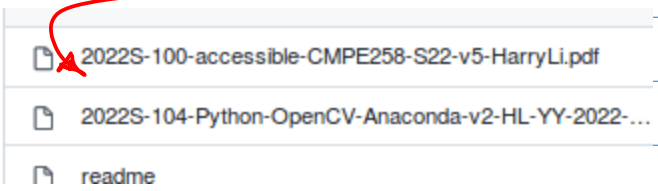
CMPE258 Spring2022

1/

Feb1st. Organizational meeting.

1. Today's Topics "Greensheet"



Naming Convention Yr+Semester+ID +Name+Date

Content Information: E-mail: hua.li@sjsu.edu

Text message to (650)400-1116.

Office Hours: M.W. 4:30-5:30pm.

Zoom (link to be shared in the email)
Join from PC, Mac, Linux, iOS or Android: https://sjsu.zoom.us/j/85616325978?pwd=MzlRbDJXVHBDQ2g1U0RPM2tYc045Zz09

Password: 451032

On-Line materials on github
https://github.com/hualili/opencv/tree/master/deep-learning-2020S
https://github.com/hualili/opencv/tree/master/deep-learning-2022s

Also, CANVAS → mostly for Assignments and projects.

All Assignment/Projects are posted on Both github & SJSU CANVAS.

Lecture Material consists of PPT. posted on github, and Lecture Notes (White-Board Written Notes)

Core Emphases of the Class: Deep Convolutional Neural Networks, And their Application in Image Analysis, Video Analysis.

1. Text Book:
https://github.com/hualili/opencv/blob/master/IP120-AI-DL/2018F/2018F-6-DeepLearningCh02.pdf

2. Computer Vision Book By Horn as a reference for Convolution & Image Segmentation, Contours Analysis (Binary Image).

3. OpenCV Reference Book (2nd Edition) together with On-Line Document (OpenCV).

Note: OpenGL (GL: Graphics Library) is just for Reference purpose, no need for this class. (But maybe helpful for the future research).

Unity is game Development platform, interactive 3D Graphics Design platform.

Programming Languages:
1' Python. 3.6 or 3.7
2. C/C++

Feb.8th.
Homework (Due A week from today)
No Submission. Submit A screen

Capture that shows OpenCV installed successfully, with Jpg or png file with Naming of the file as follows:

FirstName_LastName_SID_OpenCV.jpg

This Homework will be posted on CANVAS, Submission is on CANVAS

Homework, Installation of Tensor Flow, Due 2weeks Feb 15th.

Submission: Screen Capture that shows the installation is successful. Submission on CANVAS.

Submit jpg, png file with the Naming convention as follows:

FirstName_LastName_SID_TF.jpg

Note: Optional, for Edge AI Computing, Consider using NVDA Jetson NANO (4GB) version.

5% Bonus

Grading:

Homework, Projects : 30%
5% 25%

Project 1. Computer Vision for Preprocessing. plus Deep Convolutional Neural Nets To give Real Time Detection Result of Last 4 Digits of a Student ID.

10%.

Project 2. "Semester Long" Project, with technical requirements (List)

Team project. 4 person Team.

Each person has clear Definition of the tasks (Programming/Coding) And Balanced Contribution.

Final PPT, Demo Presentation

15%.

Midterm Exam: 30%.

Need to use your Laptop Computer, to Run/Execute code, modify the code.

Final 40%

Introduction

Topics { Neural Networks formulation (Basic Building Blocks)
Digital Images/Videos.

Example: A Single Neuron Formulation (Some kind Brain Cell)

Step 1. Summation function.

Fig 1.

"Σ" Summation function,

Note: $\sum_{K=1}^{N} x_K = x_1 + x_2 + \cdots + x_N$

Step 2. Inputs

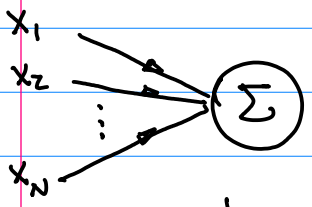
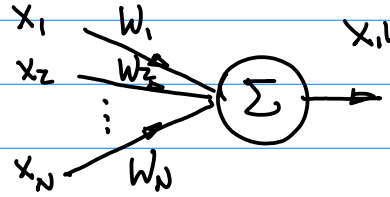


Fig 2.

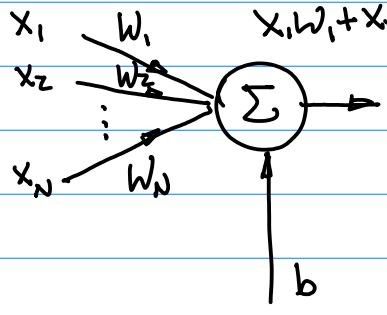
Step 3. Weights (Knowledge)



$x_1 w_1 + x_2 w_2 + \cdots + x_N w_N = \sum_{K=1}^{N} x_K w_N$

Fig 3.

$$\sum_{K=1}^{N} x_K w_N = x_1 w_1 + x_2 w_2 + \cdots + x_N w_N \quad \cdots (1)$$



$x_1 w_1 + x_2 w_2 + \cdots + x_N w_N + b$

Fig 4

$$\sum_{K=1}^{N} x_K w_N = x_1 w_1 + x_2 w_2 + \cdots + x_N w_N + b \quad \cdots (1b)$$

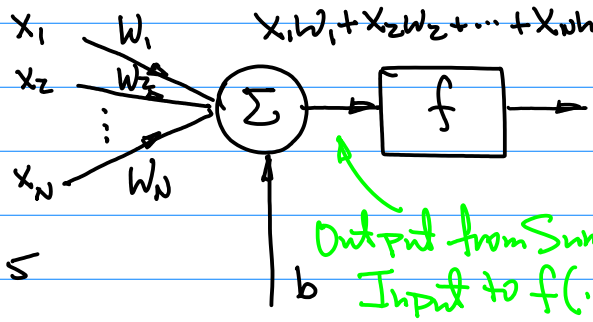


Fig 5

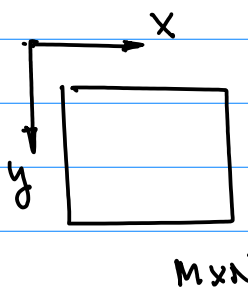
Note: Activation function f, denoted as f(·) (A function of Independent Variable "·", or A function of Input "·")

$$f(\cdot) = f(x_1 w_1 + x_2 w_2 + \cdots + x_N w_N + b)$$

$$= f\left(\sum_{K=1}^{N} w_K x_K + b\right) \quad \cdots (2)$$

Summary: The output of a Single is given by Eqn (2). Where Activation function f(·) can take different forms, it affects the Learning, Learning Speed.

Example: Digital Image, I(x,y)



$I(x,y)$ ← (x,y) Location of A picture element, "pixel"

Intensity, And/OR Color of An Image

In Case of a Single pixel, (x,y) is the Location of this pixel, I is Color/Intensity of the pixel

Note: For An Image $I(x,y)$ its features include { Resolution $M \times N$ (No. of pixels per Row, No. of Rows per frame); Pixel Depth: bpp (Bit per pixel)

$I(x,y)_{M \times N}$ OR $I(x,y)$

1024 x 768 = pixels/Row = Rows

For A color Image, A pixel depth very often is equal to 24 (bpp)

r, g, b Primitive color of red (r), green (g), blue (b) has 8 bits quantization level, e.g.

r: [0, 255], g: [0, 255], b: [0, 255].

Feb 8th (Tue)

Today's Topics: 1° Introduction, Basic Building Blocks, Math Formulation. 2° Simple python Code for OpenCV.

Example: Notations & Formulation

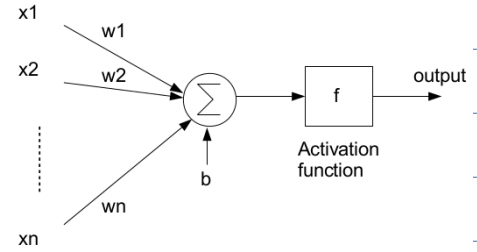


Fig.1

1. Notation for Input $x_i, i=1,2,\cdots,N$

$$\{x_i | i=1,2,\cdots,N\} \quad \cdots (1)$$

Vector form,

$$(x_1, x_2, \cdots, x_N) \quad \cdots (1b)$$

$\rightarrow X$

Introduce Superscript j for Experiment j

Input x_i^j, $i=1,2,\cdots,N$; $j=1,2,\cdots,P$

Hence Eqn (1) Becomes

$$\{x_i^j | i=1,2,\cdots,N; j=1,2,\cdots,P\}$$

$(x_1^j, x_2^j, \cdots, x_N^j)$ for Experiment j

2. Notation for Weight

w_i, for $i=1,2,\cdots,N$

Hence,

$$(w_1, w_2, \cdots, w_N) \quad \cdots (2)$$

$\rightarrow W$

3. Inputs & weights

x_i w_i

$w_i x_i$ for $i=1,2,\cdots,N$

$(x_1, x_2, \cdots, x_N) \cdot (w_1, w_2, \cdots, w_N)$

$= w_1x_1 + w_2x_2 + \cdots + w_Nx_N = \sum_{i=1}^{N} w_i x_i$

$(w_1, w_2, \cdots, w_N) \cdot (x_1, x_2, \cdots, x_N)$

$= w_1x_1 + w_2x_2 + \cdots + w_Nx_N = \sum_{i=1}^{N} w_i x_i$

4. Transfer Function, Denoted as

h

OR $h(\cdot)$

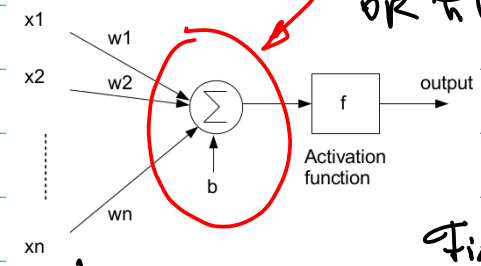


Fig. 2

$$h = \sum_{i=1}^{N} w_i x_i + b = W \cdot X + b \quad \cdots (3)$$

Offset ("Bias")

h, or $h(\cdot)$, or Eqn (3), or

$h(w_i; b)$, or $h(w_i)$

5. Activation Function. f

Acts like a switch, ON/OFF & Attenuate the Output

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

$f(h(w_i, b))$, OR $f(h(\cdot))$

The output of A Neuron is denoted as

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

$= f(h(w_i; b)) \quad \cdots (4)$

5. Outputs for A Neural Network

y_k, for $k=1,2,\cdots,Q$

if $Q=1$, y_1, also better to just y_1 for a single Neuron as y

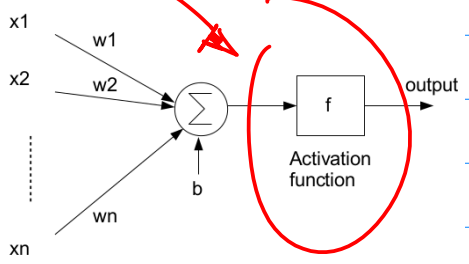
Question: what output? whose output?

Output from the Neuron (or Network)

$\tilde{y}$ (Tilde)

Ground Truth is denoted as y

6. Loss function, Objective Function, or difference



"Supervised Learning"

For Experiment j (multiple experiments)

$$\tilde{y}^j - y^j \quad , \quad j=1,2,\cdots,P \quad \cdots (5)$$

$\tilde{y}^j$: Output from the NN (Neuron or Network)

y^j: Ground Truth for the experiment

for $j=1$, $\tilde{y}^1 - y^1$

$j=2$, $\tilde{y}^2 - y^2$

$\vdots$

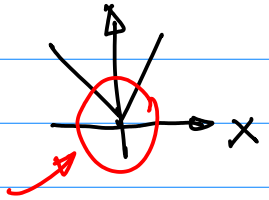
$j=P$, $\tilde{y}^P - y^P$

Therefore, put all these differences (Loss) together

$$(\tilde{y}^1 - y^1) + (\tilde{y}^2 - y^2) + \cdots + (\tilde{y}^P - y^P)$$

$$= \sum_{k=1}^{P} (\tilde{y}^k - y^k) \quad \cdots (6)$$

Note: $y = |x|$



To deal with the issue of Absolute value of a function, Let's square it.

Hence Eqn (6) becomes

$$\sum_{k=1}^{P} (\tilde{y}^k - y^k)^2 \quad \cdots (6b)$$

7. Objective function

$$L \triangleq \sum_{k=1}^{P} (\tilde{y}^k - y^k)^2 \quad \cdots (7)$$

OR,

$$L(w_i) \triangleq \sum_{k=1}^{P} (\tilde{y}^k - y^k) \quad \cdots (7b)$$

$$= \sum_{k=1}^{P} (\tilde{y}^k(f(w_i)) - y^k) \quad \cdots (7c)$$

8. To generalize the result in Eqn (7c) for multiple Output, we have the following formulation.

From Eqn (7)

$$\sum_{k=1}^{P} (\tilde{y}^k_{k_2} - y^k_{k_2})^2$$

k_2: Output k_2 for multiple output y_j, $j=1,2,\cdots,k_2,k_2+1,\cdots,M$

We can count All outputs loss.

$$\sum_{k=1}^{P} (\tilde{y}^k_1 - y^k_1), \quad \sum_{k=1}^{P} (\tilde{y}^k_2 - y^k_2), \quad \cdots$$

$$\cdots \sum_{k=1}^{P} (\tilde{y}^k_M - y^k_M)$$

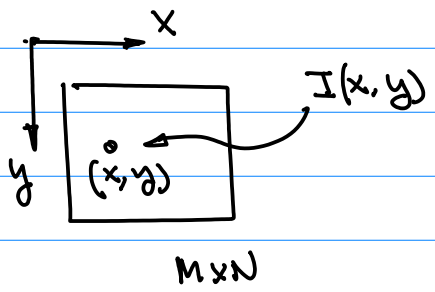
Hence

$$\sum_{k_2=1}^{m} \sum_{k_1=1}^{P} (\tilde{y}^{k_1}_{k_2} - y^{k_1}_{k_2})^2 \quad \cdots (8)$$

CmpE258 Feb8,22

$$L(W_{ij}) \triangleq \frac{1}{2} \sum_{h_2=1}^{m} \sum_{h_1=1}^{p} \left(\tilde{y}^{h_2}_{h_2} - y^{h_2}_{h_2} \right)^2 \quad \cdots (8b)$$

Example: Digital Image I(x,y)

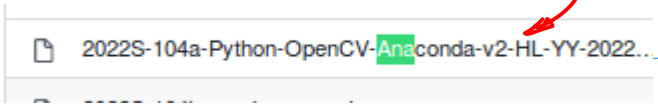


Installation of OpenCV → Python OR (C/C++, But Python is Better)

Different Packages for ML/DL, may Need different Version of Python, And different Packages

Anaconda is a well developed, Adopted tool for Package management.

1. Check github class Reference

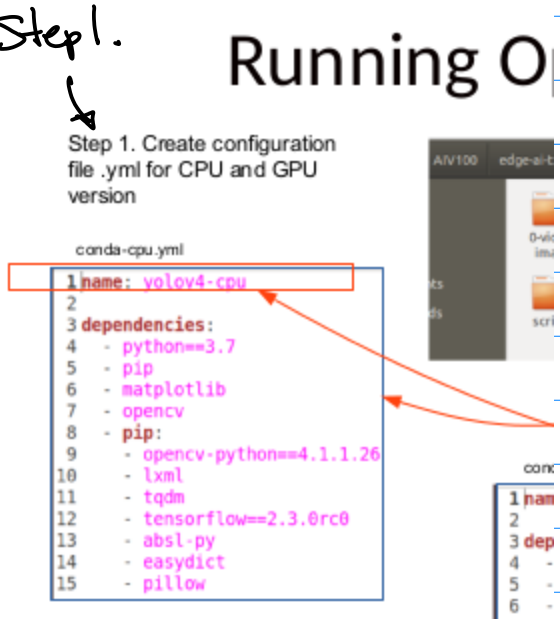


Note: For installation of Anaconda, Check github document, "2022S-104-~"

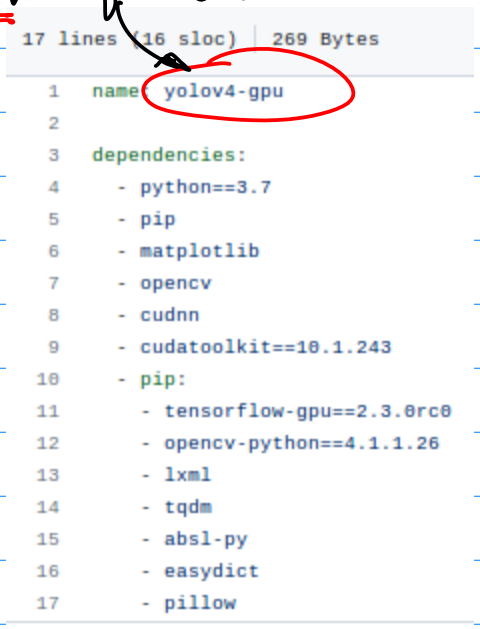
Once Anaconda is installed, then Let's take a look the configuration for

Tensorflow & OpenCV Environment

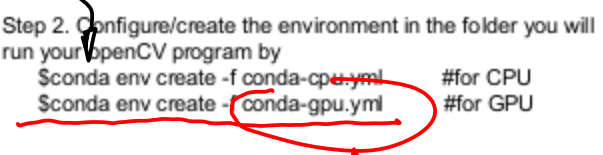
Step 1.



Name of the environment



Step2. Create the environment



file Name to Be used

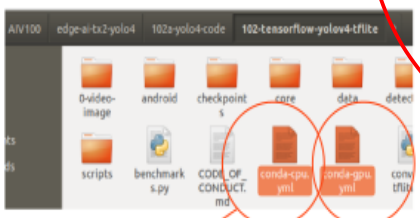
CmpE258 Feb8

Step3 Activate the enviroment

Step4. Run Your Python Code (OpenCV or T.F.)

Running OpenCV with Conda Environment

Step 1. Create configuration file .yml for CPU and GPU version



Step 2. Configure/create the environment in the folder you will run your openCV program by

$conda env create -f conda-cpu.yml #for CPU

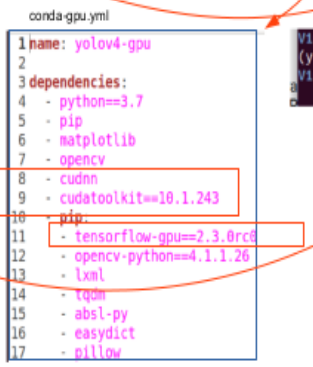
$conda env create -f conda-gpu.yml #for GPU

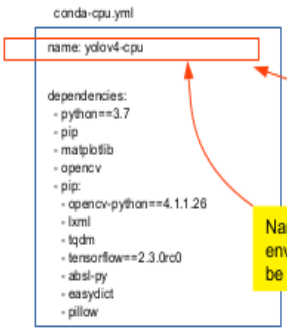
Step 3. Activate the environment with the name that you have created by

$ conda activate yolov4-cpu #for CPU

$ conda activate yolov4-gpu #for GPU

Name of the environment in the .yml file





Name of the environment to be created

Step 4. Now run your Python OpenCV program

$python myOpenCV.py image.png

Note: Install Anaconda → Create the environment for T.F. & OpenCV

Execute OpenCV ← Activate the environment

Sample Code (From the class github)

2022S-104d-#2-pdisplay-2019-1-30.py

a. Program Header — Name, Program Name, Version, Status (Debug, Release), Date. Note

b. cv2 OpenCV

Caption of the window

```
1  """
2  pdisplay.py
3  Demo read and display image
4  """
5  import sys
6  import cv2
7  import numpy as np
8
9  #main(sys.argv[1:])
10 window_name = 'Display Image'
11
12 imageName = sys.argv[1] #get file name from command line
13
14 src = cv2.imread(imageName, cv2.IMREAD_COLOR)
15
16 if src is None:
17    print ('Error opening image!')
18    print ('Usage: pdisplay.py image_name\n')
19
```

CMPE258 Feb8,22

9

2nd Ref on the github

2022S-103a-notation-neuro-loss-function-2022-2-8-1.pdf

```
22  while True:
23      cv2.imshow(window_name, src)
24
25      c = cv2.waitKey(500)
26      if c == 27:  #ESC
27          break
28
29      ind += 1
```

③

Keyboard input to exit.

Note: These Python functions are required. (memorize them!)

① import cv2 ② cv2.imread( )

③ cv2.imshow( ).

Homework (Due A week from today) Feb 15

1° Installation of OpenCV.

2° Installation of Anaconda.

3° Use your Smartphone to take photo, and save it for OpenCV Program

4° Write A Python Program (Ref. Code from the class github is OK) to Display:

a. Your Name + SID (4 Digits)

b. Your Smartphone picture.

5° Submission: One pdf file And Zipped. On CANVAS.

Feb 15

Ref: 1° 2022S-103a-Notation

2° 2022S-103C

2022S-103a-notation-neuro-loss-function-2022-2-8.pdf

VIII. Minimize the Loss Function

$$\frac{\partial L}{\partial w_{i,k}} = \frac{\partial}{\partial w_{i,k}} \frac{1}{2} \sum_{j=1}^{P} \sum_{i=1}^{M} (\tilde{y}_i^j - y_i^j)^2 \qquad (24)$$

Now, consider to optimize Neural Network Performance By minimizing the Loss function.

Mathematical Background

Derivative(s) → Partial Derivatives

a. b.

↓

c. Gradient

↓

A special kind of Gradient: Desent d.

To Reduce Loss

↓

"Steepest" ~

↓

Training A Neural Network

↓

Generalize the concept: "Learning" (Supervised learning)

Derivatives.

Definition $\frac{d}{dx} f(x) \triangleq \lim_{\Delta x \to 0} \frac{\Delta f}{\Delta x}$

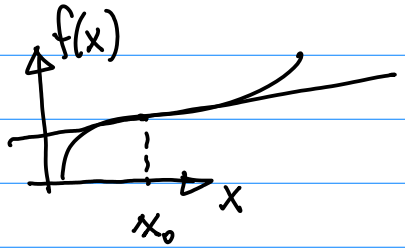
$$= \left. \frac{f(x+\Delta x) - f(x)}{\Delta x} \right|_{\Delta x \to 0} \quad \cdots (1)$$

Note Eqn(1) is Based on "Forward Difference".

$f(x+\Delta x)$

↑

forward of x



if $f'(x) = \frac{d}{dx} f(x) > 0$, then $f(x)$ at $x = x_0$ will increase

if $f'(x) < 0$, then $f(x)$ will decrease

if $f'(x) = 0$, then $f(x)$ is unchanged.

For a function $f(x_1, x_2)$, the partial Derivative of $f(x_1, x_2)$

$$\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2} \quad \cdots (2)$$

$$\frac{\partial f(x_1,x_2)}{\partial x_1} \triangleq \lim_{\Delta x_1 \to 0} \frac{f(x_1+\Delta x_1, x_2) - f(x_1,x_2)}{\Delta x_1} \quad \cdots (2a)$$

$$\frac{\partial f(x_1,x_2)}{\partial x_2} \triangleq \lim_{\Delta x_2 \to 0} \frac{f(x_1,x_2+\Delta x_2) - f(x_1,x_2)}{\Delta x_2} \quad \cdots (2b)$$

$\frac{\partial f}{\partial x_1} > 0$, $f \uparrow$ as $x_1 \uparrow$

$\frac{\partial f}{\partial x_2} > 0$, $f \uparrow$, as $x_2 \uparrow$

Define gradient of a function $f(x_1, x_2)$

(Note: Connection to Neural Network, And a Single Neuron. inputs $x_1, x_2, \cdots x_N$ And weights $w_1, w_2, \cdots, w_N$)

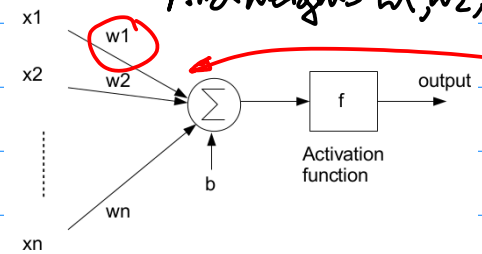


Fig1.

Defined a vector

$$\begin{pmatrix} \frac{\partial f(x_1,x_2)}{\partial x_1} \\ \frac{\partial f(x_1,x_2)}{\partial x_2} \end{pmatrix} \text{ or } \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \end{pmatrix}$$

We can use github Lecture Notes to expand Eqn(3) to N-Dimensional Case $\cdots (3)$

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix} \quad \cdots (3b)$$

For A Single Neuron, we have gradient function as follows.

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial w_1} \\ \frac{\partial f}{\partial w_2} \\ \vdots \\ \frac{\partial f}{\partial w_N} \end{bmatrix} \quad \cdots (3c)$$

Guideline for the Development of "Training" Technique:

Negative Gradient gives the steepest Descent of a loss function f.

Note: Negative gradient gives the Best way to adjust W_i in a Neural Net

to make the Neural Net to Reach to optimal solution.
e.g. minimized loss function.

The Approach to verify the Above Technique:

Generalized loss function, as f, or $f(\cdot)$, or $f(x_1, x_2, \cdots, x_N)$, or $f(\vec{x})$, or $f(w_1, w_2, \cdots, w_N)$, or $f(W)$

↓

Simple/Generalized Description to connect independent Variable $x_1, x_2, \cdots, x_N$ (or $w_1, w_2, \cdots w_N$) to the function f (loss function)

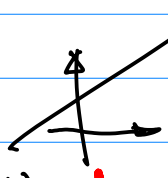
Taylor Expansion for $f(x)$

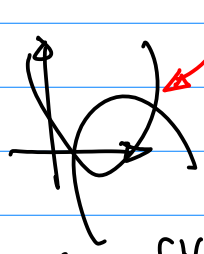
$$f(x) = f(x_0) + \frac{f'(x_0)}{1!}(x-x_0) + \frac{f''(x_0)}{2!}(x-x_0)^2 + \cdots + \frac{f^{(k)}(x)}{k!}(x-x_0)^k + \cdots \quad \cdots (4)$$

$$= \text{Constant} + \text{2nd (linear function)} + \frac{f''(x_0)}{2!}(x-x_0)^2 \quad \text{Quadratic Term.}$$

Coefficient, Slop. $y = ax + b$

$\frac{f'(x_0)}{1!}(x-x_0)$ ↑ $a(x-x_0)$ linear function





$$f(x) = f(x_0) + \frac{f'(x_0)}{1!}(x-x_0) + R_n(x)$$

↑ Higher Order terms

$$f(x) \simeq \underbrace{f(x_0) + \frac{f'(x_0)}{1!}(x-x_0)}_{\text{linear function}}$$

$$f(x) = f(x_0) + \frac{f'(x_0)}{1!}\Delta x \quad \cdots (5)$$

We expand function f (Loss) into multi-dimensional case, e.g. $x_1, x_2, \cdots$ (or $w_1, w_2, \cdots$).

Choose work on 2D case. e.g.

$$f(x_1, x_2) = f(x_{10}, x_{20}) + \frac{\partial f}{\partial x_1}(x-x_{10}) + \frac{\partial f}{\partial x_2}(x-x_{20}) + \cdots \quad \cdots (6)$$

$$f(x_1, x_2) \simeq f(x_{10}, x_{20}) + \frac{\partial f}{\partial x_1}(x-x_{10}) + \frac{\partial f}{\partial x_2}(x-x_{20}) \quad \cdots (7)$$

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

Question, if we change independant Variable x_1 and x_2 by the negative gradient of f, then the function f is decreased or increased?

Based on the Theory, we should have loss function f decreased in an optimized way, e.g. "Steepest Descent."

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

$$\nabla f = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \end{pmatrix}$$

$$(\nabla f)^t = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}\right)$$

x_1^{k+1}, x_1^k

step @ k+1 at step k

$$\begin{cases} x_1^{k+1} = x_1^k + (-\eta)\frac{\partial f}{\partial x_1} & \cdots(8a) \\ x_2^{k+1} = x_2^k + (-\eta)\frac{\partial f}{\partial x_2} & \cdots(8b) \end{cases}$$

If we use Eqn(8a), (8b) to update the independant Variables x_1, x_2 (or weights w_1, w_2), then you are sure to be able to find loss function f at this step k+1, is smaller than itself, e.g. f function at step k.

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

The Above equation, eqn(10), is Taylor Expansion

Note: from the Analysis below, we have:

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

$$f_{x_1} = \frac{\partial f}{\partial x_1}$$

$$f_{x_1}^2 = \left(\frac{\partial f}{\partial x_1}\right)^2 \geq 0$$

Similarly $f_{x_2} = \frac{\partial f}{\partial x_2}$, $f_{x_2}^2 = \left(\frac{\partial f}{\partial x_2}\right)^2 \geq 0$

Therefore $-\left[\left(\frac{\partial f}{\partial x_1}\right)^2 + \left(\frac{\partial f}{\partial x_2}\right)^2\right] \leq 0$

Hence $f(x_1, x_2) - f(a, b) \leq 0$

$f(x_1, x_2) \leq f(a, b)$ //

$$x_1 - a = -\eta\frac{\partial f}{\partial x_1}$$

$$x_1^{k+1} - x_1^k = -\eta\frac{\partial f}{\partial x_1}$$

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

$x_1^k = a$, $x_2^k = b$

$f(a, b)$

$$f(x_1, x_2) \simeq f(x_{10}, x_{20}) + \frac{\partial f}{\partial x_1}(x - x_{10}) + \frac{\partial f}{\partial x_2}(x - x_{20})$$

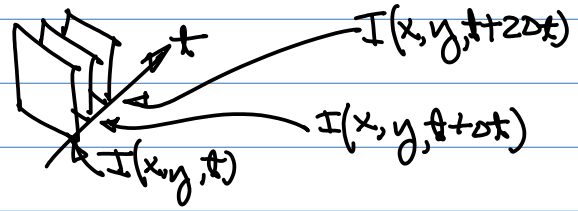
Eqn(7) pp.11

OpenCV

Note: One of the tools for Python Programming for this class is Pycharm.

Create A New Project → Configuration of the environment & packages Needed

Example: OpenCV for Video.



Exercise Use your Smartphone to Capture a video clip which has the following feature:

1. White Background, Such as Printer Paper or white Board, then, write your last 4 Digits on this white Background Paper/Board.
2. Be sure to use a marker to give adequate Contrast and width for each mark. (720P or 1080P Resolution)
3. Save the video (may have 20 ~ 60 seconds video clip) for the future use By Next Lecture.

Note: we will discuss Computer Vision preprocessing techniques in the next Lecture.

Feb 22nd (Tue.)

Today's Topics:

1. Theoretical Background on Deep Learning.
2. Preprocessing Techniques for Handwritten Student ID Recognition.

Single Neuron → Loss function → Gradient Descent Technique

Build multi-Layer feed forward NN.

Back Prop. → Deep Neural Network

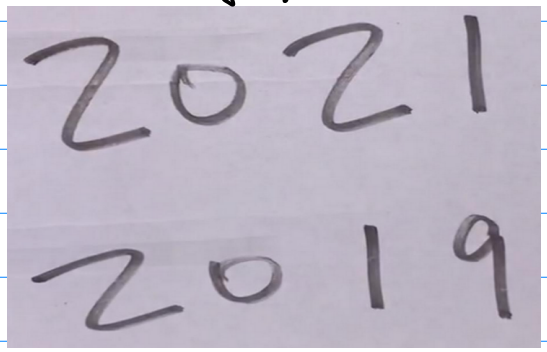
→ Convolutional NN CNN → DCNN

Example: Pre-processing Technique for handwritten Digits Recognition.

2022S-107a-video-digits1.mp4

2022S-107b-video-digits2-outdoor.mp4

Capture A Single frame from the videos



Objectives: To process video, identify ROI (Region of Interests), then
b. Remove Any Random Noise,
Enhance the quality of an image,
then c. feed it into Deep Learning Neural Network.

Example: Pre-processing Technique
Binary Image Processing.

Note: 1. A Color image, r, g, b
red, green, Blue
primitive colors, r, g, b : 8 bits per pixel.
24 Bits Color image

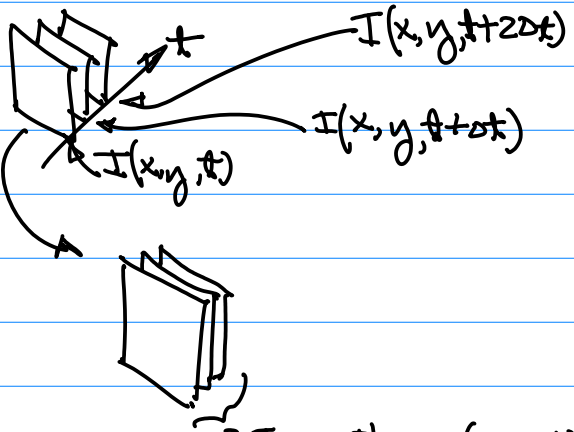


Fig. 1

3 Image Planes (r, g, b)
Pixel depth, Color Depth : 8 bits for Each Color plane
$r \in [0,255]$, $g \in [0,255]$, $b \in [0,255]$

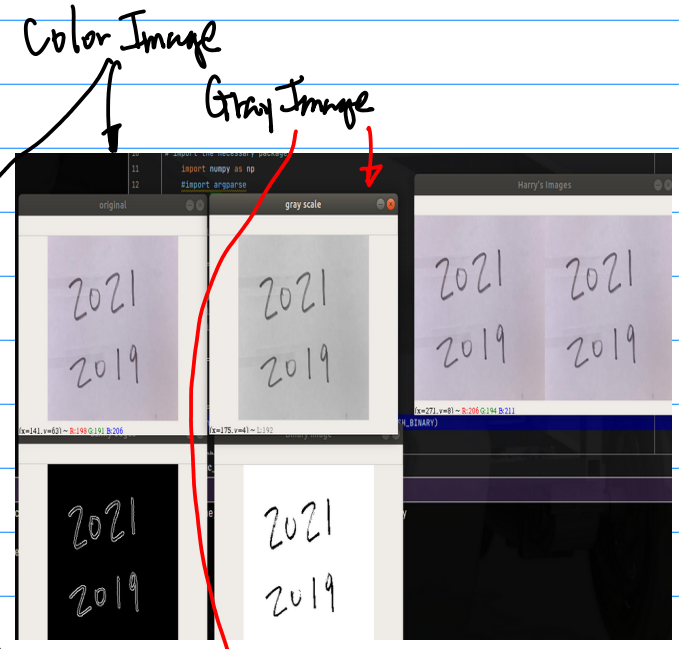
2. Gray Scale Image.
$I(x,y,t)$, drop the time t,
$I(x,y)$ 8 bit gray scale image

$I(x,y) \in [0,255]$,

$$I(x,y) = \frac{1}{3}[r(x,y) + g(x,y) + b(x,y)] \quad \cdots (1)$$

Provides a gray conversion.
Sometimes, weighted Average

$$I(x,y) = \frac{1}{3}[\alpha_1 r(x,y) + \alpha_2 g(x,y) + \alpha_3 b(x,y)] \quad \cdots (1b)$$



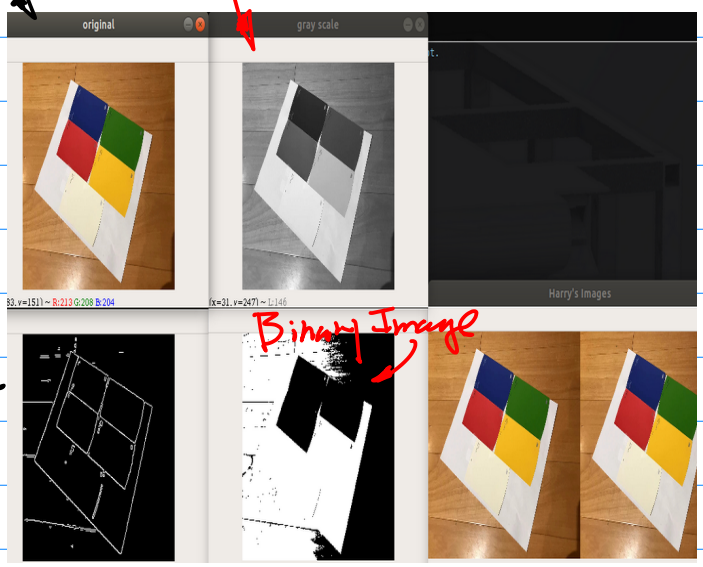


Fig 2

3. Binarize the gray scale Image (color Image can be binarized as well)

Binary image is Denoted as

$$B(x,y) = \begin{cases} C_1 & \text{if } I(x,y) \geq \text{Threshold} \\ C_2 & \text{O/W} \end{cases} \quad \ldots (2)$$

if $C_1 = 255$, $C_2 = 0$

Thresold is defined by user per his/her Processing Need.

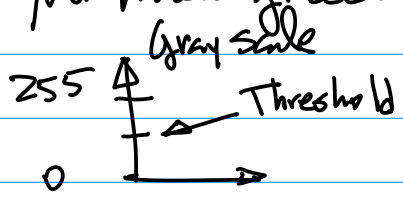


Fig3.

Ref: For Binary Image Processing

2022S-108-Intro-Binary.pdf

Task: To Identify the Regions of Interests (ROI)

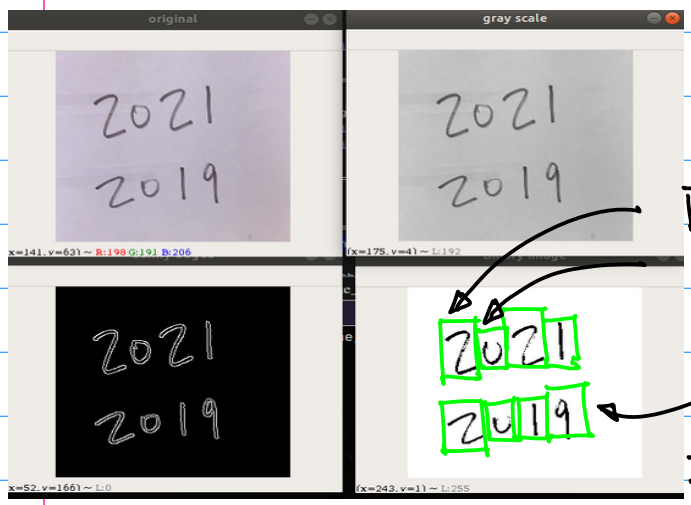


Fig4.

Note: the Aspect Ratio for Each ROI Can be different.

Define Aspect Ratio of A ROI:

Given a ROI shown in Fig3

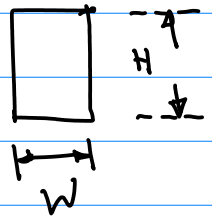


Fig5

with Height H, width W, the aspect Ratio α is defined as follows

$$\alpha = W/H \quad \ldots (3)$$

Suppose $W = 35$ (pixels), $H = 120$ (pixels)

$$\therefore \alpha = 35/120$$

Note: Color image, gray scale image, Binary image

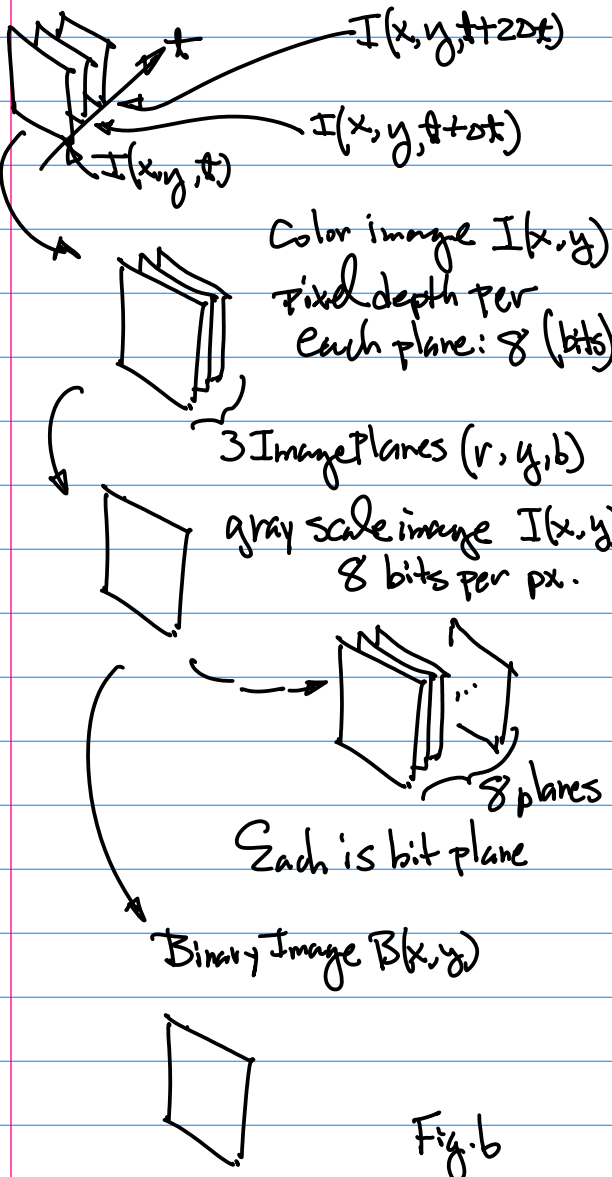


Fig.6

Question: Python Code for Tensor Representation of Each Image Types? e.g. Color, gray scale, and Binary.

Example: Binarize a gray scale image

2022S-108c-example-binary.pdf

1) Given a digital image f(x,y) find Binary Image Based on threshold T=135.

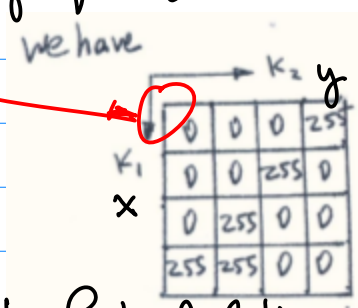
Sol

Based on T=135 and

x \ y				
	70	70	80	159
	75	85	149	124
	90	240	101	90
	211	209	115	85

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

Origin of the Coordinate System we have (0,0)



0	0	0	255
0	0	255	0
0	255	0	0
255	255	0	0

Fig7.

However, for hand calculation, we use (1,1) instead.

K_1: Index for Each Row,

K_2: Index for Each col.

Sample Code in OpenCV Python

2022S-107d-display-binary.py

```
20      image = cv2.imread(img, cv2.IMREAD_COLOR)
26      image = cv2.resize(image, (256, 256))
28      gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
30      ret, thresh1 = cv2.threshold(gray, 127, 255, cv2.THRESH_BINARY)
```

Note: 1. Binarization function, function's arguments

Input Image, threshold, intensity Value, Binarization keyword

Example: Algorithms for Binary Image Processing.

2022S-108-Intro-Binary.pdf

Pattern Recognition For Binary Images

The tool box for pattern recognition for binary images

1. Size
2. Moments
   - $\bar{x}$
   - $\bar{y}$
   - $\overline{x^2}$
   - $\overline{y^2}$ etc.
3. Perimeter
4. Orientation
5. Compositions of the above
   Perimeter and moments: vector
6. Invariant operators
   size invariant
   orientation invariant
   illumination invariant

Biologically inspired techniques

Rule 1. Proximity
Rule 2. Similarity
Rule 3. Closure
Rule 4. Good continuation
Rule 5. Symmetry
Rule 6. Simplicity

Note: 'Proximity' usage for clean up binary image and remove noise, as well as growing boundary points per 'good continuation' rule to form a better edge map.

Note: Similarity defines a interesting question, how to describe one object is similar, or somewhat similar to others, neural network and fuzzy logic may help.

Harry Li, PhD. SJSU

Shape Descriptor

Invariant "Shape Descriptor"

In this class, we would like to develop Techniques, Tools Described on the left Side of the PPT.

2022S-108b-BinaryImageFormula.pdf

Example:

$$\bar{X} \triangleq \frac{\iint_{\Omega} x\, B(x,y)\,dx\,dy}{\iint_{\Omega} B(x,y)\,dx\,dy} \quad \cdots (1)$$

Where, the Area of $B(x,y)$ is defined as

$$A \triangleq \iint_{\Omega} B(x,y)\,dx\,dy \quad \cdots (2)$$

Note 1. Binary or Gray Scale image, $I(x,y)$, $x = 0,1,2,\cdots,N-1$, $y = 0,1,2,\cdots,M-1$

No. of Row ↑ for Col.

$M \times N$

Col. (px) per Row ↑ Row.

2. Since Digital Image is Collection of Discrete Data, So Rewrite integration-Based formula By Summation.

From Eqn(1) of the handout, we have

$$\iint_{\Omega} B(x,y)\,dx\,dy$$

Ω: An Image, like the one below

70	70	80	159
75	85	149	121
90	210	101	90
211	209	115	85

$B(x,y)$: A Binary Image, like

we have

K_2 →, K_1 ↓

0	0	0	255
0	0	255	0
0	255	0	0
255	255	0	0

$\iint_{\Omega}$ — Summation w.r.t all x's; Summation w.r.t all y's

Therefore, we have

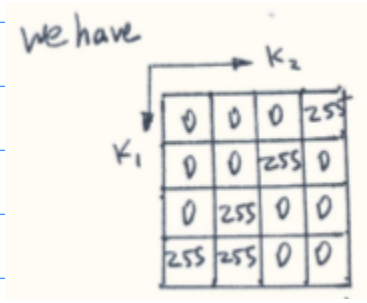
$$\sum_{y=0}^{M-1}\sum_{x=0}^{N-1} B(x,y) \simeq \iint_{\Omega} B(x,y)\,dx\,dy \quad \cdots(4)$$

$$\frac{1}{255}\sum_{y=0}^{M-1}\sum_{x=0}^{N-1} B(x,y) = \text{Area}$$

Find

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

for the given Binary Image $B(x,y)$ Below,



Sol. From Eqn (4), we have

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

Since image example is very small, we start index from 1, Not 0.

Hence,

$$\sum_{y=1}^{M-1}\sum_{x=1}^{N-1} B(x,y) = \sum_{y=1}^{4}\sum_{x=1}^{4} B(x,y)$$

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

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

$$= B(1,1)+B(2,1)+B(3,1)+B(4,1) +$$
$$B(1,2)+B(2,2)+B(3,2)+B(4,2) +$$
$$B(1,3)+B(2,3)+B(3,3)+B(4,3) +$$
$$B(1,4)+B(2,4)+B(3,4)+B(4,4)$$

$$= (0+0+0+255$$
$$0+0+255+0$$
$$0+255+0+0$$
$$255+255+0+0)$$

$$= 5\times 255$$

After multiplying $\frac{1}{255}$, we have

$$\sum_{y}\sum_{x} B(x,y) = 5$$

Exercise 1: Form 4-Person Team, elect a coordinator of the Team. to prepare for Semester Long Project Send me email By Next week with the team information.

Exercise 2. Use Existing Code Sample to write a program to display your Video Clips By Next week.

CmpE258 March 1st, 22

March 1st.

Today's Topics: Preprocessing for Digits Recognition:
2. Neural Networks (Feed Forward).

Homework Assignment Due A week from Today: March 8.

2022S-102b-homework2-building-blocks-gradient-2022-3...

Summation, Accumulative Loss

$$L_{total} = \frac{1}{2}\sum_{j=1}^{P}(\tilde{y}^j - y^j)^2$$

y^j: Ground Truth

(4.3) Suppose there are 3 experiments (training) are performed, with the known ground truth as (11.3, 0.2, 1), use the equation in Figure 3, find the total loss. You can use abstract output symbol such as y^2, and y^3 in your result provided you have evaluated y^1 output based on the given parameters in this assignment.

y^1, y^2, y^3 — $y^{(1)}, y^{(2)}, y^{(3)}$

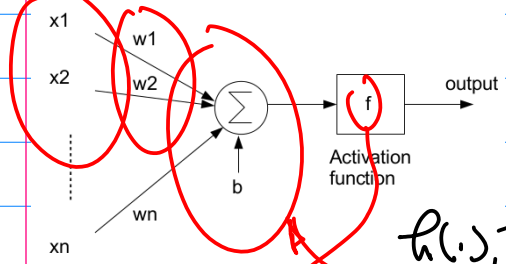
$\tilde{y}^{(1)} = 1.75$ (for Example).

$\tilde{y}^{(1)} - y^{(1)} = 1.75 - y^{(1)}$

$\tilde{y}^{(2)} - y^{(2)}, \ldots$

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

N=3, $x_1, x_2, x_3, w_1, w_2, w_3$

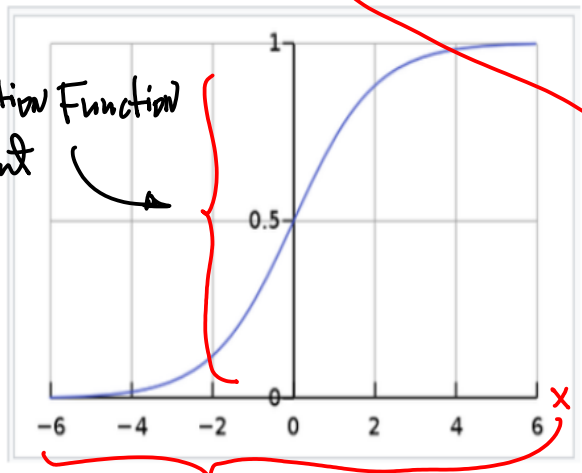


$h(\cdot)$ Transfer function.

Activation Function: Sigmoid Function

Fig1.

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



Activation Function Output

Input to the Activation function, e.g. $h(w_{ij}, b)$'s output

$$\frac{\partial L}{\partial w_{i,k}} = \frac{\partial}{\partial w_{i,k}}\frac{1}{2}\sum_{j=1}^{P}\sum_{i=1}^{M}(\tilde{y}_i^j - y_i^j)^2$$

$\tilde{y}_i^j$: ith Output

M=1, Then

$$\frac{\partial}{\partial w_{ik}}\frac{1}{2}\sum_{j=1}^{P}(\tilde{y}^j - y^j)^2\Big|_{\tilde{y}^j = f(h(w_{ik},b))}$$

where

$$\tilde{y}^j = f(h(w_{ik},b)) = f\left(\sum_{i=1}^{3} w_i x_i + b\right)$$

Therefore

$$\frac{\partial}{\partial w_{ik}}\frac{1}{2}\sum_{j=1}^{P}(\tilde{y}^j - y^j)^2 =$$

$$\frac{1}{2}\sum_{j=1}^{P}\left(f\left(\sum_{i=1}^{3} w_i x_i + b\right) - y^j\right)$$

CmpE258 March 1st, 22

$$\frac{1}{2}\sum_{i=1}^{p}\left(f\left(\sum_{i=1}^{3} w_i x_i + b\right) - y^i\right)$$

w_1, w_2, w_3 ; x_1, x_2, x_3 ; b.

then Construct feedforward NN.

Ground Truth

Example: Collecting data for training

Name	Weight (lb)	Height (in)	Gender
Alice	133	65	F
Bob	160	72	M
Charlie	152	70	M
Diana	120	60	F

Signs	Mij	Mpq	Sign
V1	133	65	Stop
V2	160	72	Right
V3	152	70	Right
V4	120	60	Stop

The Loss function

The weights for Each Neuron w_i to Replace x_i

$$\nabla f = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \cdots \\ \frac{\partial f}{\partial x_i} \\ \cdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix}$$

Fig.2

Note: Input vector (Training Data) (x_1, x_2) 2 Dimensional Vector.

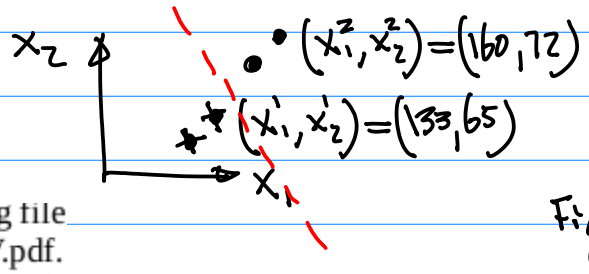
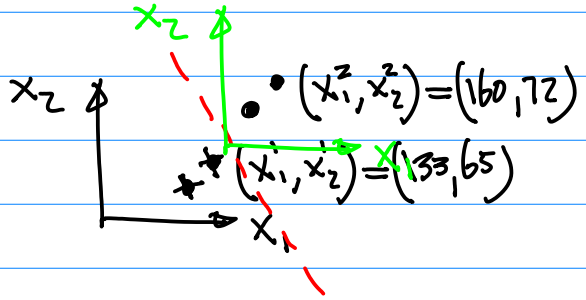


Fig.3

7. Submit your work in one PDF file, then zip it. Use the following file firstName_lastName_SID(last-4-digits)_cmpe258_~~CondaOpenCV~~.pdf.

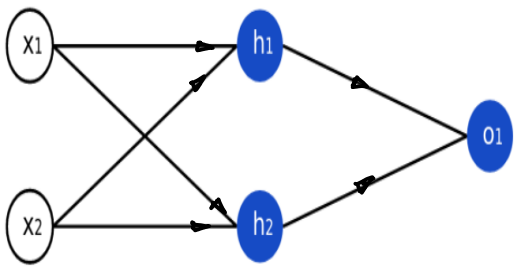
Basic Building Block

" B^3 "



Example: Build Feedforward Neural Network (Multi-Layer NN)

2022S-103c-NN-Intro-Python-v5.pdf



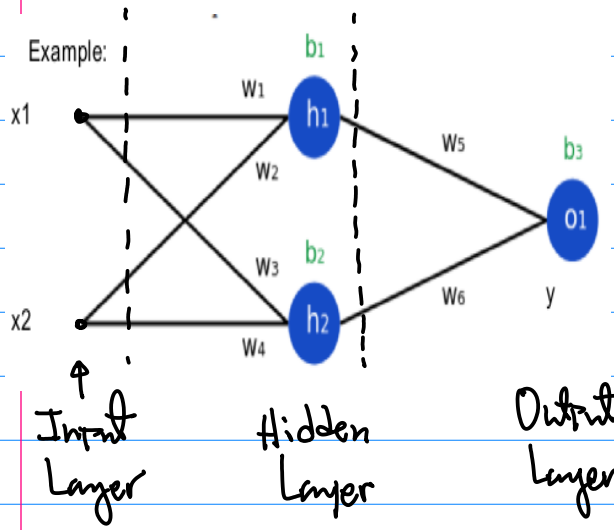
Define "Loss function", error function

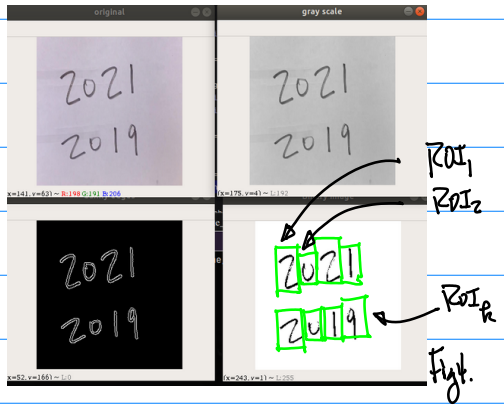
$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_{true} - y_{pred})^2$$

mean

Python Code for Basic Building Blocks,

Note: The enumeration for W_i Each Handwritten Digit.





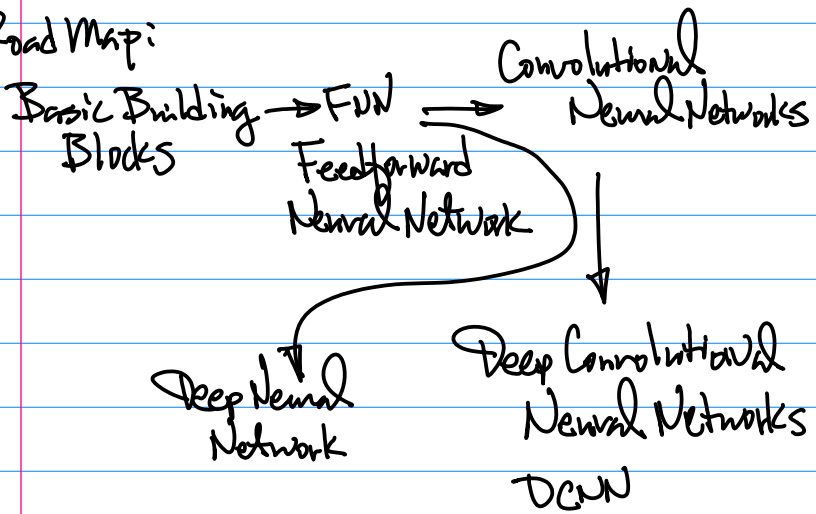
Requirements for this Python Code:

1° To be able to make connection Between the code and the Actual Neurons, Neural Network Architecture;

2° To modify the code to handle more neurons in the Hidden Layer if Needed;

3° Run the code,

Road Map:

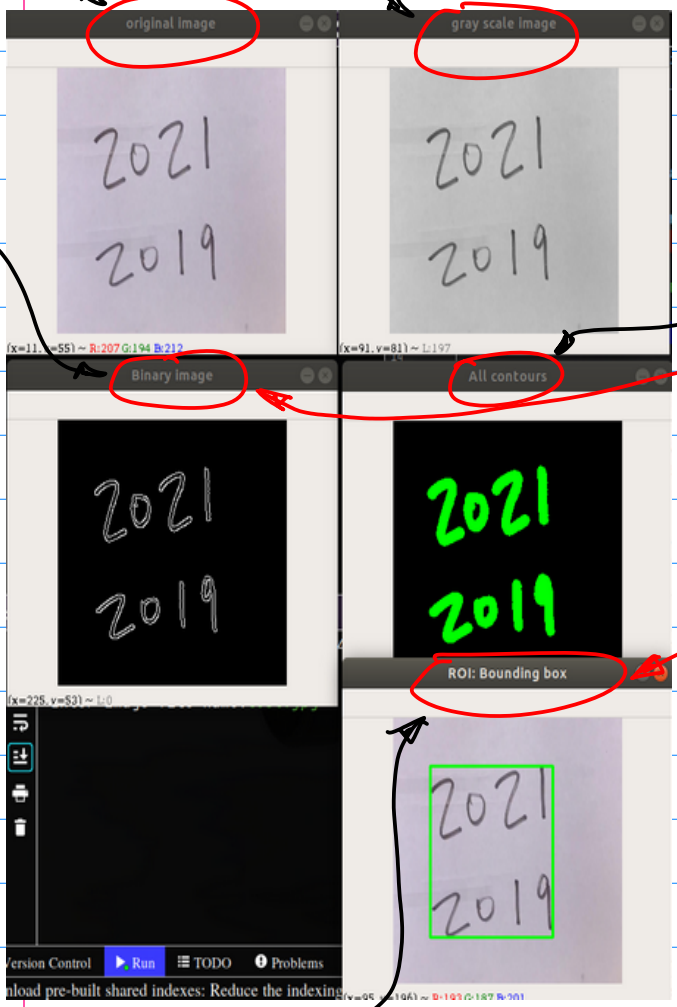


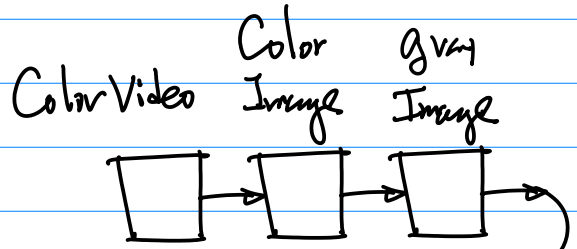
Example: Task to Develop Preprocessing Technique(s) to define ROI. (to localize

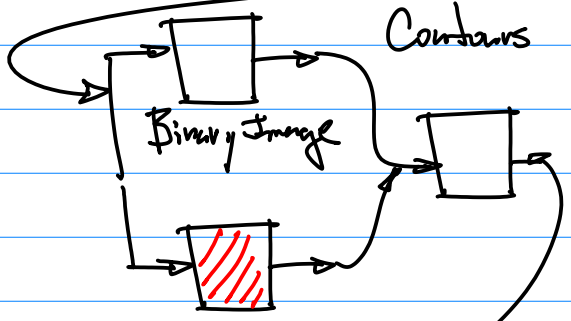
CMPE258 March 1st, 22

22

Note: a. Color Video → Color Image, b. Convert Color Image to gray scale





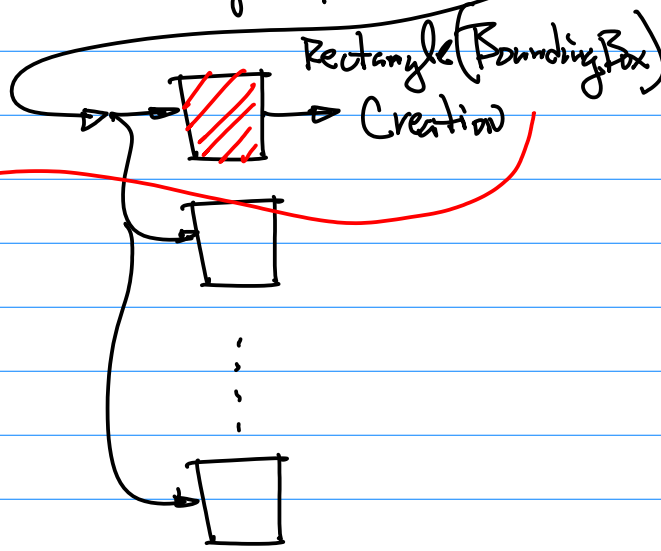


c. Edge map

Canny Edge Detector

d. Contours

Edge Map



e. Bounding Box

Ref: my Python Code

2022S-109b-countours-roi.py

Input Image, Resize it

```
18      img = cv2.imread(image, cv2.IMREAD_COLOR)
19      img = cv2.resize(img, (256, 256))
20      cv2.imshow('original image', img)
```

Convert the image to gray scale image.

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

Note: Memorize the Code!

Continued, Edge Detection, Canny Detector

```
26      thresh = cv2.Canny(imgray, 100, 200)
```

Threshold1 Threshold2

Produced Edge Map, which is a binary Image, B(x, y).

Find Contours

```
29    contours, hierarchy = cv2.findContours(thresh, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
```

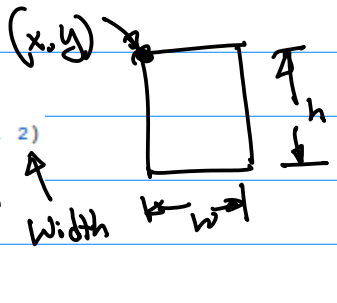
Result — Source image — Define Contour Type

Draw color contour on color image

```
32    thresh1 = cv2.cvtColor(thresh, cv2.COLOR_GRAY2BGR)
33    cv2.drawContours(thresh1, contours, -1, (0, 255, 0), 5) # all contours
```

Compute (x, y) Top left corner of the Bounding Box, w: width, h: height

```
37    x, y, w, h = cv2.boundingRect(thresh)
38    cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)
39    cv2.imshow('ROI: Bounding box', img)
```



Homework March 15, Convolution & Binary Image

Midterm Exam Practice & Exam, March 15 & 22nd

Project April 4th (Monday)

March 8th (Tue)

1. Midterm Exam: March 22nd (Tue)

Format of the midterm:

(1) Zoom Based on-Line Exam;

(2) Close-Book, Close-Notes, But One-page Formula is allowed;

(3) When entering the exam Session, Video has to Be On for the entire Session.

(4) No messaging, No Smartphone use.

(5) your Laptop, Desktop machine is ready, Part of the Exam Questions require the execution of your code.

(6) Have Screen Capture Software tool Ready, Capture Screen, Save as "jpg" format or "png" format for example.

(7) Bring Printer paper to exam to write your Calculation, then use your phone to take photos.

(8) Convert All photos into pdf Document using

online tool, then merge all the files into one pdf Document.

Note: Please control the resolution of your photos, Keep it in 720P (1024x768 pixels) 1-2mByte.

(a) Mark Each page with

a. Your First, Last Name,
b. Cmpe258
c. 4 Digits of SID.

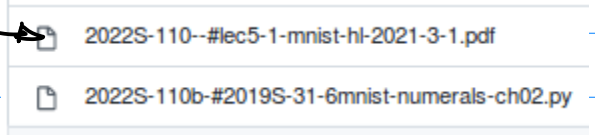
(b) Zip your file,

Name your file:

First_Name+Last Name+4Digists+Cmpe258-mid.zip

Submission to SJSU Canvas.

Ref: from the Class github



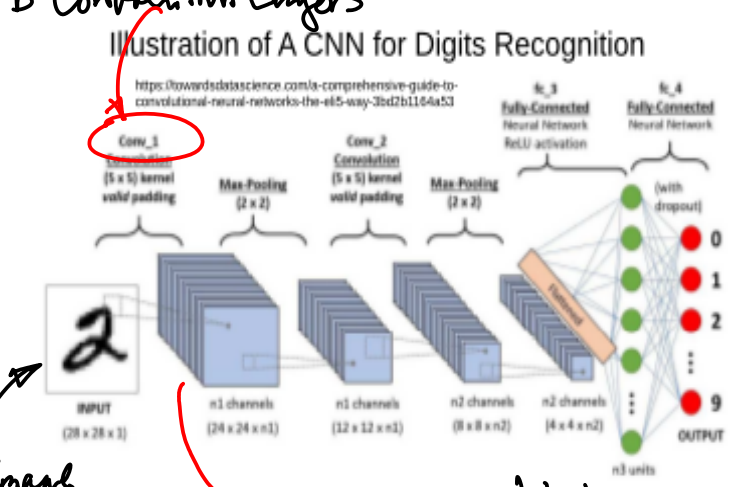
Topics: 1° Convolution Techniques

2° MNIST Neural Network Convolutional

Project Assignment: For Student ID Recognition (Hand Written ID) April 4th (Monday).

Example: Convolution

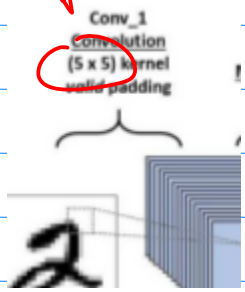
b Convolution Layers



a Image 28x28x1

Fig 1.

Note 1. Kernel ("mask") Size: KxK



Example:

K has to an "odd" Number. 1, 3, 5, ..

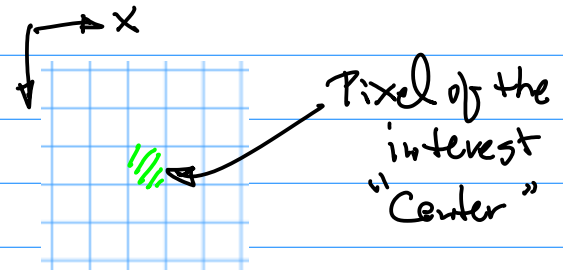


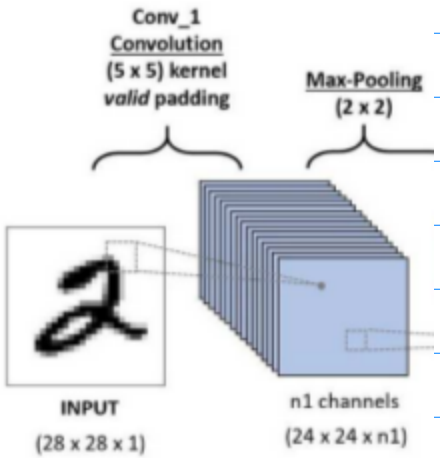
Fig 2

1° Kernel size: K x K = 5 x 5

Center of the kernel, Equal No. of Rows Above & Below the center, Equal No. of col. Left & Right from the center → K = "Odd Number"

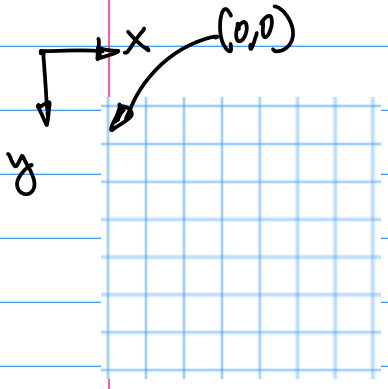
"Un-Biased" Kernel

2° Kernel v.s. given image for Convolution.



Stage i → Convolution with 5x5 Kernel → Stage i+1

Fig. 3a



I(x,y) Image

7x7

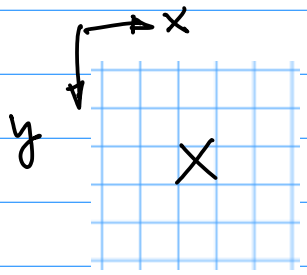
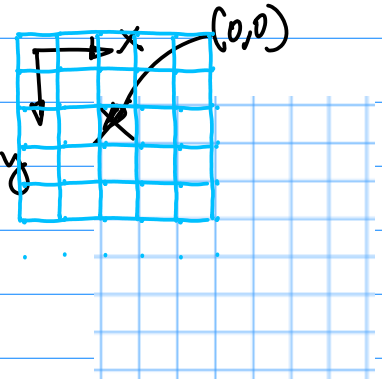


Fig. 3.b



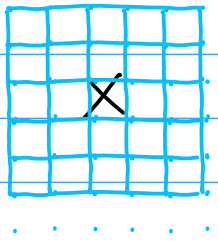


Fig. 3.c

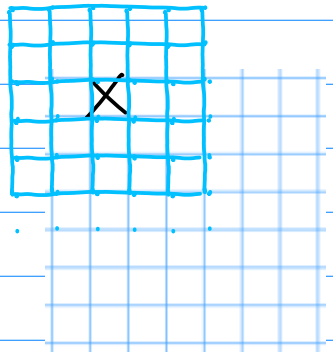


Fig 3d

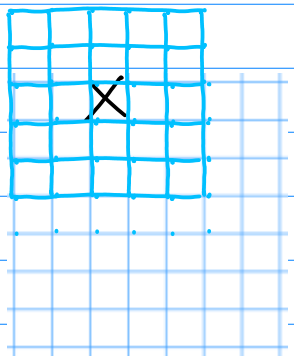


Fig 3e

Kernel movement: Starting at the top left hand corner, then L → R (Left to Right), T → B (Top to Bottom) One pixel at a time.

Note: The Basic Operations for Convolution, e.g., the movement of A Kernel

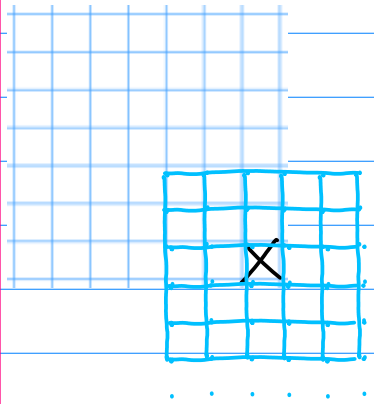


Fig. 3F

$$\iint_R I(x-u, y-v)\, k(u,v)\, du\, dv \quad \ldots (1)$$

Image $I(x,y)$

Image Domain: $m \times n$ Resolution

Kernel, Size $K \times K$

3. Mathematical Operations

Note: Each kernel ($K \times K$) has to have kernel/pixel coefficients

Example: for $K \times K$ kernel, $K=5$, we have $5 \times 5 = 25$ kernel elements, pixels, each kernel element, pixel, has its value for processing, feature extraction purpose.

$$\sum_{v=0}^{N-1} \sum_{u=0}^{M-1} I(x-u, y-v)\, k(u,v) \quad \ldots (2)$$

where $m \times N$: Resolution of a given image.

$$\sum_{v=0}^{N-1} \sum_{u=0}^{M-1} I(x-u, y-v)\, k(u,v) =$$

$$\sum_{v=0}^{N-1} \sum_{u=0}^{M-1} I(u,v)\, k(x-u, y-v) \quad \ldots (3)$$

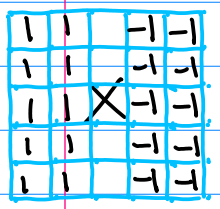
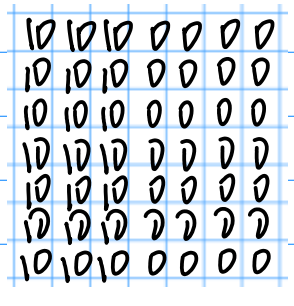


Fig. 4a.



(Edge Component Detection, for a Vertical Boundaries of Objects)

Mathematic Definition of 2D Convolution

Eqn (3):

$k(x-u, y-v)$ v.s. $k(u,v)$

a. $k(u,v)$, b. $k(-u,-v)$, c. $k(1-u, 1-v)$

...

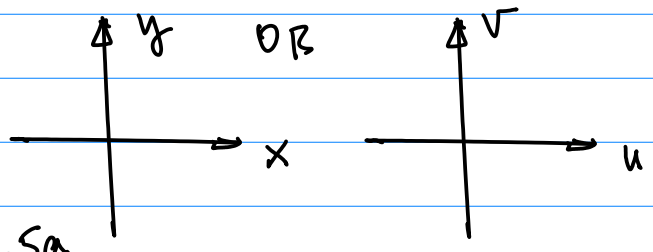
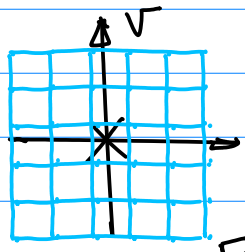


Fig 5a



$K(u,v)$

Fig 5b

From a.

place the kernel on this u-v Coordinate System

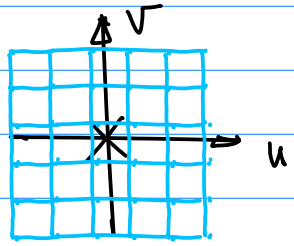
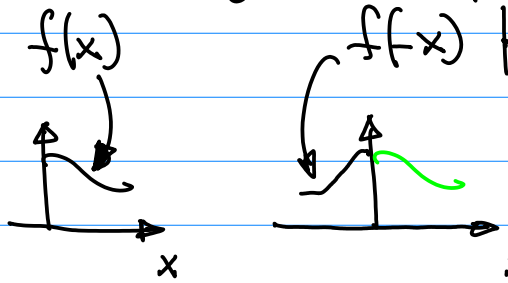


Fig 5c

$K(-u,-v)$

Flip $K(u,v)$ Twice, once along u, once along v. Example:



Note: Kernel is designed with coefficient to make the "flip". $K(1-u, 1-v)$

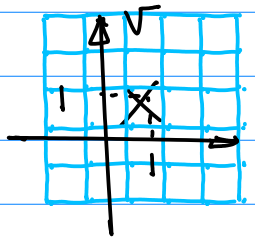


Fig 5d

We can shift Kernel $K(x-u, y-v)$ by (x, y), follow L→R, T→B one pixel @ Time starting at top left corner of the image.

Note:

$I(u,v)\,K(x-u, y-v)$

is Product

Note: From Eqn (3),

$\sum_{v=0}^{N-1} \sum_{u=0}^{m-1}$ Summation of Each Product.

Summary: To Carry out 2D Convolution in Eqn (3), we perform Operations Characterized as "3-Step" Operation, e.g.:

Shift — multiplication — Summation

Example: Given 4x4 Image Below

10	10	0	0
10	10	0	0
10	10	0	0
10	10	0	0

Fig 6a

and a kernel with K=3

1		-1
1	X	-1
1		-1

Fig 6b

Perform 2D Convolution

Sol:

Step 1. Take the kernel, place it at the starting location on the given image

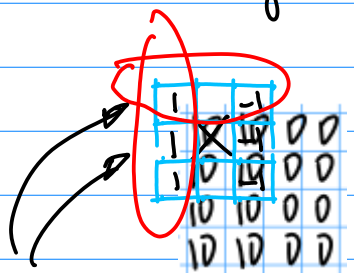
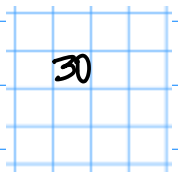


Fig. 7a

have no corresponding image pixels

Note: Always Align the kernel that non of its row or col have no corresponding image pixel.

Convolution Result



Convolution Result

Fig 7b

Carry out multiplications in "L→R, T→B" fashion

1st Row, 1st col. $I(x,y) = I(0,0) = 10$

Kernel coefficient = 1

Product $1 \times 10 = 10$

For the rest of the kernel coefficient

1st Row: $1 \times 10 + 0 \times 10 + (-1) \times 0 = 10$

→ 2nd Row of the kernel

$1 \times 10 + 0 \times 10 + (-1) \times 0 = 10$

3rd Row of the kernel

$1 \times 10 + 0 \times 10 + (-1) \times 0 = 10$

Add 1st, 2nd, 3rd All together

$10 + 10 + 10 = 30$

Continue the Calculation.

Step 2. place kernel to 1 pixel to its Right

1st Row, kernel; $1\times10+0\times0+(-1)\times0=10$

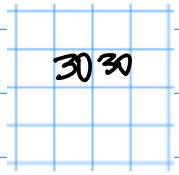
2nd " " : $1\times10+0\times0+(-1)\times0=10$

3rd " " : $1\times10+0\times0+(-1)\times0=10$

Convolution @ this pixel location

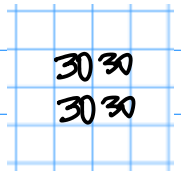
$= 10+10+10 = 30$

Convolution Result



Finally, to finish the Convolution:

Convolution Result



Dimension (Size) of the convolution Result is smaller Due to "Boundary Effect". following the formula Below:

$\frac{k-1}{2}$ Size of the Symmetric Portion (upper or lower; left or Right)

Then, $2\times\frac{k-1}{2}=k-1$ is the dimension for the Rows and col. to Be Reduced from its original size of the image.

Example: $k=3$, $k-1=2$

Original Image 4x4 is to Be Reduced to $(4-2)\times(4-2)=2\times2$ "

Homework Assigment Due Next week March 14. Check github / Canvas to Be posted on Line today.

March 15 (Tue)

Topics: 1° Convolutional Neural Network — MNIST; 2° Project for Handwritten 5x5 Recognition. (Due April 11th — Monday. 11:59PM.)

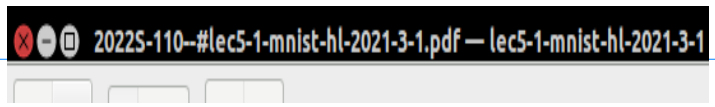


Illustration of A CNN for Digits Recognition

Reference Only

See Source Code

https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

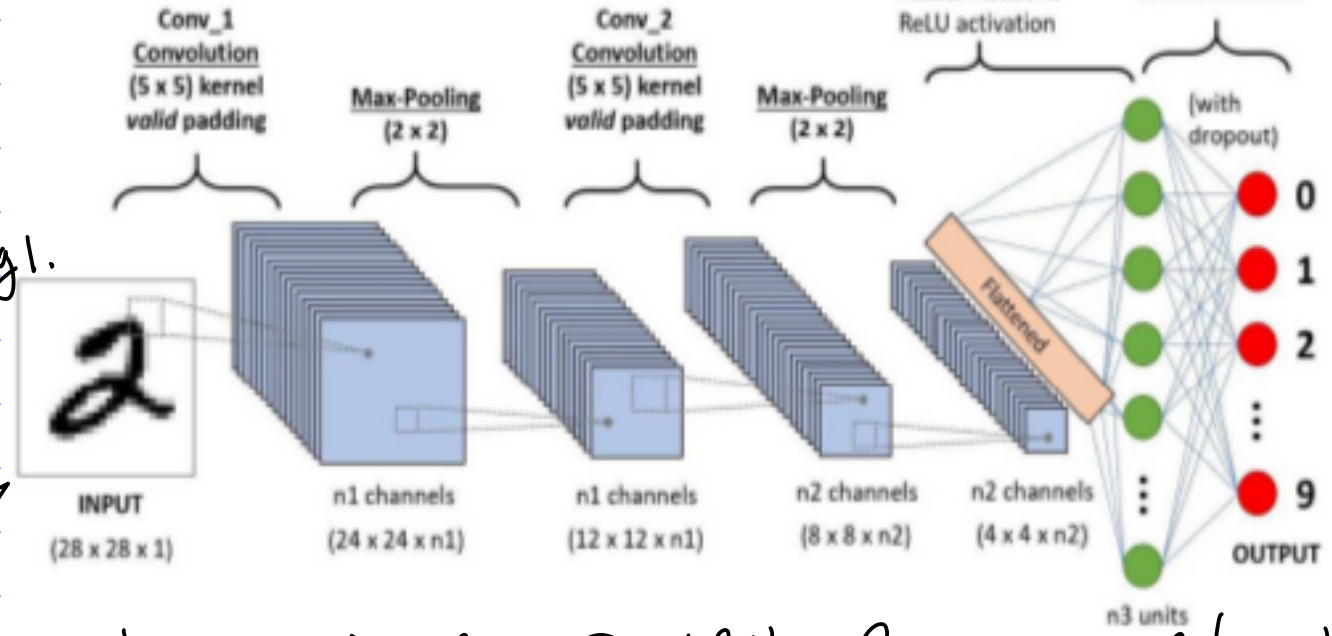


Fig 1.

a. Preprocessing Result has to be resize w/o Altering the Aspect

b. 1st 2D Conv. → Denoted it as C_1 → 32 kernels/convolutions

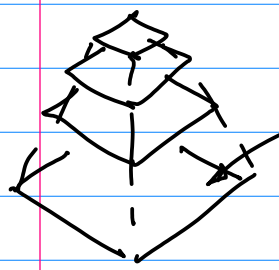
Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 26, 26, 32)	320

Input Image size 28x28
Output Convolution Result 26x26 → K=3 Kernel

c. MaxPooling → Denoted M_1
1. 2x2 kernel size
2. Find max
Use it as an output

15	34.1
2	0.1

$\max\{15, 34.1, 2, 0.1\} = 34.1$ → 34.1



Layers.

d. Verify the Dimension of Output Layer ($\frac{26}{2} = 13$ for x, y Dimen)

Layer (type)	Output Shape	Param #
max_pooling2d_1 (MaxPooling2	(None, 13, 13, 32)	0

e. Convolution Layer + Maxpooling : $C_2 M_2$ (1st: $C_1 M_1$)

Layer (type)	Output Shape	Param #
conv2d_2 (Conv2D)	(None, 11, 11, 64)	18496
max_pooling2d_2 (MaxPooling2	(None, 5, 5, 64)	0

Fig. 2

Note: C_3 (3rd Convolution Layer) No. of Kernels/Convolutions

conv2d_3 (Conv2D)	(None, 3, 3, 64)		36928

conv2d_3 (Conv2D)	(None, 3, 3, 64)	
flatten_1 (Flatten)	(None, 576)	

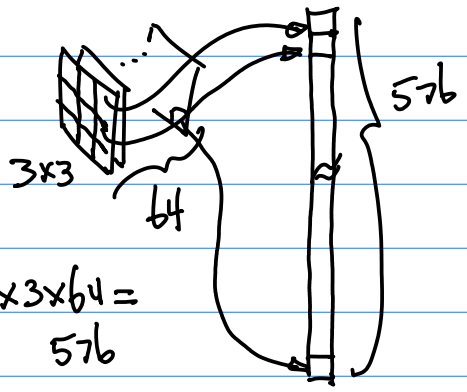


Fig 3

Note: Dense Layer (For Feedforward NN) D_1 and D_2

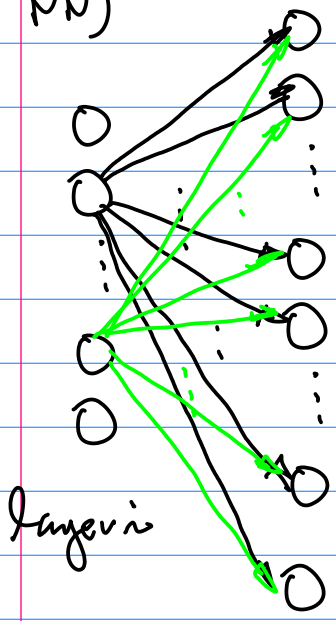
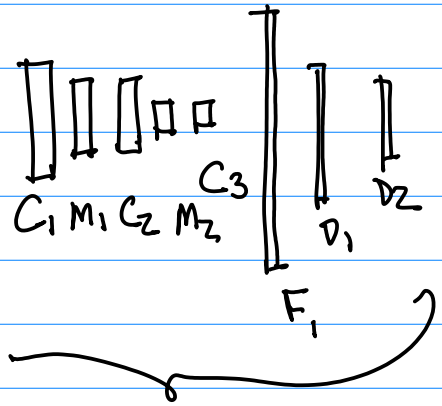


Fig. 4



Architecture for MNIST

Output, Define Activation function, to allow

(1) Output (10 of them) to match Digits 0, 1, ⋯, 9;

(2) "Probability" sort of, for each Output, the highest value will be selected for the corresponding Digit. Hence, we have,

$$f(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad \cdots (1)$$

where

$$\sum_{j=1}^{K} e^{z_j}\Big|_{k=10} = e^{z_1} + e^{z_2} + \cdots + e^{z_{10}}$$

10 Digits

Therefore

$$0 < \frac{e^{z_j}}{e^{z_1} + e^{z_2} + \cdots + e^{z_{10}}} < 1$$

in addition,

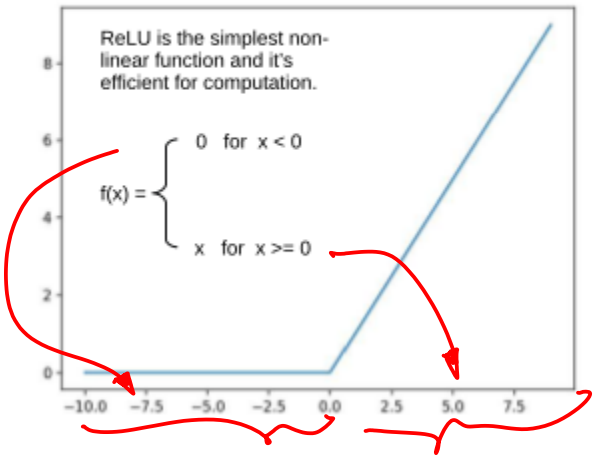
$$\frac{e^{z_1}}{e^{z_1}+e^{z_2}+\cdots+e^{z_{10}}}+\frac{e^{z_2}}{e^{z_1}+e^{z_2}+\cdots+e^{z_{10}}}+\cdots+\frac{e^{z_{10}}}{e^{z_1}+e^{z_2}+\cdots+e^{z_{10}}}$$

$$=\frac{e^{z_1}+e^{z_2}+\cdots+e^{z_{10}}}{e^{z_1}+e^{z_2}+\cdots+e^{z_{10}}}=1.$$ Softmax function

Activation Function: ReLu (Non-Linear)

Rectified Linear Activation Function

https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/



Transfer function, h, $h(\cdot)$, $h(\sum x_i w_i + b)$,

Activation Function f, or $f(h(\cdot))$ or $f(\sum x_i w_i + b)$

- Softmax,
- ReLu,
- Sigmoid ~

Code Walk-Through

```
#-------------save trained model
import h5py
model.save('harryTest.h5')
#-end
```

Building/Design Feed forward NN With Basic Building Blocks.

Block Diagram

- "Toy" Example
- MNIST

Midterm Review:

1. Scope of the Exam: Upto Today's Lecture Material;
2. 3rd Questions
   (1) Basic Concepts, Basic Building Blocks, Input (Vector X), Weight (Vector W)

Loss function Definition.

MSE $\frac{1}{N}\sum_{j=1}^{N}(g^j - h^j)^2$

$\frac{1}{M}\frac{1}{N}\sum_{j=1}^{N}\sum_{i=1}^{M}(g_i^j - h_i^j)^2$

gradient $\nabla L(w_{ij})$

(2) Practical Implementation (Coding)
Laptop/Desktop Ready to Execute your Program.

(3) Calculation, Convolution, Binary Image Processing.
Area, $\bar{x}$, $\bar{y}$, higher order terms

$$\bar{x} = \frac{\iint x B(x,y)\,dx\,dy}{\iint B(x,y)\,dx\,dy}$$

$$\frac{\iint (x-\bar{x})^2 B(x,y)\,dx\,dy}{\iint B(x,y)\,dx\,dy}, \text{ etc.}$$

Topics: Visit Preprocessing Technique.

Tools in Preprocessing → MNIST (28, 28, 1)
(Resolution, gray)

↓ CANNY Edge Thresholding → Binary Image → Contours

Contours → TREE structure or External

→ Localization. ROI

3. One Hour Exam plus 15 min. to prepare file uploading.

4. Close Book/Close Notes, One page Formula is allowed, just Formula, No Writting or Verbo Description on Formula sheet.

5. Video has to be on During the entire Session.

6. After Exam, 4:45 ~ 4:50 Back to class.

March 22nd (Tue)

Contour : TREE

l v_prev).

eme outer contours. It sets
y[i][3]=-1 for all the contours.

CV_RETR_EXTERNAL

first = c0

CV_RETR_CCOMP

first = c01000↔c01001↔c010↔c000↔c0

h0100 h0000 h01↔h00

CV_RETR_LIST

first = c01000↔c01001↔h0100↔c010↔c000↔h01↔h00↔c0

retrieves all contours without any hierarchical relationships.

CV_RETR_TREE

first = c0

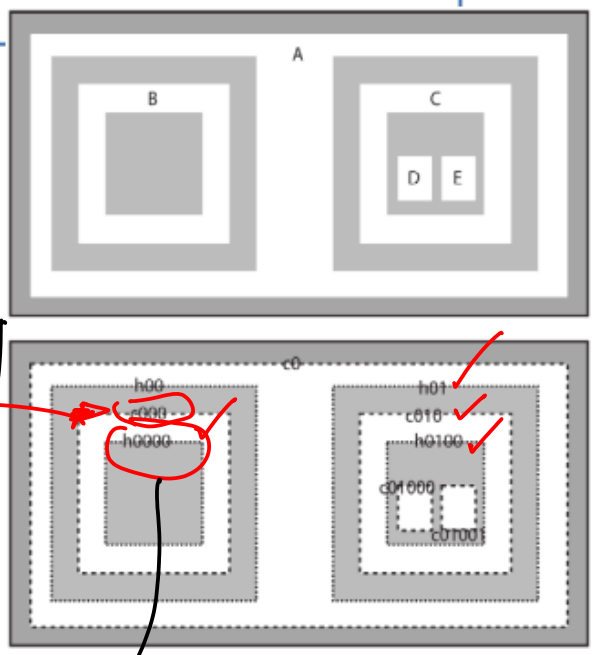
h00↔h01

c000 c010

h0000 h0100

c01000↔c01001

1st 2nd



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

Fig.1

Step1.

Background : Black

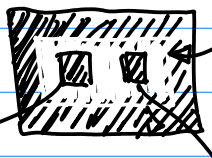
Object : White

Fig.2

Cϕ

Contour No. Enumeration of the Contour

Step2.



Cϕ

hϕ:ϕ

h : hole

hϕ:1

Fig 3

Step3.

See Fig.1. Place One Object inside

Hϕϕ

Cϕϕ:ϕ

its Parent Contour Number

1st Object

Note: To understand the tree structure, to be able to generate Contour Tree from a given Pattern, or from a contour tree to illustrate Contour Objects.

h+ No. (1st part: Reference to the No. of the Contour it belongs)

2nd part: Enumeration of the Contour)

Step4. Continue this process, we have placed a hole, hence

Hϕϕϕ:ϕ

Localization of ROI