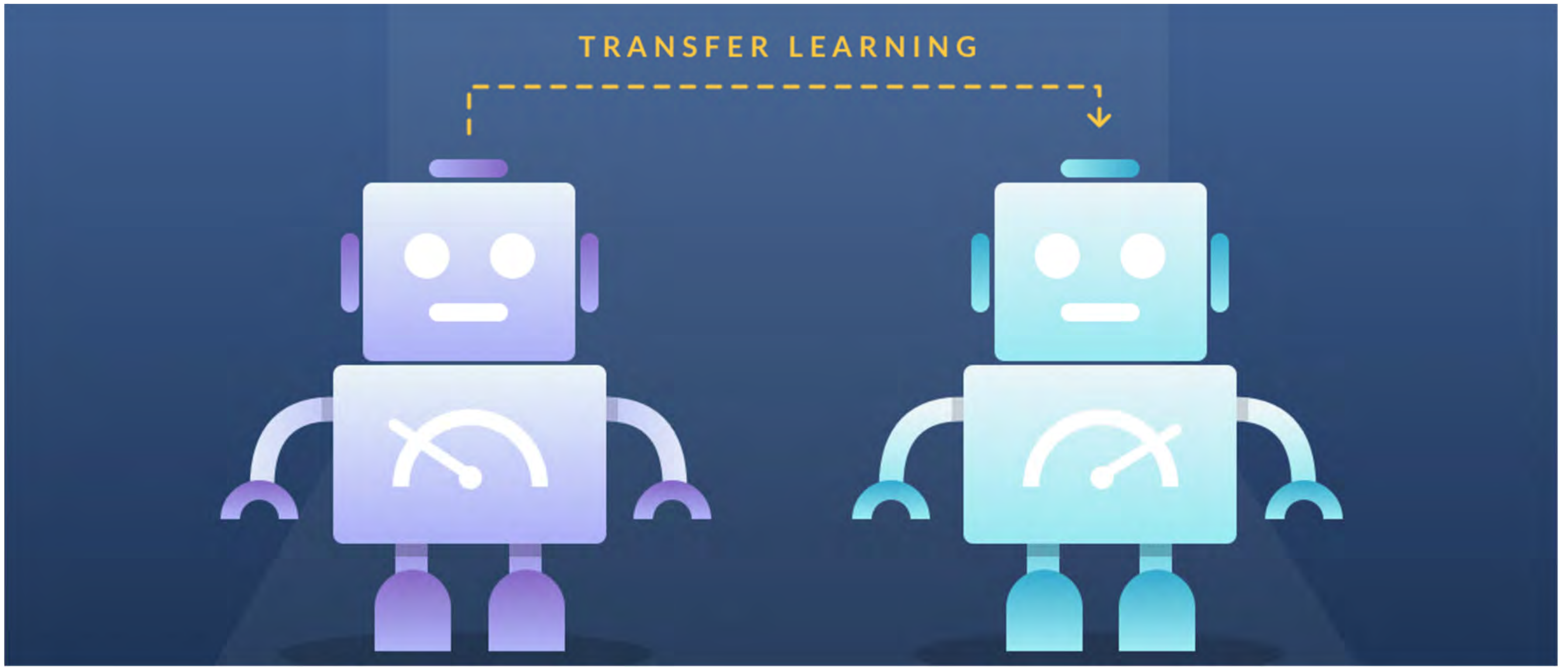


Image credit: Datacamp

Transfer Learning

What is Transfer Learning

- Transfer learning (TL) is a research problem in machine learning (ML) that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem.
  - Wikipedia

- Transfer learning is the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned.
  - Chapter 11: Transfer Learning, Handbook of Research on Machine Learning Applications, 2009.

What is Transfer Learning

- **Models are difficult to train from scratch**
  - Huge datasets (like ImageNet)
  - Long number of training iterations
  - Very heavy computing machinery
  - Time experimenting to get hyper-parameters just right

What is Transfer Learning

- **Basic idea**
  - Early layers in a Neural Network are the hardest (i.e. slowest) to train
  - Due to vanishing gradient property
  - But these ”primitive” features should be general across many image classification tasks
  - Later layers in the network are capturing features that are more particular to the specific image classification problem.
  - Later layers are easier (quicker) to train since adjusting their weights has a more immediate impact on the final result.
  - **Idea**: keep the early layers of a pre-trained network, and re-train the later layers for a specific application

Pre-trained CNN models

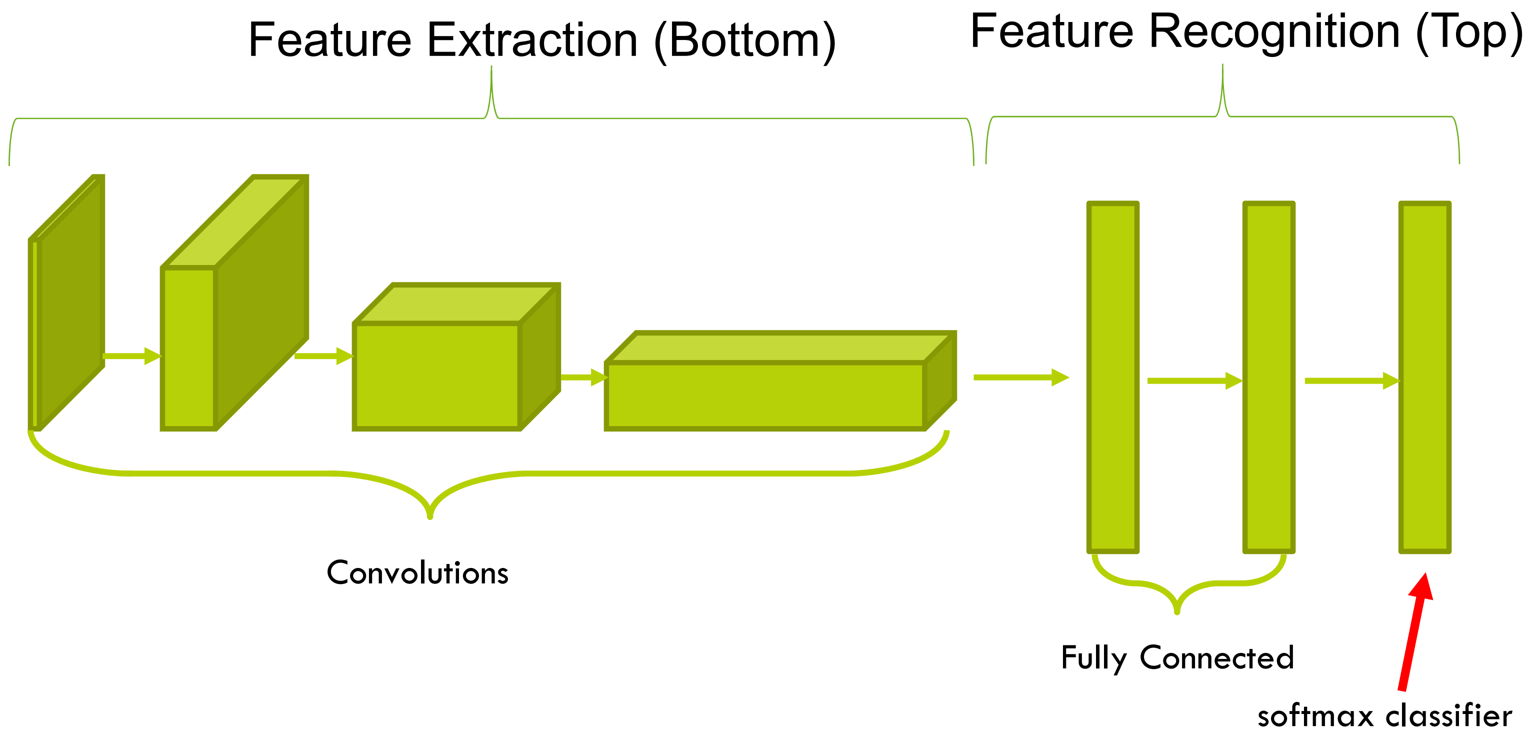
- **Models for image classification with weights trained on ImageNet:**
  - Xception
  - VGG16
  - VGG19
  - ResNet, ResNetV2
  - InceptionV3
  - InceptionResNetV2
  - MobileNet
  - MobileNetV2
  - DenseNet
  - NASNet

Pretrained CNN models

- **ImageNet (http://image-net.org/index)**
  - ImageNet is an image dataset organized according to the WordNet hierarchy. Each meaningful concept in WordNet, possibly described by multiple words or word phrases, is called a "synonym set" or "synset". There are more than 100,000 synsets in WordNet, majority of them are nouns (80,000+). In ImageNet, we aim to provide on average 1000 images to illustrate each synset. Images of each concept are quality-controlled and human-annotated. In its completion, we hope ImageNet will offer tens of millions of cleanly sorted images for most of the concepts in the WordNet hierarchy.

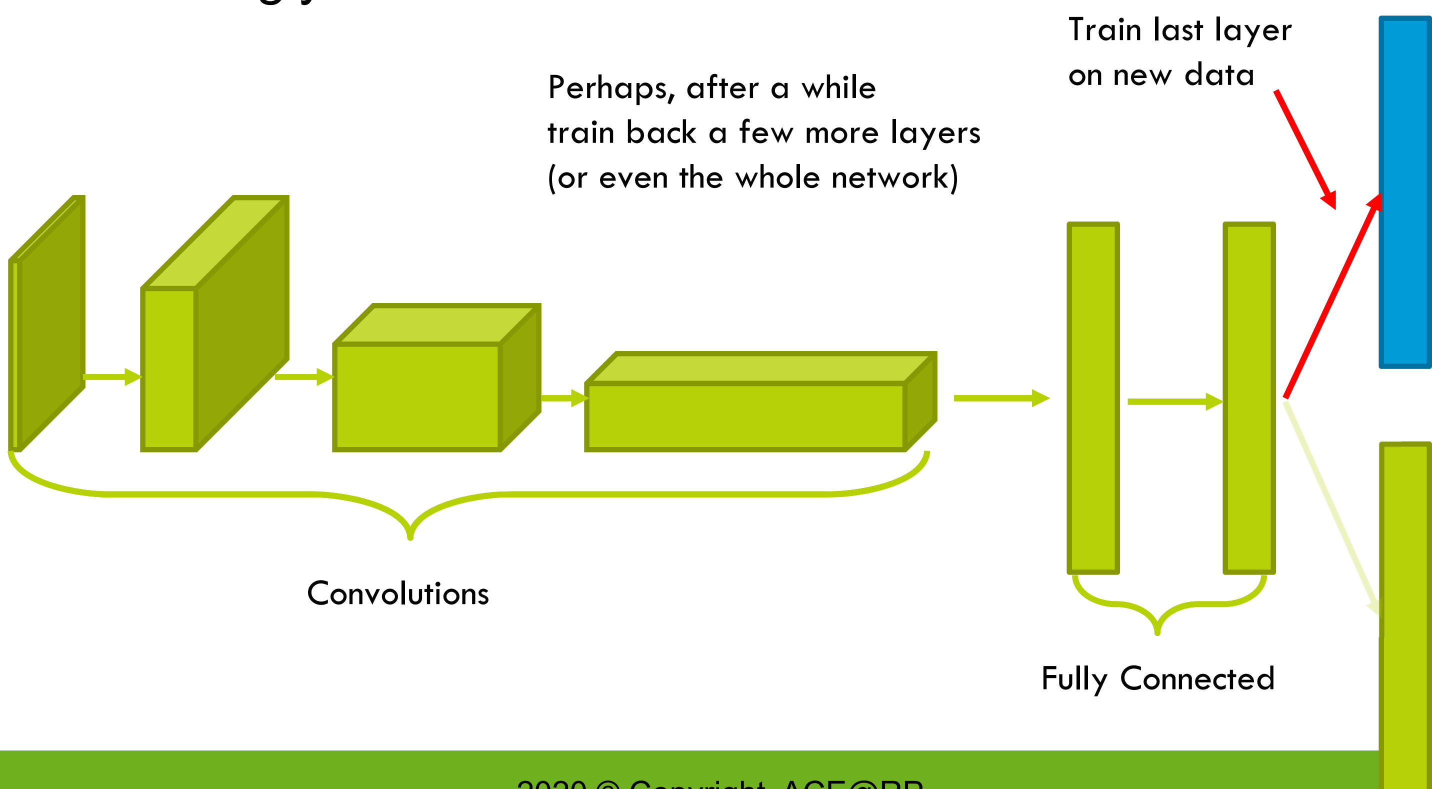
How to do Transfer Learning

- **A typical pre-trained CNN Model**



How to do Transfer Learning

- **Reconstruct Top layer**
  - Change the output layer to suit your needs
    - E.g different number of categories, detect different features
  - Adjust the number of layers and nodes of the hidden layers accordingly



How to do Transfer Learning

- **Fine Tuning**
  - The additional training of a pre-trained network on a specific new dataset is referred to as "***Fine-Tuning***"
  - There are different options on "how much" and "how far back" to fine-tune.
    - Should I train just the very last layer?
    - Go back a few layers?
    - Re-train the entire network (from the starting point of the existing network)?
  - While there are no "hard and fast" rules, there are some guiding principles to keep in mind.

When to use Transfer Learning

- Principle 1:

The more similar your data and problem are to the source data of the pre-trained network, the less fine-tuning is necessary.

- E.g. Using a network trained on ImageNet to distinguish "dogs" from "cats" should need relatively little fine-tuning. It already distinguished different breeds of dogs and cats, so likely has all the features you will need.

When to use Transfer Learning

- Principle 2:

The more data you have about your specific problem, the more the network will benefit from longer and deeper fine-tuning.

➢ E.g. If you have only 100 dogs and 100 cats in your training data, you probably want to do very little fine-tuning. If you have 10,000 dogs and 10,000 cats you may get more value from longer and deeper fine-tuning.

When to use Transfer Learning

- Principle 3:

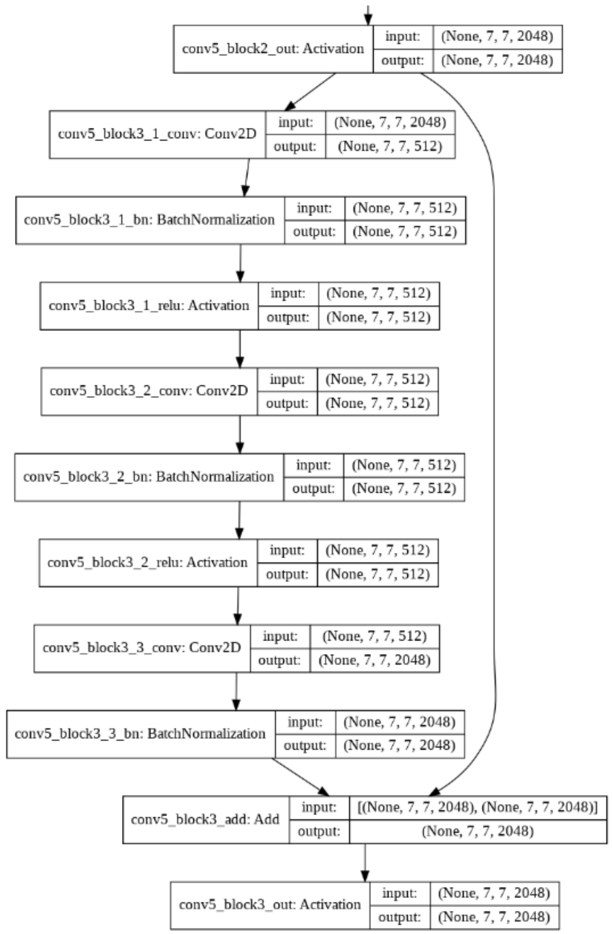
If your data is substantially different in nature than the data the source model was trained on, Transfer Learning may be of little value.

- E.g. A network that was trained on recognizing typed Latin alphabet characters would not be useful in distinguishing cats from dogs. But it likely would be useful as a starting point for recognizing Cyrillic Alphabet characters.

Resnet50

- Total 175 layers

Resnet50_plot.png



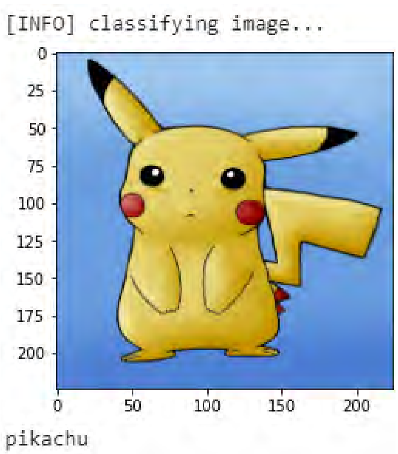
The last few layers of Resnet50

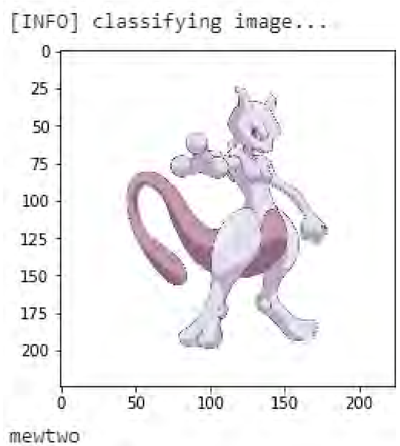
Some fine tuning comparison

- **Epoch = 200**
- **No fine tuning** test_acc: 0.1818181872367859
- **2 dense layer (128,64)** test_acc: 0.18614718317985535
- **Fine tune conv5 block (32 layers)**
  - test_acc: 0.26406925916671753
  - CPU times: user 4min 52s, sys: 2min, total: 6min 52s
  - Wall time: 13min 45s
- **Fine tune conv5 and conv4 block (94 layers)**
  - test_acc: 0.6450216174125671
  - CPU times: user 8min 24s, sys: 4min 20s, total: 12min 45s
  - Wall time: 19min 45s
- **Complete training :**
  - test_acc: 0.9134199023246765
  - CPU times: user 15min 39s, sys: 9min 32s, total: 25min 12s
  - Wall time: 32min 38s

Transfer Learning

- **Activity: 1_1_Transfer_Learning**





Exercises:
- Make the Conv5 Block3 block trainable (32)
- Add 2 dense layers with 128 & 64 neuron
- Compare the number of trainable parameters and the final performance

Step 1:
Watch and listen to the instructor's demonstration

10 mins

Step 2:
Work through the activities

30 mins

Object Detection

What is Object Detection

- Object recognition is a general term to describe a collection of related computer vision tasks that involve identifying objects in digital photographs.

- ***Image classification*** involves predicting the class of one object in an image. ***Object localization*** refers to identifying the location of one or more objects in an image and drawing a bounding box around their extent. ***Object detection*** combines these two tasks and ***localizes and classifies*** one or more objects in an image.
  - Ref: https://machinelearningmastery.com/object-recognition-with-deep-learning/

What is Object Detection

- Object Detection example:

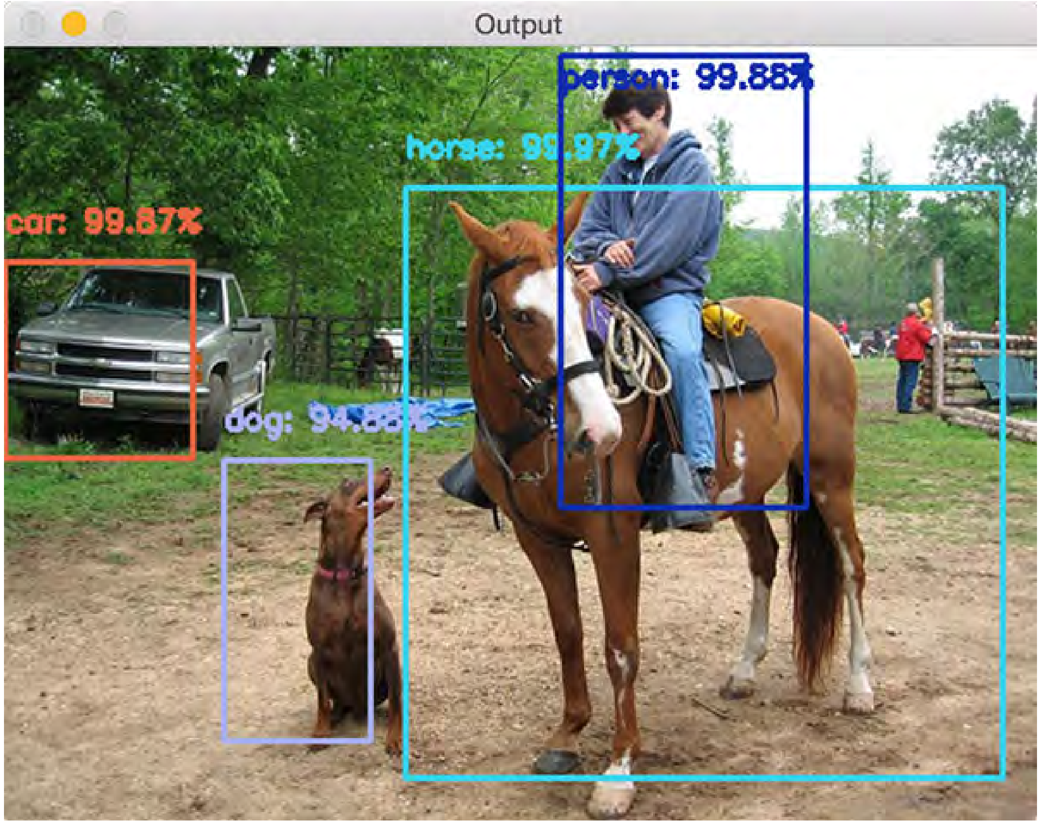


Image from: https://www.pyimagesearch.com/2017/09/11/object-detection-with-deep-learning-and-opencv/

What is Object Detection

- Object Localization example:

Image from:
https://machinelearningmastery.com/object-recognition-with-deep-learning/

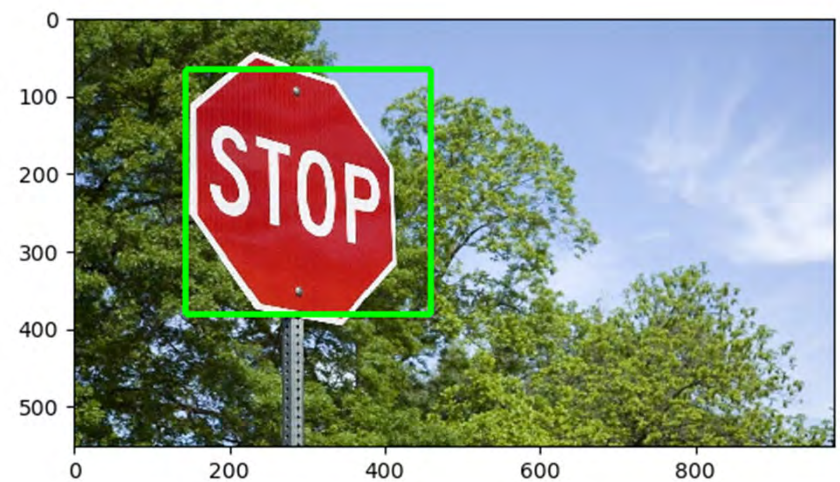


Image from:
https://machinelearningmastery.com/object-recognition-with-deep-learning/

Detection VS Localization

- Localization
  - Faster
  - Single-object of interest
  - Counting of objects
  - Using more complex recognition algorithm
    - E.g. Face recognition
- Detection
  - Slower
  - Multiple object of interest
  - Understanding the image

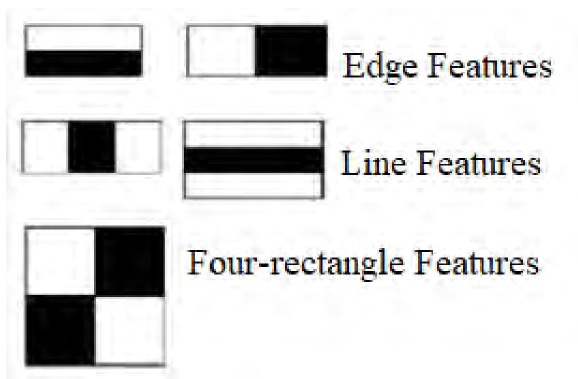
How to do Object Localization

- Haar Cascade classifier is an effective object detection approach which was proposed by *Paul Viola and Michael Jones* in their paper, "**Rapid Object Detection using a Boosted Cascade of Simple Features**" in 2001.
  - Ref: https://towardsdatascience.com/computer-vision-detecting-objects-using-haar-cascade-classifier-4585472829a9

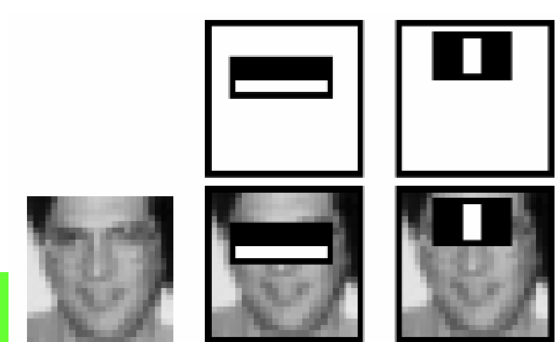
- Viola-Jones Object Detection Framework (Non-Deep Learning)
  - Ref: https://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework

How to do Object Localization

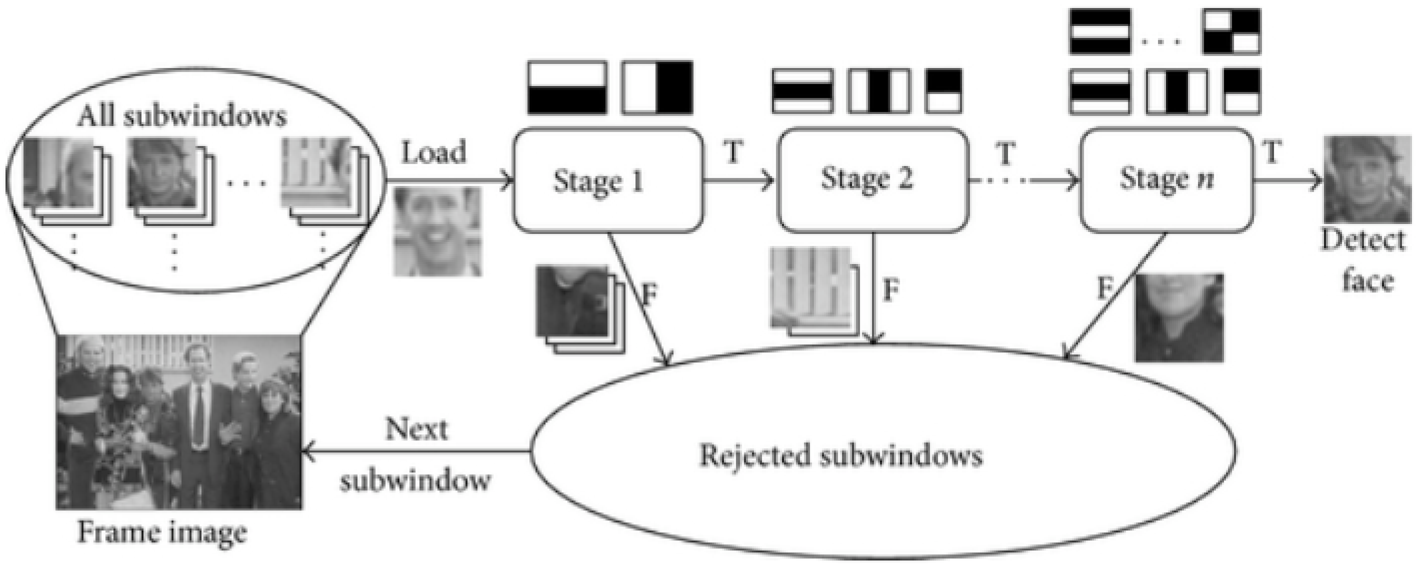
- Viola-Jones Object Detection Framework
  - Haar features
    - *Different sizes* of the Haar features very quickly scan through an image several times.



  - Haar features for Face Detection

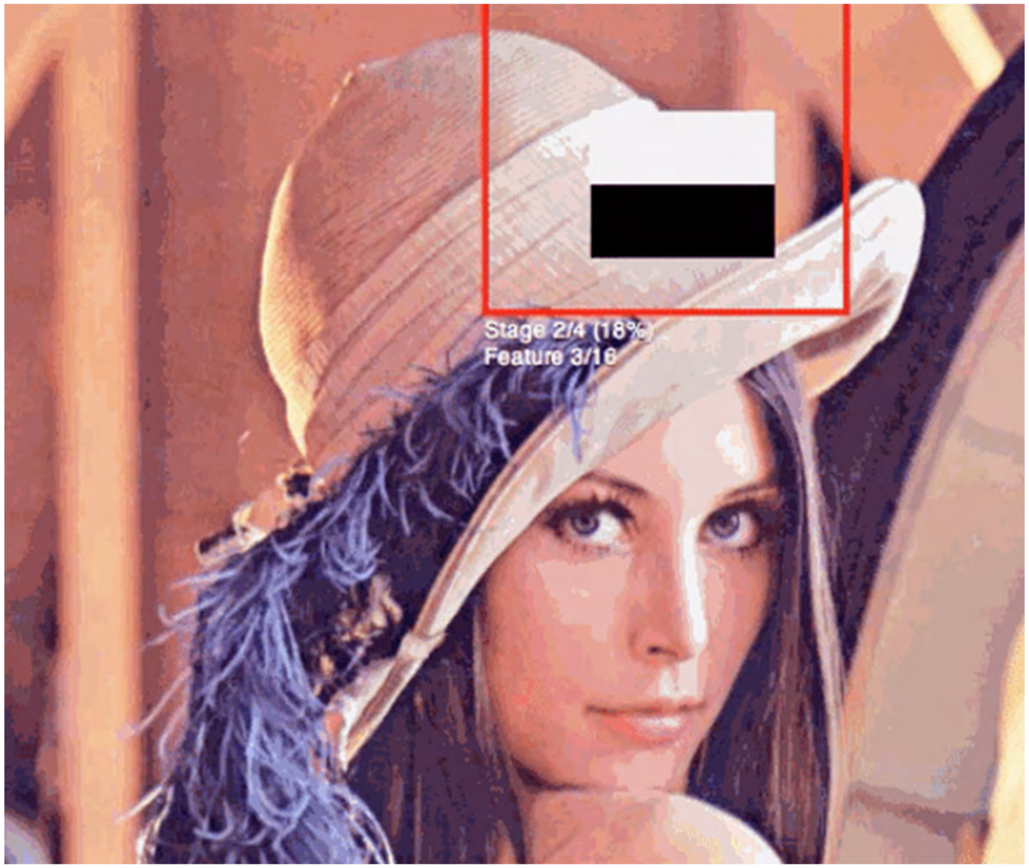


Cascade structure



https://www.researchgate.net/figure/Cascade-structure-for-Haar-classifiers_fig9_277929875

The Paul- Viola visualisation



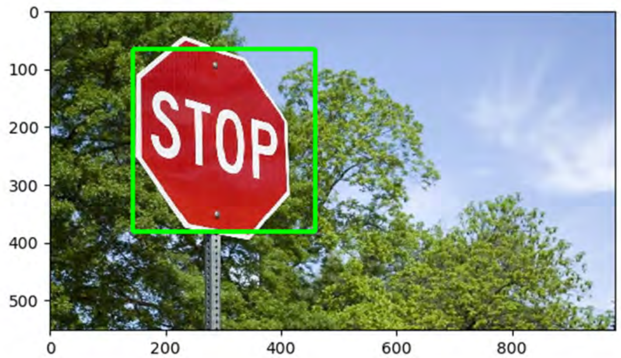
https://vimeo.com/12774628

How to do Object Localization

- Haar Features file (.xml) for different objects
  - Face, eyes, upper body, full body
  - Licence plate, car, clock, signs
  - A lot more….
- Websites to download the Haar Features file
  - https://github.com/opencv/opencv/tree/master/data/haarcascades
  - http://alereimondo.no-ip.org/OpenCV/34/
  - https://github.com/anaustinbeing/haar-cascade-files
- Train your own Haar Features file.
  - https://docs.opencv.org/master/dc/d88/tutorial_traincascade.html
  - https://pythonprogramming.net/haar-cascade-object-detection-python-opencv-tutorial/

Activity – Haar Cascades

- Activity: **2_1_Localization_using_HaarCascades**



Exercises:
- Download the Russian number plate haar cascade xml from https://github.com/anaustinbeing/haar-cascade-files
- Perform a license plate detection on cars.jpg and car2.jpg

Step 1:
Watch and listen to the instructor's demonstration

10 mins

Step 2:
Work through the activities

30 mins

Object Detection

- **Three primary object detection algorithm using Deep Learning**
  - R-CNN → RCNN, Fast-RCNN, Faster-RCNN
    - Slow
    - Difficult to understand and implement
  - YOLO → YOLO, YOLOv2, YOLOv3, YOLOv4, YOLOv5
    - Fast
    - Ease to customised
    - Works best with natural images
  - SSD
    - Even faster
    - Ease to customised
    - Works best with low resolution or ‘small’ objects

R-CNN

- **R-CNN (Region Proposal – CNN)**
  - Feature Extractor
    - To extract features from the image
  - Region Proposal Network
    - To both propose and refine region proposals
  - ROI pooling
    - To classify region of interest

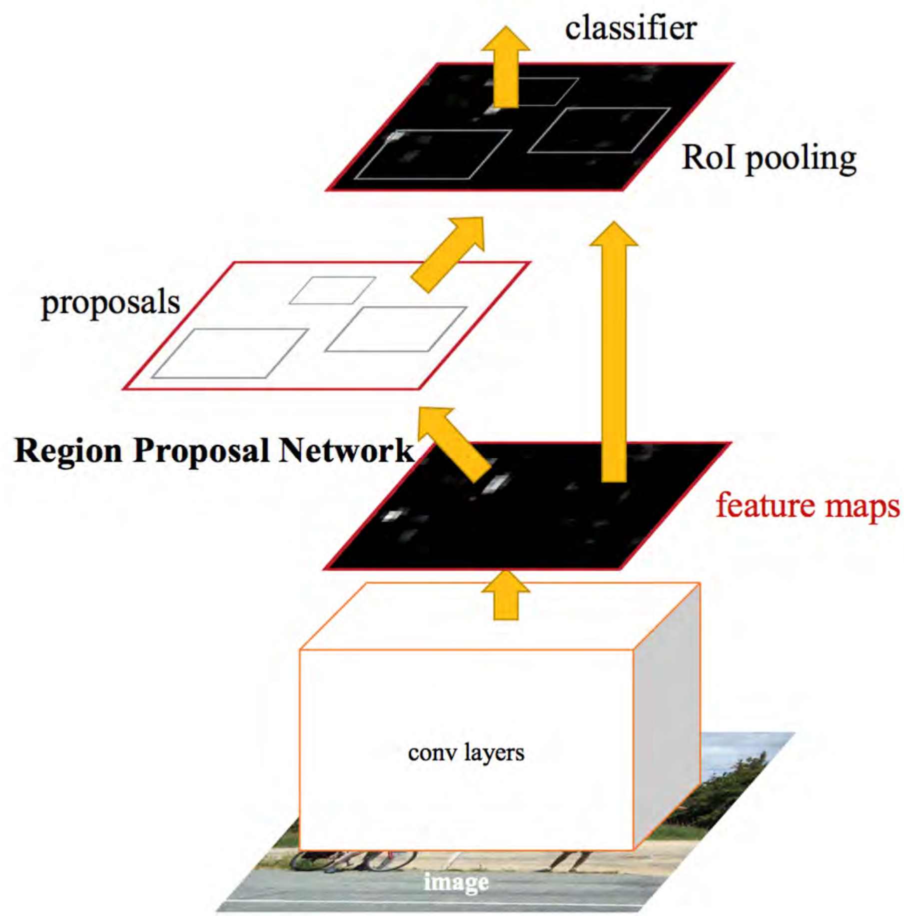
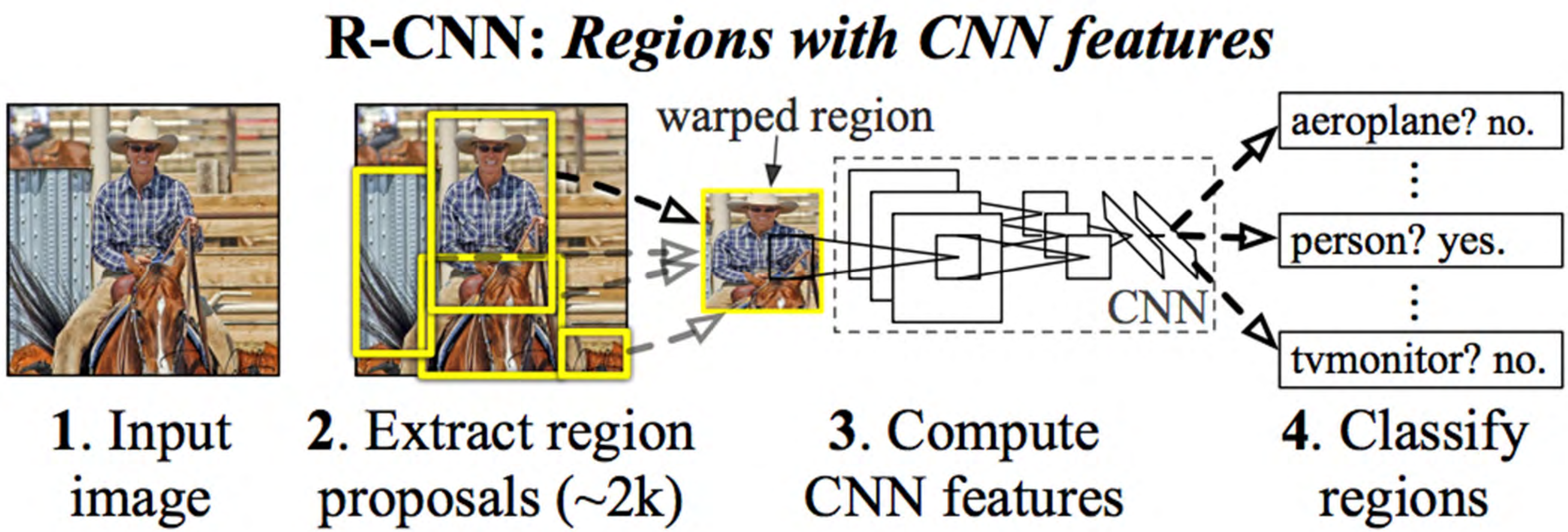


Image from: https://machinelearningmastery.com/object-recognition-with-deep-learning/

R-CNN Model Architecture

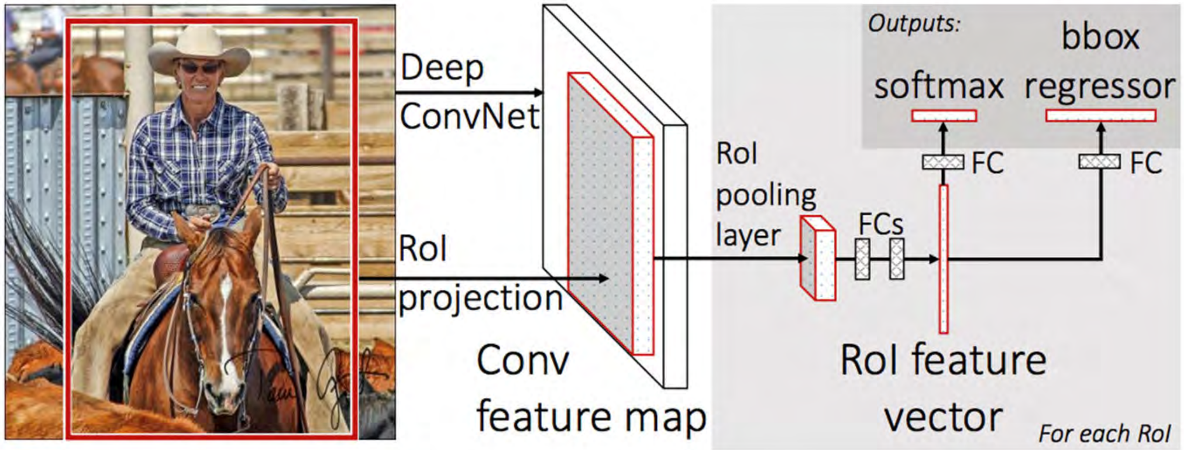


Summary of the R-CNN Model Architecture Taken from Rich feature hierarchies for accurate object detection and semantic segmentation.

Fast R-CNN

- limitations of R-CNN:
  - Training is a multi-stage pipeline. Involves the preparation and operation of three separate models.
  - Training is expensive in space and time. Training a deep CNN on so many region proposals per image is very slow.
  - Object detection is slow. Make predictions using a deep CNN on so many region proposals is very slow.

Fast R-CNN is proposed as a single model instead of a pipeline to learn and output regions and classifications directly.



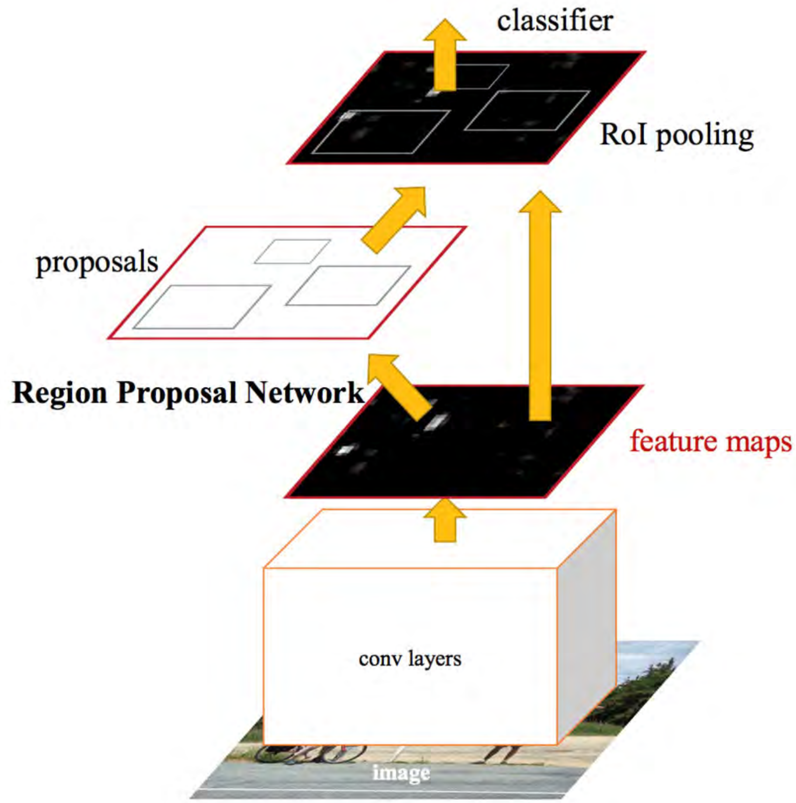
Summary of the Fast R-CNN Model Architecture.
Taken from: Fast R-CNN.

*

Faster R-CNN

The model architecture was further improved for both speed of training and detection by Shaoqing Ren, et al. at Microsoft Research in the 2016 paper titled "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks."

Although it is a single unified model, the architecture is comprised of two modules:
Module 1: Region Proposal Network. Convolutional neural network for proposing regions and the type of object to consider in the region.
Module 2: Fast R-CNN. Convolutional neural network for extracting features from the proposed regions and outputting the bounding box and class labels.



YOLO

- **YOLO (You Only Look Once)**
  - Divide an image into grids
  - Each grid
    - Bounding box prediction
    - Classification
  - Combine results
    - Non-Maximum Suppression (NMS)

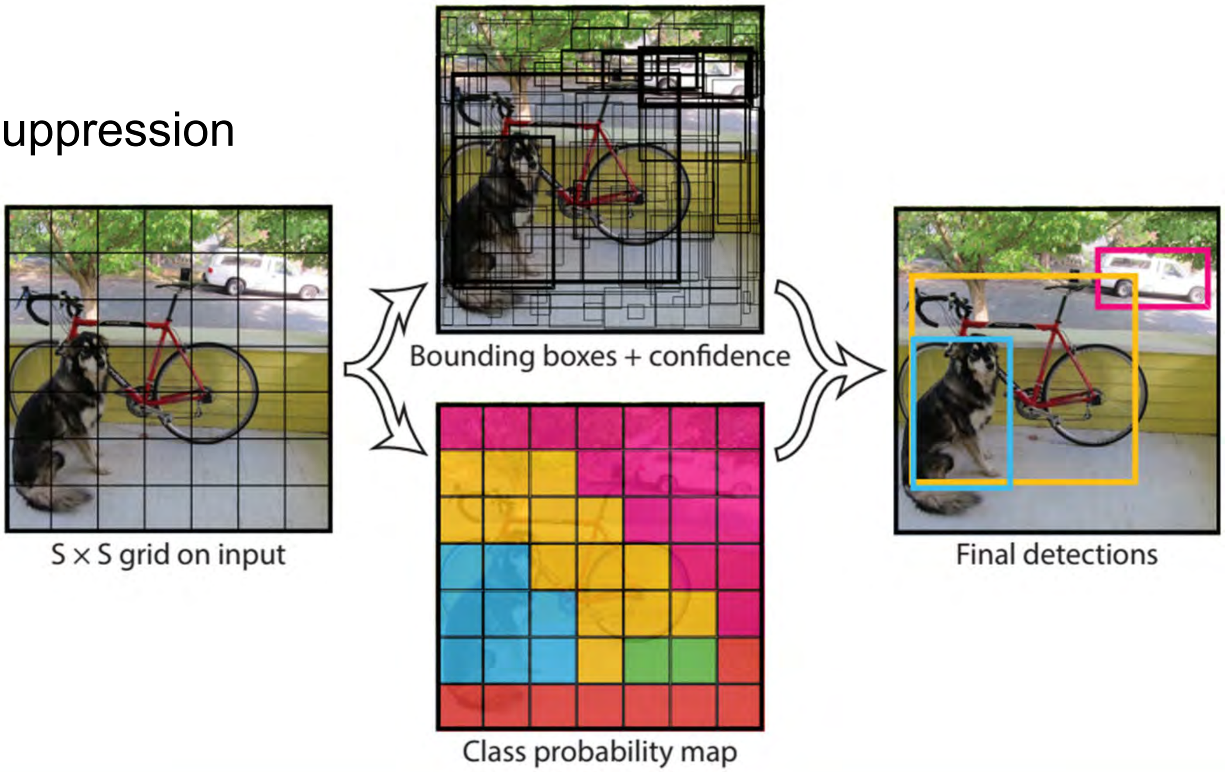
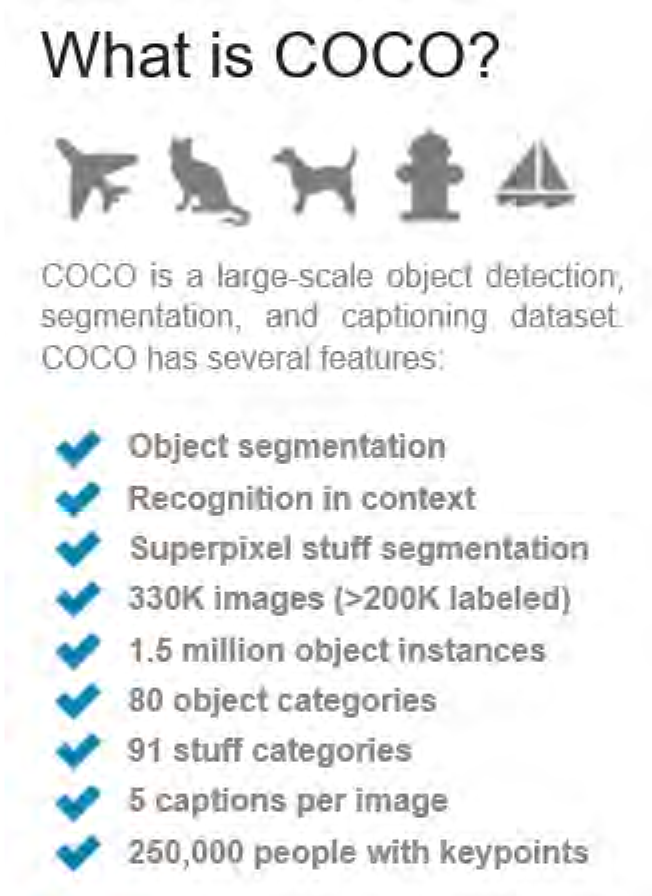


Image from: https://machinelearningmastery.com/object-recognition-with-deep-learning/

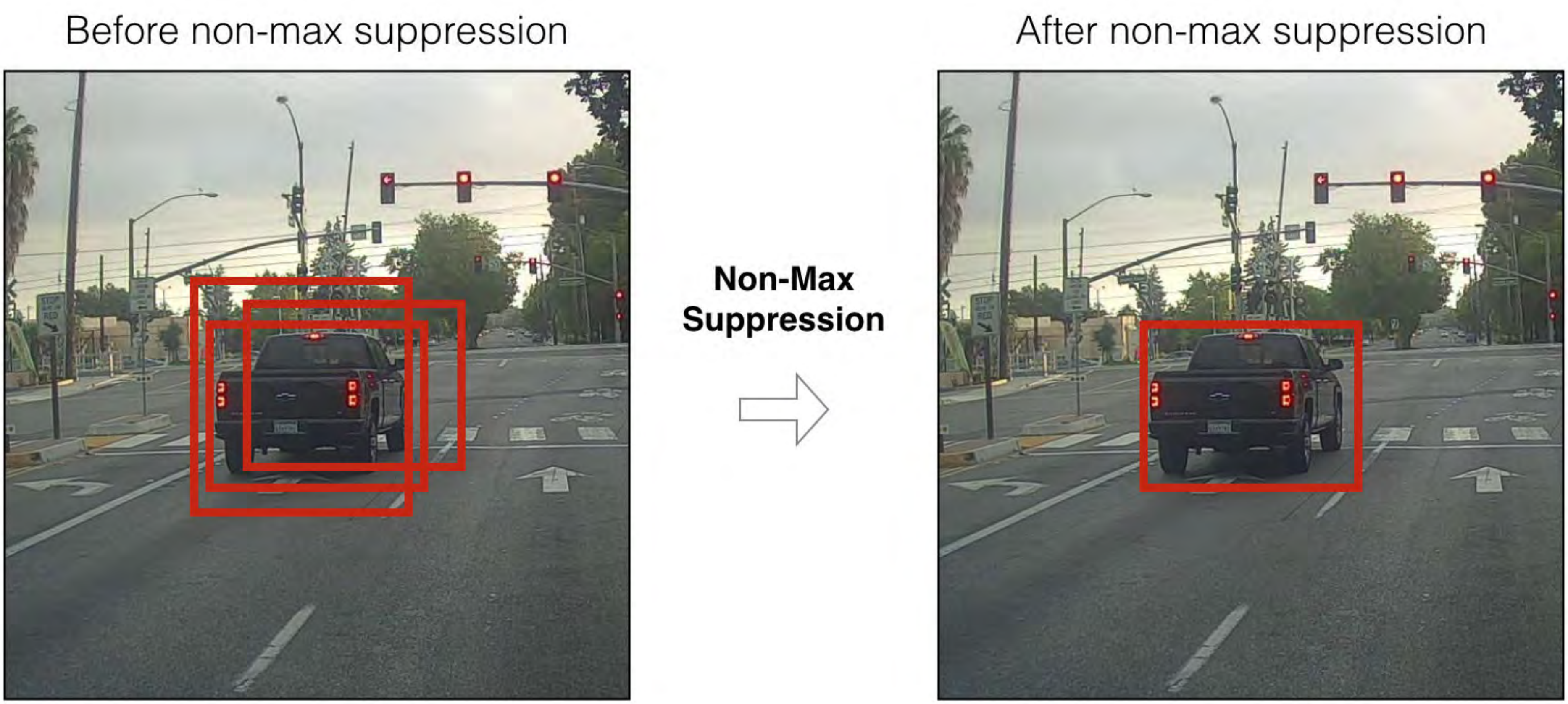
Pre-trained YOLO/SSD Model

- **Using COCO dataset**
  - Common Objects in COntext
  - COCO is a large-scale object detection, segmentation, and captioning dataset.
    - https://cocodataset.org/#home
  - 80 classes, 80,000 training images and 40,000 validation images.



- **Customized training**
  - https://blog.francium.tech/custom-object-training-and-detection-with-yolov3-darknet-and-opencv-41542f2ff44e
  - On Day 3, we will perform customised training on YOLO(V5) and Scaled-YOLO(V4) on satellite and thermal images.

Non Maximal Suppression (NMS)



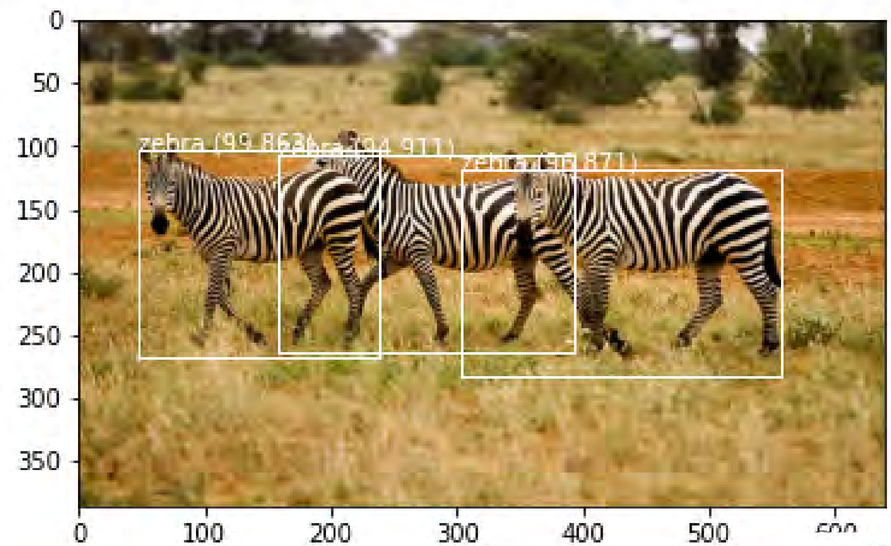
https://towardsdatascience.com/non-maximum-suppression-nms-93ce178e177c

Activity – YOLOv3

- **Activity: 2_2_Object_Detection_using_YOLOv3**

  - Load model
  - Predict
  - Decode the prediction
    - Correct box sizes
    - Perform NMS
    - Filter predicted class
    - Draw boxes

```
zebra 94.91059184074402
zebra 99.86329674720764
zebra 96.87087535858154
```



Exercises:
- Modify the code such that only zebras that the model is more than 97% confident are plotted.

Step 1:
Watch and listen to the instructor's demonstration

10 mins

Step 2:
Work through the activities

30 mins

Single Shot MultiBox (SSD)

- **SSD (Single Shot MultiBox Detectors)**
  - Designed for object detection in real-time
  - Speeds up by eliminating the need for region proposal network (Faster R-CNN)
  - Uses multi-scales features and default boxes
  - Match Faster RCNN accuracy using lower resolution images

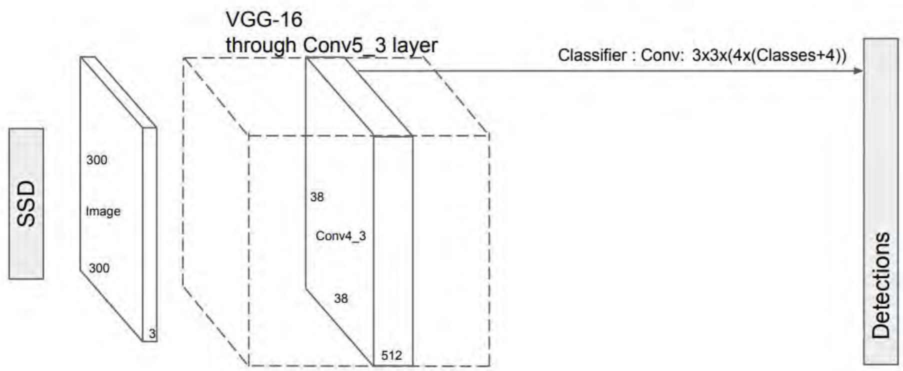
System	VOC2007 test *mAP*	FPS (Titan X)	Number of Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	~6000	~1000 x 600
YOLO (customized)	63.4	45	98	448 x 448
SSD300* (VGG16)	77.2	46	8732	300 x 300
SSD512* (VGG16)	**79.8**	19	24564	512 x 512

Source: https://arxiv.org/pdf/1512.02325.pdf

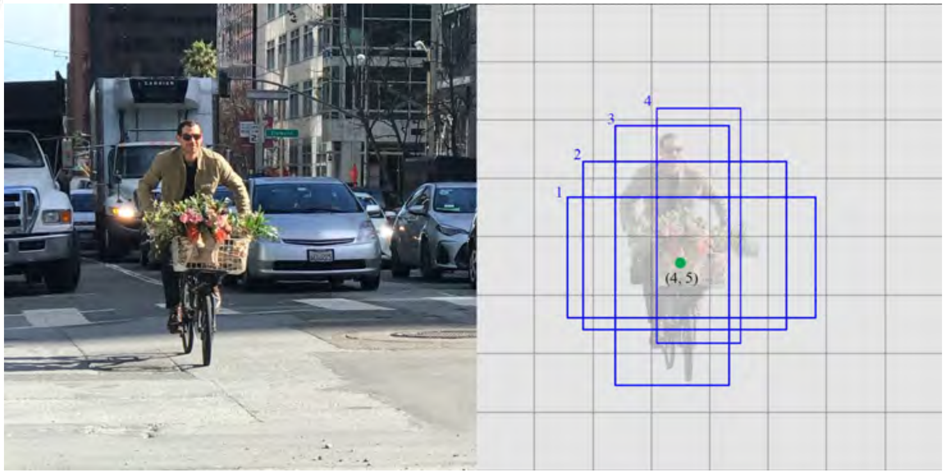
Single Shot MultiBox (SSD)

The SSD object detection composes of 2 parts:

1) Extract feature maps



SSD uses **VGG16** to extract feature maps. Then it detects objects using the **Conv4_3** layer. For illustration, we draw the Conv4_3 to be 8 × 8 spatially (it should be 38 × 38). For each cell (also called location), it makes 4 object predictions.



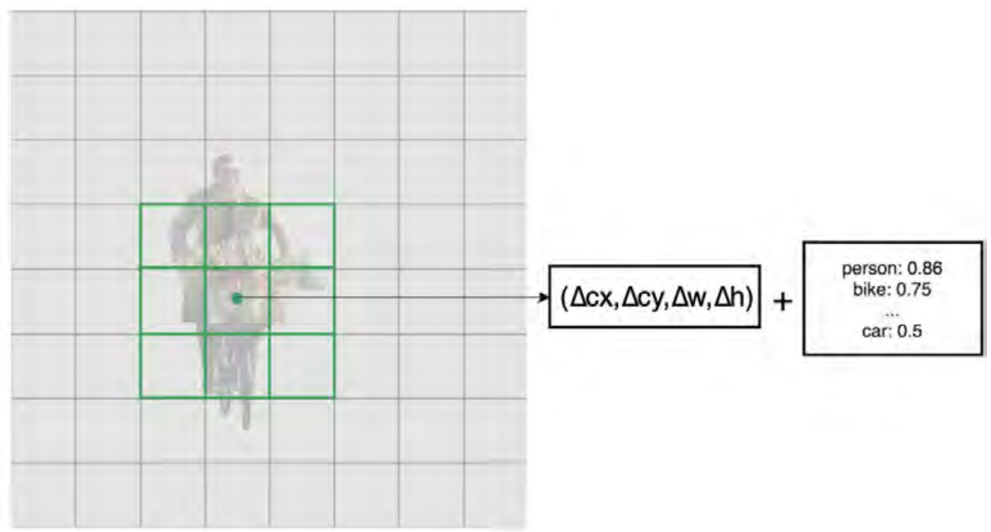
Detailed discussion:
https://medium.com/@jonathan_hui/ssd-object-detection-single-shot-multibox-detector-for-real-time-processing-9bd8deac0e06

Single Shot MultiBox (SSD)

The SSD object detection composes of 2 parts:

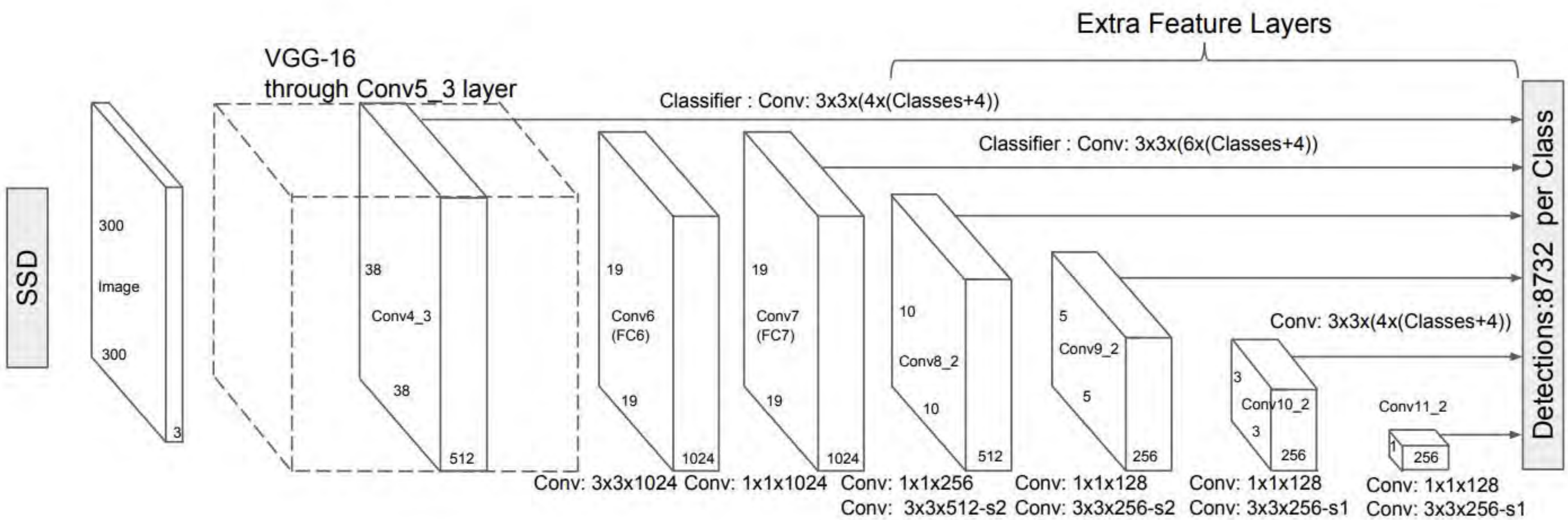
2) Apply convolution filters to detect objects.

SSD does not use a delegated region proposal network. Instead, it resolves to a very simple method. It computes both the location and class scores using **small convolution filters**. After extracting the feature maps, SSD applies 3 × 3 convolution filters for each cell to make predictions. (These filters compute the results just like the regular CNN filters.)



Single Shot MultiBox (SSD)

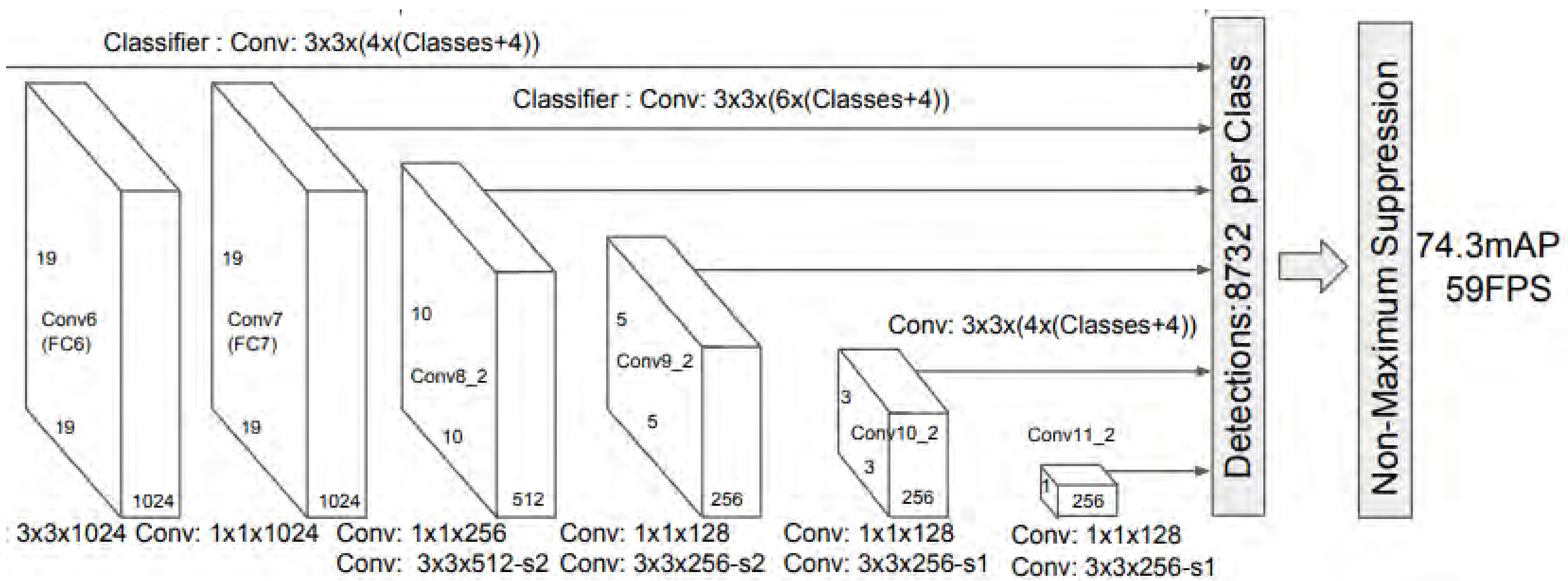
- We describe how SSD detects objects from a single layer. Actually, it uses multiple layers (**multi-scale feature maps)** to detect objects independently.
- SSD adds 6 more auxiliary convolution layers after the VGG16. Five of them will be added for object detection.



Source: [SSD: Single Shot MultiBox Detector]

Single Shot MultiBox (SSD)

- Default boundary boxes are chosen manually for matching.
- SSD uses **non-maximum suppression** to remove duplicate predictions pointing to the same object. SSD sorts the predictions by the confidence scores



Source: SSD: Single Shot MultiBox Detector

Some findings

- SSD performs worse than Faster R-CNN for small-scale objects. Accuracy increases with the number of default boundary boxes at the cost of speed.
- Design better default boundary boxes will help accuracy.
- COCO dataset has smaller objects. To improve accuracy, use smaller default boxes (start with a smaller scale at 0.15).
- SSD has lower localization error comparing with R-CNN but more classification error dealing with similar categories. The higher classification errors are likely because we use the same boundary box to make multiple class predictions.

Activity - SSD

- **Activity: 2_3_Object_Detection_using_SSD**

Exercises:
- Use SSD on the zebra image used in activity 2_2.
- Compare the speed and performance of the two models

Step 1:
Watch and listen to the instructor's demonstration

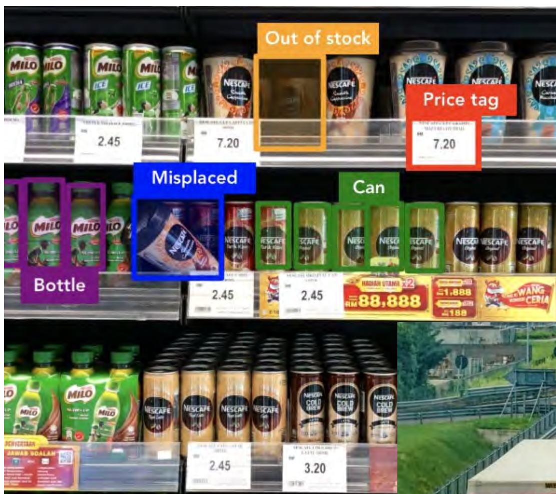
10 mins

Step 2:
Work through the activities

15 mins

Annotation of Images

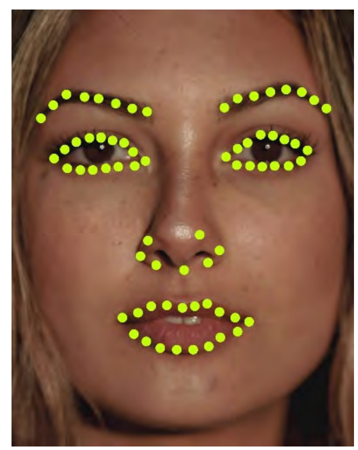
- Image annotation is a key technique used to create training data for computer vision.
- Types of Image Annotation

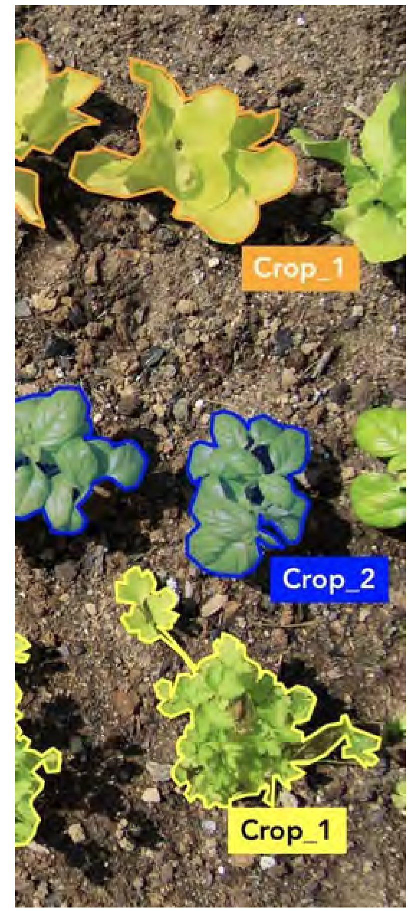


2-D Bounding Box


Segmentation/Mask


Line

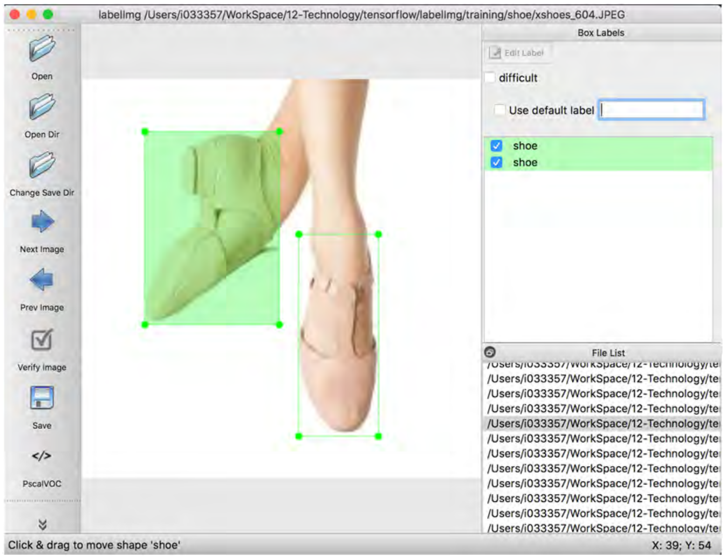

Point



Polygon

ref: https://medium.com/supahands-techblog/what-is-image-annotation-caf4107601b7

Some examples:




```
<annotation>
    <folder>shoe</folder>
    <filename>xshoes_001.JPEG</filename>
    <path>/Users/i033357/WorkSpace/12-Technology/tensorflow/labelImg/training/
    <source>
        <database>Unknown</database>
    </source>
    <size>
        <width>272</width>
        <height>272</height>
        <depth>3</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>shoe</name>
        <pose>Unspecified</pose>
        <truncated>1</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>1</xmin>
            <ymin>155</ymin>
            <xmax>228</xmax>
            <ymax>250</ymax>
        </bndbox>
    </object>
    <object>
        <name>shoe</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>1</difficult>
        <bndbox>
            <xmin>46</xmin>
            <ymin>150</ymin>
            <xmax>271</xmax>
            <ymax>238</ymax>
        </bndbox>
    </object>
</annotation>
```


Annotation of Images

- **Activity: 2_4_Annotation_of_images**
- Search on the internet and download 5 images of kick scooters.
- Follow the instructions in "2_4_Annotation_of_images.docx" to perform annotation using www.makesense.ai
- **Ref: https://towardsdatascience.com/annotate-your-image-using-online-annotation-tool-52d0a742daff**

Exercises:
- Create a folder at http://bit.ly/3qpx1WV
- Upload your images and annotations to the newly created folder

Step 1:
Watch and listen to the instructor's demonstration

10 mins

Step 2:
Work through the activities

20 mins

Quiz

https://bit.ly/kw_poll

SCAN ME

Thank you