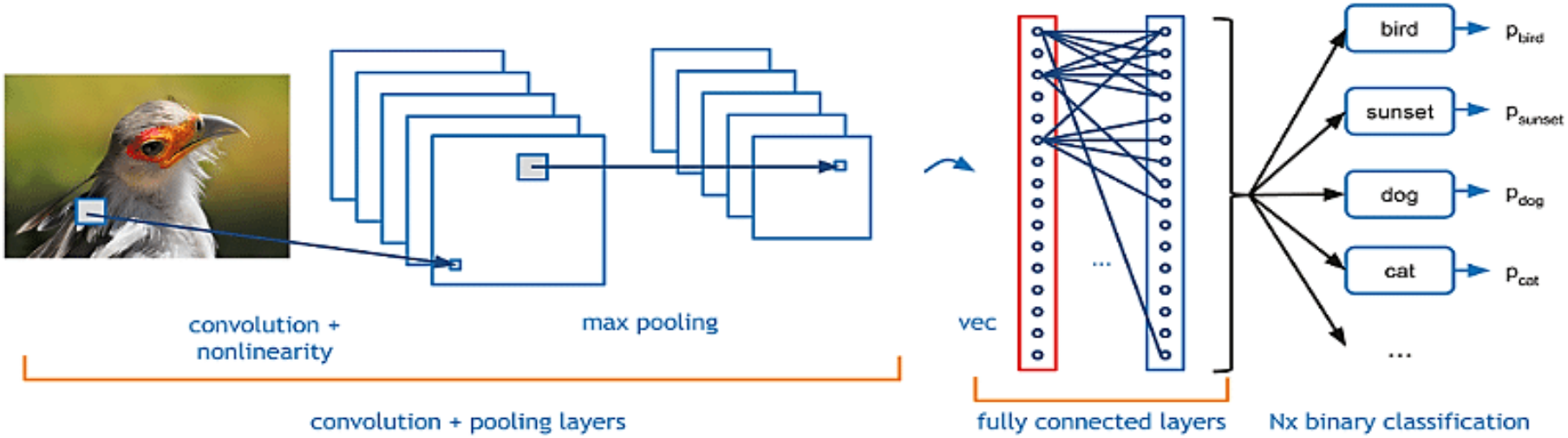
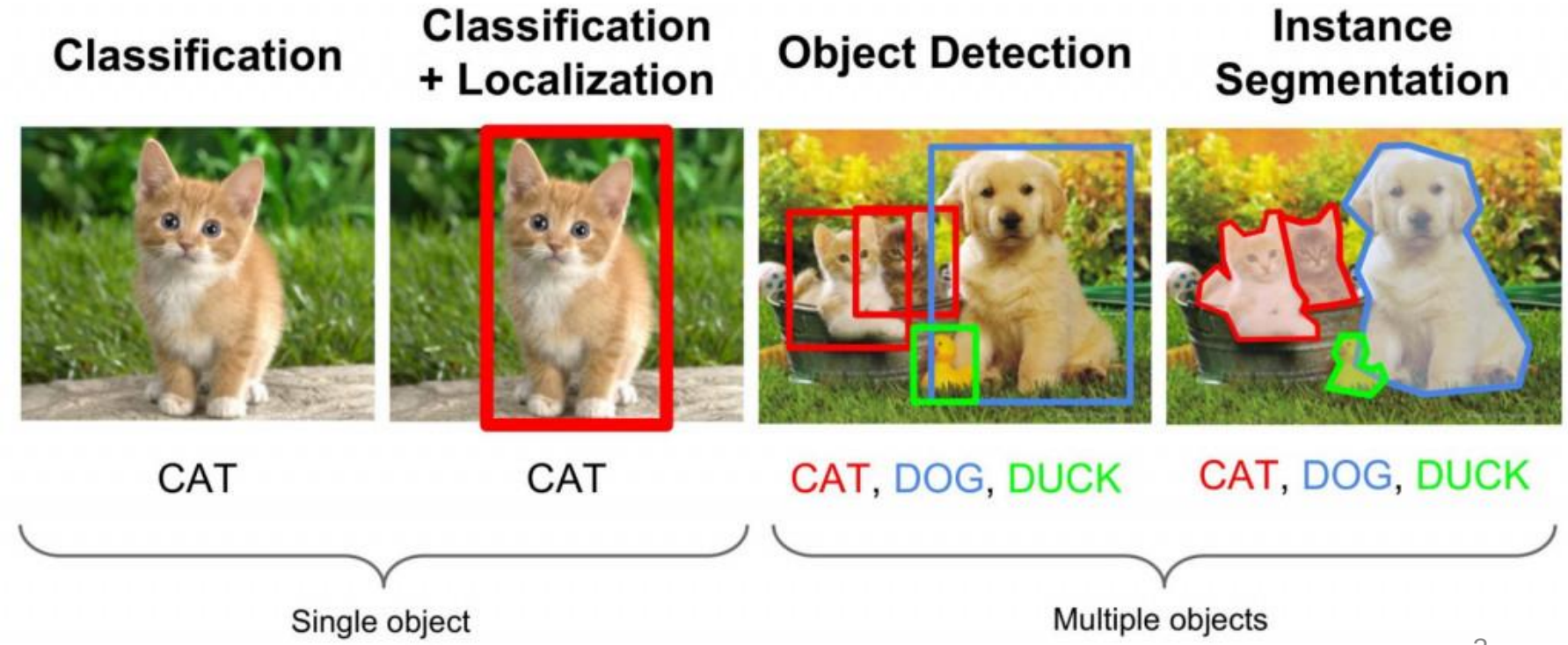
TM1117 AI Convolutional Neural Network (CNN)

Instructor: Summer Lo

Object classification with supervised learning



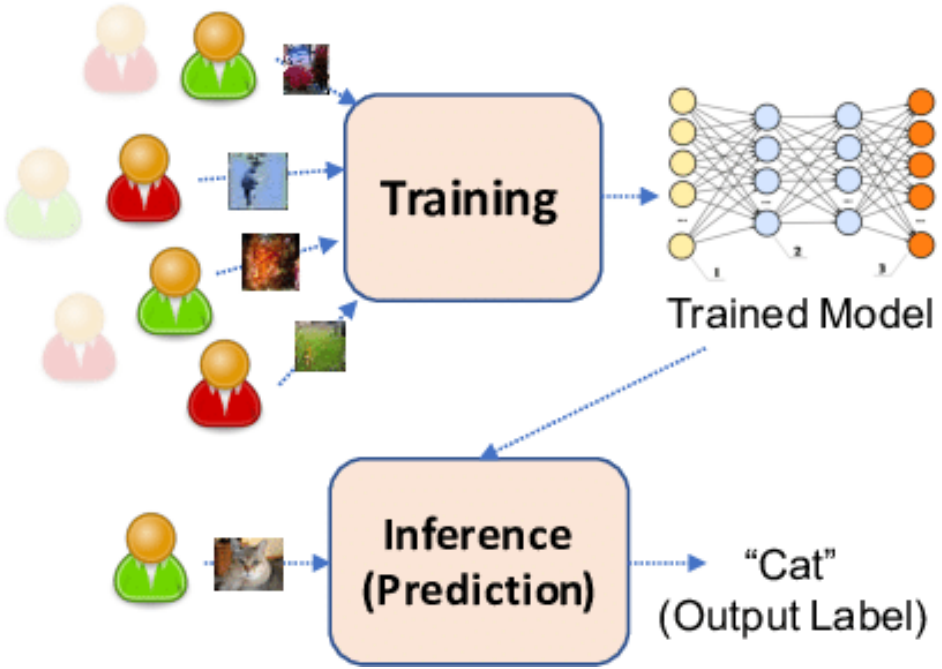
Object classification with supervised learning



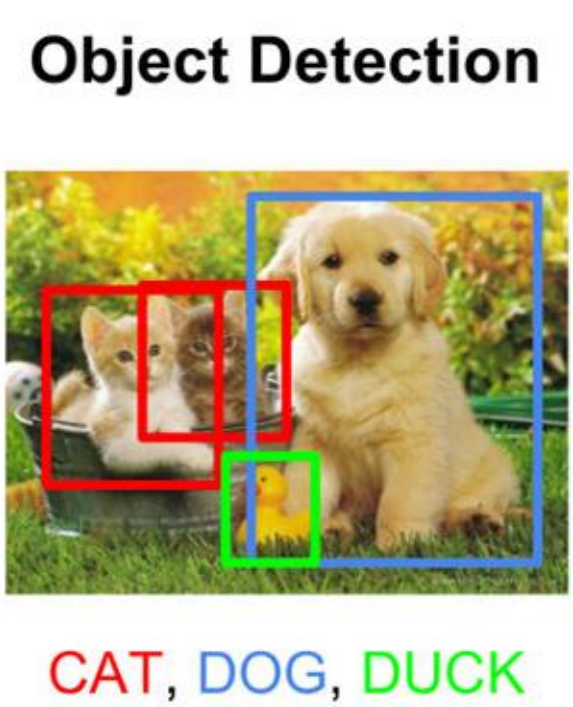
Object classification with supervised learning

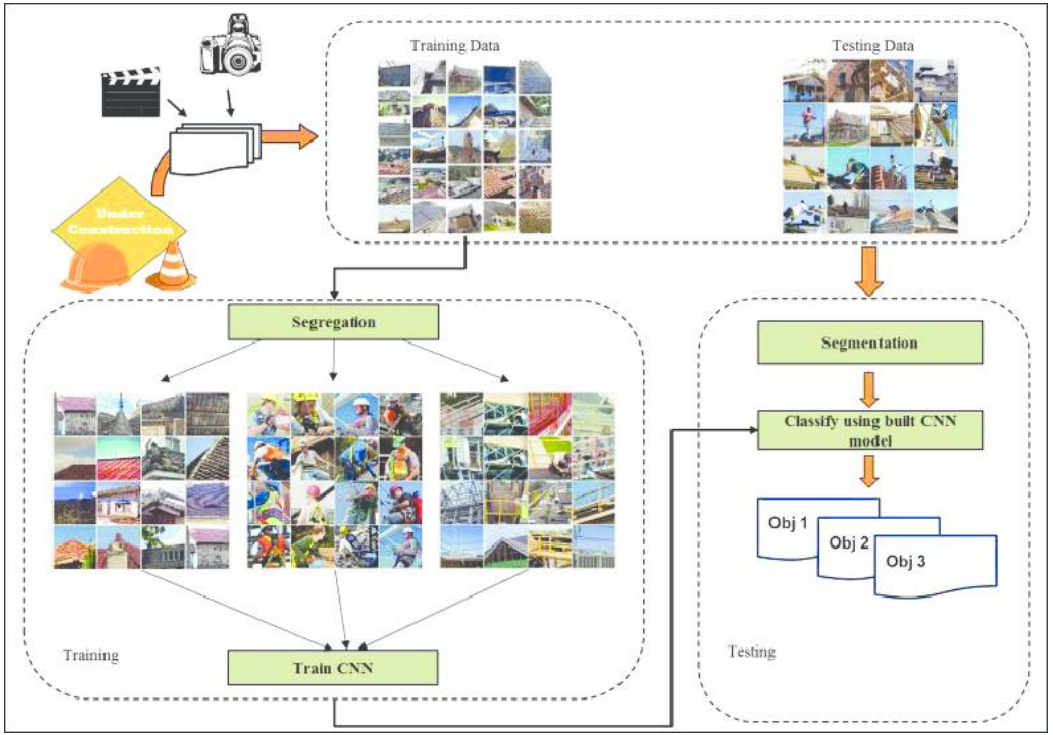
Classification

CAT

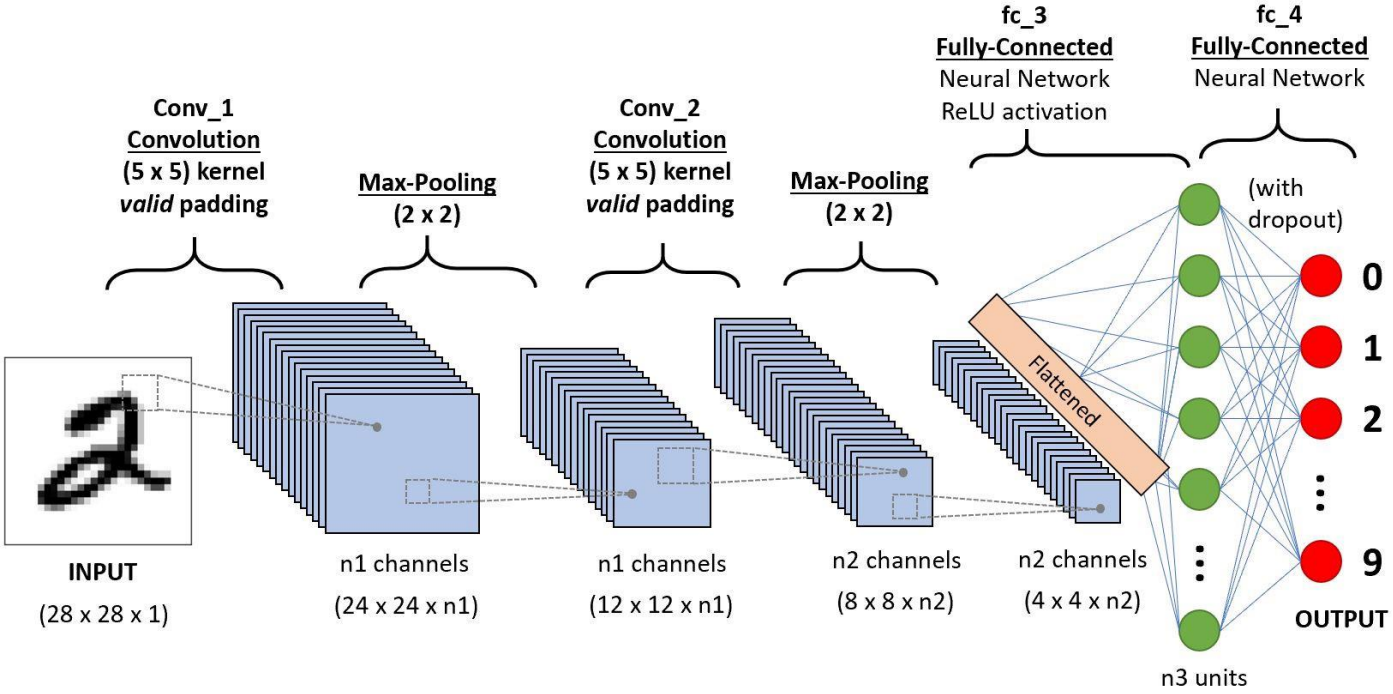


Object Detection with supervised learning

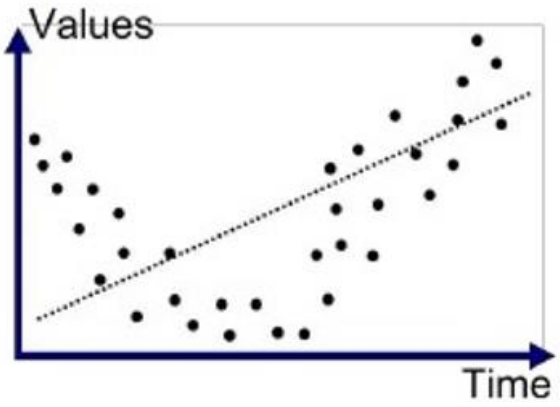




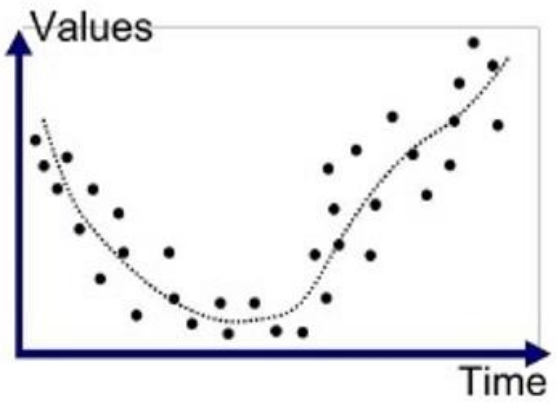
CNN (Convolutional Neural Network)



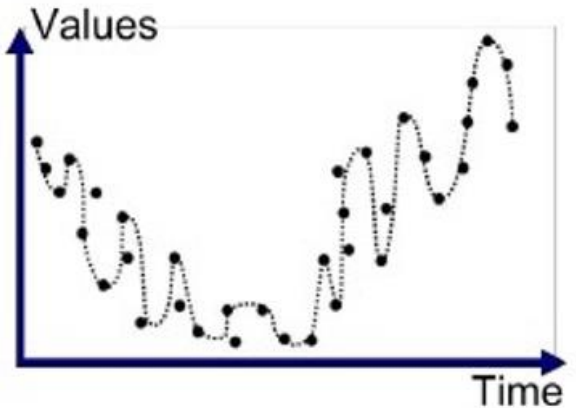
Problem of Underfitting and Overfitting



Underfitted

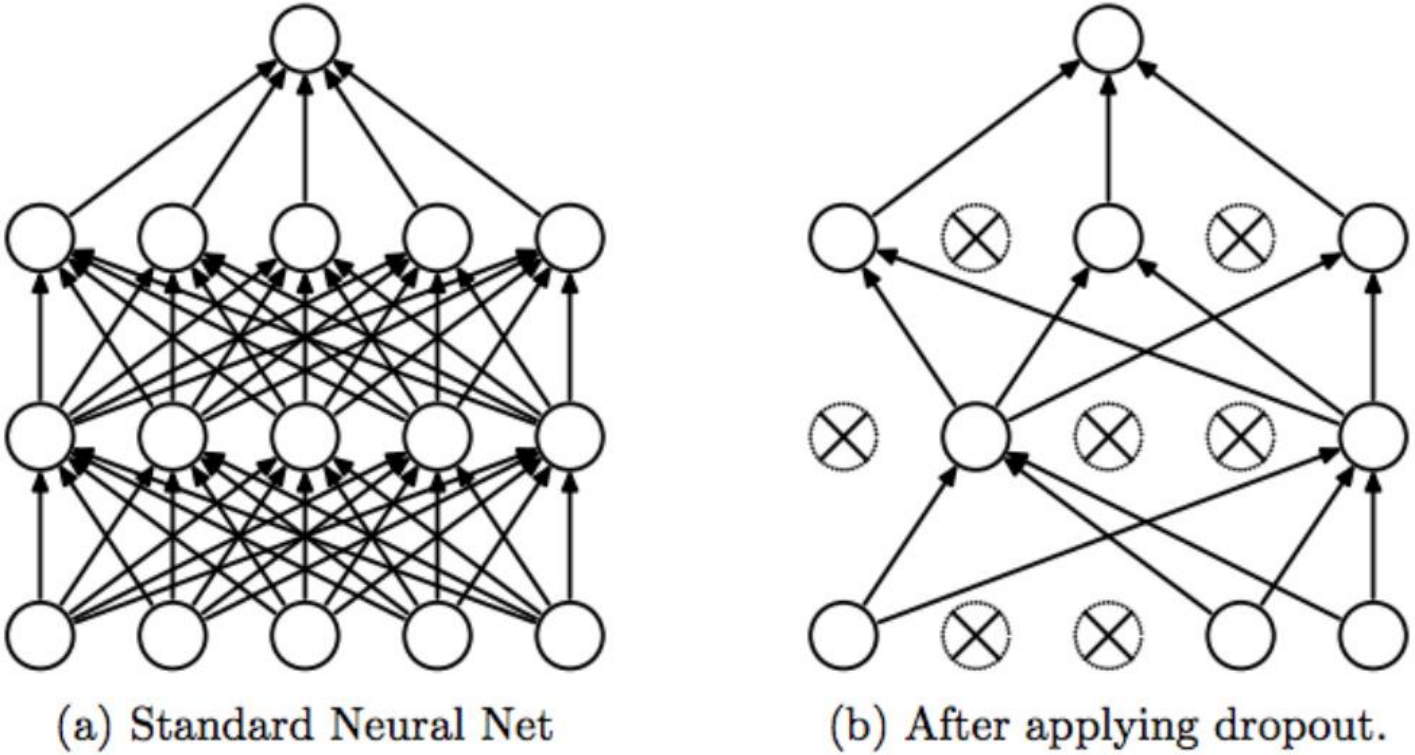


Good Fit/Robust



Overfitted

Problem of Underfitting and Overfitting



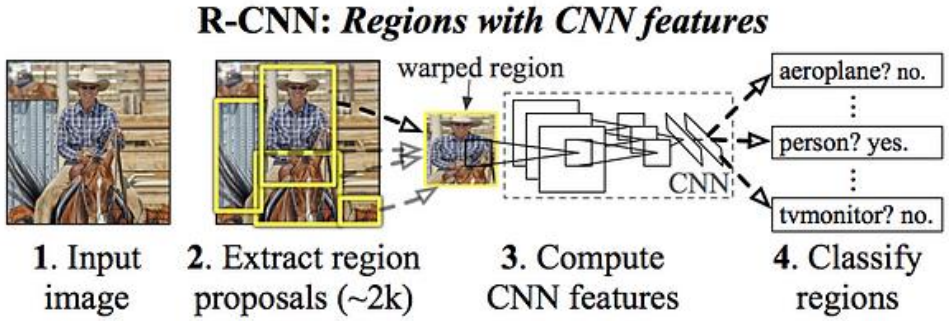
(a) Standard Neural Net

(b) After applying dropout.

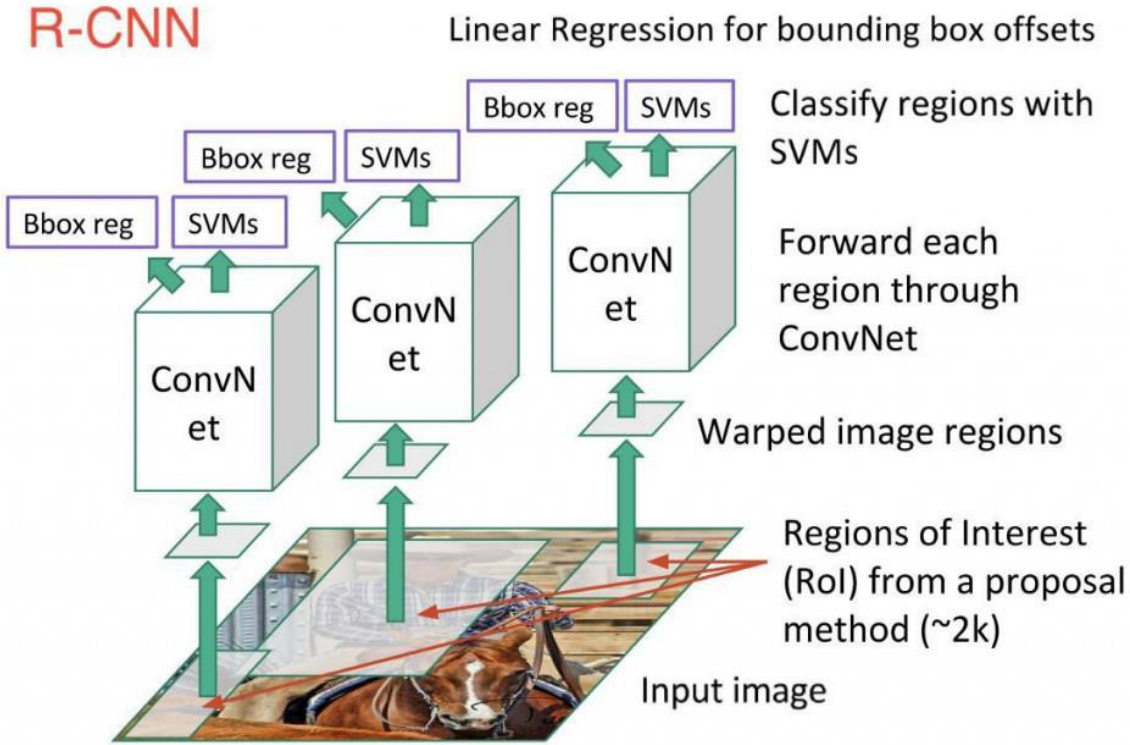
R-CNN (Region-based Convolutional Neural Network)

Selective Search:

1. Generate initial sub-segmentation, we generate many candidate regions
2. Use greedy algorithm to recursively combine similar regions into larger ones
3. Use the generated regions to produce the final candidate region proposals



R-CNN (Region-based Convolutional Neural Network)



R-CNN (Region-based Convolutional Neural Network)

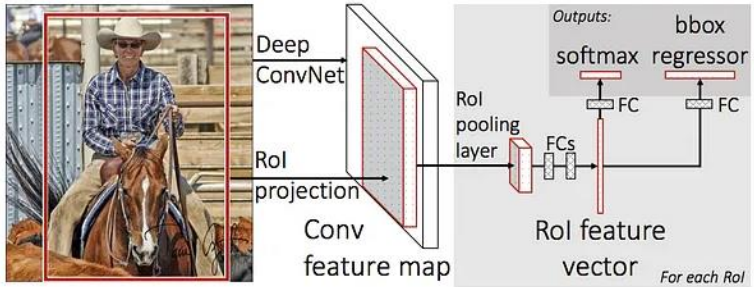
Problems with R-CNN

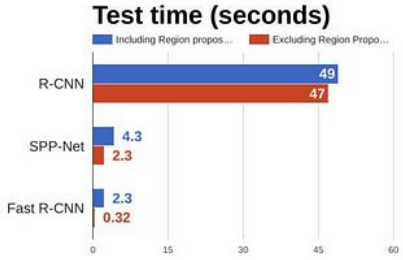
- It still takes a huge amount of time to train the network as you would have to classify 2000 region proposals per image.
- It cannot be implemented real time as it takes around 47 seconds for each test image.
- The selective search algorithm is a fixed algorithm. Therefore, no learning is happening at that stage. This could lead to the generation of bad candidate region proposals.
- The training processing requires larger hard disk space

Fast R-CNN

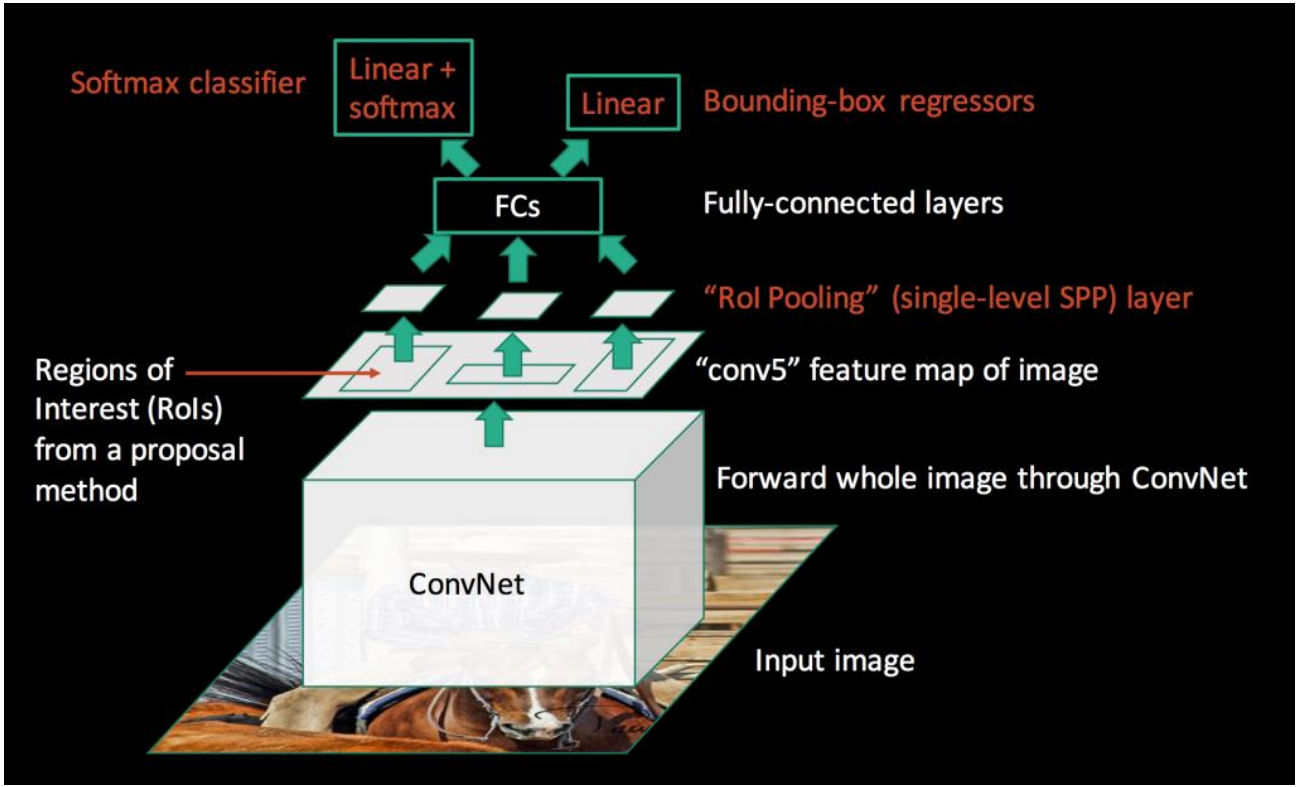
Comparing with R-CNN

1. feed the input image to the CNN to generate a convolutional feature map instead of feeding the region proposals
2. using a RoI pooling layer to reshape region of proposals into a fixed size and fed into a fully connected layer
3. softmax layer to predict the class of the proposed region

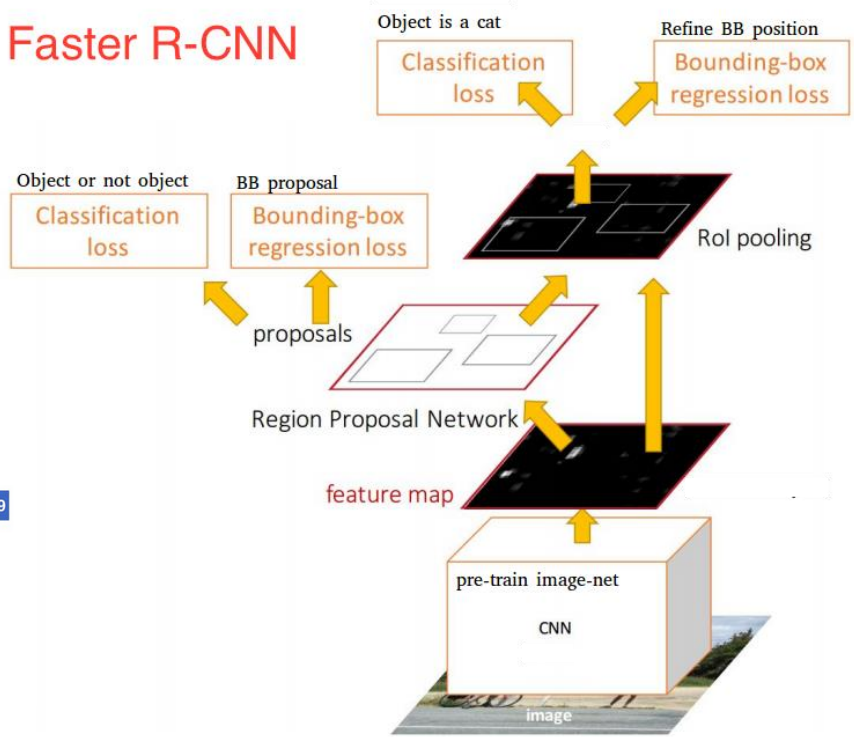


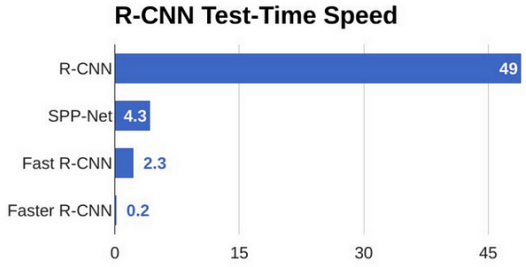


Fast R-CNN

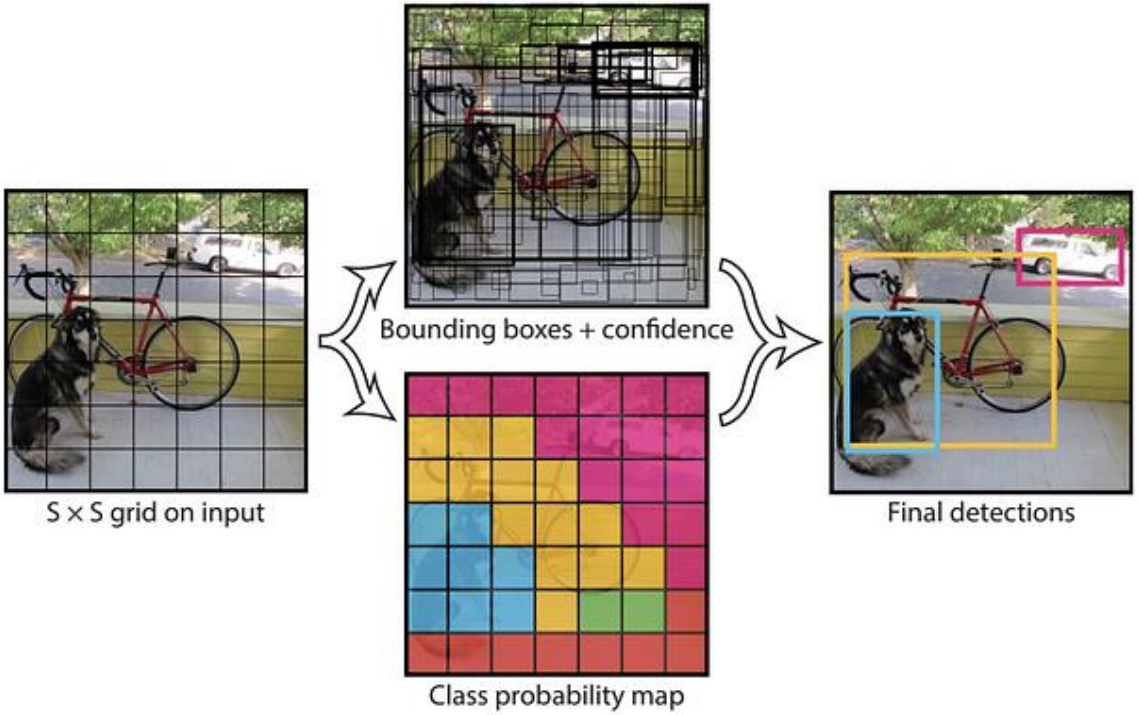

Softmax classifier
Linear + softmax
Linear
Bounding-box regressors
FCs
Fully-connected layers
"RoI Pooling" (single-level SPP) layer
Regions of Interest (RoIs) from a proposal method
"conv5" feature map of image
Forward whole image through ConvNet
ConvNet
Input image

Faster R-CNN


Faster R-CNN
Object is a cat
Classification loss
Refine BB position
Bounding-box regression loss
Object or not object
Classification loss
BB proposal
Bounding-box regression loss
RoI pooling
proposals
Region Proposal Network
feature map
pre-train image-net
CNN
image

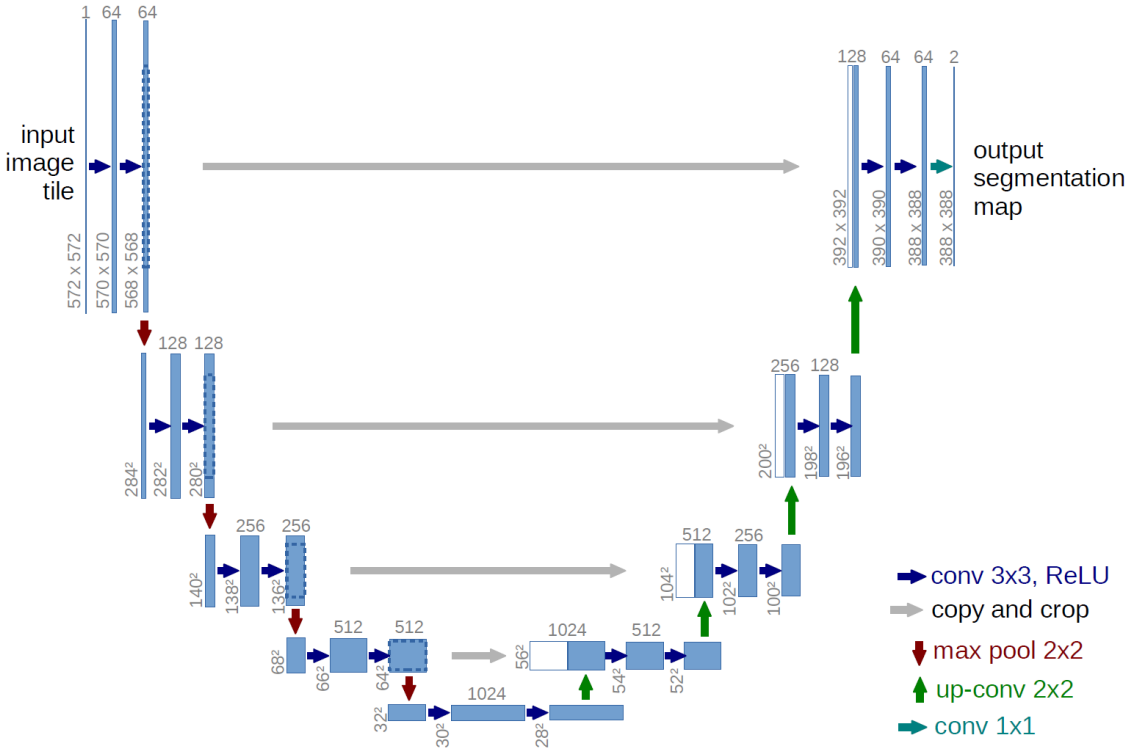

R-CNN Test-Time Speed
R-CNN
49
SPP-Net
4.3
Fast R-CNN
2.3
Faster R-CNN
0.2
0
15
30
45

YOLO (You Only Look Once)



Semantic Segmentation

Fully Convolutional Networks (FCN): U-net

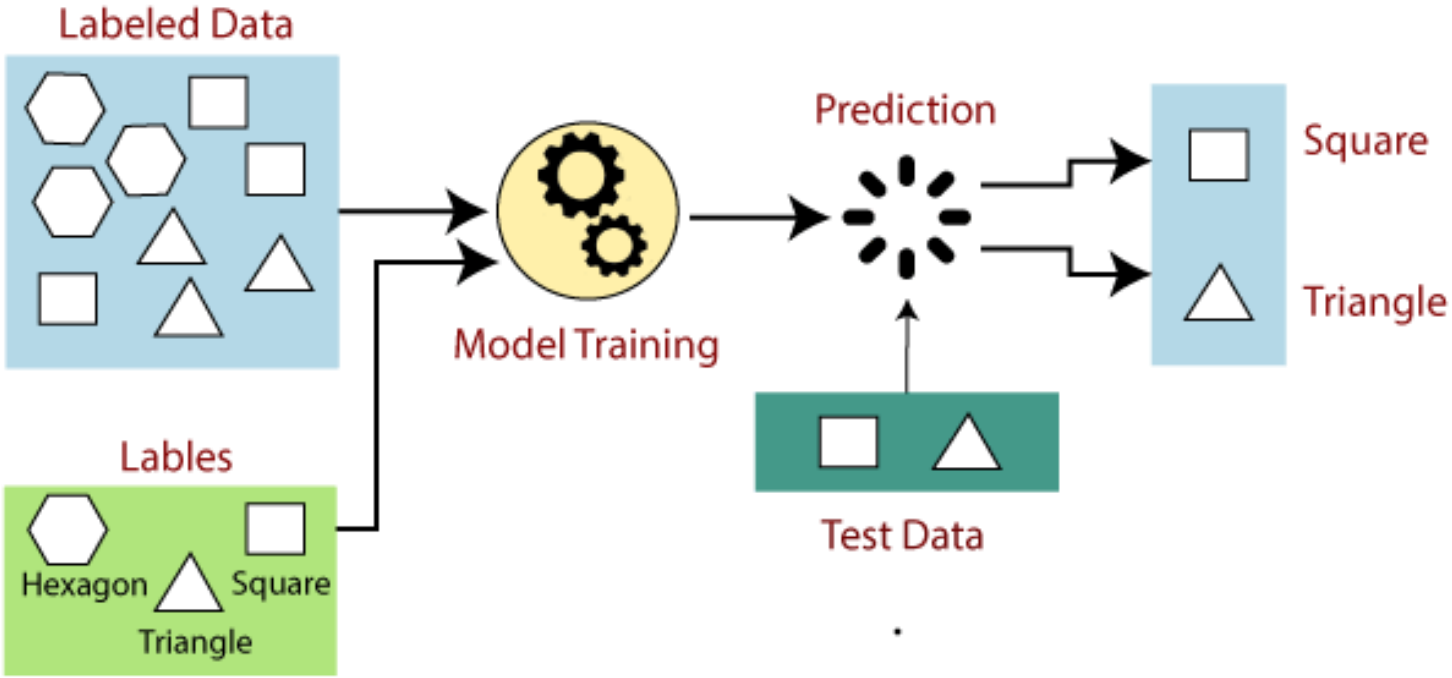


Instance Segmentation

Task 2

Using CNN for MNIST Handwritten Digit Classification

Supervised machine learning (ML) process



MNIST dataset

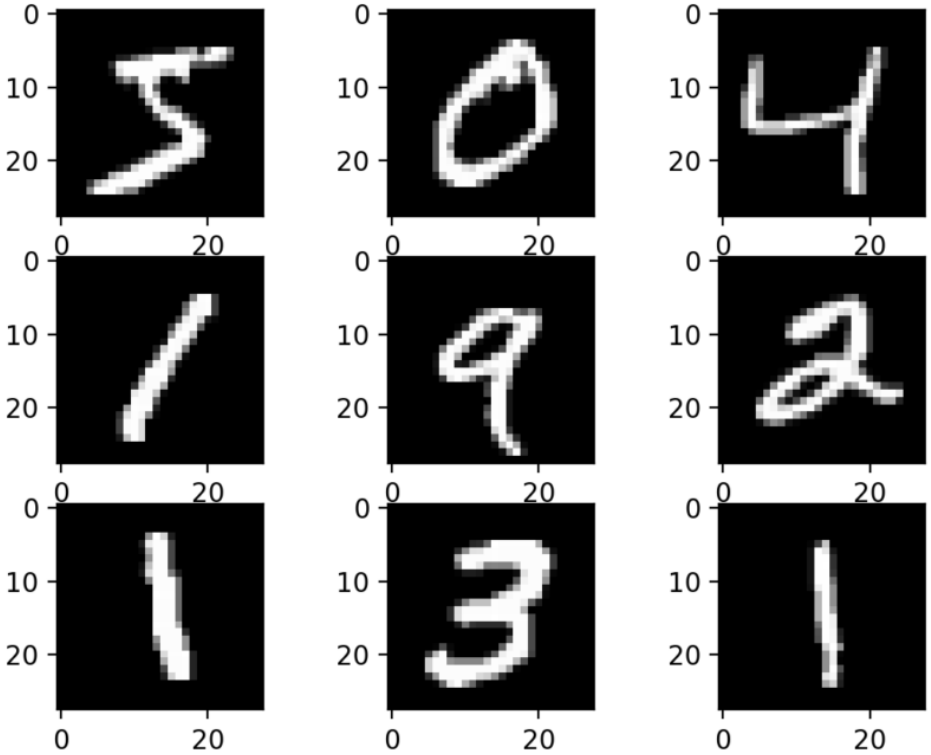
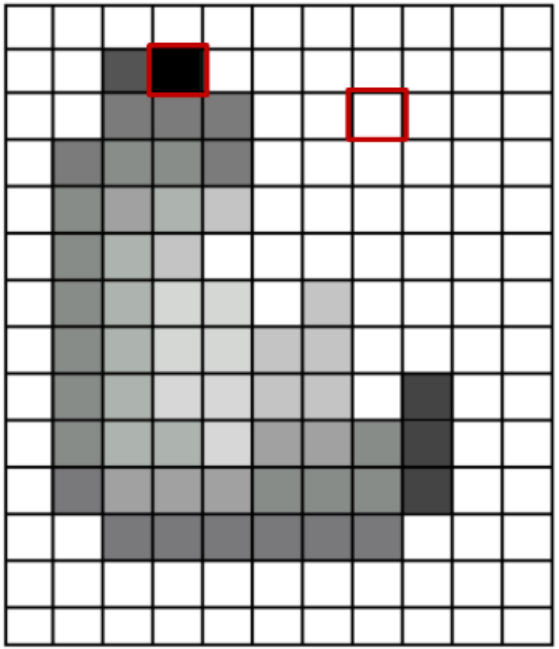


Image representation – B/W Image

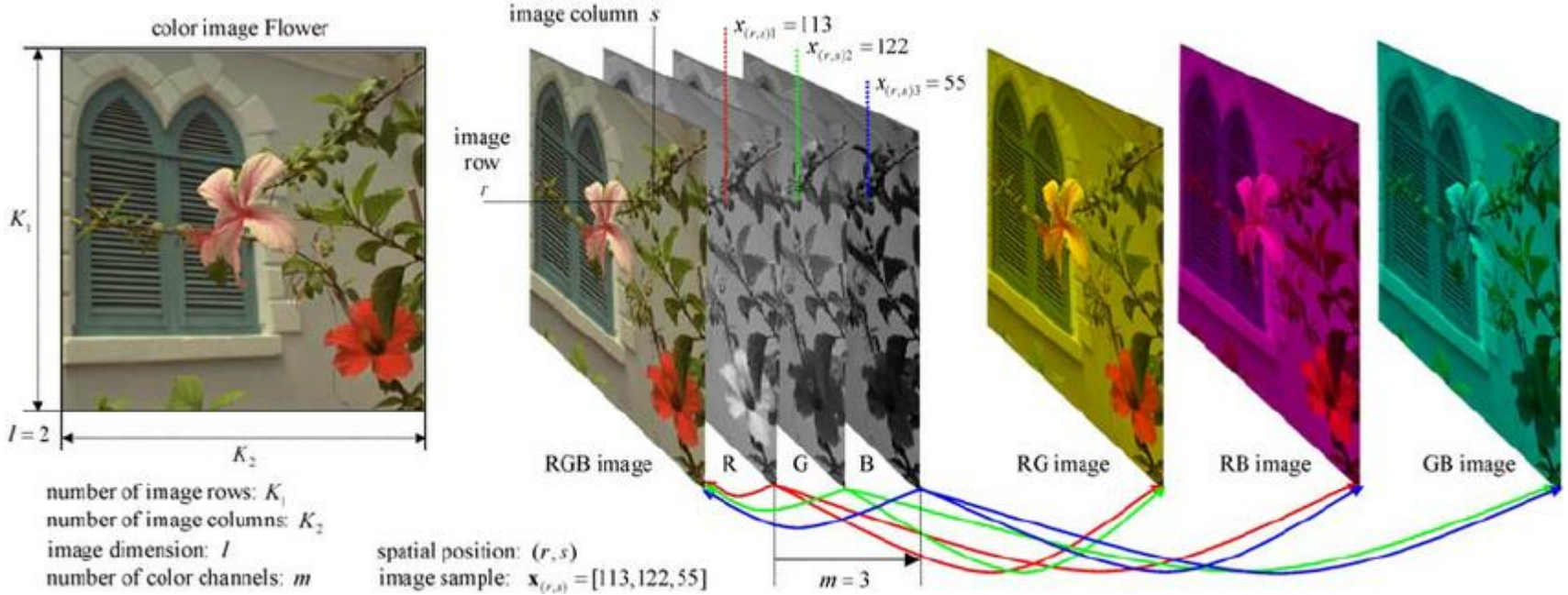


=

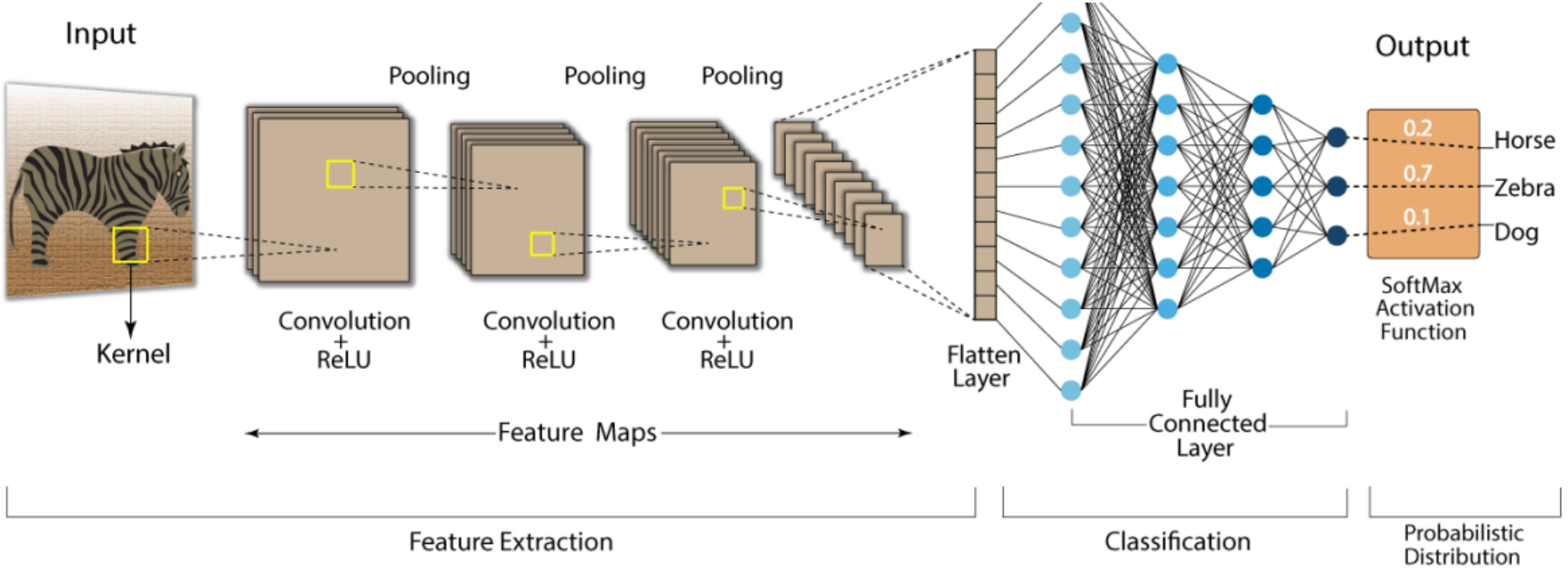
255	255	255	255	255	255	255	255	255	255	255
255	255	20	0	255	255	255	255	255	255	255
255	255	75	75	255	255	255	255	255	255	255
255	75	95	95	75	255	255	255	255	255	255
255	96	127	145	175	255	255	255	255	255	255
255	127	145	175	175	175	255	255	255	255	255
255	127	145	200	200	175	175	95	255	255	255
255	127	145	200	200	175	175	95	47	255	255
255	127	145	145	175	127	127	95	47	255	255
255	74	127	127	127	95	95	95	47	255	255
255	255	74	74	74	74	74	74	255	255	255
255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255

0 = black; 255 = white

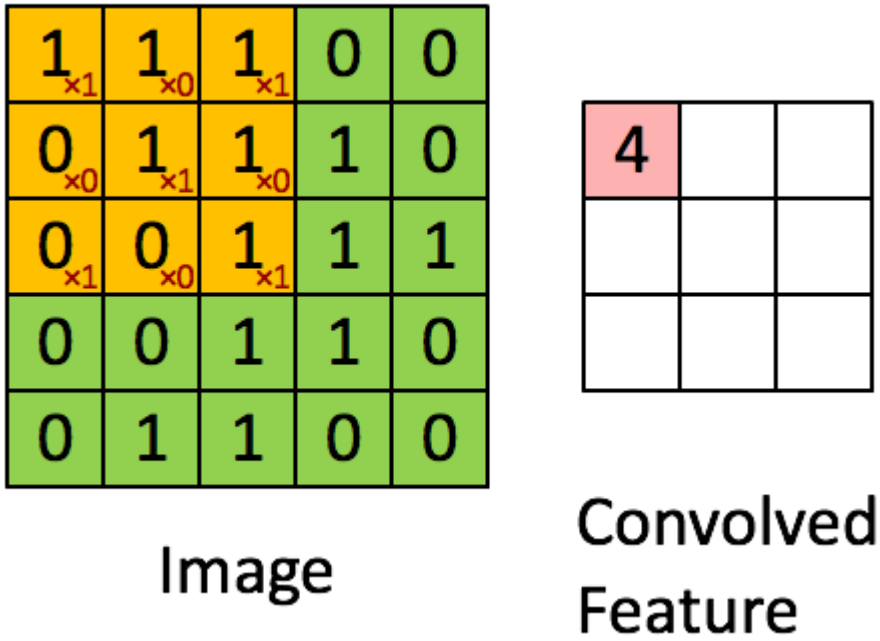
Image representation – B/W Image



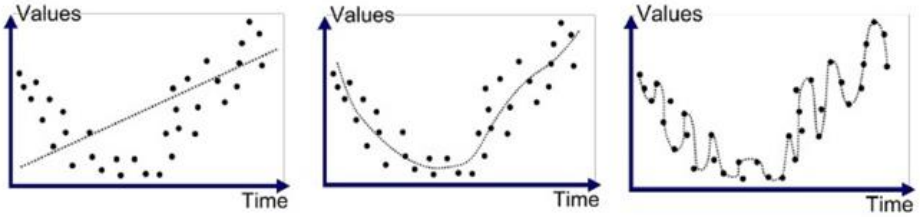
CNN (Convolution Neural Network)



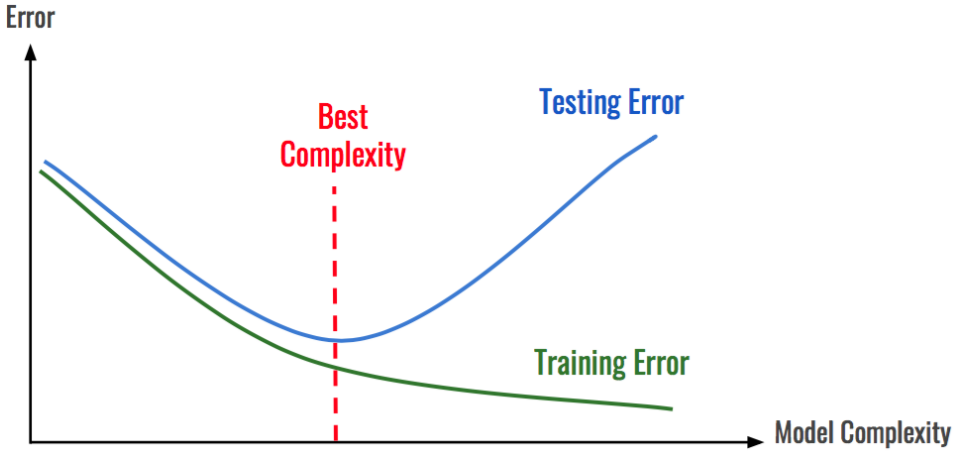
Convolutional operation



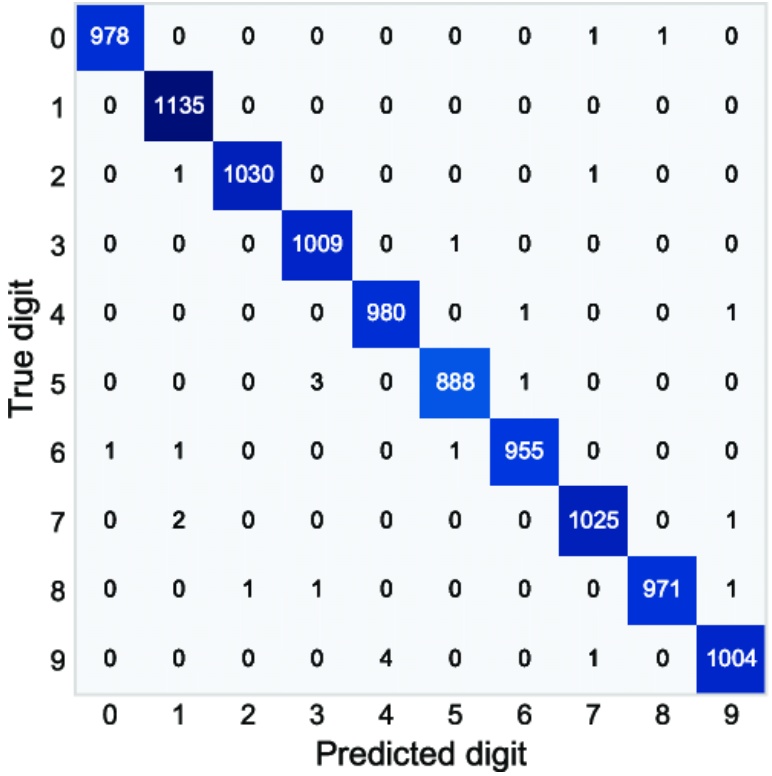
Evaulate trained network



Underfitted | Good Fit/Robust | Overfitted



Confusion Matrix



Task2

Demo

▾ Exercise - modify the network to improve the accuracy

```
model = Sequential()
# Create CN layer 1
model.add(Conv2D(filters=16,
                                    kernel_size=(3,3),
                                    padding='same',
                                    input_shape=(28,28,1),
                                    activation='relu'))
# Create Max-Pool 1
model.add(MaxPooling2D(pool_size=(2,2)))

# Add Dropout layer
model.add(Dropout(0.25))
model.add(Flatten())

model.add(Dense(128, activation='relu'))
#Prevent overfitting
model.add(Dropout(0.5))

#Ouput N-class probabilities
model.add(Dense(10, activation='softmax'))


model.summary()
print("")

model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

train_history = model.fit(x=X_Train_keras_norm,
                                                y=y_TrainOneHot, validation_split=0.2,
                                                epochs=10, batch_size=300, verbose=2)
```

```
loss, accuracy = model.evaluate(X_Test_keras_norm, y_TestOneHot)
print()
print("[Info] Accuracy of testing data = {:2.1f}%".format(accuracy*100.0))
```

```
result = np.where(test_predictions != y_Test)
print(len(result[0]),",",result)
```

▾ Prediction - Test your own handwritten digit

1. Create your handwritten digit via the Online image Editor
2. Download the file to your local drive
3. Upload the file to an online storage
4. Get the URL of your stored image

Task 3

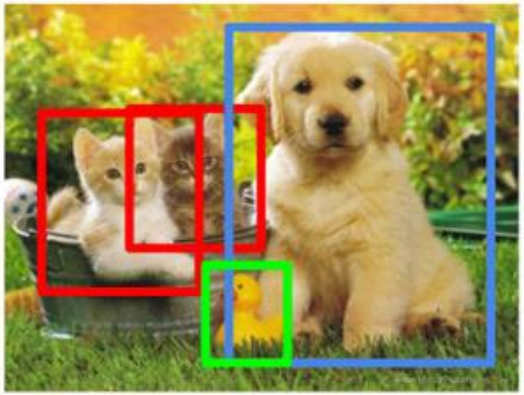
Using CNN for Object Detection

Image classification vs object detection

Classification

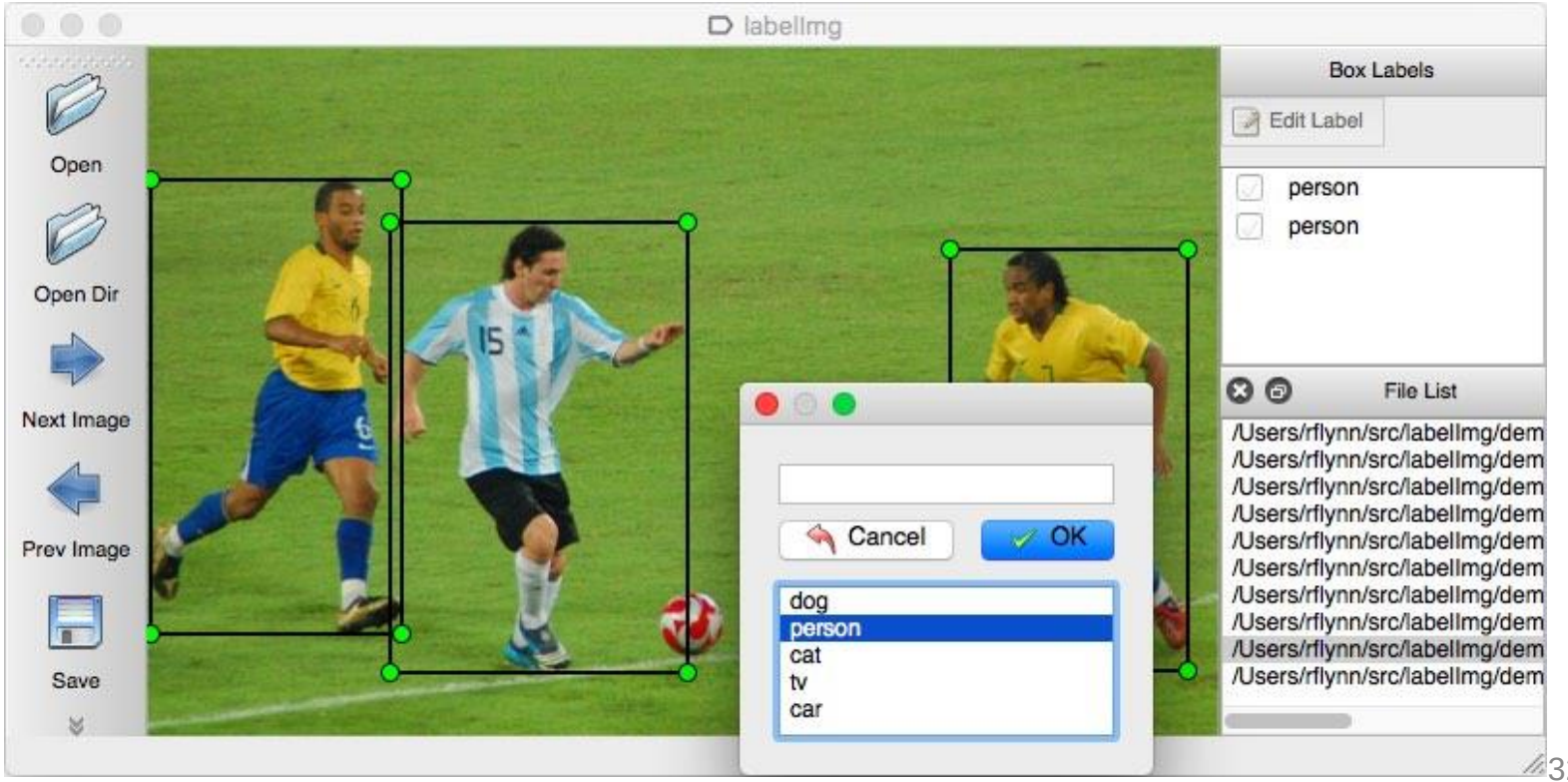
CAT

Object Detection



CAT, DOG, DUCK

Image classification vs Object detection



https://github.com/heartexlabs/labelImg.git

Task 3

Task 3

1. Try to perform the object detection for furniture shops image with the mobilenet model

- compare the inference time
- compare the accuracy with the InceptionResNet model

```
[ ] # -------------Start  editting  -------------
    # detect_img(....)
    # -------------End  editting  -------------
```

Exercise

2. Try to perform the object detection for furniture shopssame image with different model from TFHub

- compare the inference time
- compare the accuracy with the InceptionResNet model

```
[ ] # -------------Start  editting  -------------
    # detect_img(....)
    # -------------End  editting  -------------
```

Task 4

Object Detection in Construction Safety - Helmet detection

Task 4

Task 4

Using the model for detecting the "Helmet" only. If the "Helmet" and "Human hair" or "Human face" is detected, the result is true, and draw the boxes to demonstrate it.

- Refer to the previous image to check what is the class that detected
- Check whether the "Helmet" is detected. If true, append the result into "h_box" and "h_num" += 1.
- Double check the detected "Helmet" whether the "Human hair" or "Human face" is detected. If true, the detection result is true, and draw boxes for showing the class "With helmet" with the corresponding score.