X_4: A Class of Features for the size of Bounding Boxes.

Too Small → Noise, filter out

Too Big → UnLikely the Right ROI.

$X_1 \; X_2 \; X_3 \; X_4 \quad f$

✓ ✓

$f = X_1 X_3$

Note: Use OpenCV Function to Construct A Square Image.

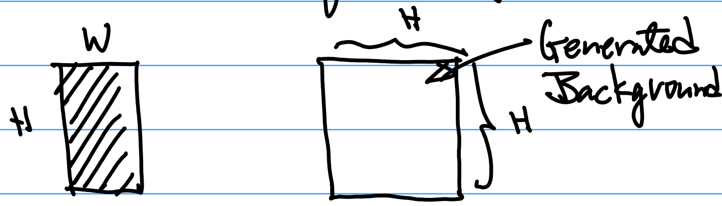
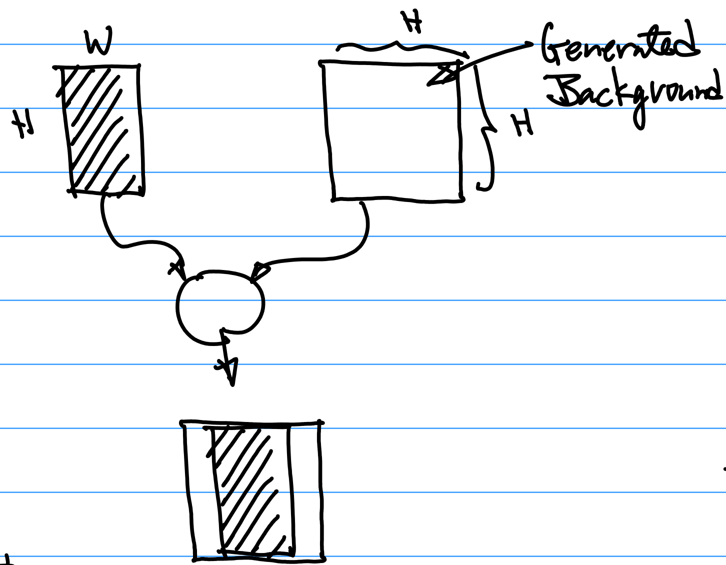


Fig 2.

$\max\{H, W\} = H$, use H as dimension for Square Image.



Then, Reduce/Resize the Image to the right size, 28x28

c. Ref: Reshape the image Format

20-2021S-3-load-deployment.py

```
from keras.models import load_model
import cv2
import numpy as np
from PIL import Image

model = load_model('mnist.h5')
model.compile(optimizer='rmsprop',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

img = Image.open('1.jpg').convert('L')
img = np.resize(img, (28,28,1))
im2arr = np.array(img)
im2arr = im2arr.reshape(1,28*28)
y_pred = model.predict_classes(im2arr)
print(y_pred)
```

d. Save Detection Result (Video Clips, 5~10 Sec), Be Sure to use your personal SID (Just Last 4 Digits)

Topic on the 2nd Part of this Class Yolo (You Only Look Once)

Source Code distribution for Yolo

Reference Paper On Yolo

2022S-112-yolo-paper.pdf

Introduction to Yolo Algorithm.

Example:

CMPE258 April 5, 22

Note: a. Input Image: Square Image, Resolution of the image: 448x448

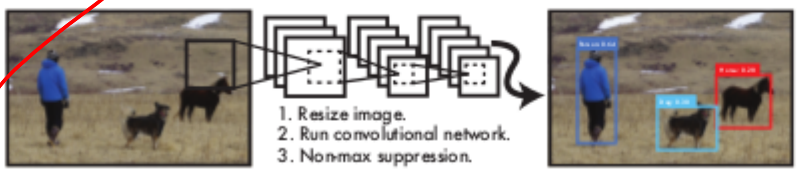
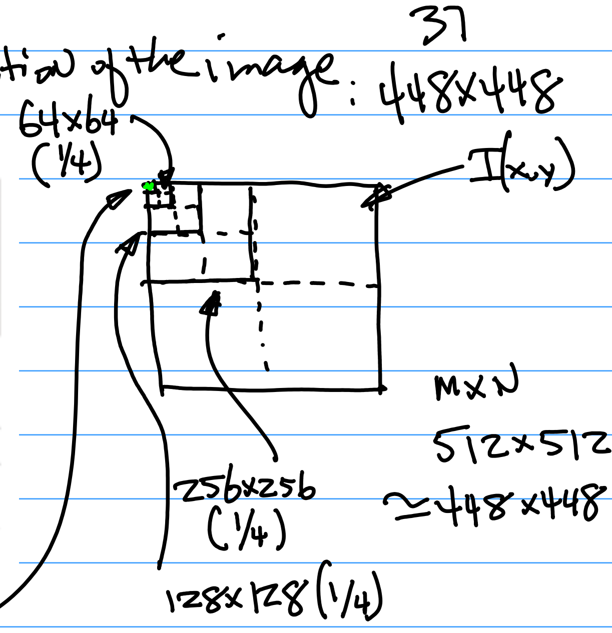


Figure 1: The YOLO Detection System. Processing images with YOLO is simple and straightforward. Our system (1) resizes the input image to 448 × 448, (2) runs a single convolutional network on the image, and (3) thresholds the resulting detections by the model's confidence.



32x32
≃(28x28)
For Handwritten Digits.

Note: Architecture, 24 Convolutional Layers. 2 fully Connected layers.

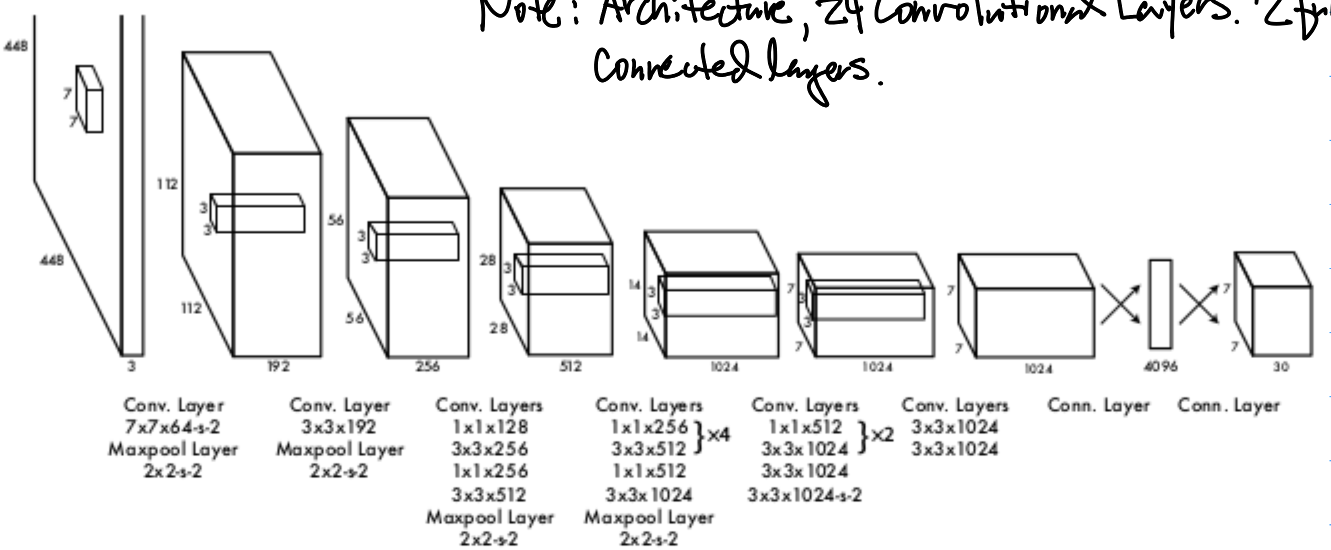


Figure 3: The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1 × 1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224 × 224 input image) and then double the resolution for detection.

Source Distribution, see Handout to Be Posted online (github).

1. Readme Document.
   a. URL for Yolo Source distribution Download

b. anaConda to Configure the environment

c. Activate the Conda environment

d. 3 Samples to test

CmpE258 April 5, 22

First Example, to perform Training
Save the training Result;
2nd works on input image
(Single Frame).
3rd program works on input
video file.

Homework: Due 2weeks from today.
(April 19th).

1° Download, Install yolo program.
(yolo4).

2° Use cellphone to record 5-10
Seconds of video clip for testing
Purpose.

3° Run Yolo Code to Perform default
Detection Task

4° Submission on CANVAS.
a. Screen Capture of yolo program
b. Processed Video Clip (Be Sure
to use your own Video Clip).

Example: Loss Function (Brief)

Note:

loss function:

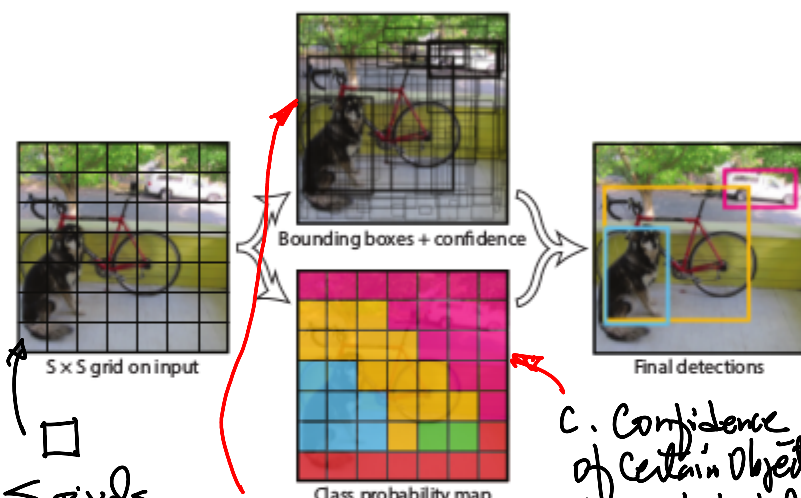
$$
\begin{gathered}
\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\
+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 \\
+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2
\end{gathered}
\tag{3}
$$

Example: On Probability Classification.

$$
\Pr(\text{Class}_i | \text{Object}) * \Pr(\text{Object}) * \text{IOU}_{\text{pred}}^{\text{truth}} = \Pr(\text{Class}_i) * \text{IOU}_{\text{pred}}^{\text{truth}} \tag{1}
$$

Definition of Conditional Probability.

Example: On Partition of a Test Image



a. S pixels Both w, h.

b. Bounding Box(es)

c. Confidence of Certain Objects to Be detected

Figure 2: The Model. Our system models detection as a regression problem. It divides the image into an $S \times S$ grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.

d. The Classification of Object Detected is Based on its Confidence Level.

April 12, 22

Topics: 1° DNN Yolo Formulation

2° Backpropagation (BackProp) Algorithm.

2022S-112-yolo-paper.pdf

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

Joseph Redmon*, Santosh Divvala*†, Ross Girshick¶, Ali Farhadi*†
University of Washington*, Allen Institute for AI†, Facebook AI Research¶
http://pjreddie.com/yolo/

$$\Pr(\text{Class}_i|\text{Object}) * \Pr(\text{Object}) * \text{IOU}^{\text{truth}}_{\text{pred}} = \Pr(\text{Class}_i) * \text{IOU}^{\text{truth}}_{\text{pred}} \quad (1)$$

Theoretical Analysis of Yolo.

Simple Example As A Starting Point of Our Discussion, from Eqn (1)

Note: 1. Prob(Class$_i$)

Probability of a given Object(s) which belongs to a class i

$i = 1, 2, \cdots, K$

Let $i=1$ for Simplicity purpose

Prob(Class$_i$) = Prob(Class)

2. IOU (Intersection of Union)

η_{IOU} Coefficient

$$\text{Prob(Class)}\,\eta_{IOU} \xrightarrow{\text{Simplify}} \text{Prob(Class)}$$

Prob(C), Denote Class as "C".

Example: Consider Object Identification, In particular, Detect Pedestrain.

Ref:

2022S-114-yolo-designGuide-2022-4-9.pdf

Red, R: Pedestrains;
Black, B: Vehicle;

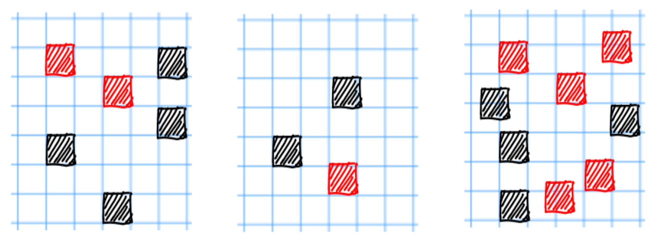


Fig 1.

1. Define An Event R
Detection of Pedestrain.

Set Relations.

$$R = RI_1 + RI_2 + RI_3 \quad \ldots (1*)$$

$$R \cap R_1, \quad I_1 \cap I_2 \cap I_3 = \phi \text{ Empty}$$

2. Formulate the Likelihood of the event R

$$\text{Prob}(R) = \text{Prob}(RI_1) + \text{Prob}(RI_2) + \text{Prob}(RI_3) \quad \ldots (2)$$

3. Conditional Probability, Bayesian Theory

$$\text{Prob}(RI_1) = \text{Prob}(R|I_1)\,\text{Prob}(I_1) \quad \ldots (3)$$

$$\text{Prob}(RI_2) = \text{Prob}(R|I_2)\,\text{Prob}(I_2)$$

$$\text{Prob}(RI_3) = \text{Prob}(R|I_3)\,\text{Prob}(I_3)$$

4. $\text{Prob}(R) = \text{Prob}(R|I_1)\text{Prob}(I_1) + \text{Prob}(R|I_2)\text{Prob}(I_2) + \text{Prob}(R|I_3)\text{Prob}(I_3)$

$$= \sum_{i=1}^{3} \text{Prob}(R|I_i)\text{Prob}(I_i) \quad \cdots (4)$$

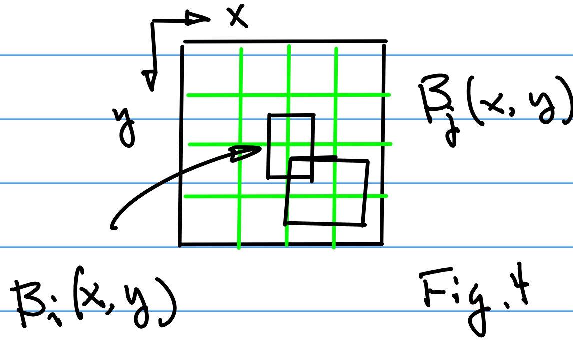


Fig.4

5. Notations for Yolo

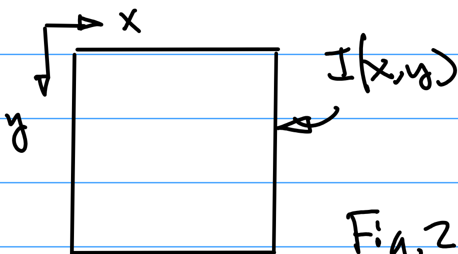
① 

Fig.2

② Define Grids $\underset{S \times S}{G(x,y)}$

↑

S pixel Width, S pixel Height

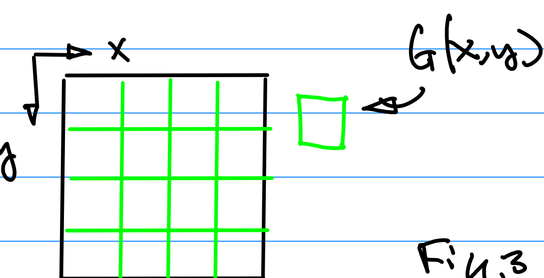


Fig.3

we have multiple grids, denoted as

$$G_p(x,y), \quad p = 1, 2, \cdots \quad \cdots (5)$$

③ Define Bounding Box for ROI.

as $B(x,y)$

Note:

$B_i(x, y; W, H; f)$, for $i = 1, 2, \cdots, M$

Location, shape, Confidence, No. of B. Boxes

$$B_j(x, y; W_j, H_j; f_j) \quad \cdots (6)$$

Note: Among them, we have one ground Truth Bounding Box

$B_{GT}(x, y; W, H, f)$, $f = 1$ for the Ground Truth.

④ IOU Definition.

Example: Illustration of IOU

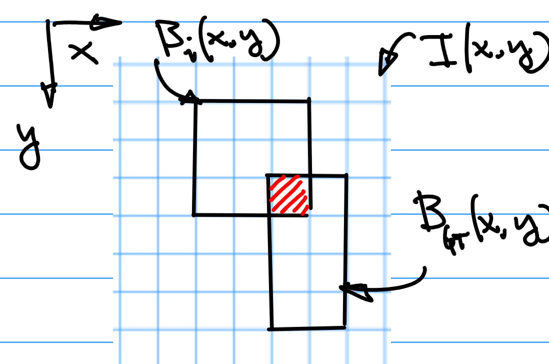


Fig.5

Intersection of Union

$$\text{IOU} = \frac{\text{Intersection}}{\text{Union}} \quad \cdots (7)$$

$$IOU = \frac{\text{Intersection}}{\text{Union}}$$

$$= \frac{B_i(x,y)\,B_{GT}(x,y)}{B_i(x,y)+B_{GT}(x,y)-B_i(x,y)\,B_{GT}(x,y)}$$

$$= \frac{1}{9+8-1} = 1/16 \;//$$

Physical meaning of IOU.

If $B_i(x,y)$ & $B_{GT}(x,y)$ Are Identical then

$$IOU = \frac{B_i(x,y)\,B_{GT}(x,y)}{B_i(x,y)+B_{GT}(x,y)-B_i(x,y)\,B_{GT}(x,y)}$$

$$= 1 \;//$$

Denote IOU as η_{IOU}

From Eqn (1)

$$Prob(C)\,\eta_{IOU} \quad \ldots (8)$$

Note if more than One class,

$$Prob(C_i) = Prob(C_i)\,\eta_{IOU,i}$$

6. Consider Multiple Classes.

One class, from one Camera

$$Prob(C)\,\eta_{IOU}$$

One class, from multiple Cameras, from Eqn

$$Prob(C) = \sum_i Prob(C/I_i)\,Prob(I_i)$$

Hence

$$Prob(C_i)\,\eta_{IOU,i}$$

$$= \sum_{i=1}^{K} Prob(C/I_i)\,Prob(I_i)\cdot\eta_{IOU,i} \quad \ldots (9)$$

Note: Now Expand the Above Analysis Beyond Multiple Cameras, treat Each Camera as input Grid $G(x,y)$ from a given image, the Above Analysis holds good.

Homework: Due 1 week from today. (April 19th).

1° Download, Install yolo program. (yolo4).

2° Use cellphone to record 5-10 Seconds of video clip for testing Purpose.

3° Run Yolo Code to Perform default Detection Task

Team Project:

1° Presentation PPT. 7 Slides 10 Minutes for each team entire entire presentation. PPT.; Demo; Q&A Session

2° Topics, Deep Learning CNN Applications.

Consider Back Propagation Algorithm:

Background

1. Multiple Layer Feedforward Neural Networks. MNIST

$\underbrace{CM_1 \cdots F.D.D}_{\text{Layers}}$

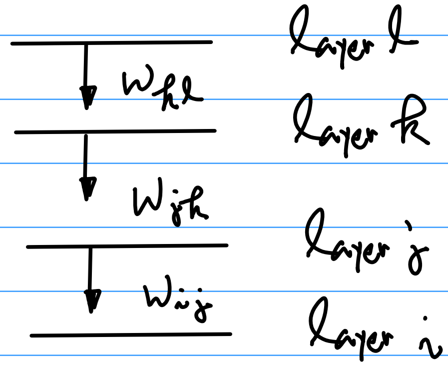


Fig 6

2. For Each Layer.

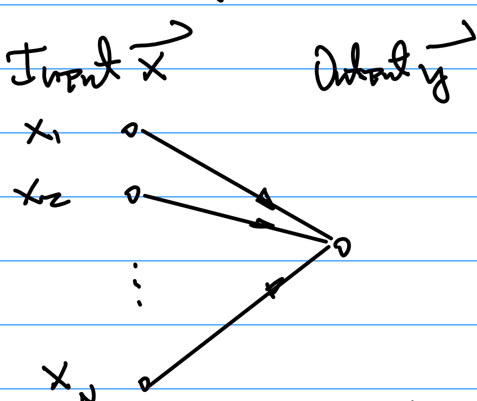
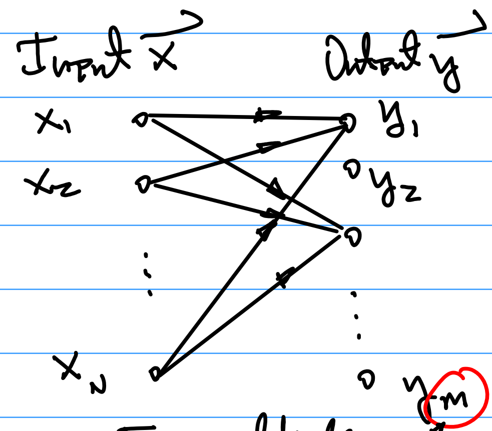


Fig. 7

$$y = f\left(\sum_{i=1}^{N} w_i x_i + b\right) = f(h(w_i; x_i; b)) \quad \cdots (1)$$



For multiple Output Neurons.

Fig. 8

3. Notation for Weights at Each Layer, From the Output layer i towards the input j, k, and l

W_{ij}

The Index of the Output @ the Current Output layer i

The Index of the Input @ layer j

So, we have

W_{ij} W_{jk} W_{kl}

3. Output

$y_j = f\left(\sum_{i=1}^{N} w_i x_i + b\right)$ for One Output Neuron. $\cdots (2)$

Output for multiple Neurons per Each Layer, and multiple Layers as well.

In Addition, Let's Denote Experiment (CNN output y) as $\tilde{y}$ ("tilde"), Let ground Truth denoted as y.

Error for one experiment then is defined as

$$\tilde{y} - y \quad \text{or} \quad y - \tilde{y} \quad \ldots(4)$$

Now. Loss function :

4. Loss function.

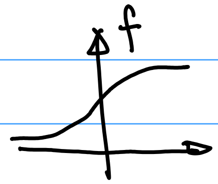
$$L = \frac{1}{2}\sum_{M=1}^{M}\sum_{i=1}^{N}\left(\tilde{y}_i^M - y_i^M\right)^2 \quad \ldots(5)$$

No. of Experiments   No. of Output

5. Now, Consider Eqn(1) & (2)

Eqn(1): has Activation function f( ); has Transfer function h( );

$$y = f\left(\sum_{i=1}^{N} w_i x_i + b\right) = f(h(w_i; x_i; b)) \quad \ldots(1)$$



$$\sum_{i=1}^{N} w_i x_i + b \triangleq h(w_i; x_i; b) = h(\cdot)$$

Based on the Notation for the Experiments and Ground Truth, Rewrite Eqn(1) as

$$\tilde{y} = f\left(\sum_{i=1}^{N} w_i x_i + b\right) = f(h(w_i; x_i; b))$$

OR

$$\tilde{y} = f(h)$$

5. The weights update equation

$$W_{ij}(t+1) = W_{ij}(t) + \delta W_{ij} \quad \ldots(6)$$

$$\frac{\partial L}{\partial W_{ij}} = \frac{\partial}{\partial W_{ij}} \frac{1}{2}\left[\sum_{M=1}^{M}\sum_{i=1}^{N}\left(\tilde{y}_i^M - y_i^M\right)^2\right]$$

From Here, Apply "Chain-Rule" for multiple Layers. i, j, k, l.

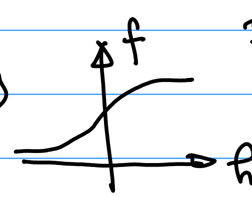
April 19.

Topics: 1. Bounding Boxes Selection.
2. K-mean Cluster Algorithm.

2022S-114a-BoundingBoxSelection-Yolo4-2022-4-19.pdf

2022S-114b-boundingBox-Selection-yolo-tutorial-2022-...

Example: NMS Algorithm to select Bounding Boxes

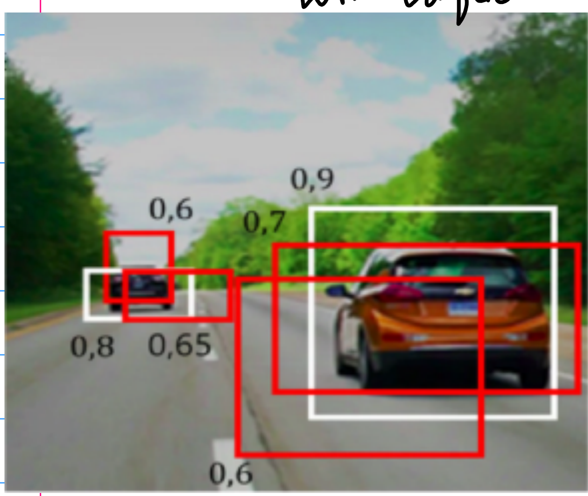


Note: the Notation for our discussion

Image Partition into
$S \times S$ Grids $G_i(x, y)$.
$B_i(x, y)$, $i = 1, 2, \cdots, K$
Bounding Boxes.

B_i, x, y, w, h, C_i Confidence
Center point of B_i Shape of B_i

(B_i, C_i) Pair Bounding Box with Confidence



Note: Goal is to Select/Decide one Bounding Box for Each Object with the highest Confidence Level.

Calculation of Bounding Box Selection.
See Handout Example.
IOU Computation (see PP40)

Note this Process selected the Bounding Box with the highest Confidence Level By Elimination process.

Now, consider the K-mean algorithm
A Technique that allows us to define Probability of Classes/Objects.
Ref:

- 2022S-114c-Kmean-handCalculation1.jpg
- 2022S-114c-Kmean-handCalculation2.jpg
- 2022S-114c-KmeanCluster-v3-2022-4-19.pdf

Example:
Feature Vectors

$$\vec{x}_1, \vec{x}_2, \cdots, \vec{x}_n \quad \ldots (1)$$

where

$$\vec{x}_1 = \begin{pmatrix} x_{11} \\ x_{12} \end{pmatrix} \quad \ldots (1\text{-}b)$$

K-mean Clustering Algorithm is to Partition these feature vectors into K Classes.

$$S_1, S_2, \cdots, S_K \quad \ldots (2)$$

in such a way to minize the within-Class Variation, e.g.

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \left\| \mathbf{x} - \boldsymbol{\mu}_i \right\|^2 \quad \ldots (3)$$

$$\|\vec{x} - \vec{M_i}\|^2 \quad \dots (4)$$

M_i Cluster for S_i

Class i.

$$\vec{M_i} = \begin{pmatrix} M_{i1} \\ M_{i2} \end{pmatrix}, \quad \vec{x} - \vec{M_i} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - \begin{pmatrix} M_{i1} \\ M_{i2} \end{pmatrix}$$

$$= \begin{pmatrix} x_1 - M_{i1} \\ x_2 - M_{i2} \end{pmatrix}, \quad \dots (5)$$

$$\|\vec{x} - \vec{M_i}\| = \sqrt{(x_1 - M_{i1})^2 + (x_2 - M_{i2})^2} \quad \dots (6)$$

minimization of the difference, e.g. Variation from its class' Cluster $\vec{M_i}$

Arg min

Minimization for all feature Vectors in the Class S_i

$$\sum_{\vec{x} \in S_i} \quad \dots (7)$$

The minimization will have to Be carried out for all the Classes, therefore

$$\sum_{i=1}^{K} \quad \dots (8)$$

K Classes

The Algorithm:

"for Any" j

$$S_i^{(t)} = \{x_p : \|x_p - m_i^{(t)}\|^2 \le \|x_p - m_j^{(t)}\|^2 \; \forall j, 1 \le j \le k\}, \quad \dots (a)$$

Mean given set $x_1, x_2, \dots, x_N$

$$m = \frac{1}{N}\sum_{i=1}^{N} x_i \text{ for scalar} \quad \dots (10)$$

for vector, we have feature vectors.

$$\vec{x_1}, \vec{x_2}, \dots, \vec{x_N}$$

$$\vec{m_i} = \frac{1}{N}\sum_{i=1}^{N} \begin{pmatrix} x_{i1} \\ x_{i2} \end{pmatrix} \quad \dots (11)$$

Class i

Note: Eqn (a) defines the Algorithm which make selection of cluster $\vec{M_i}$ (for Class i) satisfy the condition, such that each of every $\vec{x_p}$ in the Class will have

$$\|\vec{x_p} - \vec{m_i}\| \le \|\vec{x_p} - \vec{m_j}\| \quad \dots (12)$$

We conduct the Computation Based on Eqn (a).

First, Select Number of Class K Based on Heuristics

Then, the initial process,

Step 2 Pick mean value for each class,

$$\vec{m}_i = \frac{1}{N}\sum_{j=1}^{N} \vec{x}_{ij} \quad \dots(13)$$

Note: the class i is arbitrarily assigned. ($i=1,2,\dots,K$).

Step 3. Use Equation Below to check if the Equation satisfied, if yes, then keep the vector in the class i,

$$\|\vec{x}_p - \vec{m}_i\| \le \|\vec{x}_p - \vec{m}_j\| \quad \dots(14)$$

O/W, reassign the vector to its smaller distance group.

Step 4. Update the Cluster By Re-evaluating mean.

$$\vec{m}_i = \frac{1}{N}\sum_{j=1}^{N} \vec{x}_{ij}$$

for all i classes.

Step 5. Check and verify the termination condition. by

Comparing the current means to their previous mean value.

e.g.

$$\vec{m}_i^{(t)} = \vec{m}_i^{(t-1)} \quad \dots(15)$$

Step t and step $t-1$ for mean i, $i=1,2,\dots,K$

If Eqn(15) holds good, then, the process is done, Clusters are given as $\vec{m}_i$, for $i=1,2,\dots,K$.

Otherwise, Continue the Process as in Step 3. Continue till the termination condition satisfied.

Homework (Due A week from Today, April 26).

1. Write a python code to implement K-mean cluster Algorithm. (No OpenCV necessary)
2. Test Data Pattern is given in the previous gender prediction Python Code; (Note the height,

Weight of Each person from 2 Classes are given)

3. Submission of the Source code
4. Screen Capture of the Success execution of the program. (Submission to CANVAS)
5. please make one zip file for your Submission.

April 26

Note: Final Exam. May 18 (Wed).

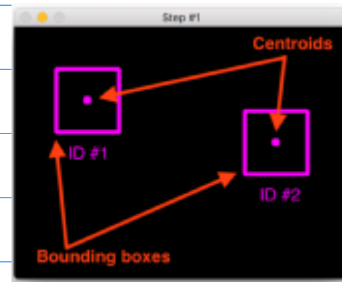
Team Presentation: May 10.
a. 7~8 min. for Presentation & Demo.
b. 5~6 Slides
c. Demo, Show & Tell
d. $1 \sim 1\frac{1}{2}$ min Q&A

Topics: 1. Tracking Algorithm.
2. Semantic Segmentation

2022S-116a-semantic-segmentation-rcnn-2022-4-26.pdf

2022S-115-object-tracker-hand-calculation-HL-2020-8-3...

Objective: To keep Track object movement/Position from time t_1 to t_2, to establish one-to-one relation Based on the shortest distance principle.



1. Denote Objects as $\vec{P_i}(x_i, y_i)$, and $\vec{P_{i+1}}(x_{i+1}, y_{i+1})$

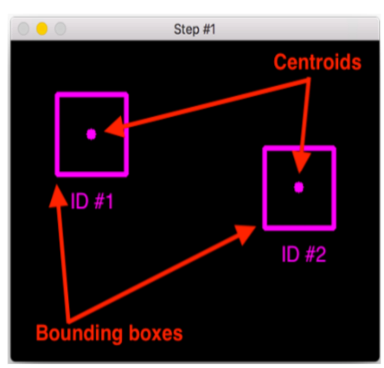
$\vec{P_i} \to \vec{P_i}(x_i, y_i), (x_i, y_i)$

2. distance

$$d(\vec{P_i}(x_i, y_i), \vec{P_{i+1}}(x_{i+1}, y_{i+1}))$$
$$= d(\vec{P_i}, \vec{P_{i+1}}) = \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2} \quad \ldots (1)$$

Step 1. $\bar{x}, \bar{y}$ Computation.

Step 1. Compute Centroid



The algorithm:

1. Compute centroid, is given in my lecture image processing an computation

2. Assign ID to each

Centroid computation re 2019S-24-2018S-114-C Inference-final-2018-4-3

Gray Scale Image → Binarization

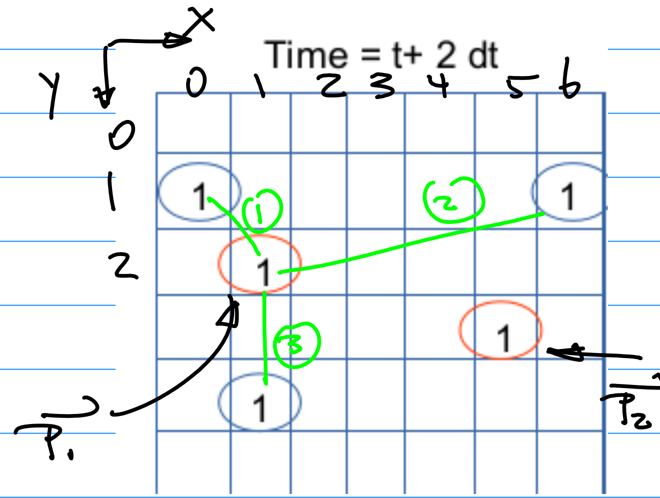
T=3

	4					
	3	2				
	3			5	3	7
				1	2	

Time = t

	1					
	1					
	1			1	1	1

Time = t



Compute $\bar{x}, \bar{y}$ for Each object
And Create A Registration Table
& place the Object ID, and its
$\bar{x}, \bar{y}$ in the table

$$dist_1 = \sqrt{(x_1 - 0)^2 + (y_1 - 1)^2}$$
$$= \sqrt{1^2 + 1^2} = \sqrt{2} = 1.414$$

✓ D(o1, o1_new) = sqrt(2); D(o1, o2_new) = sqrt(26); D(o1, o3_new) = sqrt(4); D(o2, o1_new) = sqrt(29); D(o2, o2_new) = sqrt(5); D(o1, o3_new) = sqrt(17);

Obj. No.	ID	x-bar, y-bar
Object 1.	1	1, 2
Object 2.	2	5, 3

$$dist_2 = \sqrt{(1-6)^2 + (2-1)^2}$$
$$= \sqrt{26}$$

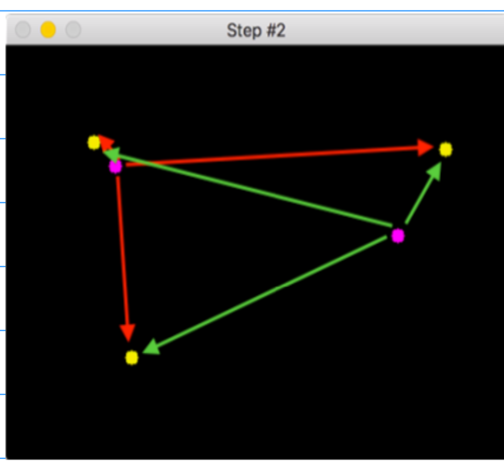
Next, Compute All Distances

Note: Pink Dots (t_1), Yellow Dots (t_2)
the Distance should be Computed
for Time t_1 & t_2.

$$dist_3 = \sqrt{(1-1)^2 + (2-4)^2}$$
$$= 2$$

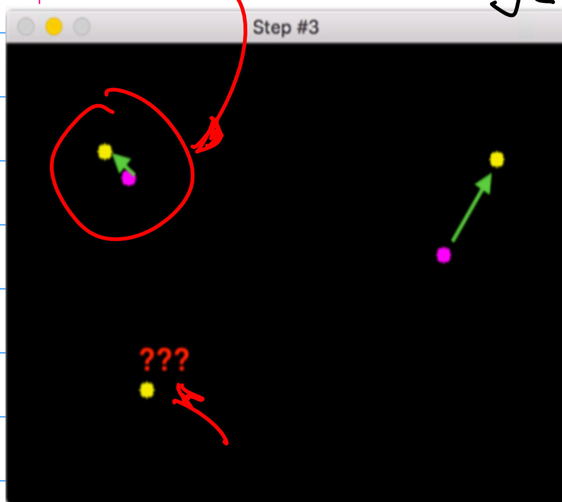
$$\min\{dist_1, dist_2, dist_3\} = dist_1$$

Shortest Distance,
therefore Object $\vec{P_1}(x_i, y_i; t)$ Now is
at Location $\vec{P_1}(x_i', y_i'; t+1)$,
$(x_i', y_i') = (0, 1)$



CMPE258 April 26, 22 49

Verified, Shortest Distance $= \sqrt{2}$



New Point Appeared

Note 1: at time t_2, Generally Speaking this new point does not provide shortest distance (!) So place this new point in the Registration Table.

Note 2: Once the matching is established, then Update the Registration Table.

Homework (1st, Optional)

Implement the tracking Algorithm in Python or C++ to Verify the Example given in the class (PPT).

Due A week from today May 3rd.

Example: Semantic Segmentation. 4 Steps

Fully Connected

1. Replace FC layers with convolutional layers.
2. Convert the last layer output to the original resolution.
3. Do softmax-cross entropy between the pixelwise predictions and segmentaion ground truth.
4. Backprop and SGD

a.

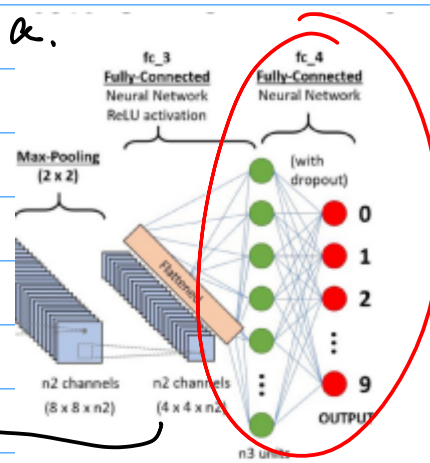


Fig. 1

First, Convolutionalization of F.C. Layer.

1x1 Convolution

Same Process as KxK Convolution.

1x1 Convolution

Image 5X5

-5	3	2	-5	3
4	3	2	1	-3
1	0	3	3	5
-2	0	1	4	4
5	6	7	9	-1

-10	6	4	-10	6
8	6	4	2	-6
2	0	6	6	10
-4	0	2	8	8
10	12	14	18	-2

Kernel 1X1

2

$-1 * 2 = -2$

Kernel 1x1

Scaling Operation.

Fig. 2

Note: the Number of Kernels for Convolution And the matching Output Layers.

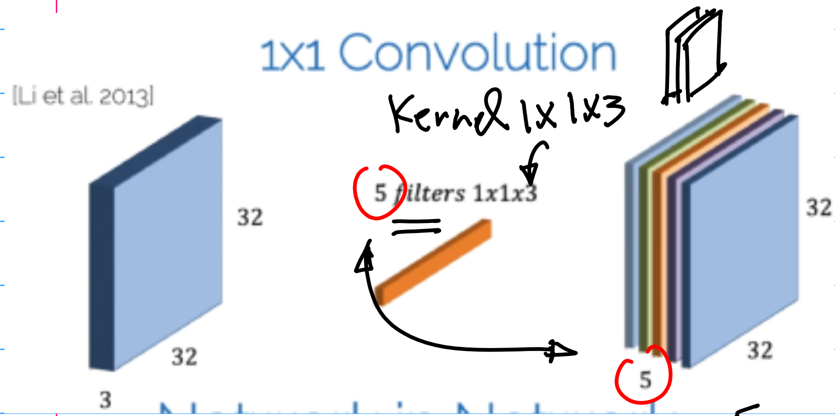


Fig 3.

Interpolation.
un-pooling.

Convolve Feature map 32x32x3 once by 1x1x3. → Output 32x32x1

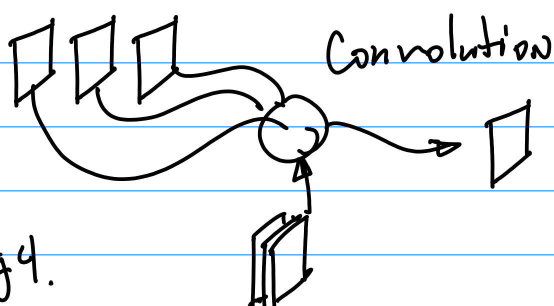


Fig 4.

1x1x3 Kernel

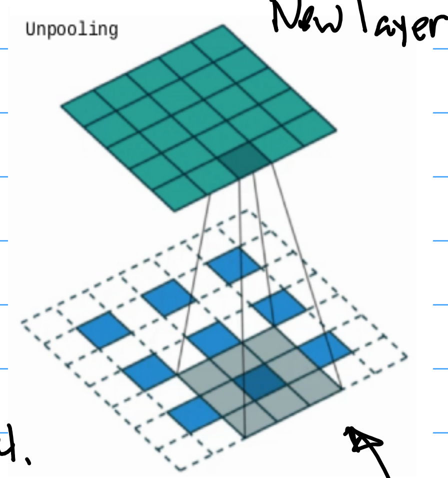


Fig 4.

Pad "0"s Around each element (pixel)

Observation: 1x1xK Convolution allows us to convert a Single Neuron to 2D Convolutional Layer (point).

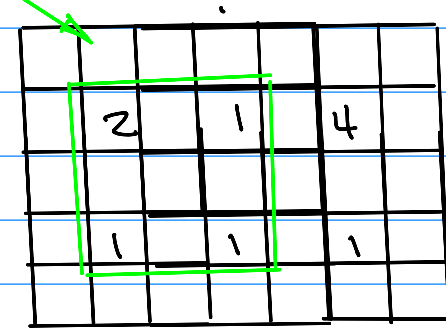
1x1xK Convolution allows us to implement filtering operation as illustrated Above.

Suppose

1	-2	3
2	1	4
1	1	1

$\frac{1}{9}(2+1+1+1)$

⋮



Consider Up Sampling (Super-Sampling) Techniques.

Then, Choose un-pooling Operation, such as Average if more than 1 Non-zero elements under the position of 3x3 Kernel.