CMPE259

F22, HL

1/

August 23 (Tue)

First Day of the Class

1. Organizational meeting

"Green Sheet"

Repo: github/hualili/opencv/deep-learning-2022s/

Email: hua.li@sjsu.edu

(650) 400-1116 Cellphone for Text message Only.

Office Hours: M.W. On Zoom. (See Syllabus for the Zoom Link)

2 Software Tools:

Anaconda — Install it By the end of this week;

TensorFlow, TF 2.0

OpenCV.

3. Prerequisites: CMPE255 & CMPE257

Homework: To upload a copy of your un-official transcript to show the required courses satisfied. On CANVAS.

4. Textbook. Deep Learning with Python. Keras (API) for TF.

Robot Vision Book By Horn. (Heavy Theoretical Book, Good Reference for OpenCV Algorithms. Good Theoretical Foundations)

5. Projects { Mandatory Assigned Project
{ Team Project (Mandatory)

4-Person Team. Presentation By the End of the Semester.

August 23 (Wed)

Note: 1. The lecture note will be posted on to the github.

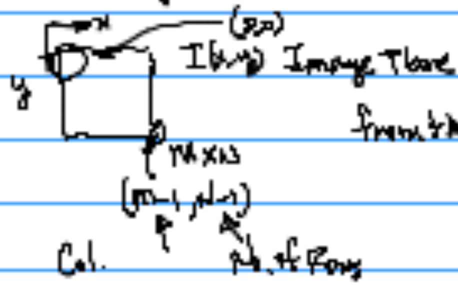
2. Zoom Recording will be posted on the the github.

Homework: By A week from today 1. Anaconda Installation; 2. OpenCV Installation. Submission On Canvas. Jpg/Png Image → pdf → zip

2 pts.

Example: (github: 2022F-103-...)

MNIST Architecture for Handwritten Digits Recognition.



Single channel "gray scale" Image from the PPT. 28x28x1 → Square Image

Gray scale image → 1 Channel → 8 bit → [0, 255]

First Layer of the MNIST Architecture

n1 Channel/plane of the 1st Convolution layer C1

Next, pooling → Reduction of Resolution → 2x2 becomes 1

M1

From the Architecture diagram:

C1 M1 → C2 M2 → Flatten → FFNN (FF)

To generalize the quick inspection of the the CNNs, we have to investigate the Behavior of Each Single Neuron as the Basic Building Block.

CmpE258

Aug. 25, 22

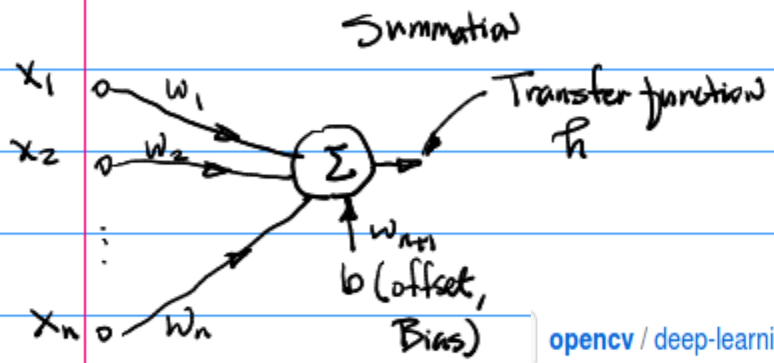
August 30. Tue

2/

Ref. 1

https://github.com/hualili/opencv/blob/master/deep-learning-2022s/2022F-103b-NN-Intro-Python-v5-2022-8-25.pdf

opencv / deep-learning-2022s / 2022S-103a-notation-neuro-loss-function-2022-2-8.pdf



Input/Excitation in Vector Form: $X=(x_1,x_2,\cdots,x_n)$ ...(1)

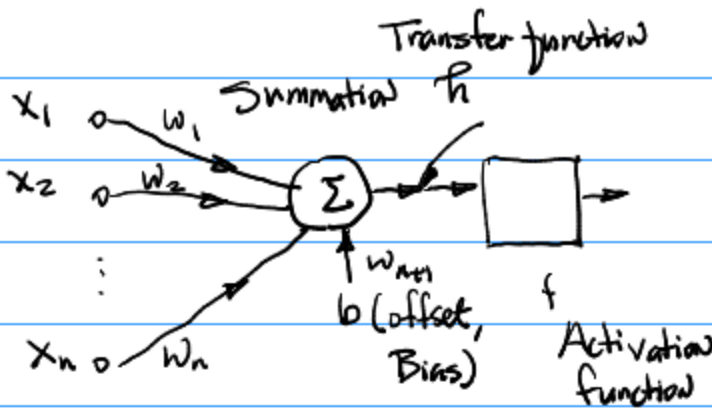
Weights, links each excitation to the Neuron

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

$$x_1w_1+x_2w_2+\dots+x_iw_i+\cdots+x_nw_n+bw_{n+1}=h$$

$$\sum_{i=1}^{n} x_iw_i+bw_{n+1}=h(x_iw_i) \text{ or Simply } h(XW) \quad \cdots (3)$$

$$h(XW;b),\ h(X,W)$$



$$f(h(XW)) = f\left(\sum_{i=1}^{n} x_iw_i+bw_{n+1}\right) \quad \cdots (4)$$

$$h(XW)=\sum_{i=1}^{n} x_iw_i+bw_{n+1}$$

Examples of Different Activation functions include RELU. A piecewise Linear.

Note: Be Able to Build A Single Neuron per a technical specification, such as well, Activation $f(\cdot)$, Draw a Block

Ref 2. Code

https://github.com/hualili/opencv/blob/master/deep-learning-2022s/2022S-110b-%232019S-31-6mnist-numerals-ch02.py

Example:

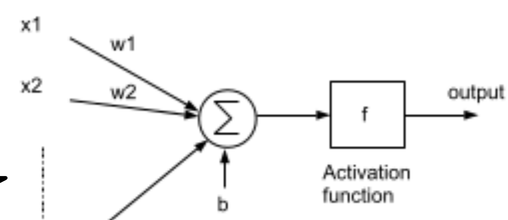


Fig. 1

$(x_1,x_2,\cdots,x_n)$ Feature Vector with Dimension N.

$$h=\sum_{i=1}^{N} w_ix_i = W\cdot X+b \qquad (11)$$

Transfer function $h(\cdot)$.

$$w_{n+1}b=b'$$

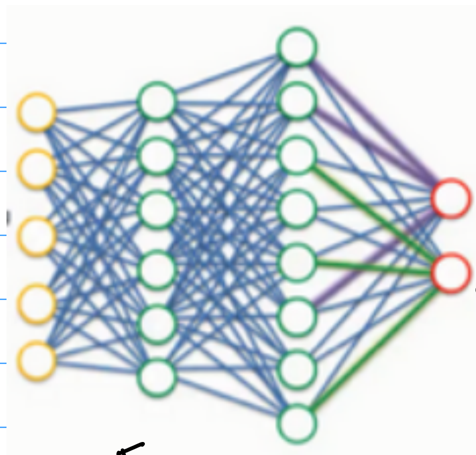
$$y=f\left(\sum_{i=1}^{N} w_ix_i = W\cdot X+b\right). \qquad (17)$$

Activation function. Its output is the Response of the Neuron.

Consider the output of the Neuron y from Eqn(1).

Output of A Single Neuron.

For multiple Neuron Output, see Fig.2



$\nearrow y_1$

$\nwarrow y_2$

Fig.2

y_i ...(1)

Subscript: $i=1,2$

No. of Output at the Output Layer.

y_i, $i=1,2,\cdots,M$.

In practical Application,

y_i^j ...(2)

Superscript

$j=1,2,\cdots,P$ No. of Experiments performed, Training Performed.

Look at the Concept & Definition of Loss function.

Mathematically To Compare a Neural Network Output (Single Neuron Output)

function f. function g

Comparision of the Similarity or diffence between f and g.

$$\begin{cases} f-g \\ f/g \end{cases}$$

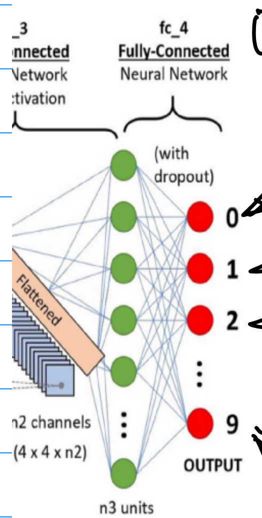
Difference Between Two Functions.

Take this Approach to define Loss function,

$$y - \hat{y} \quad \cdots (3)$$

Ground Truth. Output (prediction) from the Neuron

Outputs



y_0, y_1, y_2, $\vdots$, y_9 $\Big\}$ $\hat{y}_i$

$i=0,1,2,\cdots,9$

Fig 3.

$y_i - \hat{y}_i$ ... (4-a)

To measure All the output for Each Training/Experiment

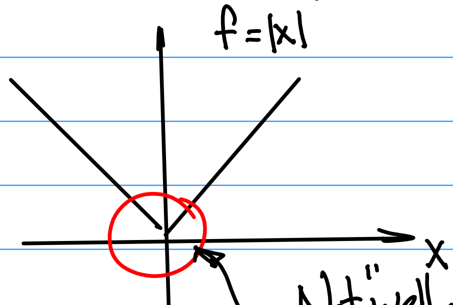
$$\sum_{i=0}^{q} (y_i - \hat{y}_i) \quad \ldots (4\text{-}b)$$

Expand this to Experiment/Training up to "P" Times

$$\sum_{j=1}^{P}\left[\sum_{i=0}^{q}\underbrace{(y_i^j - \hat{y}_i^j)}\right] \quad \ldots (4\text{-}c)$$

Note: Eqn(4-c) may Lead to positive & Negative Terms Cancellation.

Fix: Absolute Value ? → Squared Instead.



Not "well-behaved"!

"Well-Behaved" System (Function) → derivative/Partial Derivative up to order "K".

!

$$L = \sum_{j=1}^{P}\left[\sum_{i=0}^{q}(y_i^j - \hat{y}_i^j)\right]^2 \quad \ldots (4\text{-}d)$$

→ J, or Φ

$$L_{total} = \frac{1}{2}\sum_{j=1}^{P}(\tilde{y}^j - y^j)^2. \qquad (23)$$

For a Single Neuron @ the Output Layer

Training Based "Steepest Gradient Descent" SGD.

Example: Given A function $f(x) = x^2$, Find its Derivative

$$\frac{df}{dx} = 2 \cdot x$$

To get rid of Coefficient from the derivative, Let's define $f(x) \triangleq \frac{1}{2}x^2$

then, $\frac{d}{dx}f(x) = \frac{1}{2}\cdot 2 \cdot x = x$

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