CMPE258
Fall 2023

1/

August 22 (Tue)

Organizational meeting.

1. Class Material on github

github/hualili

aiv100	Add files via upload
deep-learning-2020S	Add files via upload
deep-learning-2022s	Add files via upload
deep-learning-2023s	Add files via upload
facial-detect	Add files via upload
lec1Capture/CMakeFiles	Delete CMakeDirectoryInformation.
lecOpenCV_GL	openGL and openCV sample commi
riscv	Create readme.txt
20-2021S-0-7-1convnets-NumeralD...	Add files via upload

Course and Contact Information

Instructor(s): Harry Li
Office Location: Engineering Building, Room 267A
Telephone: (650) 400-1116 for text messaging only
Email: hua.li@sjsu.edu
Office Hours: M.W. 3:00-4:00 pm

IN-Person.

Class Days/Time: Tuesdays and Thursdays 4:30-5:45 pm.
Classroom: Engineering Building Room 337
Prerequisites: CMPE 255 or CMPE 257 or instructor consent. Computer Engine
Engineering majors only.

Course Description

2. Prerequisites Requirements

Bring your Proof to the next Class.

3. Emphasis on "Deep Neural Networks" & Semantic Segmentation

Course Description

Deep neural networks and their applications to various problems, e.g., speech recognition, image segmentation, detection and recognition of temporal and spatial patterns, and natural language processing. Covers underlying theory, the range of applications to which it has been applied, and learning from very large data sets.

Note: Definition (HI): (Human Intelligence)
is Symbolic Representation of
Learnt Experience.

CMPE258

Fall 2023

4. Projects. 2

plus 1 team project (Semester Long) } ~~30%~~ 25% ~~30~~ pts

5. In-Person Class; CANVAS is utilized to post Homework/Project Requirements, and to collect the submission of the Homework, as well as for the exams.

This course is an online course. NO The students must have Internet connectivity and Zoo
his/her machine. The students must participate in the class activities and submit all as:
exams to SJSU CANVAS. The syllabus, faculty contact information on the syllabus, I
projects, and exam papers are all available on CANVAS. See University Policy F13-2

6.

Grading Information

Quiz, Homework, Projects	30%
Midterm Examination	30%
Final Examination	40%

Note:

1° CANVAS To Be up by the end of the day, Friday;

2° Tools & Software To be installed (will provide "Readme" as Ref)

OpenCV

Python.

T.F. Version 2.0 or higher

7. Textbooks & References

Textbook

- Deep Learning with Python, 1st or 2nd Edition, by François Chollet, ISI
  ISBN-10: 9781617294433, https://github.com/hualili/opencv/blob/mast
  2018F-6-DeepLearningCh02.pdf
- Robot Vision by B.K. P. Horn, the MIT press, ISBN 0-262-08159-8, c
  Hill).
- Reference textbook Learning OpenCV, Computer Vision with the Ope
  Kaebler, O'Reilly Publisher, ISBN 978-0-596-51613-0, 2011.

Today's Topic: Intro. to Deep Convolutional Neural Networks.

Ref: github.

2022F-103-NN-Intro-Python-v5-2022-8-25

Note: Lab Space for the class Rm268.

8. Software Tools & Dev. Environment

Python. Pycharm.

AnnoConda.

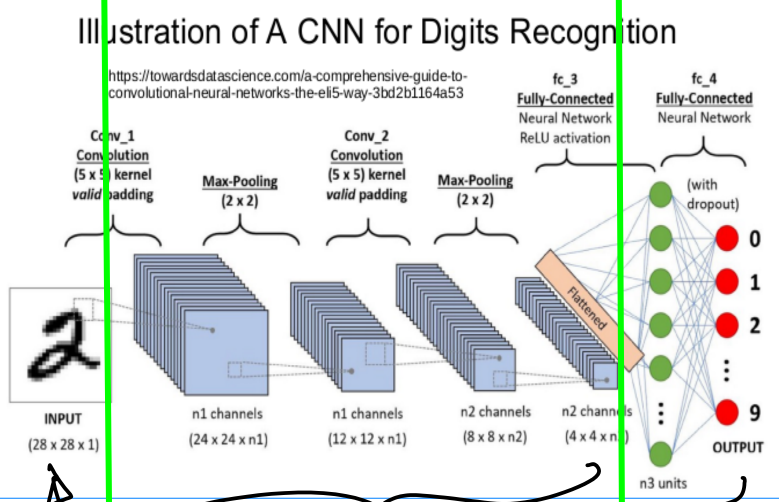
TensorFlow.

Note: Rm268 Available ~~Per~~ ON Approved Basis.

August 24 (Thursday)

Example: Architecture Overview.

Note 1.



Preprocessing
Computer Vision.

Convolution Layers

Feature Extraction

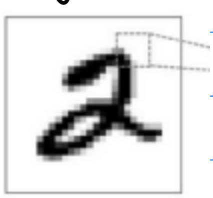
To produce Feature Vectors.

$\vec{u} = (u_1, u_2, \cdots, u_N)$

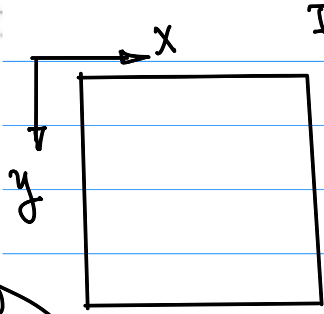
Feed Forward Neural Networks.

Decision Making

2. Image Definition



M N Channel

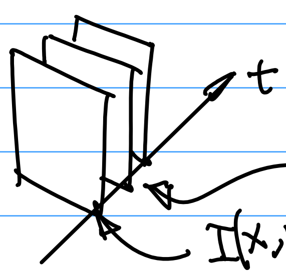


$I(x, y; t)$

Note: for x & y definition see Ref. Next page.

$M \times N$

No. of col. Per Row. No. of Rows per frame.



$I(x, y; t+\Delta t)$

$I(x, y; t)$

CmpE258

Fall2023

Ref: on the github.

2023S-101-Note-cmpe258-2023-03-16.pdf

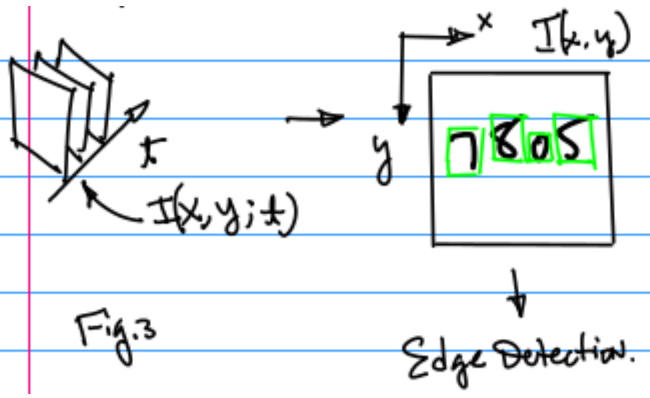


Fig.3

Comparison to Web Cam.

1080p Resolution: 1920x1080

$1920 \to \sim 2^{11} = 2 \cdot 2^{10} = 2K$

$1080 \to \sim 2^{11}$

$2^{22} = 2^2 \cdot 2^{20} = 4$ Meg

$\simeq 4$ million.

3. For the Convolutional Lay, we denote it as C_i, where $i = 1, 2, \cdots, N$

for Max pooling Layer, we denote it as M_j, where $j = 1, 2, \cdots, K$

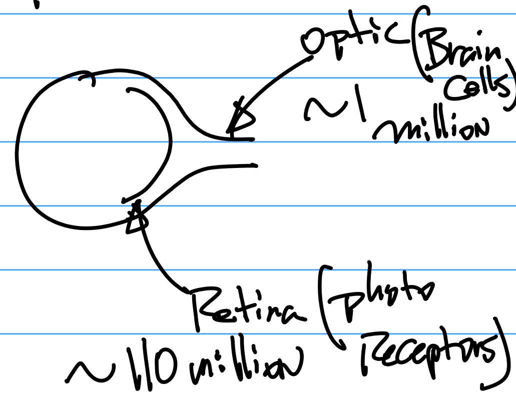
So, for the example, we have

$C_1 M_1 \to C_2 M_2$

Maxpooling for Resolution Reduction / Feature Reduction.

$$I(x,y;t)_{M \times N} \to \boxed{M_i} \to I_{\frac{M}{2} \times \frac{N}{2}}(x,y;t)$$

Note: BioLogic Inspiration, Human Visual Perception System, Retina, ~110 million photo Receptors



4. At the 3rd Segment (Blocks) of the Architecture, we denote Feed Forward Neural Network as F_M

M: No. of Neurons / Nodes

for example, F_{10} (10 Nodes) for the output Layer.

August 29 (Tue)

Note: 1° CANVAS is up.

2° Homework Assignment

a. Honesty Pledge to Be Signed / Signed Copy has to be uploaded to CANVAS.

b. Software Tools Installation. (Opt)

① OpenCV. By Friday

Next Tuesday. Bring Your Laptop w/ OpenCV installed.

Please use Smartphone to take a photo, And upload the photo to your laptop to display.

Note: Sample Code (Python) was posted on the github.

② Anaconda Installed on your Laptop.

Note: Readme for Anaconda installation was posted on the github.

③ Create ChatGPT Account.

Python Interface to ChatGPT API (3.5 Version) is to be utilized in your Team Project.

3° Form A Team for the Semester Long Project.

Example: Consider 2D Convolution Technique.

Ref: github/hualili/opencv

2022

Note 1:

$$c(n_1, n_2) = \sum_{k_1=-\infty}^{\infty} \sum_{k_2=-\infty}^{\infty} a(k_1, k_2) b(n_1 - k_1, n_2 - k_2) \quad \ldots (1)$$

Image Kernel

$n_1 = x$, $n_2 = y$

Summation Index:

k_1, k_2 are for x and y.

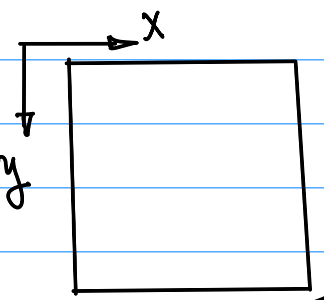


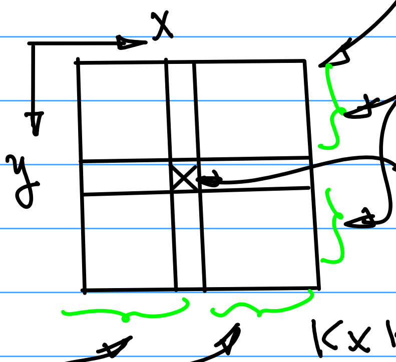
Fig. 1

Note 2. $a(x,y)$ or $a(k_1, k_2)$ as an Image. $I(x,y)$

$b(x,y)$: a kernel for 2D convolution, $b(k_1, k_2)$

Note 3. Kernel $b(x,y)$ can be rewritten $K(x,y)$

Size of A kernel is denoted as $K \times K$



K(x,y)

The No. of Rows Above the P.O.I. = No. of Rows Beneath the P.O.I.

Pixel of Interests.

Fig.2

K x K

K = 3, 5, 7, ... (odd Number)

The No. of. Col. Right to the P.O.I. = The No. of Col. Left to the P.O.I.

I(x,y)

0	0	100	100
0	0	100	100
0	0	100	100
0	0	100	100

Fig.3a

4x4

8 bit Grayscale: $2^8 - 1 = 255$

-1	0	+1
-1	0	+1
-1	0	+1

Fig.3b

Step1. Take the Kernel, and place it at the initial position.

0	0	100	100
0	0	100	100
0	0	100	100
0	0	100	100

Fig.3c

$-1 \times 0 + 0 \times 0 + 1 \times 100 +$
$-1 \times 0 + 0 \times 0 + 1 \times 100 +$
$-1 \times 0 + 0 \times 0 + 1 \times 100$
$= 300$

I(x,y) * K(x,y)

	300		

Fig.3d

Step2. Shift K(x,y) to the Right By 1 pixel (on the Same Row).

0	0	100	100
0	0	100	100
0	0	100	100
0	0	100	100

Fig.3e

$-1 \times 0 + 0 \times 100 + 1 \times 100 +$
$-1 \times 0 + 0 \times 100 + 1 \times 100 +$
$-1 \times 0 + 0 \times 100 + 1 \times 100 = 300$

CmpE258

F2023

7/

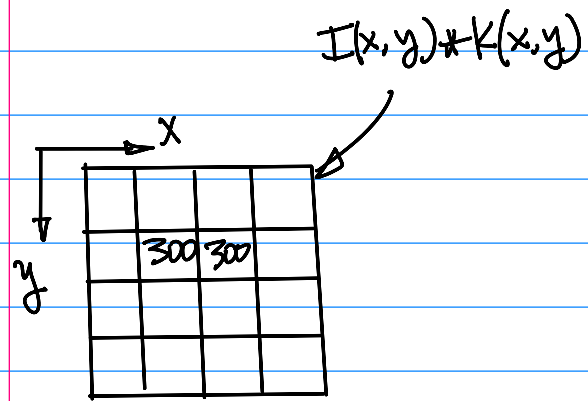


Fig.3F

Step3.

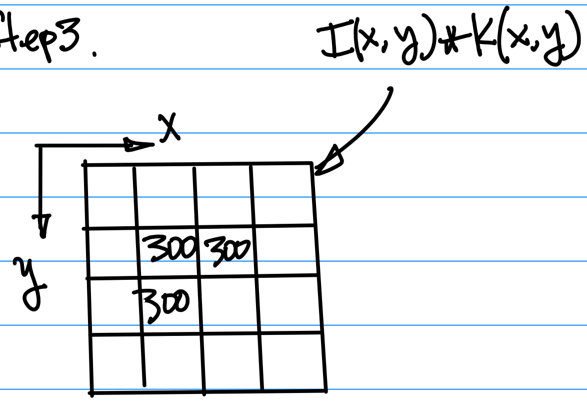


Fig.3G

Step3.

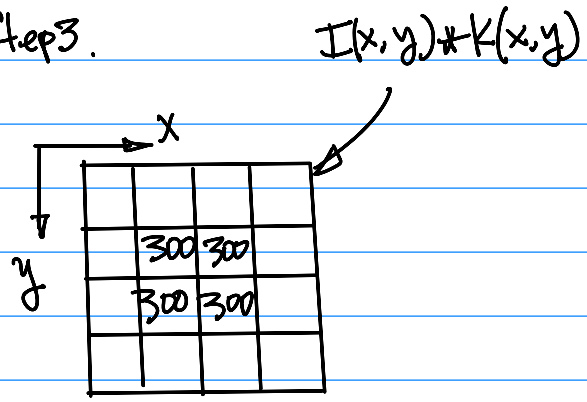


Fig.3H

Note: 2D Convolution (Continuous Case)

given Image $f(x,y)$
a Kernel $h(x,y)$

$$\iint_{\Omega} f(u,v)\, h(x-u, y-v)\, du\, dv \quad \cdots (2)$$

August 31 (Th).

Note: 1° Honesty Pledge on CANVAS, Signed pdf due this Friday (ON CANVAS). / Homework ON CANVAS

2° Will post Anaconda Installation & OpenCV Installation, display an image,

(1) Screen Capture of the Activated CONDA Environment, with personal identifier;

(2) Screen Capture of the OpenCV Display with P.I.D.

Sample Code for the display to Be Re-posted on github, 2023F

Example: Theoretical Aspects for 2D Convolution.

Ref: 1° PPT on the github.

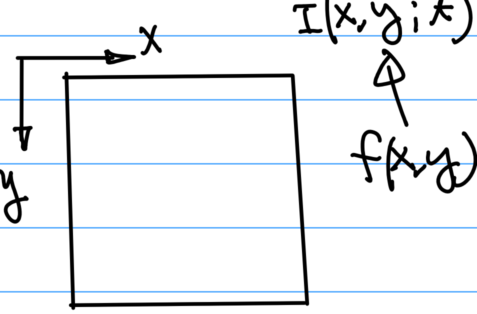
2° 2023S-100-note-... 2023-03-21.pdf

$$\iint_{\Omega} f(u,v)\, h(x-u, y-v)\, du\, dv \quad \cdots (1)$$

$f(u,v)$ ← Image

$h(x-u,y-v)$ ← Kernel

Note 1. Ω: Image Plane(s)



VR

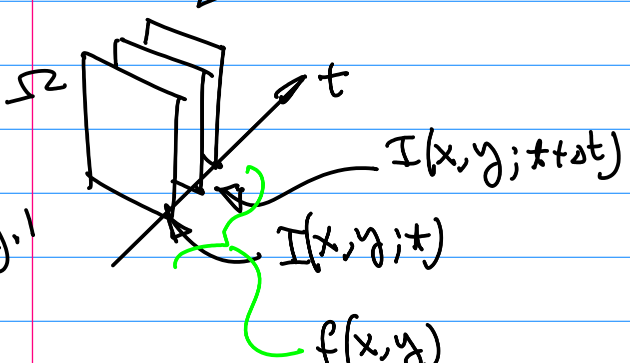
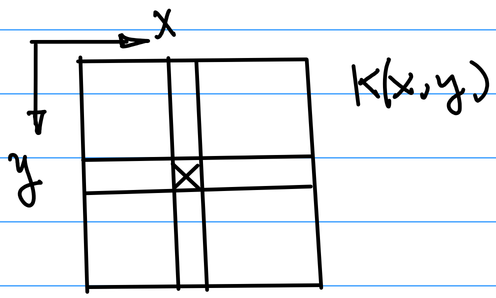


Fig.1

Note 2: Kernel



K×K

Fig.2a

↓ (1)

K(-x, y)

Flip the Kernel w.r.t y-axis

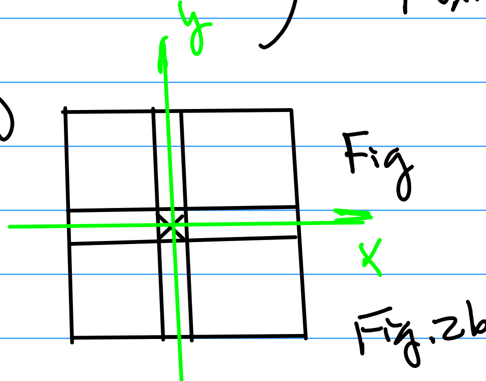


Fig.2b

(2) Flip the Kernel w.r.t. x-axis.

(since K(-x,y) changed to K(-x,-y))

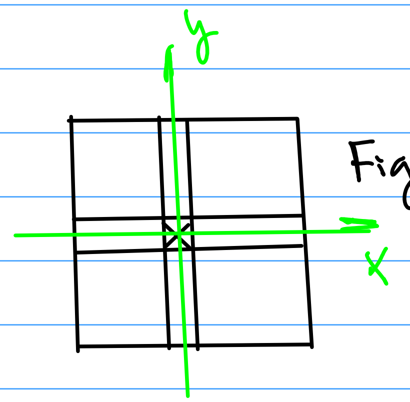
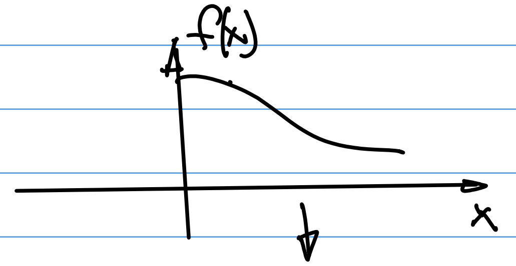
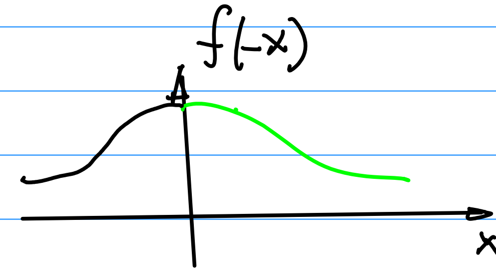


Fig.2C

Note: for 1D Case



↓



Next, make $K(-x,-y)$ to $K(x_0-x, y_0-y)$

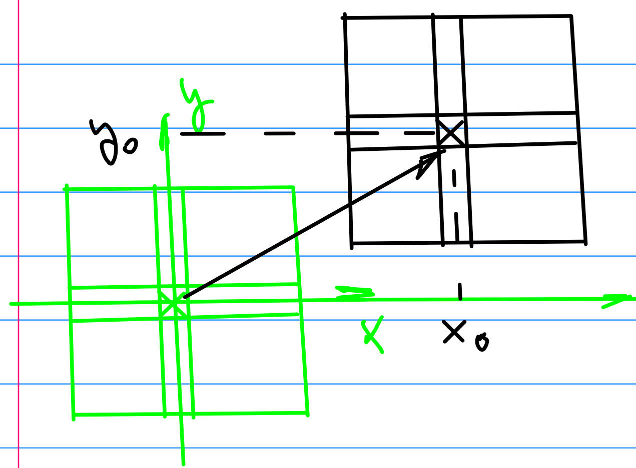


Fig. 2d

Note: $\frac{k-1}{2}$ for the top Rows
$\frac{k-1}{2}$ for the Bottom Rows
Lost

$$2 \times \frac{k-1}{2} = k-1$$

Similarly for the col.s.

Therefore,

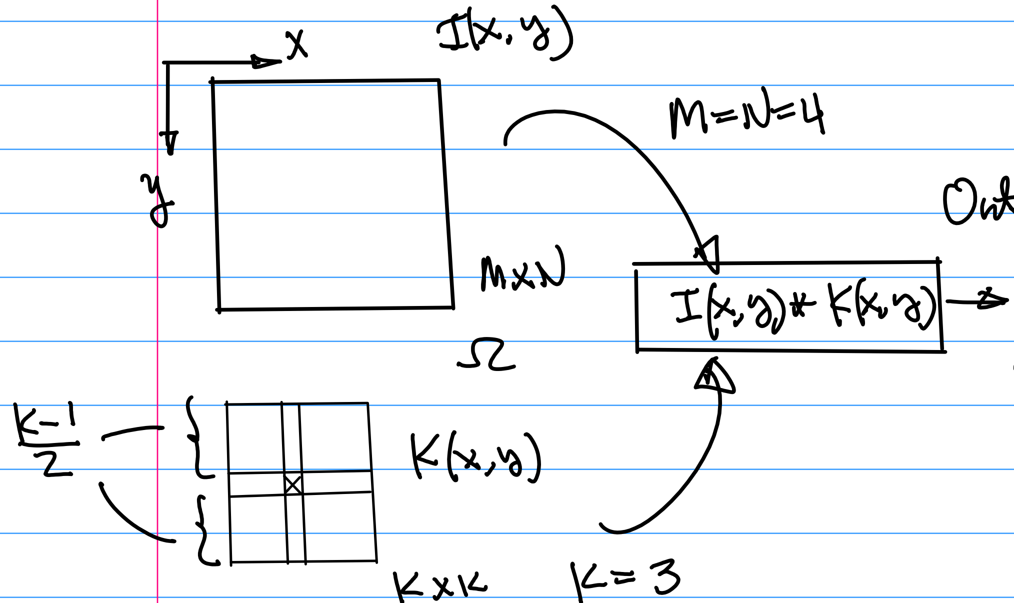
Input Dimension $M \times N$ → Output After Convolution $M-(K-1) \times N-(K-1)$ ... (3)

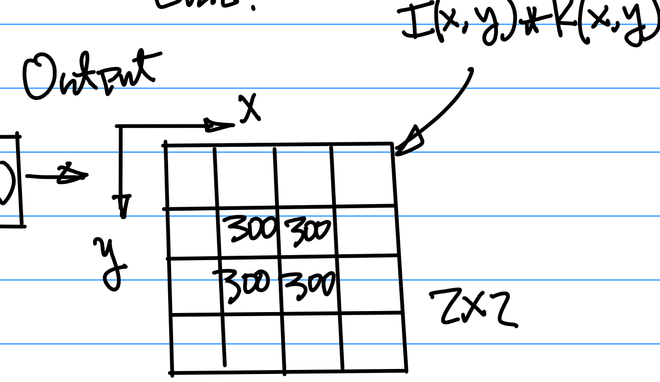
Now, for Discrete/Digital Image (OR Feature plane)

Replace $\iint$ → by $\sum_{k_1}\sum_{k_2}$

$$c(n_1,n_2) = \boxed{\sum_{k_1=-\infty}^{\infty}\sum_{k_2=-\infty}^{\infty}} \underbrace{a(k_1,k_2)}_{\text{Image}}\underbrace{b(n_1-k_1,n_2-k_2)}_{\text{Kernel}} \quad \cdots (2)$$

Dimension Change Due to the Convolution.





Sept. 5 (Tue)

Note: 1° CANVAS OpenCV, Anaconda Installation;

2° Homework Coming On Thursday.

a. Installation of T.F. Version 2.0 or Higher;

b. ChatGPT 3.5 API Down Load.

Example: Continuation on Convolutional Layers and Architecture for Hand written Digits Recognition.

Based on Eq(3), pp.9.

$M \times N$ Image $\xrightarrow[\text{Convolution}]{K \times K}$ $M-(K-1) \times N-(K-1)$

From MNIST Architecture, the Input Image $I(x,y)$ with $M \times N$ $(M=N=28)$. the output feature Layers Resolution 24×24. $\rightarrow$ $K = \underbrace{(28-24)}_{\text{Resolution difference.}} + 1$

Now, the total No. of feature layers $= n|$

Since one Convolution (e.g. using one $K \times K$ Kernel to perform Convolution) produces 1 feature Layer,

Therefore

$n|$ Convolutional feature Layers is the result of $n|$ Convolutions, e.g. $n|$ different Kernels.

Note: Multiple feature Layers are due to the inspiration of Human Visual Perception System, e.g. "Early" Vision System.

LoG Laplace of Gaussian.

Gabor Operators

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \quad \cdots (2)$$

$$G(x,y) = \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{(x-\mu_x)^2+(y-\mu_y)^2}{\cancel{2\sigma^2}\; 2(\sigma_x^2+\sigma_y^2)}} \quad \cdots (1)$$

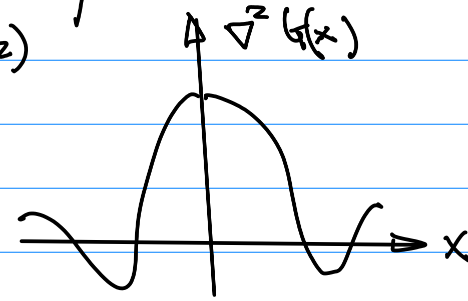


Fig 1.

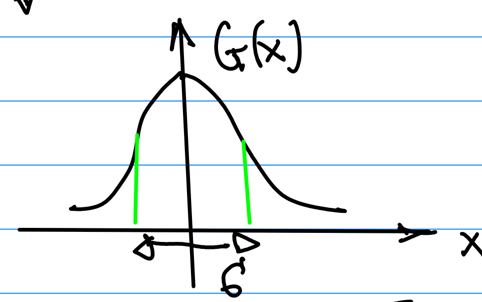


Fig 2

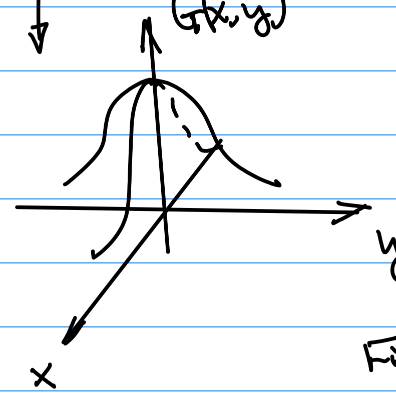
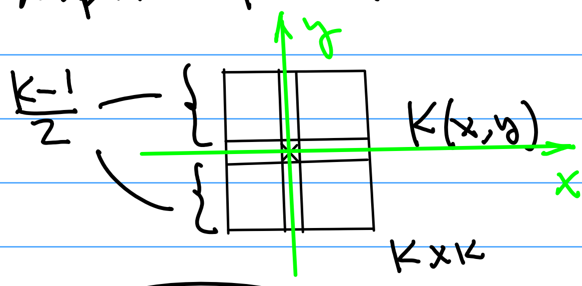


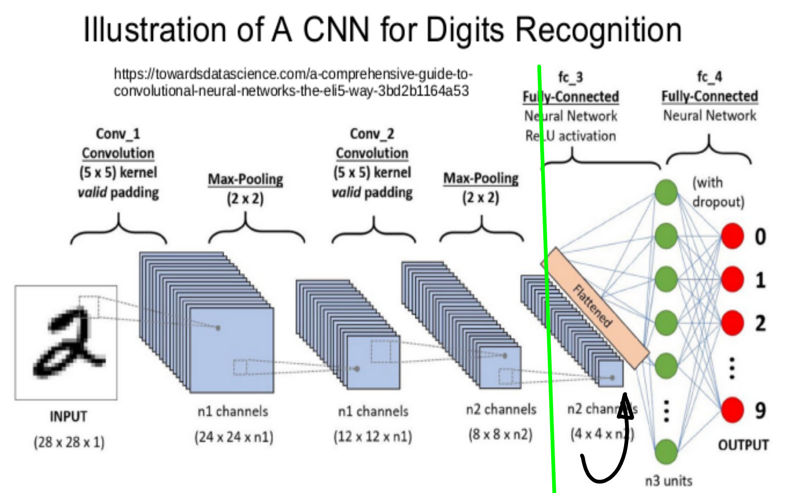
Fig 3

Take G(x,y) in Fig.3, Map it to KxK.



n1 Kernels Produce n1 Layers.

Now, consider the Architecture of MNIST



Consider Re-arrange 2D feature Layers / Each into 1D Neurons.

Now, consider a technique for feature Reduction,

Given an image I(x,y)

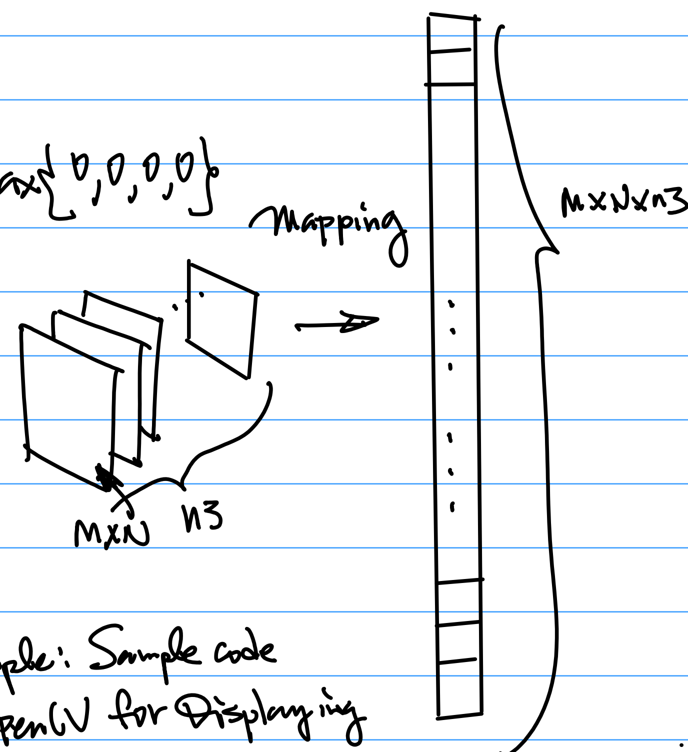
0	0	100	100
0	0	100	100
0	0	100	100
0	0	100	100

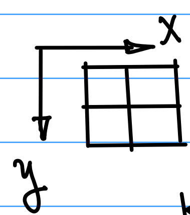
Feature Layer

Maxpooling →

0	100
0	100

Max{0,0,0,0}





KxK where K=2

Example: Sample code OpenCV for Displaying an image

```
cv2.imread(imageName, cv2.IMREAD_COLOR)
cv2.imshow(window_name, src)
```

Sept. 7 (Th).

Note: 1° About OpenCV Reference/Sample Code.

a) On github.

opencv / deep-learning-2022s / 2022S-109c-v2-localization-contours.py

hualili Add files via upload

for Video Capture to get A Single Frame.

b) Sample for Single frame, e.g. .jpg, .png etc Image Input.

2022S-104d ~ Python Code

c) Script for Conda Environment On github:

2022S-104b ~
" -104c ~

d) Readme for Creating Conda Environment.

Note 2. Homework Due A week from today (Sept. 17, Sun)

Requirements:

a) Installation of T.F. Version 2.0 or higher

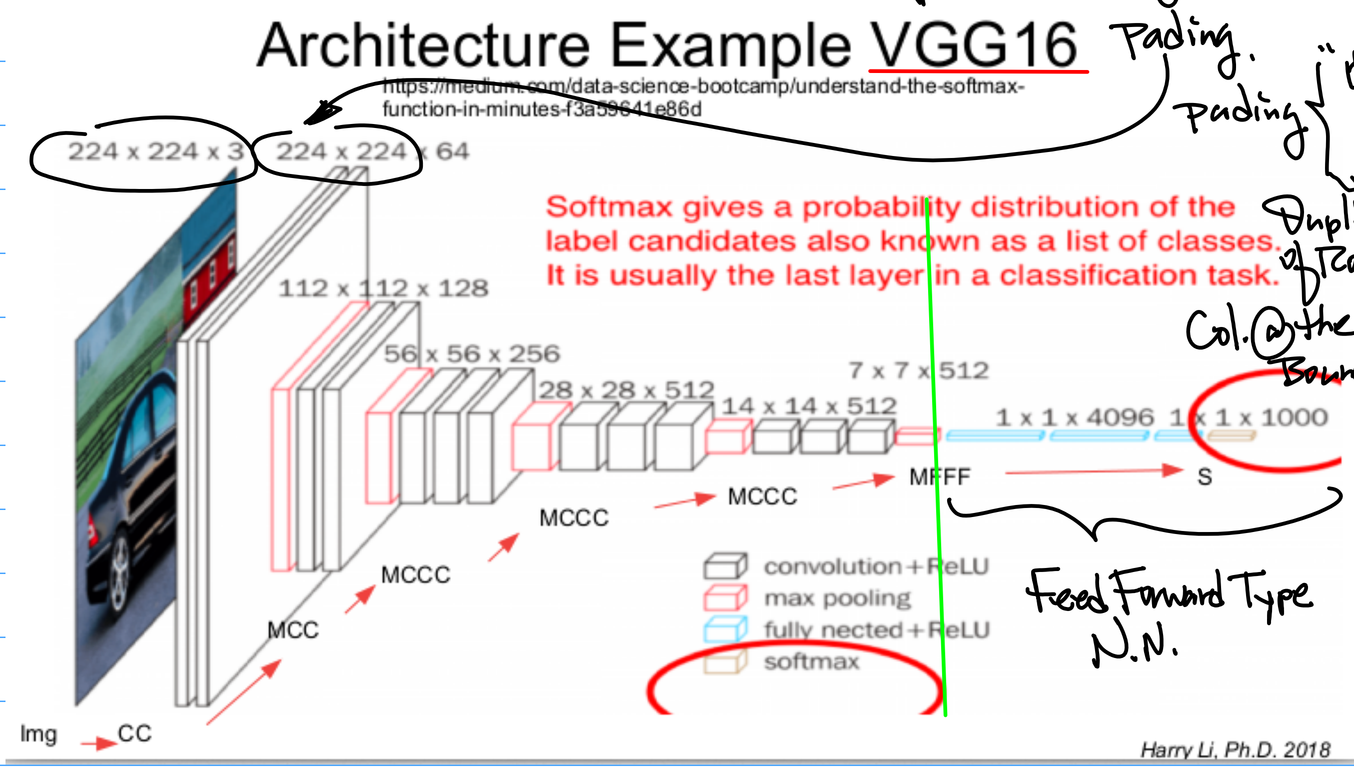
b) Screen Capture that shows T.F. installed Successfully. (with Personal Identifier)

Submission on CANVAS.

Example: Convolutional Layer, Architecture Analysis.

Note 1. Architecture; Note 2: Pading v.s. No Pading.

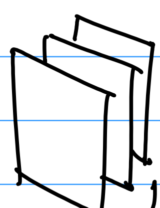
"φ"s pading; Duplication of Const + Col. @ the Boundary



Feed Forward Type N.N.

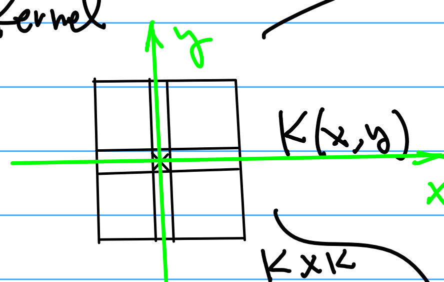
Note 3: Convolution Discussion.

One Example



Colour Image

Kernel



$K(x,y)$

$K \times K$

	x		
y	-1	0	+1
	-1	0	+1
	-1	0	+1

$I(x,y) = (r(x,y), g(x,y), b(x,y))$

LoG $\nabla^2 G(x,y) \rightarrow$ PP.10.

$\nabla^2 G(x,y;\mu_x,\mu_y,\delta_x,\delta_y)$

$\rightarrow \left(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}\right) G(x,y;\mu_x,\mu_y,\delta_x,\delta_y)$

To perform Convolution. $\rightarrow K(x,y)$

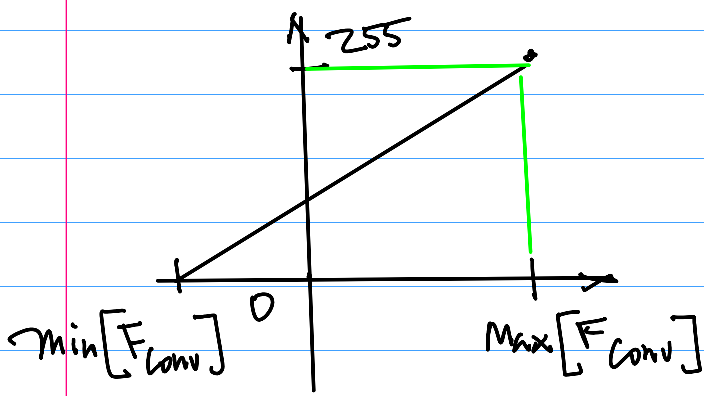
First, kernel is one channel in this case;

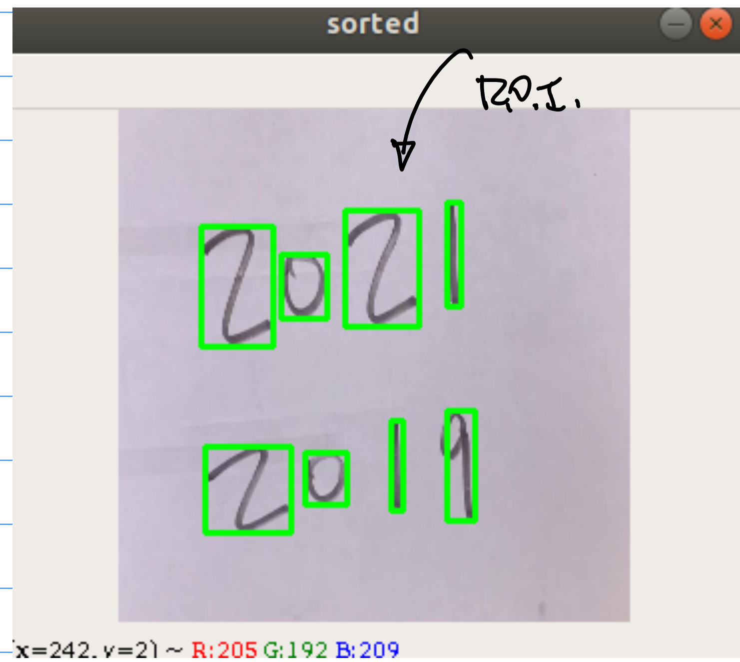
Then, Convolutions. $r(x,y) * K(x,y)$, $g(x,y) * K(x,y)$, $b(x,y) * K(x,y)$

CmpE258

F2023.

$$F_{conv}(x,y) = \frac{1}{3}\left[r(x,y) * k(x,y) + g(x,y) * k(x,y) + b(x,y) * k(x,y)\right]$$





Region of Interests.

Consider A Design of S.I.D Recognition System Using T.F & MNIST CNN as a processing engine.

Requirements:

1° Live CAM Input,
720P or 1080P
1280x720 (M x N)
Col. Row.
1920x1080 (M x N)
Col. Row.

2° Printer Paper (Blank/White) Black mark to Writ 4 Digits

3° Localize R.O.I. ON Each Digit

Sept. 12 (Tue).

Note 1. For Homework 1 Extended to the Saturday, Requires the Submission of Python Code.

Note 2. Team Project.
Ref: 2023F-103-project-team-2023-9-12.pdf
ON CANVAS as well.
Due on Nov. 27 (Monday) 11:59 P.M.

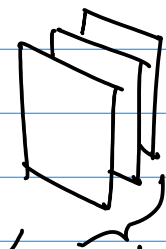
Example: Suppose a color image $I(x,y)$ is given below, and a Convolution Kernel $K(x,y)$

is given as follows

A color Image $I(x,y)$

$$\frac{1}{6}\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix}$$

3x3.



Colour Image

Find Convolution Result.

$$I_r(x,y) = \begin{bmatrix} 0 & 0 & 0 & 255 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad I_g(x,y) = \begin{bmatrix} 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 100 \\ 0 & 0 & 5 & 100 \\ 0 & 0 & 2 & 0 \end{bmatrix} \quad I_b(x,y) = \begin{bmatrix} 0 & 0 & 255 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 255 & 0 \end{bmatrix}$$

$$I_r * \frac{1}{6}\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & -255 \\ 0 & 1 \end{bmatrix}$$

$$I_g * \frac{1}{6}\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix} \rightarrow \begin{bmatrix} -95 & 5 \\ 2 & -98 \end{bmatrix}$$

$$I_b * \frac{1}{6}\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix} \rightarrow \begin{bmatrix} -255 & -255 \\ 255 & 255 \end{bmatrix}$$

Note: In the future, we have Need to Reduce the Number of feature Layers

By introducing Newer Kernel(s) for Convolution.

CmpE258

F2028

Homework: Due 1 week from Today.
(Sept. 19th).

1° Installation of T.F., Screen Capture the installation Result.
(w/ Personal ID).

2° OpenCV Code to Handle
a. Live Web CAM Input Video, and Display it;
b. file input, MPEG4 format Video and display it.

Note: Please use A Printer Paper, Write with a Black marker, 4 Digits of your SID.

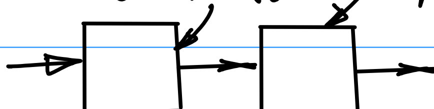
Submission: the code & Video Clips.

Preprocessing for Hand Written SID Recognition System Design.

Live Web CAM Input

Output(I)

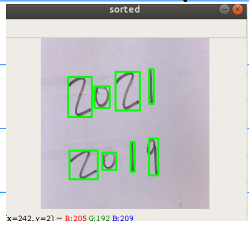
Output stage II



Make I(x,y) to match the Resolution & Channel Requirements of the DCNN Engine Input (e.g. 28x28x1) without Distortion!

Processed

Localized R.O.I.



Region of Interests

Ref:

Homework: Due 1 week from today. Sept 21st.
Use your OpenCV for the following preprocessing functions.

1° Convert the color Image $I(x,y)$ to a gray scale Image. $I_G(x,y)$

2° Binarize the gray scale image to Obtain a Binary Image $B(x,y)$.

3° Perform CANNY Edge Detection ON $I_G(x,y)$, and display the edge map.

4° Perform Gaussian Blur ON the Gray Scale Image.

Submission:

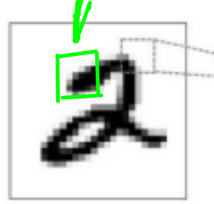
5° Screen Capture of 1° to 4°. with Personal Identifier.

Example: Continuation of project Discussion. Preprocessing.

The First Objective: To get all the Bounding Boxes

Color Video → Gray Scale Image.

255		
	200	
		11



INPUT
(28 x 28 x 1)

C. Filtering Operation, Conducted by Convolution w/ Predefined Kernel Coefficient.

B. To Be Continued.
Edge Detection CANNY, LoG, Gabor
Note: Input Should be a grayscale (8 bits), Output should be 8 bit (1 Channel)

A.
Binarization. To Obtain A Representation of A Digit.
Note: Binarization May lead to the Loss of useful information.

Example: Image Binarization.

Given $I_G(x,y)$ or Simply $I(x,y)$
Binarization is defined as

$$B(x,y) = \begin{cases} 255 & \text{if } I(x,y) \geq T \\ 0 & \text{o/w} \end{cases} \quad \cdots (1)$$

Note: T is set for the entire image for Now.

Example from P.18, Lecture Notes on Github, 2023S.

Note: Treat the top Left Corner as (1,1) for the Binarized Image Descriptors Calculation (to Void skewed Result).

Most Important Descriptors are those Built Based on Moments.

$$\frac{\iint_{\Omega} (x-\bar{x})^m (y-\bar{y})^n B(x,y)\,dx\,dy}{\iint_{\Omega} B(x,y)\,dx\,dy} \quad \cdots (2)$$

order

Image Plane. $m \times N$

Binary Image

Area, Size

$$A = \sum_{y=0}^{N-1} \sum_{x=0}^{M-1} B(x,y) \quad \cdots (3)$$

$$\sum_{y=0}^{N-1} \sum_{x=0}^{M-1} (x-\bar{x})^m (y-\bar{y})^n B(x,y) \quad \cdots (4)$$

Where

$$\bar{x} = \frac{\sum_{y=0}^{N-1} \sum_{x=0}^{M-1} x\,B(x,y)}{A} \quad \cdots (4\text{-a})$$

and

$$\bar{y} = \frac{\sum_{y=0}^{N-1} \sum_{x=0}^{M-1} y\,B(x,y)}{A}$$

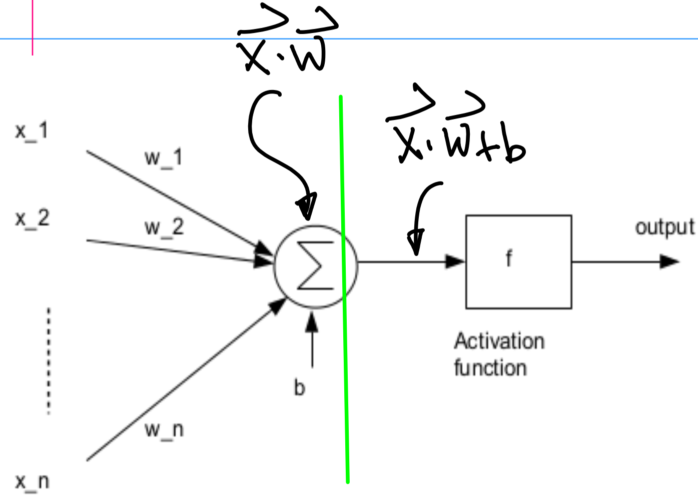
Sept 19 (Tue).

Example: Consider the Architecture of MNIST CNN.

Roadmap:

MNIST CNN with Preprocessing. To generate Bounding Boxes → Yolo. Automate Bounding Boxes Creation R.O.I. → Yolact Semantic Segmentation

Ref: On the github 2023S-103C ~



$$\sum_{i=1}^{n} x_i w_i \text{ or } \sum_{i=1}^{n} w_i x_i$$

$$= \vec{X}\cdot\vec{W} \quad \cdots (3)$$

Together with b (Bias), we have

$$\sum_{i=1}^{n} w_i x_i + b = \vec{X}\cdot\vec{W}+b \quad \cdots (4)$$

Note 1. Feature vector, or Input, Excitation

$$\vec{X} = (x_1, x_2, \cdots, x_n) \quad \cdots (1)$$

$$\vec{W} = (w_1, w_2, \cdots, w_n) \quad \cdots (2)$$

Weights. $w_i \in [0,1]$

where $i = 1, 2, \cdots, n$;

b: Bias, e.g., offset

Note 2. $x_1 w_1 \rightarrow$ the Neuron input 1 to

$x_2 w_2$: input 2 to the Neuron.

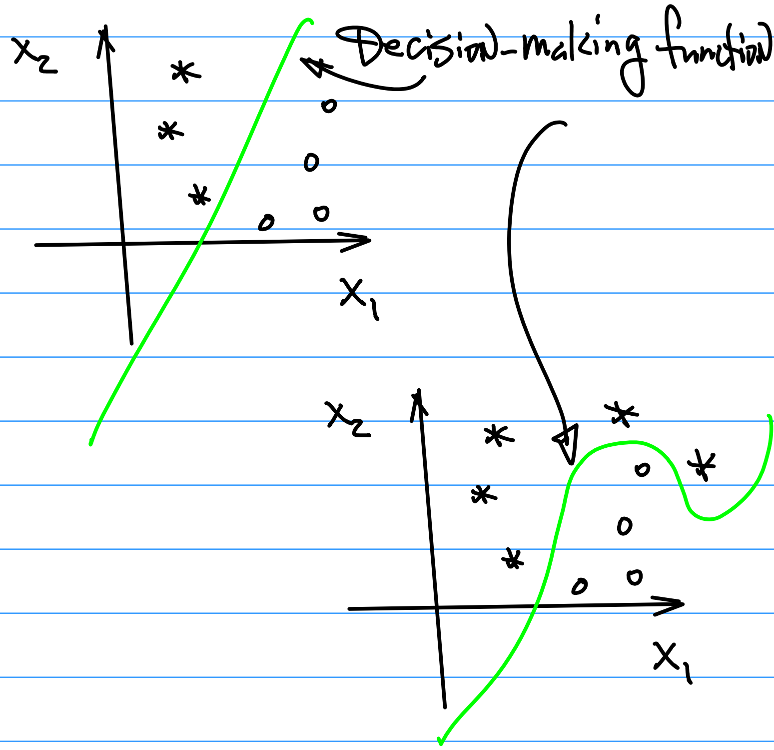
⋮

$x_i w_i$: " i "

$x_1 w_1 + x_2 w_2 + \cdots + x_i w_i + \cdots + x_n w_n$

Note 2: The "Bigger Picture" of this Part of the Architecture.

Feedforward NN.

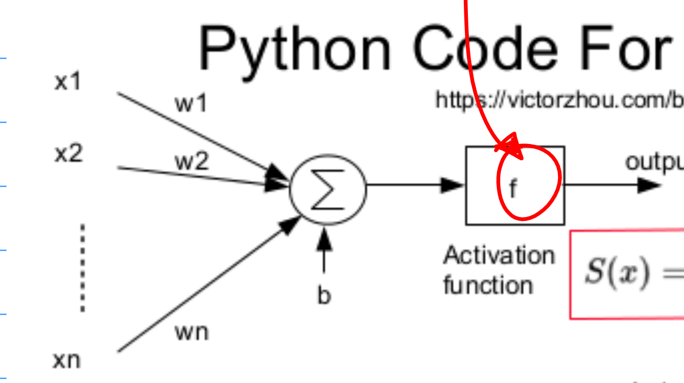


Note 3. Activation Function.

CmpE258

F2023.

$$f\left(\sum_{i=1}^{n} w_i x_i + b\right) \quad \cdots (5)$$

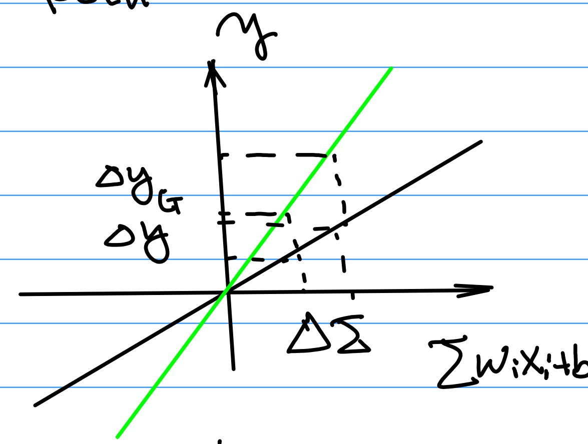


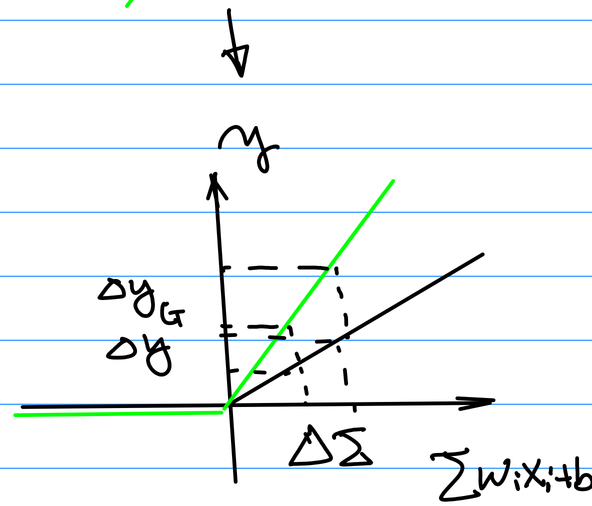
$$O = f\left(\sum_{i=1}^{n} w_i x_i + b\right), \text{ OR}$$

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right)$$

Output of the Neuron.

Note: RELU





Note 4. Sigmoid function

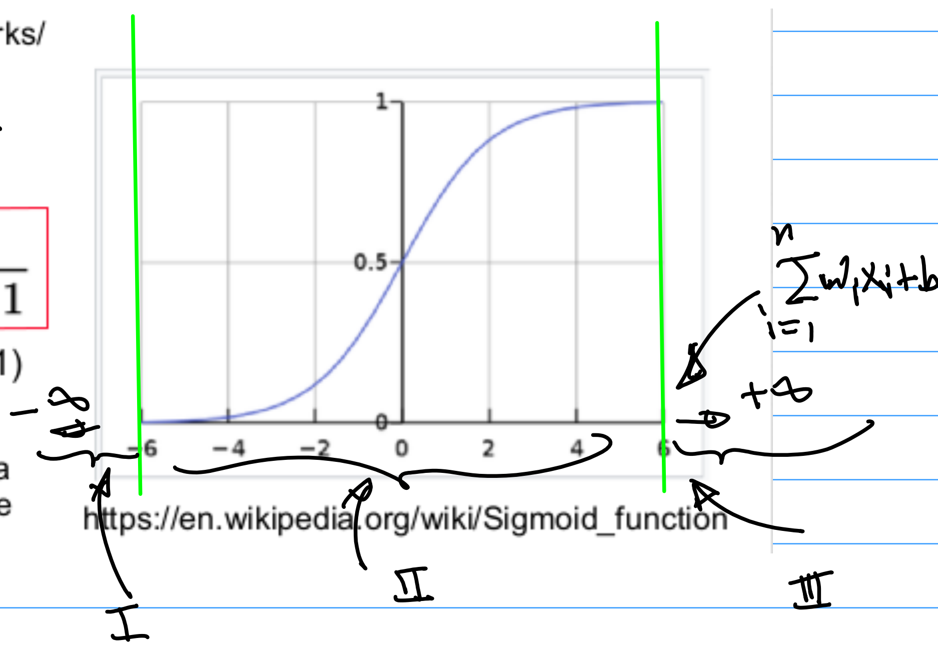
orzhou.com/blog/intro-to-neural-networks/

output

$\sum w_i x_i + b$.

$$S(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1} \quad (1)$$

A sigmoid function is a mathematical function having a characteristic "S"-shaped curve



https://en.wikipedia.org/wiki/Sigmoid_function

CMPE258
F2023

Sept. 21 (Th).

Continuation on the Formulation of Feedforward NN.

Ref: github.

Sample plain python code on github.

Example.

Note: 1. define activation function

```
def sigmoid(x):
  # Our activation function: f(x) = 1 / (1 + e^(-x))
  return 1 / (1 + np.exp(-x))
```

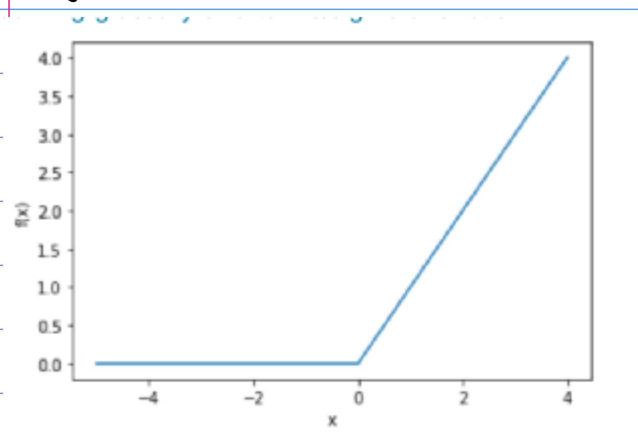
2. define a single neuron

```
class Neuron:
  def __init__(self, weights, bias):
    self.weights = weights
    self.bias = bias

  def feedforward(self, inputs):
    # Weight inputs, add bias, then use the activation function
    total = np.dot(self.weights, inputs) + self.bias
    return sigmoid(total)
```

A sigmoid
mathemat
characteri

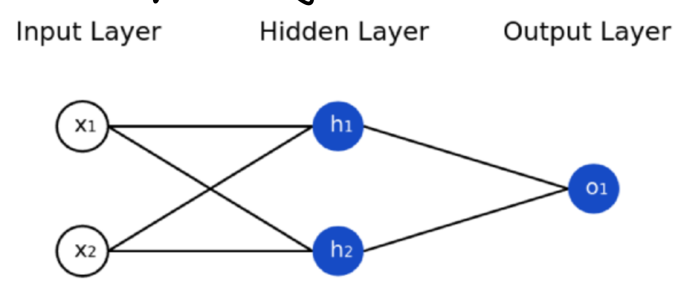
Fig. 1.



$$f(x) = \begin{cases} \alpha x & x \geq 0 \\ 0 & o/w \end{cases} \quad \ldots (1)$$

Note: $y = f\left(\sum_{i=1}^{n} w_i x_i + b\right)$

Can be applied to the Neurons in the following illustration



Note: Comparison to Sigmoid in terms of its Computation

$$S(x) = \frac{1}{1+e^{-x}} \quad \ldots (2)$$

from Taylor Expansion.

$$f(x) = f(x_0) + \left.\frac{df}{dx}\right|_{x=x_0}(x-x_0) + \left.\frac{d^2 f}{dx^2}\right|_{x=x_0}(x-x_0)^2$$

$$+ \cdots + \frac{d^k f}{dx^k}(x-x_0)^k + \cdots + R_n(x) \quad \ldots (3)$$

Note 1. y output of a Neuron. $\begin{cases} y_{pred} \\ y_{true} \end{cases}$ Compare the two, to evaluate the performance of a Network.

Define Loss Function

https://victorzhou.com/blog/intro-to-neural-networks/

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_{true} - y_{pred})^2$$

The Mean Square Error Function as a loss function, very often it is also defined as objective function, e.g., minimize the loss function becomes objective function, as shown below step by step.

$$\downarrow$$

$$y_{true} - y_{pred}$$

$$\downarrow$$

$$(y_{true} - y_{pred})^2$$

$$\downarrow$$

$$\sum_{i=1}^{n}(y_{true} - y_{pred})^2 \longrightarrow \text{Exp}\left[\sum_{i=1}^{n}(y_{true} - y_{pred})^2\right]$$

Note 2. y_{pred}/y_{True}?

$y_{pred} - y_{true}$ $= 0$ the same

$y_{pred} - y_{true}$ $\begin{cases} > 0 & y_{pred} > y_{true} \\ < 0 & \text{o/w} \end{cases}$

Note 3. Comparision for a large dataset

$\tilde{y}_1 - y_1$, $\tilde{y}_2 - y_2$, $\tilde{y}_3 - y_3$, ...

$\tilde{y}_n - y_n$.

Where $\tilde{y}_i$: Experiment Output. Actual Output.

y ground Truth.

$$\sum_{i=1}^{n} \tilde{y}_i - y_i$$

$\downarrow$ To Avoid Cancellation of errors. Let's Square them.

$$\sum_{i=1}^{n} (\tilde{y}_i - y_i)^2 \quad \cdots (4)$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\tilde{y}_i - y_i) \quad \cdots (4\text{-}b)$$

Average of the Accumulative Comparision for the entire Data Set.

M.S.E (mean square error).

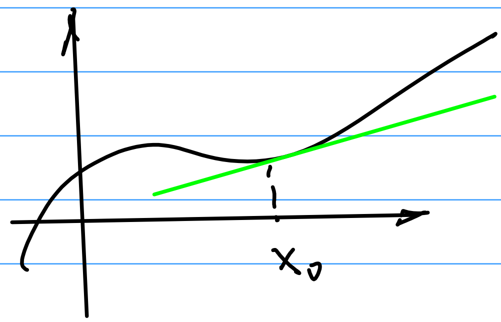
Example: Gradient Descend

Given $f(x_1, x_2, \cdots, x_n)$, such as the objective function in Eqn. (4-b).

To Evaluate its performance for training purpose, we use derivative.

Consider one dimensional Case first.

$$\frac{d f(x)}{dx} = \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x} \quad \cdots (5)$$



if Eqn(5) > 0

$$f(x+\Delta x) > f(x)$$

if Eqn(5) < 0, then

$$f(x+\Delta x) < f(x).$$

The goal is to make the objective function, e.g., loss function Reduced to its minimum through the training.

For $f(x_1, x_2, \cdots, x_n)$, we have partial derivatives.

$$\frac{\partial f(x_1, x_2, \cdots, x_n)}{\partial x_1},$$

$$\frac{\partial f(x_1, x_2, \cdots, x_n)}{\partial x_2}$$

$$\vdots$$

$$\frac{\partial f(x_1, x_2, \cdots, x_n)}{\partial x_n}$$

Now, to put each partial derivative together to get a whole view of the loss (e.g. Objective function), we define a gradient.

$$\begin{pmatrix} \frac{\partial f(x_1, x_2, \cdots, x_n)}{\partial x_1} \\ \frac{\partial f(x_1, x_2, \cdots, x_n)}{\partial x_1} \\ \vdots \\ \frac{\partial f(x_1, x_2, \cdots, x_n)}{\partial x_n} \end{pmatrix} \quad \cdots (6)$$

CMPE258
F2023

Sept. 26 (Tue).

Note 1. New Homework Posted on the CANVAS.

CMPE258
A Single Neuron Basic Building Blocks and Gradient Descent Function
Homework
HL

$$y = f(\sum_{i=1}^{N} w_i x_i = W \cdot X + b) = f(h(w_i, b)).$$

Example: On the gradient descent Technique.

Ref: 2022S-105c-#20-2021S-4gradie

gradient.

$$(x_1^{k+1}, x_2^{k+1}) = (x_1^k, x_2^k) + [-\eta(\nabla f)^t] \qquad (5)$$

Note 1. Weights, $\vec{W} = (w_1, w_2, \cdots, w_n)$ where $\vec{w}$ matches to the feature vector $\vec{X} = (x_1, x_2, \cdots, x_n)$

Superscript "k+1", index for the training at step k+1

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial w_1} \\ \frac{\partial f}{\partial w_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix} \quad \text{"} t \text{"} \text{ transpose} \rightarrow \qquad \cdots (1)$$

$$\nabla f^t = \left(\frac{\partial f}{\partial w_1}, \frac{\partial f}{\partial w_2}, \cdots, \frac{\partial f}{\partial w_n}\right) \qquad \cdots (1\text{-}b)$$

$$w_1^{k+1} = w_1^k + (-\eta)\frac{\partial f}{\partial w_1} \qquad (1\text{-}c)$$

$$w_2 = w_2^k + (-\eta)\frac{\partial f}{\partial w_2}$$

$$\vdots$$

$$w_i = w_i^k + (-\eta)\frac{\partial f}{\partial w_i}$$

$$\vdots$$

$$w_n = w_n^k + (-\eta)\frac{\partial f}{\partial w_n}$$

Now, verify the equation.

From Eqn (6).

$$f(x_1, x_2) \simeq f(a, b) + \frac{\partial f}{\partial x_1}(x_1 - a) + \frac{\partial f}{\partial x_2}(x_2 - b) \qquad (6)$$

Consider 1D Case

$$f(x) = f(x_0) + \frac{f'(x)}{1!}(x - x_0) + \frac{f''(x)}{2!}(x - x_0)^2 + \frac{d^3 f(x)}{dx^3}(x - x_0)^3 + \cdots + R_n(x). \qquad \cdots (2)$$

Fig. 1

The multidimensional Taylor Expansion can be realized Based on the Similar Principle.

CMPE258

F2023

Simplify it, so just use the constant and the linear term → Linear Term.

$y = ax + b \rightarrow y' = ax + \underbrace{b + c}_{b'}$

$$f(x_1, x_2) \simeq f(a,b) + \frac{\partial f}{\partial x_1}(x_1 - a) + \frac{\partial f}{\partial x_2}(x_2 - b) \qquad (6)$$

Start from here

Note: Denote $\frac{\partial f}{\partial x_1}$ as $f_{x_1}(a,b)$

$\frac{\partial f}{\partial x_2}$ as $f_{x_2}(a,b)$

$$f(x_1,x_2) - f(a,b) \simeq f_{x_1}(a,b) * (x_1 - a) + f_{x_2}(a,b) * (x_2 - b) \qquad (8)$$

Or,

$$f(x_1,x_2) - f(a,b) = (x_1 - a, x_2 - b)\begin{pmatrix} f_{x_1}(a,b) \\ f_{x_2}(a,b) \end{pmatrix} \qquad (9)$$

$\rightarrow \nabla f(x_1, x_2)$

hence,

$$f(x_1,x_2) - f(a,b) = (x_1 - a, x_2 - b)\nabla f \qquad (10)$$

$$\underbrace{f(x_1,x_2) - f(a,b)}_{} = (\Delta x_1, \Delta x_2)\nabla f = -(f_{x_1}^2 + f_{x_1}^2) \qquad (12)$$

The error becomes smaller each iteration as long as $\Delta x_1, \Delta x_2$ defined by Eqn (5)

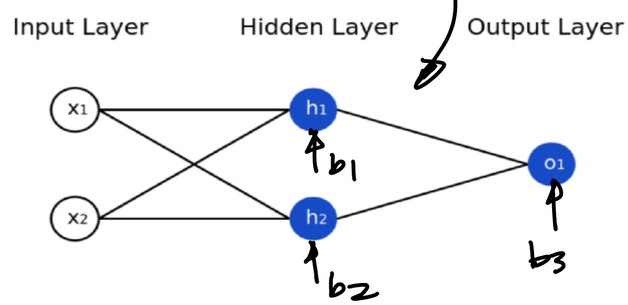
!! "<"

$$f(x_1,x_2) - f(a,b) = -(f_{x_1}^2 + f_{x_1}^2) < 0 \qquad (13)$$

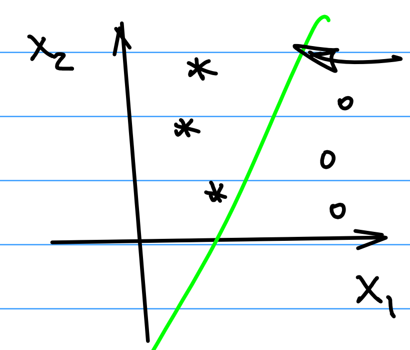
Now, Review the Python Sample Code

Ref → 2022S-103c-#nn_sample_2022.py

Sample code for this NN.




```
#-----------------------------------
data = np.array([
    [1, 2.5],      # person A
    [1, 3],        # person A
    [2.1, 3.4],    # Person A
    [2.1, 1], # person B
    [3.3, 1], # person B
    [3, 2.3], # person B
# HL 2020-9-7 Part E
# for the testing of adaptive learning
# (1) create a new program based on this o
```



Sept. 28 (Th).

Note 1. LLM (Large Language Model) Based ON Chat GPT. focus ON API Package,

CMPE258

F2023

Note: the 1st of the 3 PPT.

Deploy API with Python.

Ref: ON the class github.

2023F-104-#190c-9a-chatGPT-API-YZ-HL-2023-9-7 (copy).pdf

CTI

ChatGPT API from OpenAI

https://community.openai.com/t/is-chat-gpt-provided-for-free/86249

Step 1. Log in ChatGPT, then click on API

OpenAI

Step 2. Design your prompt to program your model

ChatGPT →
Interact with our flagship language models in a conversational interface

DALL·E →
Create realistic images and art from a description in natural language

API →
Integrate OpenAI models into your application or business

Example: Conversation with API: a conversation is formatted with a system message first, followed by alternating user and assistant message, see below:

```
import openai

openai.ChatCompletion.create(
  model="gpt-3.5-turbo",
  messages=[
      {"role": "system", "content": "You are a assistant."},
      {"role": "user", "content": "Who won the world series in 20?"},
      {"role": "assistant", "content": "The Los Angeles Dodgers."},
      {"role": "user", "content": "Where was it played?"}
    ]
)
```

GPT-3.5 Turbo

GPT-3.5 Turbo is optimized for dialogue.

Learn about GPT-3.5 Turbo ↗

GPT-3.5-Turbo Pricing:

Model	Input	Output
4K context	$0.0015 / 1K tokens	$0.002 / 1K tokens
16K context	$0.003 / 1K tokens	$0.004 / 1K tokens

https://openai.com/pricing#language-models

harry@harrys-gpu-laptop: ~/PycharmProjects/chatGPT

File Edit View Search Terminal Help

```
(base) harry@harrys-gpu-laptop:~/PycharmProjects/chatGPT$ tree -L 2
.
├── API_Key.json
├── ChatHistory.json
├── fine-tuning
│   ├── fine-tuning.jsonl
│   ├── finetuning.py
│   └── sample
├── fine-tuning-10QA-2023-9-14
│   └── GPT_Fine_Tuning.zip
└── GPTWithHistorySaved.py

3 directories, 6 files
(base) harry@harrys-gpu-laptop:~/PycharmProjects/chatGPT$
```

Note 1. Be sure to save it on a separate file from the ChatGPT (OpenAI) page.

```
harry@harrys-gpu-laptop: ~/PycharmProjects/chatGPT
File  Edit  View  Search  Terminal  Help
├── fine-tuning
│   ├── fine-tuning.jsonl
│   ├── finetuning.py
│   └── sample
├── fine-tuning-10QA-2023-9-14
│   └── GPT_Fine_Tuning.zip
└── GPTWithHistorySaved.py

3 directories, 6 files
(base) harry@harrys-gpu-laptop:~/PycharmProjects/chatGPT$ tree -L 3
.
├── API_Key.json
├── ChatHistory.json
├── fine-tuning
│   ├── fine-tuning.jsonl
│   ├── finetuning.py
│   └── sample
│       └── train_data.jsonl
├── fine-tuning-10QA-2023-9-14
│   └── GPT_Fine_Tuning.zip
└── GPTWithHistorySaved.py

3 directories, 7 files
(base) harry@harrys-gpu-laptop:~/PycharmProjects/chatGPT$
```

Note2: ".jsonl" file that ~~user~~ Created Developer to provide training data for fine-tuning Purpose

Note: Create ".jsonl" file to provide Training Data for fine tuning.

```
harry@harrys-gpu-laptop: ~/PycharmProjects/chatGPT/fine-tuning
File  Edit  View  Search  Terminal  Help
{"messages": [    {
      "role": "system",
      "content": "CTI One Test:  W100 is a smart walker to listen to your instru
ction and drive up to you when you call it"
    },
    {
      "role": "user",
      "content": "Tell me how can I use W100 walker."
    },
    {
      "role": "assistant",
      "content": "Make sure you have powered-up W100 and initialized it already.
 Then you can use your smartphone to call for its attention."
    }
  ]
}
~
~
~
~
```

Homework, Due 1 week, Oct. 5th.

1° Install openAI to use ChatGPT API 3.5;

2° Create and Run "Hello, the world" Python Code;

3° Screen Captures. for 1° and 2° with Personal ID.

4° Submission: ON CANVAS. as zip file. use the following Naming Convention:
firstName_LastName_4SID_ChatGPT1.zip.

Consider Back Propagation, or, "Back Prop" Technique.

Ref: on the github

2022S-106a-#2019S-38-lec13-BackProp-activation.pdf

CMPE258
F2023

Note 1. Generalize the Architecture from a Single Neuron.

Back Propagation (1)

1. input and output neurons

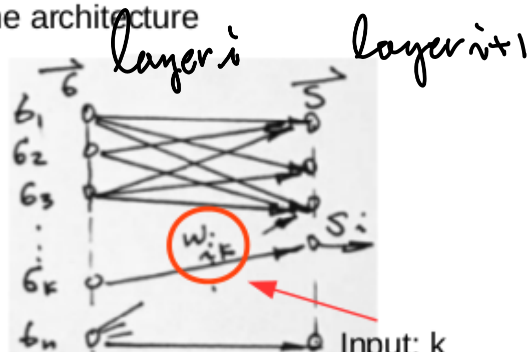
Input

$$\vec{\sigma} = \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_n \end{bmatrix}_{n\times 1}$$

Output

$$\vec{S} = \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_{n_0} \end{bmatrix}_{n_0\times 1}$$

The architecture



2. weights w_ik

i for output k for input

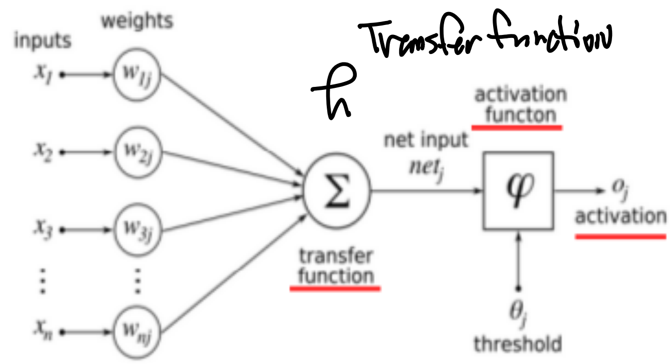
Note 2: for the Weights Notation We adopted 2 subscripts

$$\vec{W} = (W_1, W_2, \cdots, W_n)$$

$$(\ldots, W_{ik}, \ldots)$$

Other popular notation

$$Y = \sum (weight * input) + bias$$



3. Activation function

$$\boxed{S_i = f\left(\sum_k \omega_{ik} \sigma_k\right)} \quad \ldots (1)$$

Note the activation function f(.) and the bias (offset can be written in the unified summation term)

Harry Li, Ph.D.

Note 3. Activation function.

$$S_i = f\left(\sum_k W_{ik} \sigma_k\right)$$

"i" for the output layer.

Input, feature vector, Excitation.

4. Transfer function h_i

let

$$\boxed{h_i = \sum_k \omega_{ik} \sigma_k} \quad \ldots (2)$$

Index i for the output neuron

Neuron output function S_i

$$\boxed{S_i = f(h_i) = f[h_i(\omega_{ik})]} \quad \ldots (2\text{-}1)$$

Note: The New Notation By Substitute Eqn (2) into Eqn (1).

5. Error at each neuron output (the difference between the true output Zeta (desired true output) and the current output S_i) at the experiment Mu

$$\boxed{\zeta_i^{\mu} - S_i^{\mu}} \quad \text{… (2-2)}$$

Harry Li, Ph.D.

6. total error for all output neuron I and all experiments Mu

$$\boxed{D = \frac{1}{2} \sum_{\mu} \sum_{i} \left(\zeta_i^{\mu} - S_i^{\mu} \right)^2} \quad \text{… (3)}$$

MSE ? (mean Square Error)

N : No. of Neurons

M : No. of Experiments | EPOCHS .
μ

$\frac{1}{NM}$ Scaling factor

Note: Rewrite the output S_i by its functional form.

7. minimize the error wrt to w_ik

Activation function (Transfer function)

$$\frac{\partial D}{\partial w_{ik}} = \frac{\partial}{\partial w_{ik}} \cdot \frac{1}{2} \sum_{\mu} \sum_{i} \left[\zeta_i^{\mu} - f(h_i^{\mu}) \right]^2$$

$$= \sum_{\mu} \left[\zeta_i^{\mu} - f(h_i^{\mu}) \right] f'(h_i^{\mu}) \frac{\partial h_i}{\partial w_{ik}}$$

Derivative Calculation is Based on "Chain Rule"

Note: Eqn (5) on P. 24.

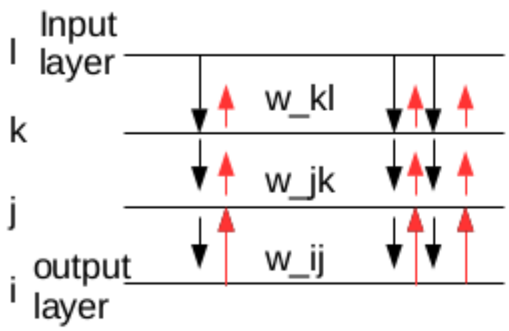
8. Training the NN by updating the weights

$$\boxed{w_{ik}(t+1) = w_{ik}(t) + \delta w_{ij}(t)} \quad \text{…(4)}$$

where

$$\boxed{\delta w_{ij}(t) = -\epsilon \frac{\partial D}{\partial w_{ij}}} \quad \text{…(5)}$$

10. feed forward NN (4 layers)



Black: for Experiments.
Red: Training Update

Black for the input; red for the back prop training direction

Math. Description.

Back Propagation (4)

11. chain rule, updating the weights (training)

Note: the training described here all related to the derivative to the activation function, f'(.). So selection of the activation function is important and will be discussed in details next.

For layer i

$$\boxed{\frac{\partial D}{\partial w_{ij}} = \sum_{\mu} [\zeta_i^{\mu} - f(h_i^{\mu})] f'(h_i^{\mu}) \frac{\partial h_i^{\mu}}{\partial w_{ij}}}$$

For layer j

$$\boxed{\frac{\partial D}{\partial w_{jk}} = \sum_{\mu} \sum_{i} [\zeta_i^{\mu} - f(h_i^{\mu})] f'(h_i^{\mu}) \frac{\partial h_i^{\mu}}{\partial S_j} \cdot \frac{\partial S_j}{\partial w_{jk}}}$$

For layer k

$$\boxed{\frac{\partial D}{\partial w_{kl}} = \sum_{\mu} \sum_{i} \sum_{j} [\zeta_i^{\mu} - f(h_i^{\mu})] f'(h_i^{\mu}) \frac{\partial h_i^{\mu}}{\partial S_j} \cdot \frac{\partial S_j}{\partial S_k} \cdot \frac{\partial S_k}{\partial w_{kl}}}$$

Project 1 Due 2 weeks
from Next Monday.

CmpE258

F2023

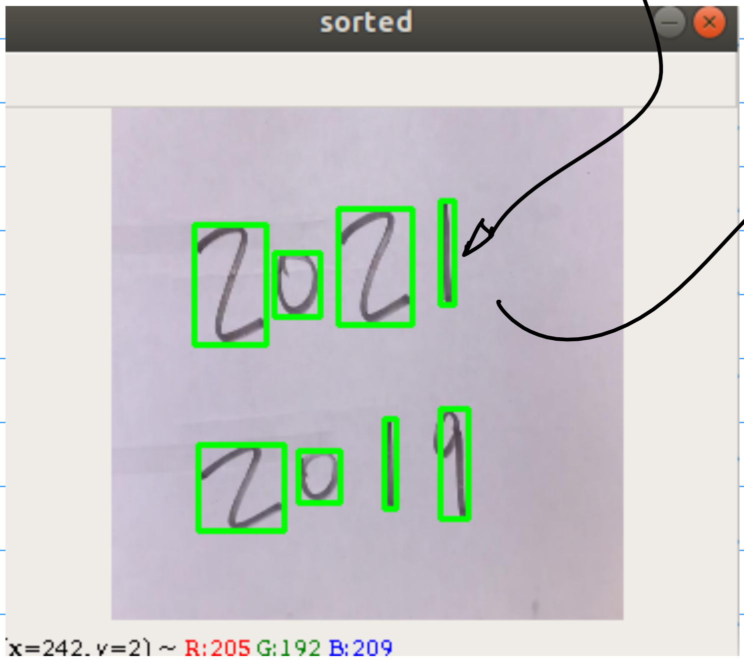
Oct. 3rd (Tue).

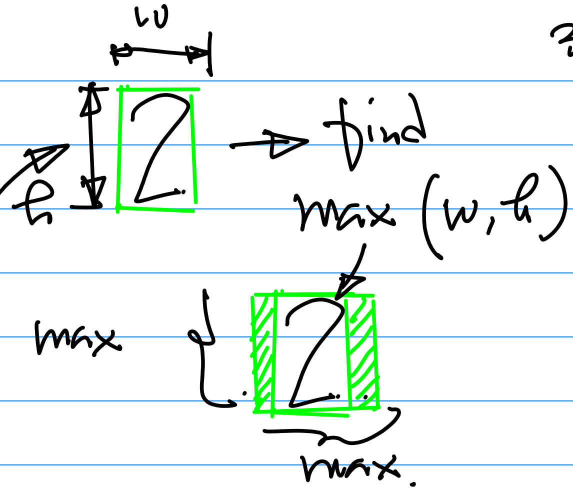
Note 1: Move the Homework to the Sunday.

Note 2: Project 1 Assignment for Handwritten SID Recognition System.

1. Objective To use either Live CAM Input or Video file input to Recognize Hand Written SID. Last 4 Digits
   Please keep Handwritten Digits as the inputs, No printed digits.

2. Preprocessing of the input image, to get Bounding Boxes





Use OpenCV Bitwise Operations

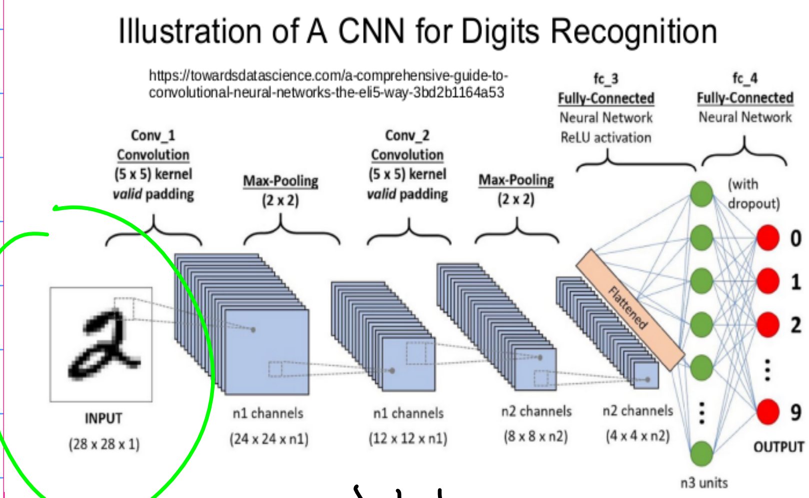
3. Input Image Resolution (1pt).

Bounding Box $B_i(x, y; w, h)$

width height

Make sure in the preprocessing you will produce Square image to preserve the Aspect Ratio of the digits.

$$\alpha = h/w \quad \cdots (1)$$



28x28x1 is to be Changed to Higher Resolution. such as 56x56x1 by selecting Proper Convolution Layers / Maxpooling Layers.

please preserve the Rest of the MNIST Architecture.

4. Comparison of your implementation with the Conventional MNIST.

5. Provide Readme file, (good quality document).

Due Oct. 21st, Sat (11:59 pm).

Colab, and On Line Tools are encouraged, However the submission of the projects, Homework Require Python code.

Example: Preprocessing.

Ref: 2023S-101- ~ -03-21.

Binarized Image, Contour Analysis. Sample Code

~~Ref: 2023S-101-Part2-~ 04-18.~~

Step1. Capture Image from a Video, Convert it to a Gray scale image I(x,y).

Step2. Canny Edge Detection to produce Edge Map. $I_{Canny}(x,y)$

Step3. Contour Analysis. 19~22 primitive Contour features.

Draw the Contours on the input preprocessed image.

a) w,h of a given object
b) Bounding Box of an object
c) Shape Descriptor
d) moments
e) others.

Step 4. Filtering Operation. Based on the Contour Map. $I_{count}(x, y)$

Filter Out the Artifacts. Look into the features of $I_{count}(x, y)$

Step 5. Prepare the image ROI. And Capture the Image(s) from the ROI, then form Square image, to feed one image (digit) at time to the CNN Engine.

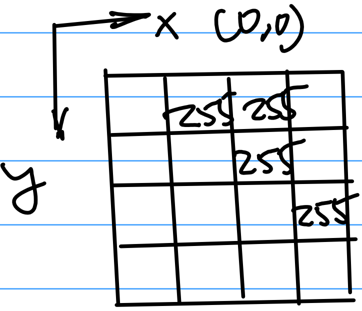
Example: Continuation of Examples on Binarized image pp. 17 & 18

Motivation for Binarization: To Extract Objects.

$$\bar{x} = \frac{\iint_\Omega x B(x,y)\,dx\,dy}{\iint_\Omega B(x,y)\,dx\,dy} \quad \ldots (1)$$

AREA

$$\sum_{y=0}^{N-1}\sum_{x=0}^{M-1} x B(x,y) \Big/ \sum_{y=0}^{N-1}\sum_{x=0}^{M-1} B(x,y)$$



Note: Redefine the top Left Corner Coordinate as (1,1) for Hand Calculation. !

$$\sum_{y=\cancel{0}\,1}^{\cancel{N-1}} \sum_{x=\cancel{0}\,1}^{\cancel{M-1}} B(x,y) =$$

$$\sum_{y=\cancel{0}\,1}^{\cancel{N-1}} \left[B(\cancel{0}\,1,y) + B(\cancel{1}\,2,y) + B(\cancel{2}\,3,y) + B(4,y)\right]$$

$$= \cancel{B(1,1)} + B(2,1) + B(3,1) + \cancel{B(4,1)} +$$
$$\cancel{B(1,2)} + \cancel{B(2,2)} + B(3,2) + \cancel{B(4,2)} +$$
$$\cancel{B(1,3)} + \cancel{B(2,3)} + \cancel{B(3,3)} + B(4,3) +$$
$$\cancel{B(1,4)} + \cancel{B(2,4)} + \cancel{B(3,4)} + \cancel{B(4,4)}$$

$$= 255 * 4$$

Similarly,

Now, $\sum_{y=1}^{4}\sum_{x=1}^{4} x B(x,y)$

$$= \sum_{y=1}^{4} \left[0 + 2*B(2,1) + 3*B(3,1) + 3*B(3,2) + 4*B(4,3)\right]$$

= 255 * (2*1 + 3*2 + 4*1)

= 255 * 12

Oct. 5 (Th)
Note 1: Check the CANVAS for the project 1, due Oct. 21st Sat;

Example: X-bar calculation
Continuation from the previous example

hence,

X_ bar = 12/4 = 3;

Similarly, for Y_bar , see the equation from the PPT of the class ref on github.

Denominator from the equation for Y_bar is the same as for X_bar (Area)

$$\ldots (4\text{-}a)$$

and

$$\bar{y} = \frac{\sum_{y=0}^{N-1} \sum_{x=0}^{M-1} y B(x,y)}{A}$$

from pp. 18 of this note.

Then the top part of this equation is evaluated:

255 * (1 * B(2,1) + 1 *B(3,1)
2 *B(3,2) + 3 * B(4,3))
= 255 * 7

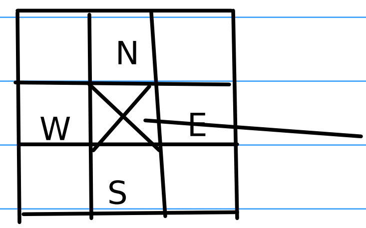
Therefore

Y_bar = 7/4

Note: keep the float for the coding implementation if needed.

Consider edge detection technique. ref. lecture note on the github

Consider edge detection, e.g., detection of sudden changes of pixle values



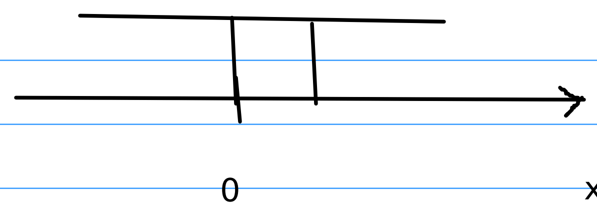
Define 4 connected neighbors N, S, E, W then the sudden change of the intensity value is defined as the difference between the PoI and any of one of its neighbors.

Similarly, we can also define 8 connected neighbors, by adding NE, NW, SE, and SW

To pick up the sudden change, we use derivative, see pp. 22 of the lecture note

From eqn (2a) for x and (2b) for y, we have forward difference to compute derivatives

Now, let us derive convolutional kernel for this computation



Form a small grid with index 0 to map the equation (2a)

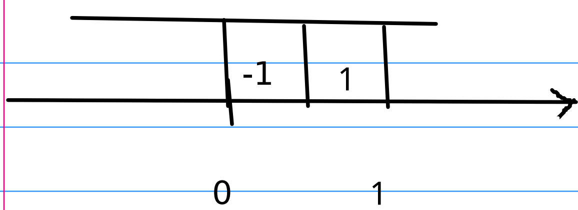
let delta x = 1 for digital image and we have

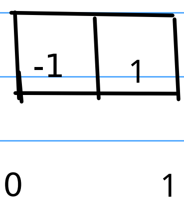
f(x+1) - f(x) for the right hand side of the equation,

we can rewrite the above form into
1 * f(x+1, y) + (-1) * f(x,y)

Map the cofficients to the template, e.g., kernel

so we have



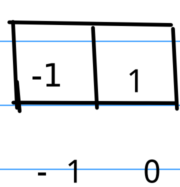


Note this is a kernel but it is not

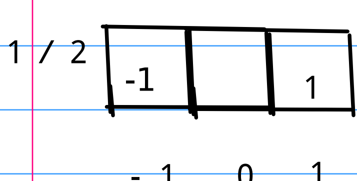
skewed, not balanced.
So with the modification of derivative from the forward difference to backward difference,

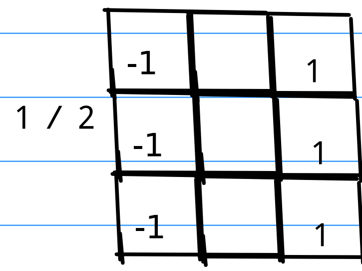
f(x,y) - f(x-1,y)

which give the kernel



Add this kernel (BW) to the kernel (FW) and divide the result by 2, we have





Similarly, we can do the same for y direction.

Note for LoG(x,y)

Oct. 10th (Tue).

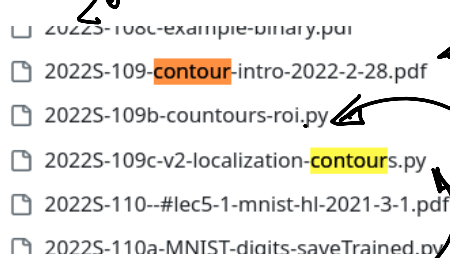
Example: Consider Contour Analysis

Ref.

1° 2023S-101-note ~ 03-24.pdf

Pp. 23.

2°



PPT.

Python Sample for Contour Analysis

Python Sample Code which integrates all the preprocessing task Except the Last one which is the task to Build a squared Bounding Boxes

Note:

1° Notation for Contours

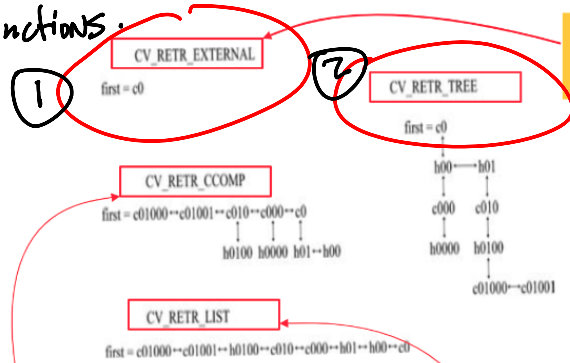
C — Contour

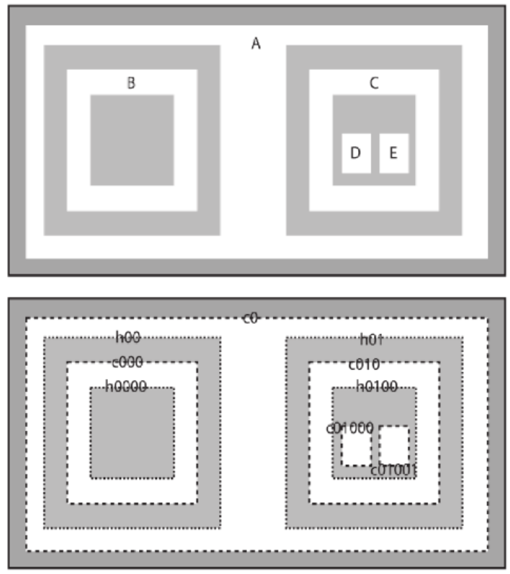
h — Hole

Note 1. Pay Attention to 2 of the following 4 OpenCV Contour functions.

4 Contours Models

Retrieves only the external outer contours. It sets hierarchy[i][2]=hierarchy[i][3]=-1 for all the contours.





CV_RETR_CCOMP
Retrieves all the contours into two-level hierarchy, top-level for external boundaries and the 2nd level for the holes.

retrieves all contours without any hierarchical relationships.

Harry Li, Ph.D.

http://opencvexamples.blogspot.com/2013/09/find-contour.html

Note 2: CV_RETR_TREE is very widely adopted. So we will emphase how to use it. To be able to Carry out hand Calculation as well.

Note 3: CV_RETR_EXTERNAL can be utilized for the SID Hand Written digit Recognition. Which provides Simpler, But Adequate Contours for Bounding Boxes finding.

Example:

CmpE 258

F2023

See the Example ON PP. 24,
Lecture Notes 2022S-109 ~

Find external Contours.

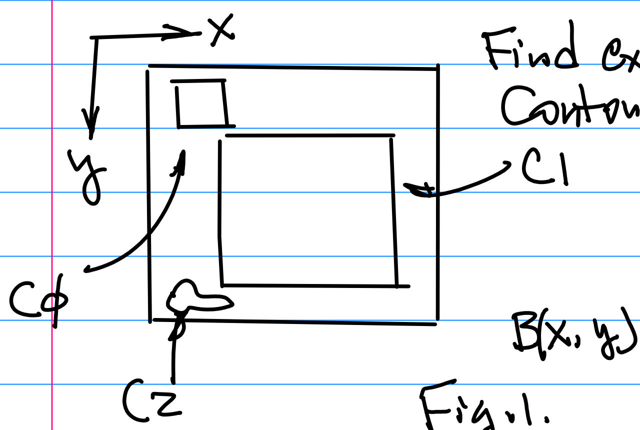
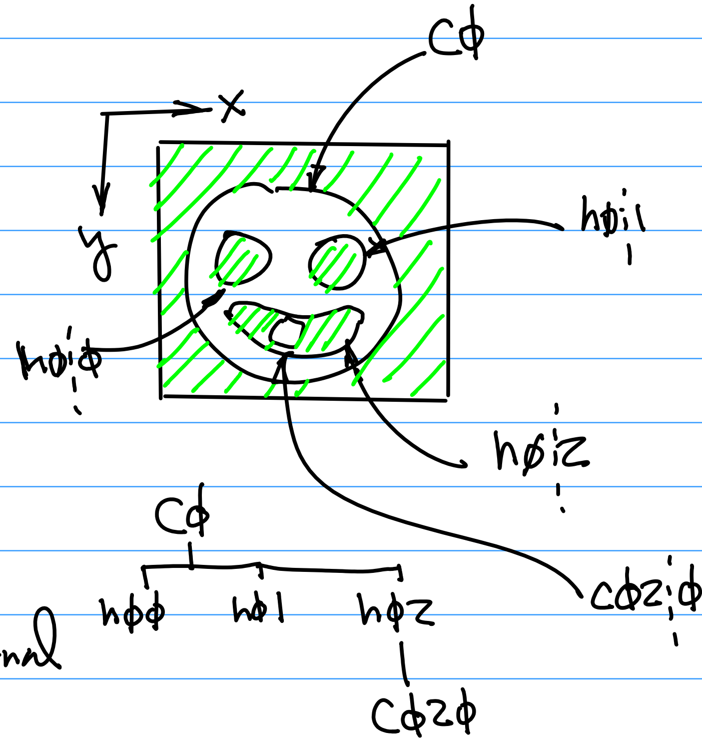


Fig. 1.

Now, find CV_RETR_CCOMP

Consider the following contour.



Find external Contours.

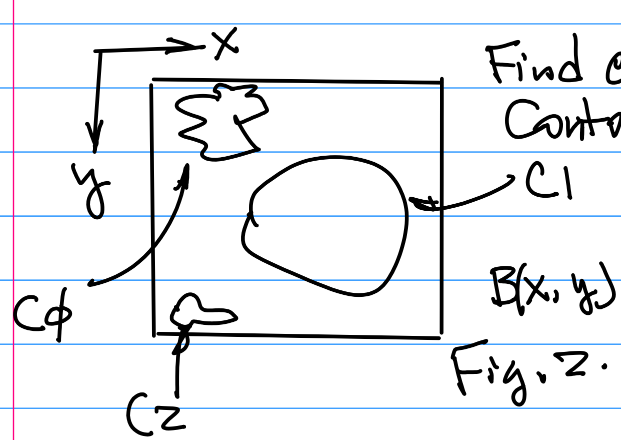


Fig. 2.

Note: Cv2.cvtColor( );
Cv2.Canny( );

Sample code.

opencv / deep-learning-2022s / 2022S-109c-v2-localization-contours.py

hualili Add files via upload

Note: This Python Code Shows Contour Finding.

Code Blame 53 lines (39 loc) · 1.5 KB Code 55% faster with GitHub Copilot

```python
# This is a sample Python code for preprocessing for contour analysis
#

def contour_example(name):
    # Use a breakpoint in the code line below to debug your script.
    print(f'Harry, {name}')  # Press Ctrl+F8 to toggle the breakpoint.

def get_distance(x,y):
    return math.sqrt(x*x+y*y)

# Press the green button in the gutter to run the script.
if __name__ == '__main__':

    contour_example('Contour Example')
```

Note 1. Be Careful for cv2.resize( ) : It forces the image to Resize to 256x256, which distore the Original Image, Not Recommendated, should use w/2, h/2 as a better practice to preserve the Aspect Ratio

```
import cv2
import math

#img = cv2.imread('test.jpg')
image = input('Enter image file name:')

img = cv2.imread(image, cv2.IMREAD_COLOR)
img = cv2.resize(img, (256, 256))
cv2.imshow('original image', img)

imgray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
cv2.imshow('gray scale image', imgray)

thresh = cv2.Canny(imgray, 100, 200)
cv2.imshow('Canny', thresh)

contours, hierarchy = cv2.findContours(thresh, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
# Draw contour
thresh1 = cv2.cvtColor(thresh, cv2.COLOR_GRAY2BGR)
cv2.drawContours(thresh1, contours, -1, (0, 255, 0), 5) # all contours
cv2.imshow('All contours', thresh1)
#Bounding box
```

Note 2. OpenCV Color Image Format "BGR" Not RGB.

memorize this function!

Note 3. cv.Canny( ) edge Detection, with Upper and Lower Thresholds

Note 4. contours, hierarchy = cv2.findContours( )

Keywords: cv2.RETR_EXTERNAL
cv2.CHAIN_APPROX_SIMPLE

Note 5: cv2.drawContours( ) memorize this function!

Note 6. Filtering

```
    for i in range(len(contours)):
        [x, y, w, h] = cv2.boundingRect(contours[i])
        # Filter out smaller noise regions
        if (w > 5) & (h > 25):
            print("i:",i)
            cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)
    cv2.imshow('sorted', img)

print(img.shape)

cropped2 = img[10:20, 10:30]
cv2.imshow('cropped2:',cropped2)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Note 6. Bounding Boxes

Note 7. Crop Image To Prepare for Bitwise Operation in the Square Image Construction.

Oct 12 (Th).

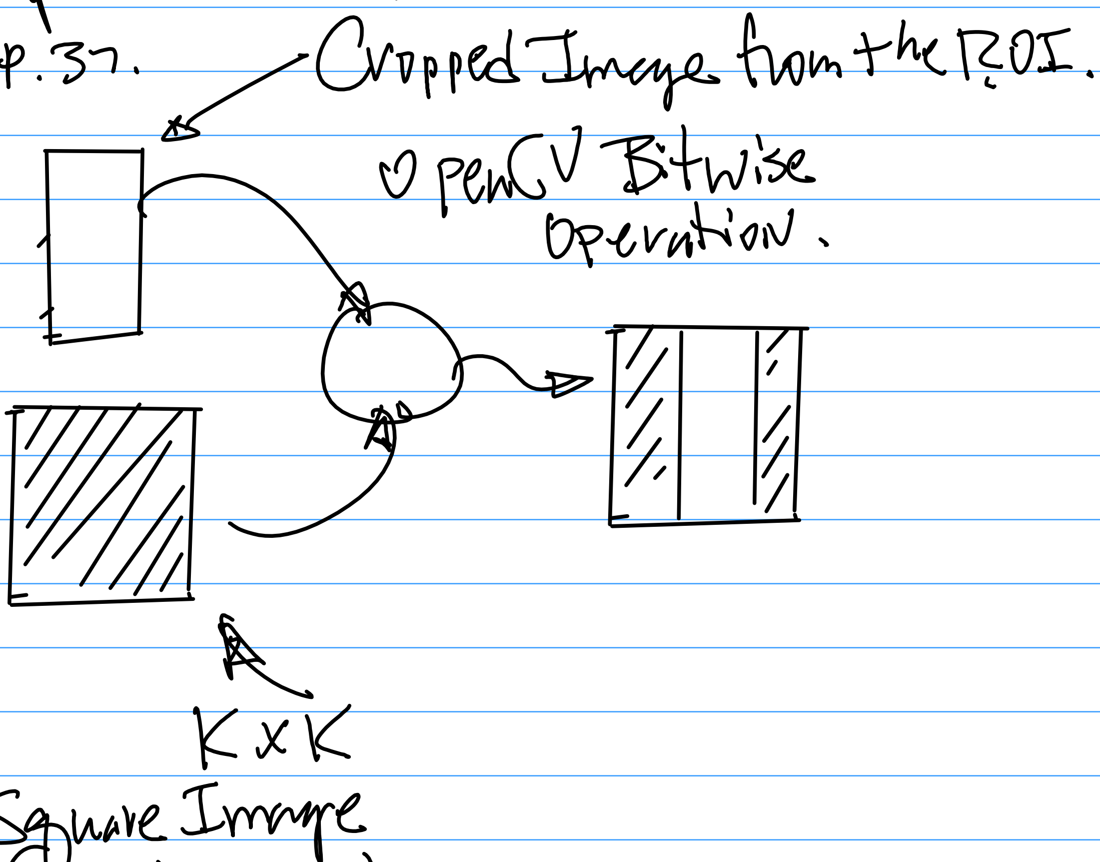
Today's Topics:

1° Sample Code
for ROI localization.

2° Yolo

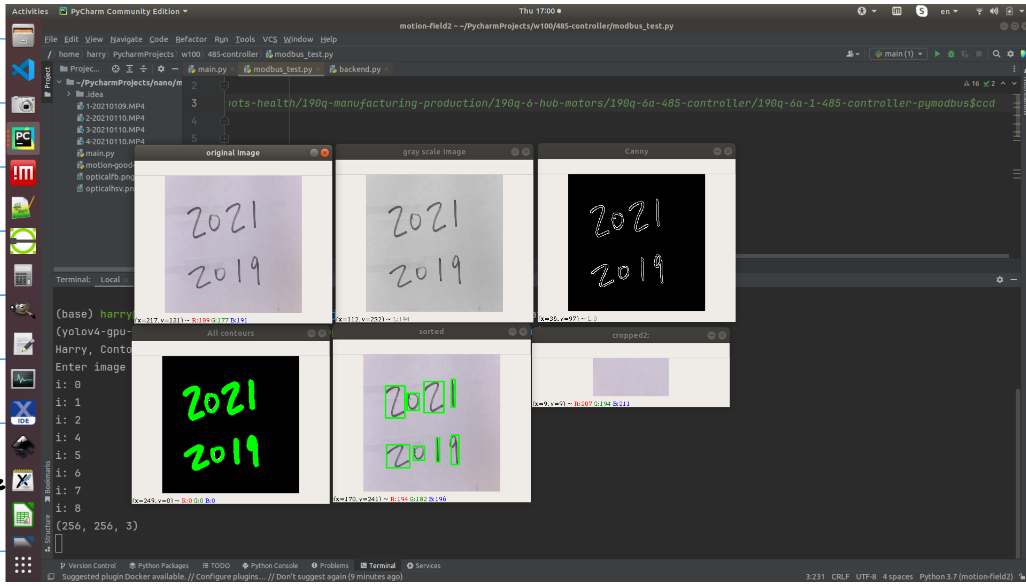
Example: Same Code, Line 50.

pp. 37.



$K = \max(w, h)$

Example:



Note: The 2nd Sample code Processed Results Displayed with Caption.

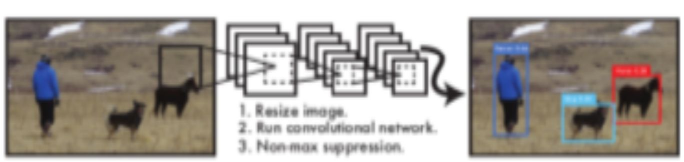
Now, Consider YoLo Technique.

Note1: YOLO (You Only Look Once)

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

Research paper: https://arxiv.org/pdf/1506.02640v5.pdf

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



Base YOLO model runs at 45 FPS. A smaller version, Fast YOLO, runs astounding 155 FPS second, outperforms DPM (deformable parts models) and R-CNN.

Figure 1: The YOLO Detection System. (1) resizes image to 448 × 448, (2) runs a convolutional network, and (3) thresholds the result by the model's confidence.

YOLO github Code and readme https://github.com/hualli/opencv/blob/master/deep-learning-2022s/2022F-106-YOLO4-README-Tiny-Yolo4-GPU-YY-2022-9-28.pdf

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

GitHub, Inc. (US) `git clone https://github.com/pjreddie/darknet`

Harry Li, Ph.D.

Note2: Motivation. Among other Objectives 2 of them plays Significant Roles here.

1. To Automate the R.O.I Bounding Boxes Creation.

2. To Expand the Object Recognition Capability Beyond what we have so far.

The Approach:

1° Research Paper.

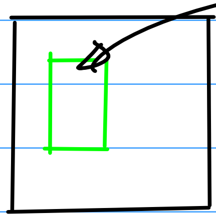
2° Github Code With Readme Provided from the Class. ON the Class github.

2022S-112-yolo-paper.pdf

Oct. 17 (Tue).

Q&A. Bounding Boxes Filtering.

Objective: To Detect Those Boxes which do not Contains a Digit



Any Digits.
↓
Single color
↓
Hence, Any Pattern inside the Box has more 2 Colors, than we want to Remove it.

"Floodfill" for coloring operation

① "Seed" point is Needed.

② feed the seed to the floodfill.

Note 1. Homework ON Yolo Installation.

Note 2. Oct. 24 (Tue). Quiz, 15 + 15 minutes.

Prep & Submission of your work.

① Bring Your Laptop. Run the Code including OpenCV, NN Introduction, MNIST.

② Screen Capture of the Code Execution. With personal identifier.

③ Please Bring 1 Blank Paper to Write your Answer.

④ Then take a photo. → pdf, then integrate All pdf files into One pdf. With following Naming Convention.

FirstName_LastName_SID_Quiz2_258.zip.

⑤ Submission to the Canvas.

YOLO (You Only Look Once).

Ref: Yolo Paper, see the github.

Overview. of the Paper.

Example: Notations & Formulation.

CMPE258

F2023

40

Ref on the github.

Paper

2022F-101-cmpe258-note-2022-11-1 (4).pdf

2022F-108a-Yolo-architecture-loss-function-2022-10-10 (1).pdf

2022S-112-yolo-paper.pdf

2023S-101-Note-part2-cmpe258-2023-04-18 (1).pdf

PPT.

overview.

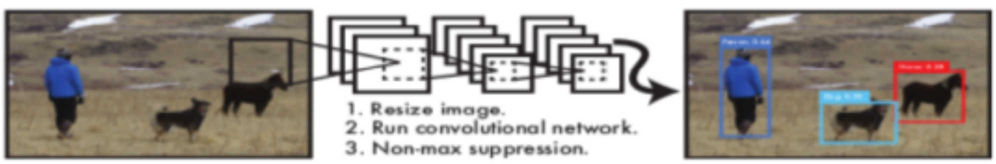
You Only Look Once: Unified, Real-Time Object Detection

Research paper: https://arxiv.org/pdf/1506.02640v5.pdf

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

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/



Base YOLO model runs at 45 FPS. A smaller version, Fast YOLO, runs astounding 155 FPS second, outperforms DPM (deformable parts models) and R-CNN.

Figure 1: The YOLO Detection System. (1) resizes image to 448 × 448, (2) runs a convolutional network, and (3) thresholds the result by the model's confidence.

YOLO github Code and readme

https://github.com/hualili/opencv/blob/master/deep-learning-2022s/2022F-106-YOLO4-README-Tiny-Yolo4-GPU-YY-2022-9-28.pdf

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

GitHub, Inc. (US)

```
git clone https://github.com/pjreddie/darknet
```

Harry Li, Ph.D.

Notations for the formulation.
See Ref Below.

CmpE258

F2023

CmpE258

Oct.13,22

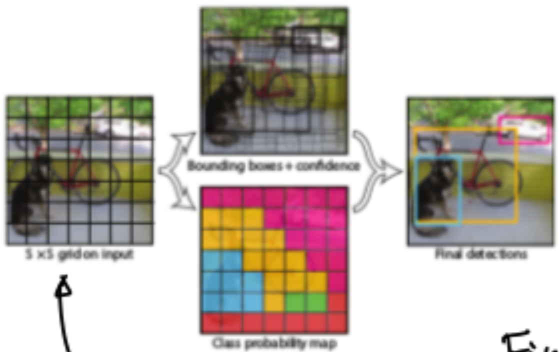


Fig.2C

Note:

1. Image $I(x,y)$ is divided into $S \times S$ Grids.

Denote it as $G_{p,q}(x,y)$, ... (1)

where $p, q = 0, 1, 2, \cdots$ indicate the location of the grid. (see Ref. Lecture Note, pp40)

So, we have $G_{0,0}(x,y)$, $G_{0,1}(x,y)$, etc. from Left to the Right.

$G_{p,q}(x,y)$ where $p, q = 0, 1, \cdots$, ... (1)

2. Bounding Box:

$B_{i,j}(x,y)$. $i, j = 0, 1, 2, \cdots$ ... (2)

Note: A $B_{ij}(x,y)$ may be inside $G_{p,q}(x,y)$, or Overlaps with $G_{p,q}(x,y)$, OR

Could be Across the Boundary of a $G_{p,q}(x,y)$, occupy multiple $G_{p,q}(x,y)$

Note: Parameters to Characterize $B_{ij}(x,y)$.

Location (x,y). Top Left Corner.

Shape, w, h width, height ;

$B_{ij}(x, y, w, h)$

Object Classification for the Bounding Box.

See Example Next Page.

CmpE258

F2023

Compare to the Objective function, loss function, $B_{ij}(x, y, w, h)$

$$\lambda_1 \quad \lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$\lambda_2 \quad + \lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$\lambda_1 + \lambda_2 = 1.0$

Assume $\lambda_1 = \lambda_2 \rightarrow$ Equal weight, Equally Important.

Note: Suppose make Location of $B_{ij}(x, y)$ twice as important as the shape. So, we have

$\lambda_2 = 2\lambda_1$

Hence, $2\lambda_1 + \lambda_1 = 1.$

$\lambda_1 = 1/3.$

$\therefore \lambda_2 = 2/3.$

Oct 11a (Th).

Note: A week from today, Presentation (Abstract) for the Semester Long Project Proposal. Please Read the team Project Requirements on CANVAS, Prepare the Presentation. to Address.

1° Title of the object;

2° Technical Challenge(s).

3° Proposed Work / Approach to address the challenges.

4° Existing github code. and primary Research Paper.

5° Expected Results & Deliverables

Example: Continuation of the formulation for YOLO Technique. Ref: github

2023S-101-Note-part2-cmpe258-2023-04-18 (1).pdf

$B_{ij}(x, y, w, h, f)$

Confidence Index to Describe how like the object Detected inside the Bounding Box Belongs to Object Class i.

CMPE258
F2023

From the Lecture Note, see Ref.

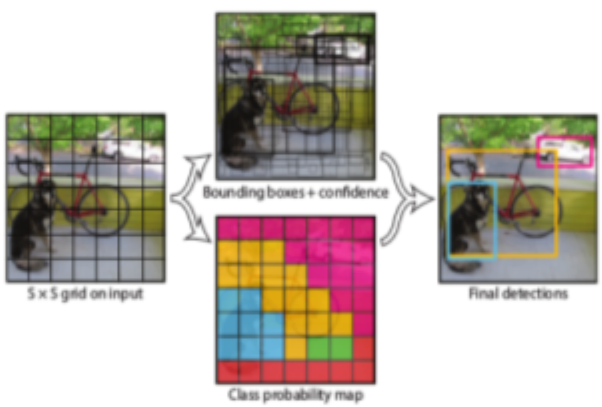
3. Five Parameters to define each Bounding Box.

(x, y) : Location of the top Left Corner of $B_{ij}(x, y)$

w, h : Width and Height of $B_{ij}(x, y)$

f : Confidency Level, Probability distribution to Describe the likelihood of the B.B (B^2) belongs to a certain Class of objects.

(x, y, w, h, f) ... (3)



Spri

Note 1. a. $G_{pq}(x, y)$ Grid.

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.

b. $B_{ij}(x, y)$

c. (x, y, w, h, f)

Class probability.

Probability

Confidency.

April (Th).

Example: Discussion on Notation/Formulation

Ref. 2022F-101-cmpe258-note-2022-11-1.pdf pp38

CMPE258
F2023

44/

From the Lecture Note. Ref. on github.

36

Note 1. a. $G_{pq}(x,y)$ Grid.

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.

Class probability.

b. $B_{ij}(x,y)$

c. (x, y, w, h, f)

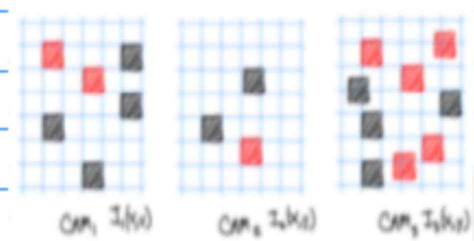
Probability

Confidency.

April (Th).

Example: Discussion on Notation/Formulation.

Ref. 2022F-101-cmpe258-note-2022-11-1.pdf pp38



Camera 1: $I_1(x,y)$ Camera 2 $I_2(x,y)$ Cam 3. $I_3(x,y)$

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

R: Red Squares, Persons.
B: (Black) Vehicles.

for "Union" " $\cup$ "

Intersection. " $\cap$ "

Consider the probability of the event "R" (meaning Person(s) being Capture on any one of these images).

$$Prob(R) = Prob(RI_1) + Prob(RI_2) + Prob(RI_3) \quad \cdots (2)$$

$I_1 \cap I_2 \cap I_3 = \phi$ (Empty set)

Consider Each Individual Camera:

$$Prob(RI_1) = Prob(R/I_1)\, Prob(I_1) \quad \cdots (3a)$$

Similarly,

$$Prob(RI_2) = Prob(R/I_2)\, Prob(I_2) \quad \cdots (3b)$$

$$Prob(RI_3) = Prob(R/I_3)\, Prob(I_3) \quad \cdots (3c)$$

Rewrite Eqn (2):

$$Prob(R) = Prob(R/I_1)\, Prob(I_1) + Prob(R/I_2)\, Prob(I_2) + Prob(R/I_3)\, Prob(I_3) = \sum_{i=1}^{3} Prob(R/I_i)\, Prob(I_i) \quad \cdots (4)$$

Ref: 2022S-112-yolo-paper.pdf

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

Example: pp36

Note 1. Conditional Probability. Prob(C_i)

Prob(C_i | Obj_j) Prob(Obj_j)

1. IOU (Intersection of Union)

Index for the purpose of Comparing 2 Bounding Boxes at a time

5. IOU (Intersection of Union).

Example, Illustration of IOU

Fig.3

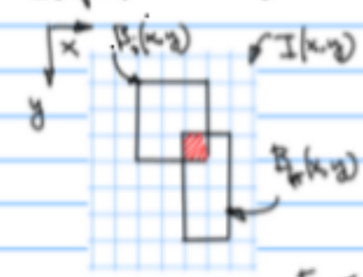


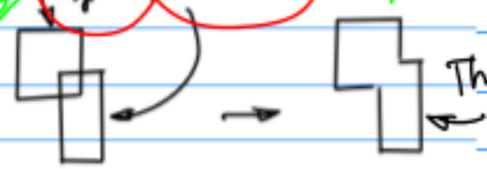
Fig.5

pp.37

Note: "GT" Ground Truth.

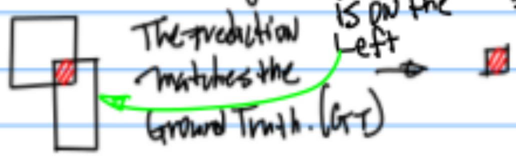
The Union of $B_{ij}(x,y)$ $B_{GT}(x,y)$

Note: $B_{ij}(x,y)$ for the i,j B.B.



The Union

The Intersection of $B_{ij}(x,y)$, $B_{GT}(x,y)$ Fig.1



The prediction matches the Ground Truth. (GT) is on the Left

$$\text{IOU} = \frac{\text{Intersection of } B_{ij}(x,y) \text{ and } B_{GT}(x,y)}{\text{The Union of } B_{ij}(x,y) \text{ and } B_{GT}(x,y)} \quad \cdots (5)$$

Sol. First, find the Number of pixels of the $B_{ij}(x,y) \cap B_{GT}(x,y)$;

$B_{ij}(x,y) \cap B_{GT}(x,y) = 1$

then, Find the Union

$B_{ij}(x,y) \cup B_{GT}(x,y) =$

$N[B_{ij}(x,y)] + N[B_{GT}(x,y)]$

$- N[B_{ij}(x,y) \cap B_{GT}(x,y)]$

$= (3\times3) + (2\times4) - 1 = 9+8-1$

$= 17-1 = 16$

$\therefore$ IOU $= 1/16$

pp37

C = 

(IOU) Confidence, denoted as C

~~…(1)~~

Now, from Eq (1) of the Ref. (Research Paper)

Note

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

Hand Calculation of IOU.

IOU truth ← "G.T." B.B.

pred ← Predicted B.B.

CMPE258

F2023

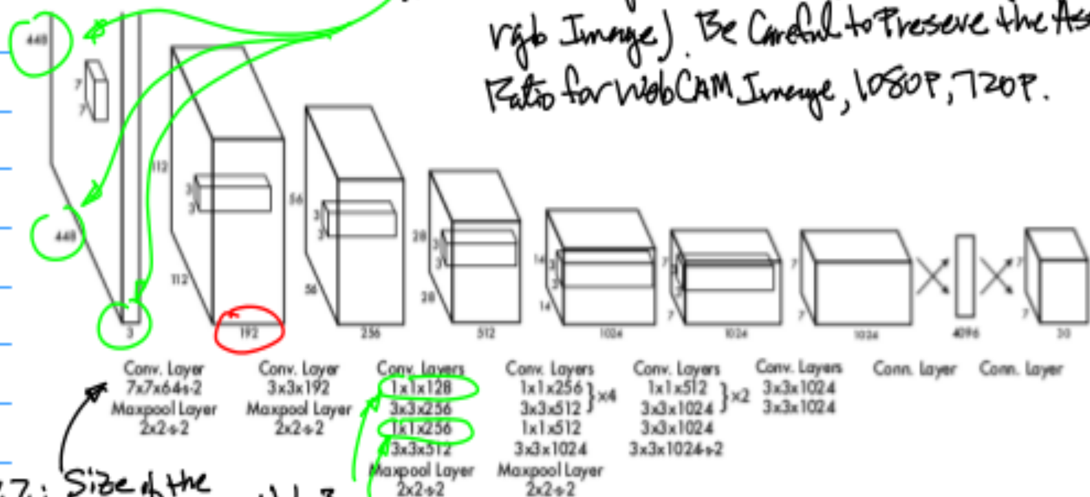
46/

YOLO Spring 2023

38

Example: Architecture:

Note 1. Input Image Resolution: 448x448x3 (Channels: rgb Image). Be Careful to Preserve the Aspect Ratio for WebCAM Image, 1080P, 720P.



Note 2: Size of the Kernel: 7x7, 64 of them.

Note 3. 1x1 Convolution is utilized here

Example: Continuation ON Math Formulation. 1x1 Convolution.

Note 1. Background ON KxK Convolution.

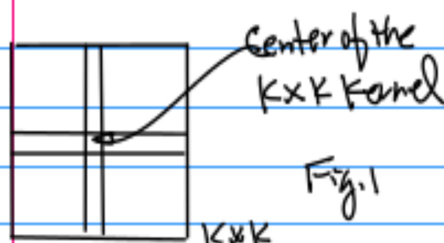


Fig. 1

For KxK 2D Convolution, the output of the convolution. "Spatial Information", Neighbouring pixels under the KxK Kernel are Counted for (for feature extraction @ the Center of the Kernel)

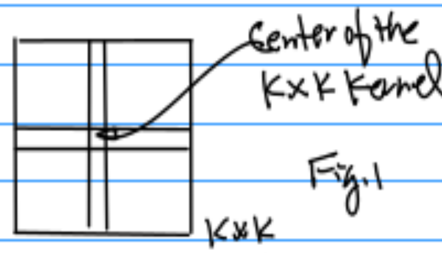


Fig. 1

Output: 1 pixel
Input: KxK pixels.

Captures All Neighbouring pixels at a time, And Produce One pixel Output.

April 11 (Tue).

Presentation (Brief) ON Each Team Project.

Note 2. for each convolution Kernel, the Convolution Conducted will result in One Output Feature Layer

/ 2022F-108a-Yolo-architecture-loss-function-

Oct. 24 (Tue).

Note 1: Semester Long Project
Abstract Presentation
On Oct. 26th.

1° Check the CANVAS
Requirements. On Abstract.

2° One page Abstract.
Title,
Technical Challenges
Methodology / Techniques
Hardware / Software Resources
Deliverables.
Table of Responsibility.

Consider the Architecture for YoLo.

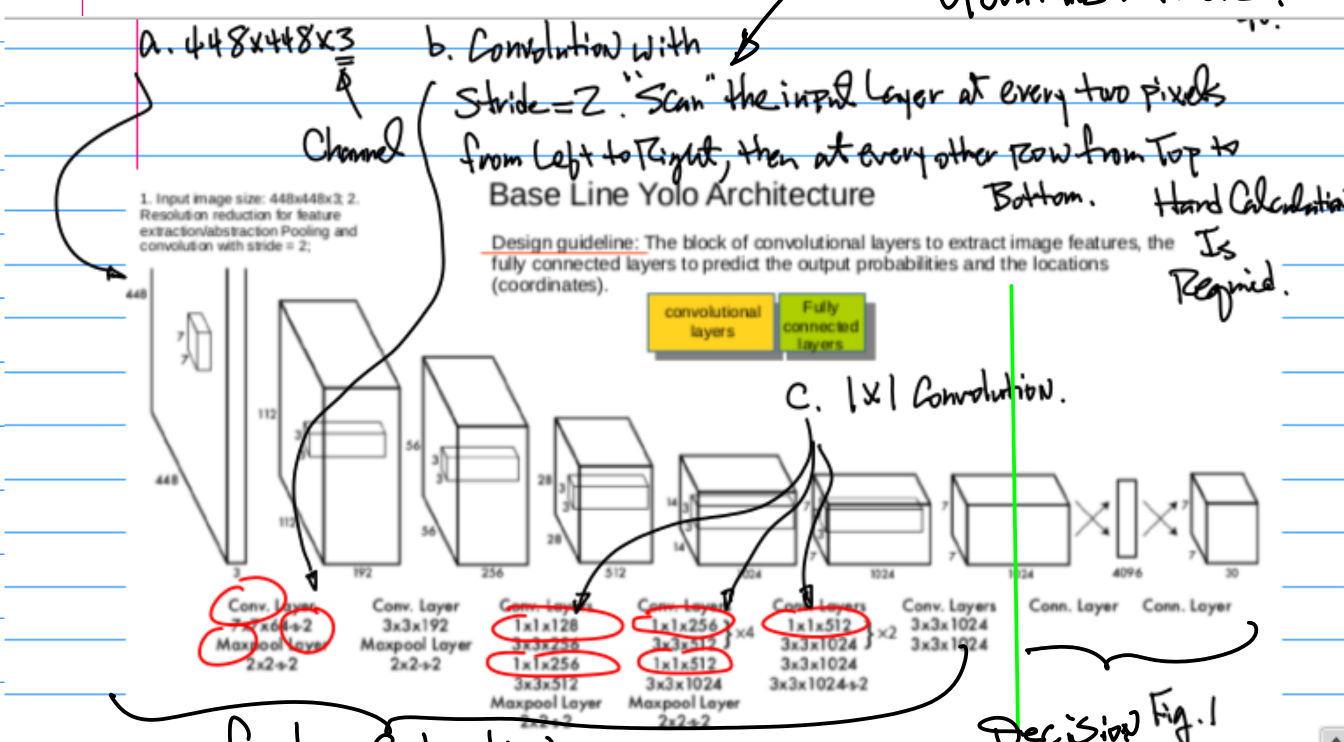
Example: Continuation. Ref. PP41.

Note: Combine Convolution + Subsampling Operations into One.

a. 448x448x3
Channel

b. Convolution with Stride = 2. "Scan" the input layer at every two pixels from Left to Right, then at every other row from Top to Bottom.

Hand Calculation Is Required.



c. 1x1 Convolution.

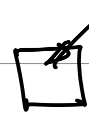
Feature Extraction

Decision making.

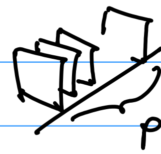
Fig. 1

For 1x1 Convolution.

Option 1. 1x1 Kernel



Option 2 1x1xP



Convolution with the Kernel in Option 1 produces duplication of the feature layer but weighted by its Coefficient α.

Convolution with Option 2 on a set of feature Layers will allows us to integrate multiple (high Dimensional) features layers into One feature layer.

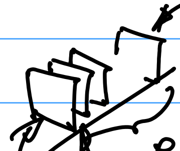
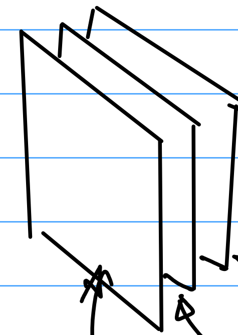


Fig.1.

$$\sum_i Coef_i = 1$$

In addition, we can assign different Coefficient to emphasize the Contributions from Some feature Layers than the others.

Suppose P=3.



Oct.26 (Th).

Project Presentation in Class.

Oct.31 (Tue)

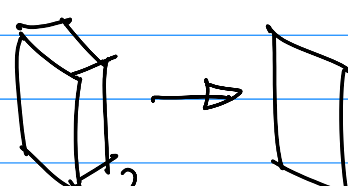
Subject today: 1x1xK Convolution

Example: General Guideline:

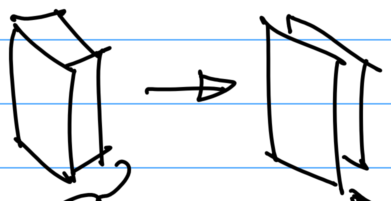
To Reduce the Number of feature layers to A manageable Level.

e.g. from K channels (Layers) to 1 from Fig.1 where the Layers = P.

$1\times1\times P|_{P=192}$



for the equal contribution from each Layer, we can make Coefficient $\alpha_i = 1/192$. $i = 1, 2, \cdots, 192$.



For the above requirements,

Two 1xK Kernels

$1 \times 1 \times P_1$ and $1 \times 1 \times P_2$

$P_1 + P_2 = 192$ AND

$P_1 = P_2$

Ref: On the Class github.

2022F_108a_...

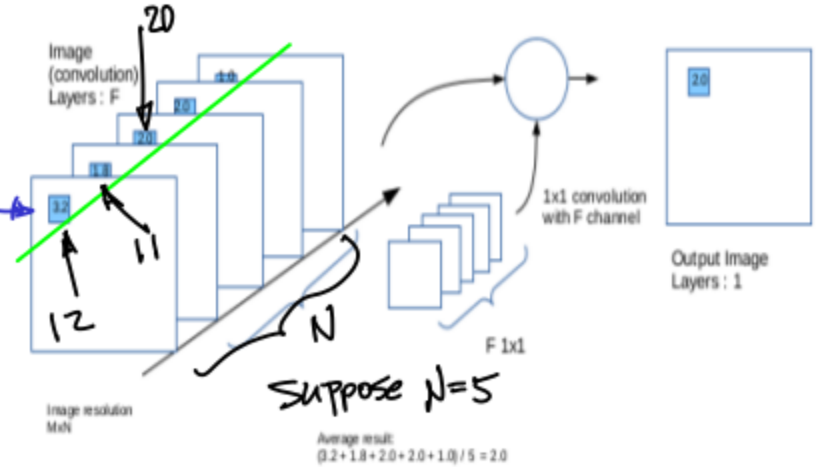
Note: To Combine N=5 Layers (Feature Layers) into 1 layer. We have 1x1x5 Kernel.

The 1x1 convolution enables dimension reduction by reducing the number of channels in convolution layers

1. Suppose the input layers is C*H*W, where C is its channels. The 1x1 convolution generates one average result in shape H*W. The (filter) is a vector of length C.
2. Now if you have F 1x1 filters, you get F layers of output, the output shape is F*H*W. For input layer C*H*W with F 1x1 convolution (with channel is C), you will get F*H*W layers.

Note1.

One pixel across the entire stack of the feature layer.



Note2. Suppose N=6, No. of feature Layers. we would like to have 2 output feature Layers.

a) the 1st 3 features is defined as Group 1.

b) the 2nd 3 feature Layers is defined as Group 2.

c) Output 1 feature Layer each from Group 1 and 2 Respectively.

Design $1 \times 1 \times P_1$ and $1 \times 1 \times P_2$ Convolutions.

d) Suppose Group 2 has Twice as much Contribution as Group 1.

$$\begin{cases} \alpha_2 = 2\alpha_1 & \cdots (1) \\ \alpha_1 + \alpha_2 = 1 & \cdots (2) \end{cases}$$

Substitute (1) into (2) we have

$\alpha_1 + 2\alpha_1 = 1$

$\therefore \alpha_1 = 1/3$

$\alpha_2 = 2/3$ //

CMPE258

F2023

Once the coefficients are set, Convolution can be conducted as follows. for Example for $\alpha_1=\alpha_2\cdots=\alpha_5=1/5$
$N=5$.

PP.40. from the Ref.

$$\frac{1}{N}\left[I_1(x_1,y_1)+I_2(x_1,y_1)+\cdots+I_5(x_1,y_1)\right] \quad \cdots (1)$$

$$\frac{1}{N}I_1(x_1,y_1)+\frac{1}{N}I_2(x_1,y_1)+\cdots+\frac{1}{N}I_5(x_1,y_1)$$

Example: Ref. PP.43 Equal contribution By each feature Layer.

Given 3 Feature Layers

$$\underset{F_1(x,y)}{\begin{matrix}1&1&3\\2&0&-5\\1&1&0\end{matrix}}\qquad \underset{F_2(x,y)}{\begin{matrix}2&0&-1\\1&0&2\\0&0&1\end{matrix}}\qquad \underset{F_3(x,y)}{\begin{matrix}2&3&2\\1&0&2\\6&0&1\end{matrix}}$$

3x3

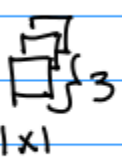


Fig.1a

↓ Computation.

For Step1. PP.52

$$\begin{matrix}1&1&3\\2&0&-5\\1&1&0\end{matrix}\qquad \begin{matrix}2&0&-1\\1&0&2\\0&0&1\end{matrix}\qquad \begin{matrix}2&3&2\\1&0&2\\6&0&1\end{matrix}$$

Fig.1b

$$1*C_1+2*C_2+2*C_3=\frac{1}{3}(1+2+2)$$
$$=5/3$$

5/3		

Step2.

$$\begin{matrix}1&1&3\\2&0&-5\\1&1&0\end{matrix}\qquad \begin{matrix}2&0&-1\\1&0&2\\0&0&1\end{matrix}\qquad \begin{matrix}2&3&2\\1&0&2\\6&0&1\end{matrix}$$

$$1*C_1+0*C_2+3*C_3$$
$$=\frac{1}{3}(1+0+3)=4/3$$

Fig.1c

5/3	4/3	

Then, Carry out the Rest of the Convolution.

Consider Probability Distribution map.