Jan 28, 2021

Welcome To CMPE258 !

First Day of the Class

Harry LI, github/hualili/opencv/deep-Learning-2020S
20-2021S Email: hua.li@sjsu.edu

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

Zoom Based

(650) 400-1116 Text Only

On-Line Material { github/hualili
CANVAS { Homework Assignment
Collect Submission of Homework

HANDS-ON:

Programming. Python

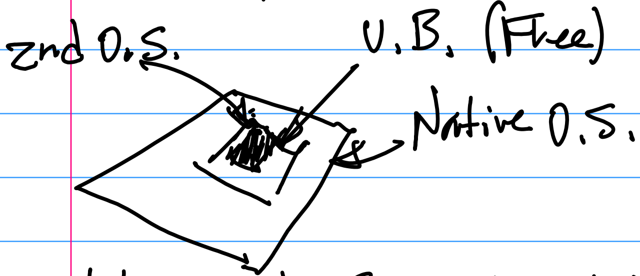
① OpenCV, T.F. (Tensor Flow) Keras API

google Colab

Action 1: Installation of OpenCV. Version 4.2

Note: Use/Adopt Linux Ubantu 18.04

Virtual Box



Note: Python 3. Python Virtual Enviroment

3 major Areas { ① Handwritten Numerals Recognition. MNIST { C.V. R.O.I. Deep NN
② TimeSeries Prediction LSTM
↑
Subjects

③ FaceNet, ResNet
④ Deep Reinforcement Learning
DRL
Action — Policy — Reward

Work to be done:

1. Programming / Code Development from Repo.

2. Write/Submit PseudoCode (Brief Summary) Report 1 page
Note: Post a Sample on github Latex

3. Homework Submission

4 Submission (Including Semester Long team Project).

Action 2: Form 4-person Team By Feb 14 week; work has to Individual / Encourage team Discussion.

Grading Policy: { Mid: 30%
Homework: 30%
Final: 40%

* Introduction

Neural Nets
Biological System
Human Brain
Neurons (cells)

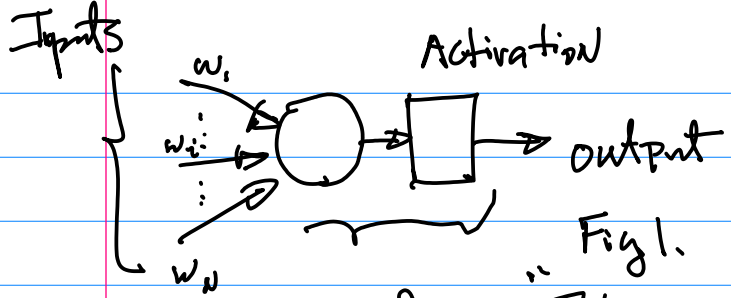


Fig 1.

Prof. Carvard Mead — "Silicon Brain"
Intel Processor ~1992-94, ↓ Father of VLSI"
~1994-95
↓
Autonomous System
↓
2005-2006
Standford Group + Google
Self Driving Market Hype
↓
2013 Alex Net (Deep convolutional Neural Network)
NVDA GPU Architecture

Recently: FaceNet, ResNet
Time Series Analysis (LSTM)
Deep Reinforcement

OpenCV
1997

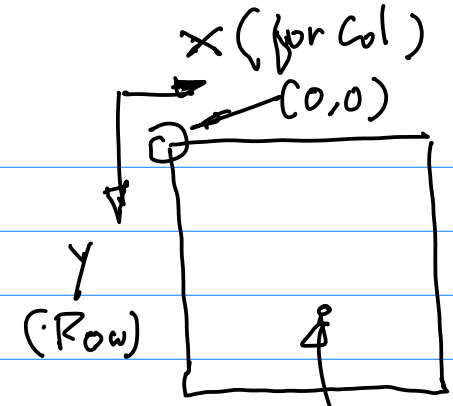
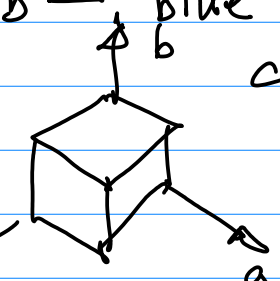


Fig.3

Note:
1° "Scanning" ~ From L to R, top to B
"L2R, T2B"
2° Resolution 1024x768
M x N
Resolution
No. of column
No. of Row
M x N → 1024x768
M N
Colum Row
X y

3° Color Image Vector
Color Space (r, g, b)
Primitive Color
r — red;
g — green;
b — blue



color Cube

Fig.4

(0,0,0) Black (1,1,1) White
grey scale as Traveling from (0,0,0) towards (1,1,1)

Computer Vision

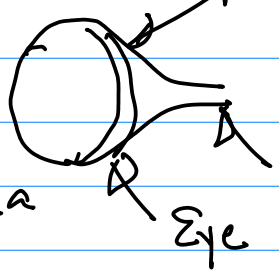


Fig.2a

multiple Layers of Photo Receptors
~10 layers.
110 ~ 120 million P.R.
Optic N. (Brain cells)
~ 1 million

Image Pyramid

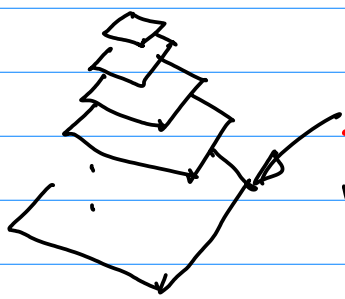


Fig.2b

Highest Red $(r,g,b) = (1,g,b)$

" Green $(r,g,b) = (r,1,b)$

" Blue $(r,g,b) = (r,g,1)$

$I(x,y)$ → Intensity in terms of (r,g,b)

① Location On the Image Plane

$I(100, 201)$

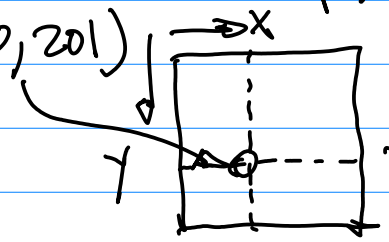


Fig 5

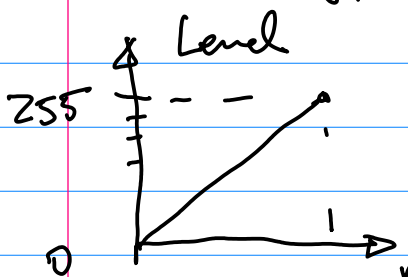
r: 8 bit [0, 255] $(2^8 = 256)$

g, b; 8 bit, " "

Quantization Level

Pixel Depth (BPP: Bit Per Pixel)

For 8 bit Grayscale Image



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

r,g,b. Convert Color Image to Grayscale Image

Note: 1° Installation of Python, OpenCV, and T.F.

Python { 2.7
3.5 or higher

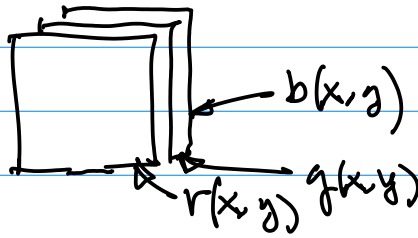
Anaconda — Python Distribution → Python Virtual Environment

OpenCV 4.2 Version 3.0 or higher

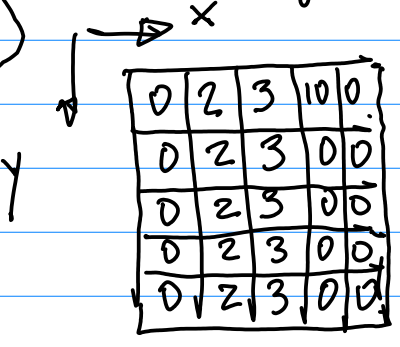
T.F. Keras (API)

Given a digital image

$I(x,y) \rightarrow r(x,y), g(x,y), b(x,y)$



Example: Suppose an $I(x,y)$ is given below,



Action 3. Enable OpenCV { colab / Locally (Install)

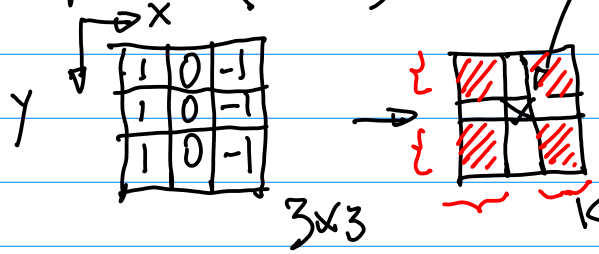
.... Display A "Jpg" Color Image → From Your Smart

Feb 4th, CMPE 258

Today's Topics: 1° Convolution, Two Dimensional Convolution; 2° Intro to Neural Network

Performan Convolution on $I(x,y)$

Convolution kernel (mask)



Ref: [_2020S_ ... 2DConvolution

From 1D Case, given a function $f(x)$ and a kernel $g(x)$

$$\int_{\Omega} f(u)\, g(x-u)\, du \quad \dots (1)$$

$$\sum_{i \in \Omega} f(i)\, g(x-i) \quad \dots (2)$$

Now, 2D Case

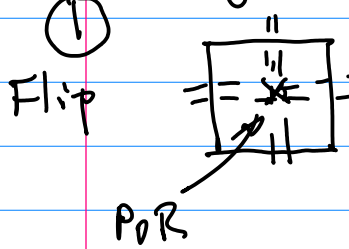
$$\iint_{\phi} f(u,v)\, g(x-u, y-v)\, du\, dv \quad \dots (3)$$

$$\sum_{j} \sum_{i} f(i,j)\, g(x-i, y-j) \quad \dots (4)$$

Image, Kernel, Index of pixel location

Kernel $g(x,y) \rightarrow g(-x,-y)$

① Flip



a) Flips the Kernel

b) $g(x,y) = g(-x,-y)$ if Symmetric

② $f(i,j)\, g(x-i, y-j)$ product

③ shift

$g(x-i, y-j)$

$g(1-i, 1-j)$

$g(2-i, 1-j)$

$g(2-i, 2-j)$

and so on ...

Summary: 2D convolution consists of shift-product-summation

Compute 2D convolution:

Step 1. Place the kernel @ the initial condition, e.g. the top left Hand corner of the image, in such a way its boundary rows and columns are aligned with the image boundary row and column.

$f(0,0)*1 + f(1,0)*0 +$
$f(2,0)*(-1) +$

$f(0,1)*1 + f(1,1)*0 +$
$f(2,1)*(-1) +$

$f(0,2)*1 + f(1,2)*0 + f(2,2)*(-1)$

$= 0*1 + 2*0 + 3*(-1) +$
$0*1 + 2*0 + 3*(-1) +$
$0*1 + 2*0 + 3*(-1)$

$= -3 - 3 - 3 = -9$

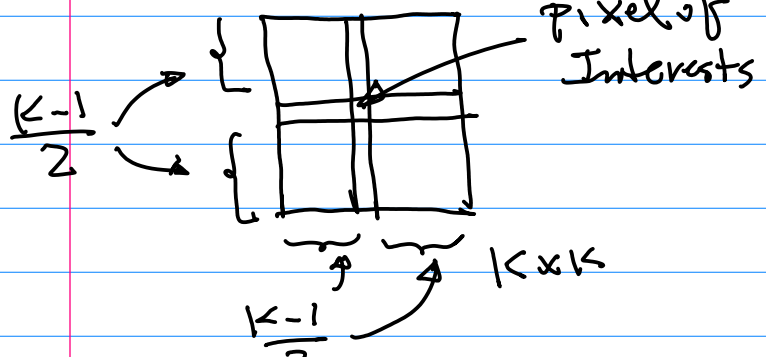
Note: This convolution resulted in a new processed image plane $I_{New}(x,y)$ whose Rows is less than the original image,

No. of Rows = Original No. of Rows − 2

for 3x3 kernel, for KxK kernel (K is odd Number), in this case

No. of Rows Reduced by $2*\left(\frac{K-1}{2}\right)$

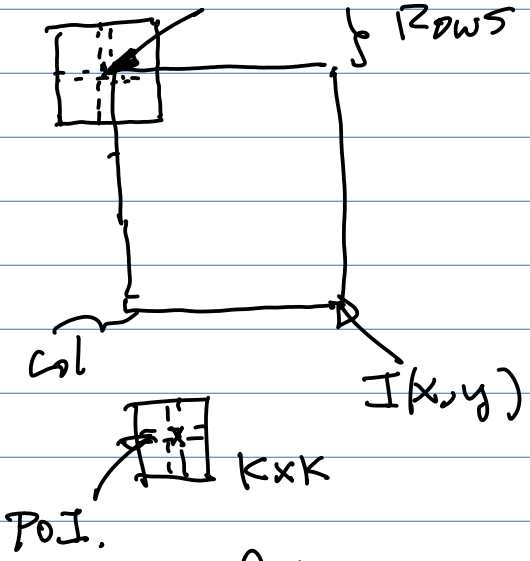
$= K-1$.



Homework (Exercise — No Submission)

Based on the given I(x,y) Image from PPT @ github, perform hand Calculation of 2D convolution

Note follow the example in Class.



Consider Neural Networks.

Supervised Learning

Reference: ZR-20215-Z

Example: $\bar{x}$

Input $(x_1, x_2, \cdots, x_n)$ $\bar{w}$

Weights $(w_1, w_2, \cdots, w_n)$

$\bar{x} \cdot \bar{w} = (x_1, x_2, \cdots, x_n) \cdot (w_1, w_2, \cdots, w_n)$

$= x_1w_1 + x_2w_2 + \cdots + x_iw_i + \cdots + x_nw_2$

$= \sum_{i=1}^{n} x_i w_i \quad \cdots (1)$

Define Transfer function $f(\cdot)$ as follows

$$f(\bar{x}) = \begin{cases} 1 & \sum_{i=1}^{n} x_i w_i + b > 0 \\ 0 & \text{O/W} \end{cases} \quad \cdots (2)$$

Feb 11,

Today's Topics: 1° Introduction to NN; 2° Coding (Python) Examples.

Action 1: Form 4-person Team. Submit your team member information By Friday 5:00 pm.

Subject: CmpE258 First Last Name of the group coordinator

First, Last Name, Last 4 Digits of your SID; E-mail Contact information;

Homework 1: Due A week from Today; Submission to CANVAS.

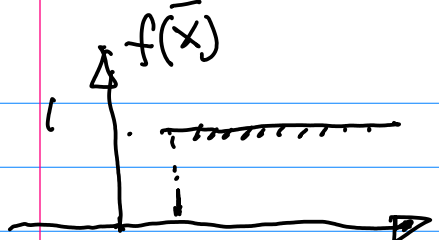


Fig 1.

Consider a Simple Feedforward NN

Example:

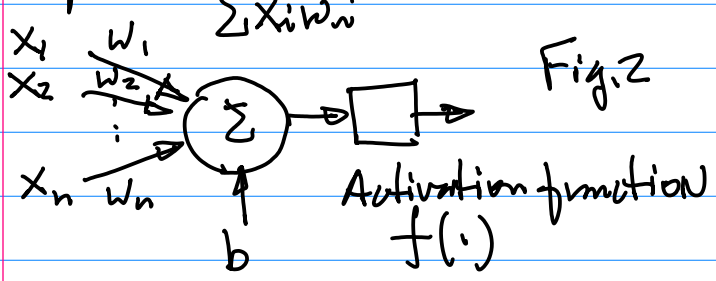


Fig.2

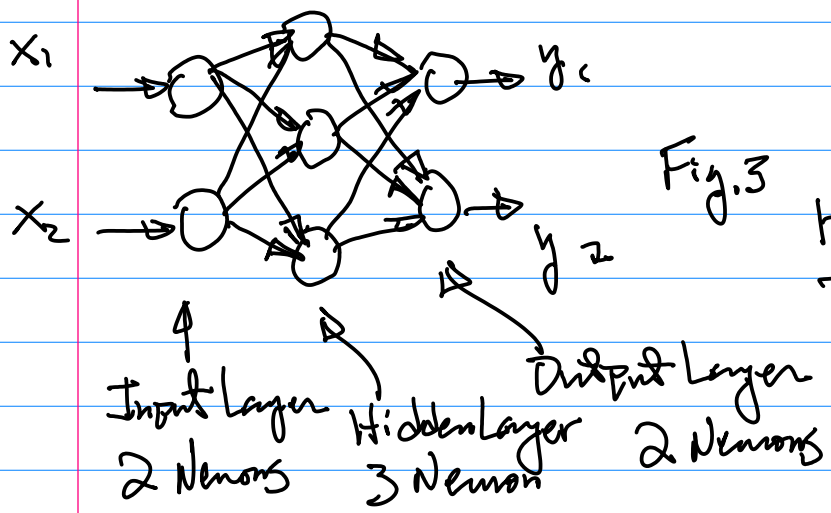


Fig.3

Consider Dense Feedforward NN From the PPT.

1. A Single Neuron { Block Diagram, Notation

$\bar{x} = (x_1, \cdots, x_2)$

$y = f(\bar{x}) = f(x_1, x_2, \cdots, x_n)$

Activation function

FeedForward

Have Training Dataset ϕ or Ω

Two Classes C_1, C_2 Representing 2 Patterns

Feature Vector $X = (x_1, x_2)$, up to N_1 of them for C_1 upto N_2 for C_2

N_1, N_2 do not have to be equal;

Training (Learning)

$$w_i^+ = w_i^- + \Delta w_i = w_i^- + \eta(d-y)x_i \quad \cdots(3)$$

w_i^+ updated, w_i^- Current, η Rate, d desired Output, y Actual Output

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

3. Python Implementation
Create A template for your program header,

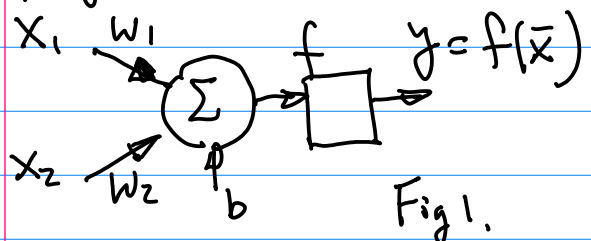


Fig 1.

$X_1 = 2, X_2 = 3, W_1 = 0, W_2 = 1.$
$b = 4$, $f(x) = \text{Sigmoid}(x)$

4. Feedforward NN

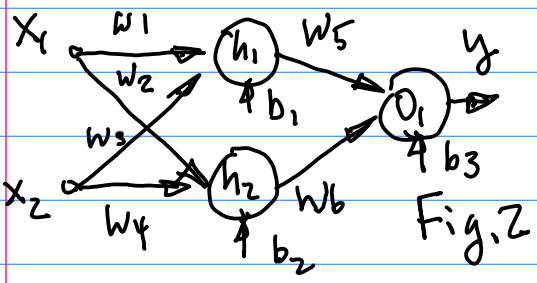
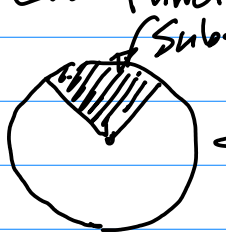


Fig. 2

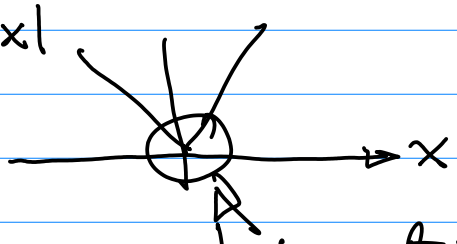
5. Data set → Pre-processing to allow Activation function to Better handle the input values

Supervised Learning
Data → Its Categories Known
Labeling Images/Videos → Annotation.
ImageNet ~1.4 million Test Images

6. Loss Function → Objective Function



y_{true}, y_{pred} — Notation & meaning
gives the information for which Category the data belongs to
→ use squared difference to handle potential error (Loss)
Cancellations due to opposite Signs
Note: Be Careful Not Absolute value!



No derivatives at this point.

8. Define Loss function
↓
Behavior of Loss Function → minize the Loss/error function in the negative direction of the gradient
Gradient gives the fastest increase of the error

9, Denote the Loss as

$$L(W,b) = \sum (y_{True} - y_{Pred})^2 \cdots (2)$$

L is a function of W and b

$b = (b_1, b_2, \cdots, b_M)$

$W = (w_1, w_2, \cdots, w_N)$

$M \neq N$

∇

θ