Basic DL

tensorflow

Deep learning

= (Image / video)

Deep learning unstructure

ANN :- Unstr → DL → Code

CNN :- Computer vision :- Image classification

Object Detection

" " segmentation

" " tracking

RNN :- (NLP)

(voice / sound / text)

BCR

each and every ANN

Machine learning.

Structure

① Reg.

② Class.

= H W Rgb

180	50	22
150	60	18
200	70	19
120	90	20

matrix

ANN :-

=

(Regression or Classification)

vector → input layer

(h number of) Hidden layer

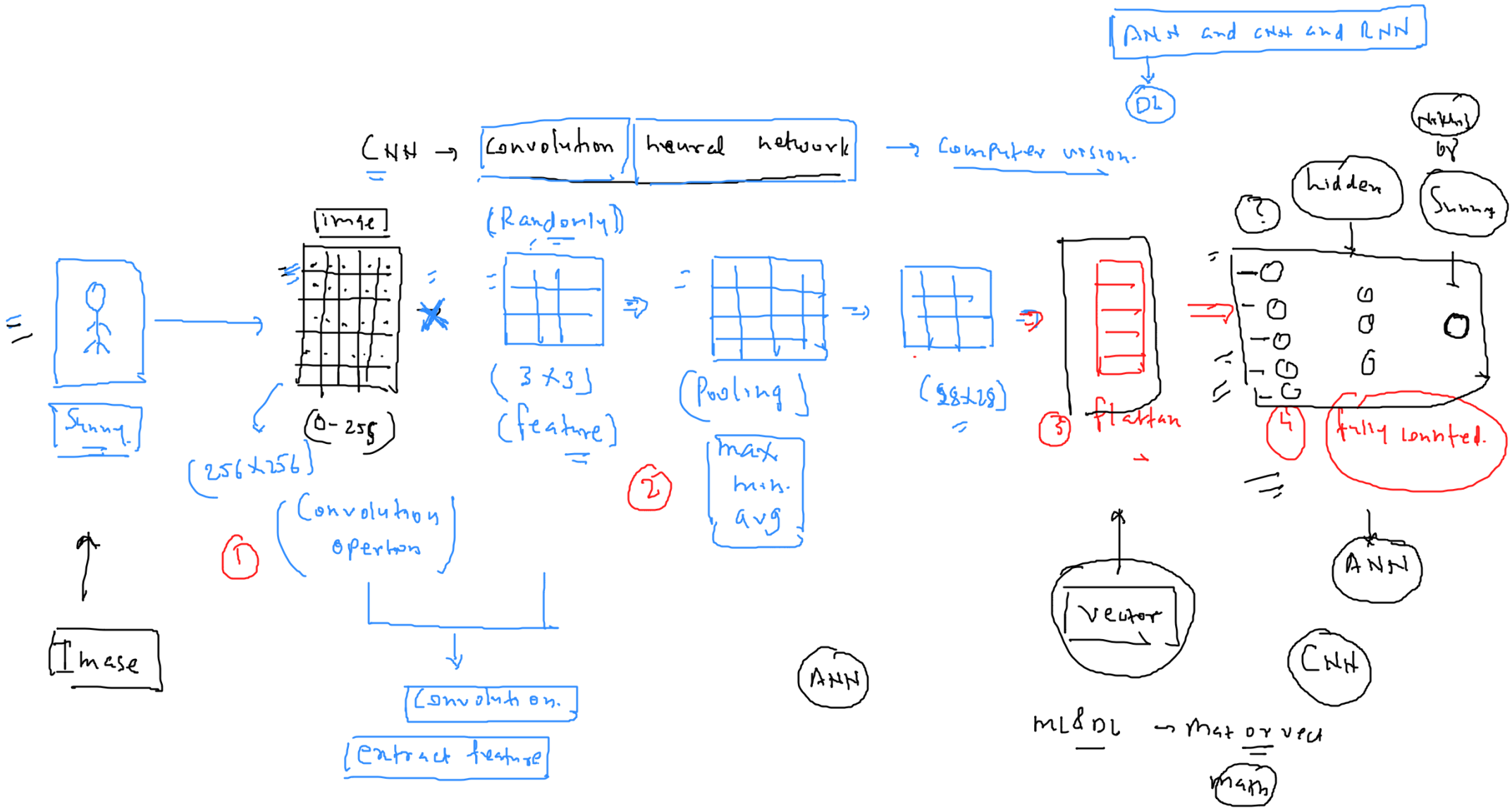
(Reg.)(Class) Output layer

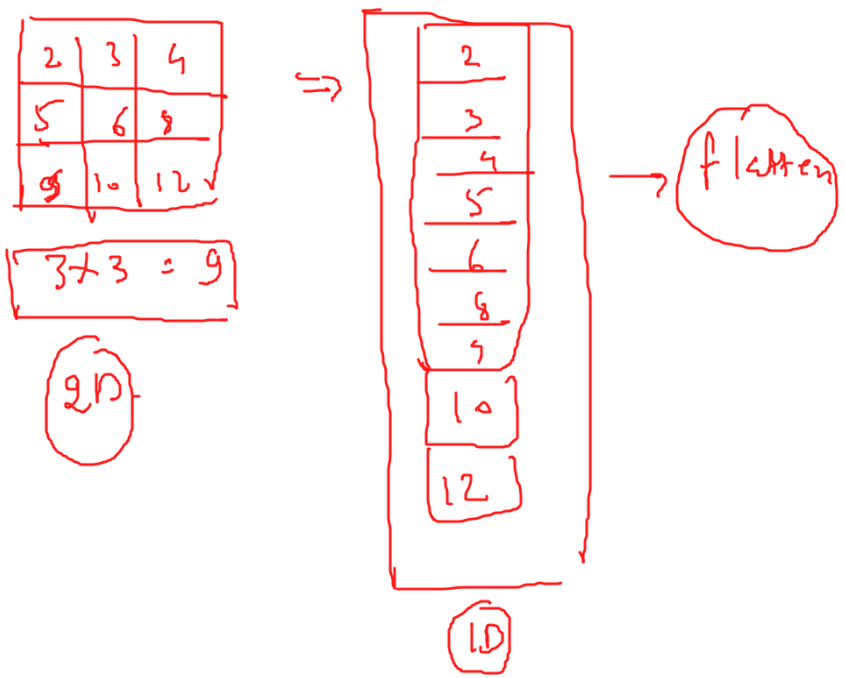
180 H

50 W

Rgb

?


ANN and CNN and RNN
DL
CNN → Convolution neural network → Computer vision.
image
(Randomly)
Sunny
(0-255)
(256×256)
(3×3)
(feature)
(Pooling)
max
min.
avg
(28×28)
flatten
hidden
Sunny
fully connted.
ANN
vector
CNN
Image
Convolution operation
Convolution.
Extract feature
ML & DL → matrix
maths


2
3
4
5
6
8
10
12
3*3 = 9
2D
⇒
2
3
4
5
6
8
9
10
12
1D
flatten

NLP → ANN

text :- "Sunny is a data scientist Sunny like math and he like coding also"

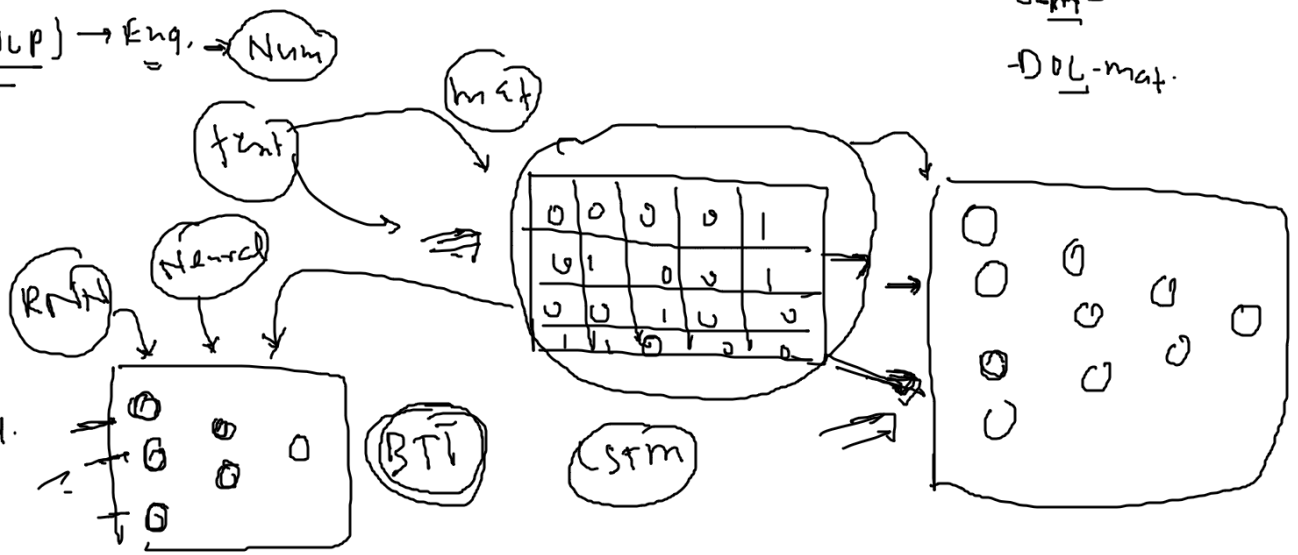
→ Preprocessing → Numeric form

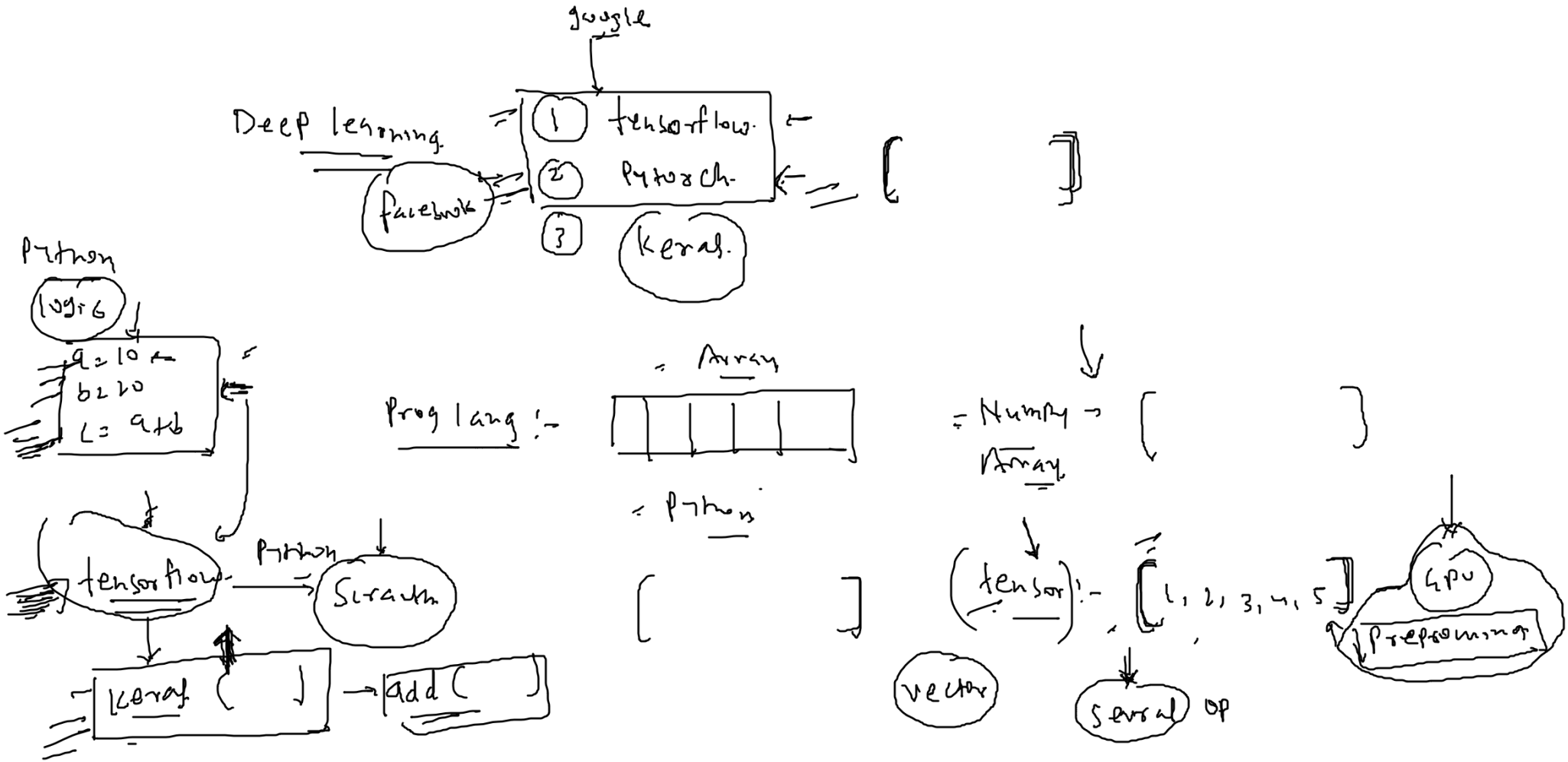
- one hot encoding
- TF-IDF
- Word 2 vec.
- Elmo
- DOL-mat.

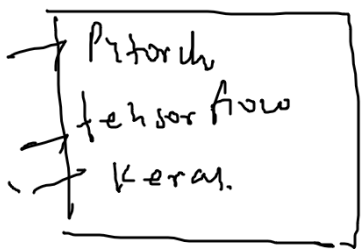
my computer (NLP) → Eng. → Num

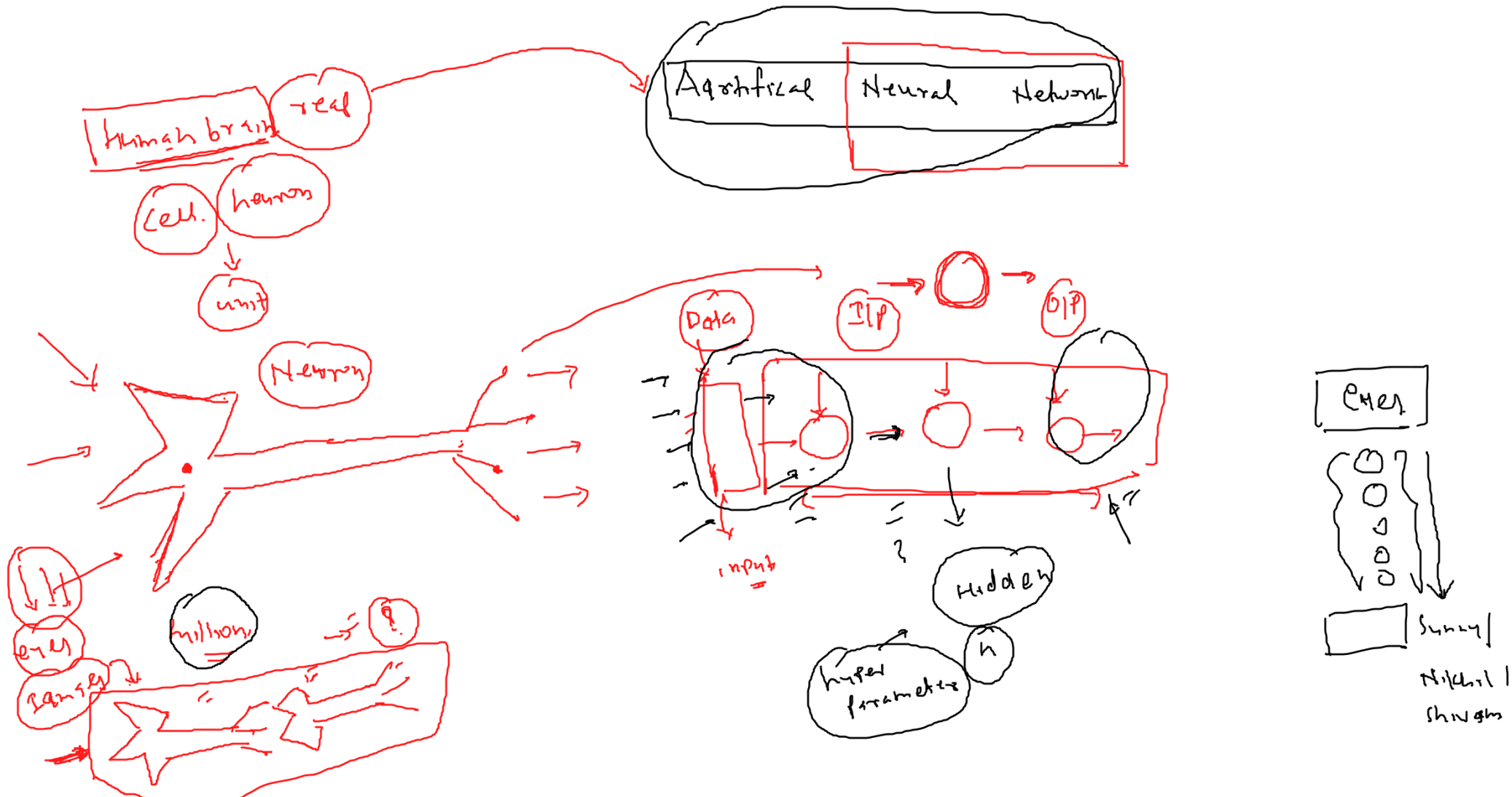
Processing → ANN → Predict.

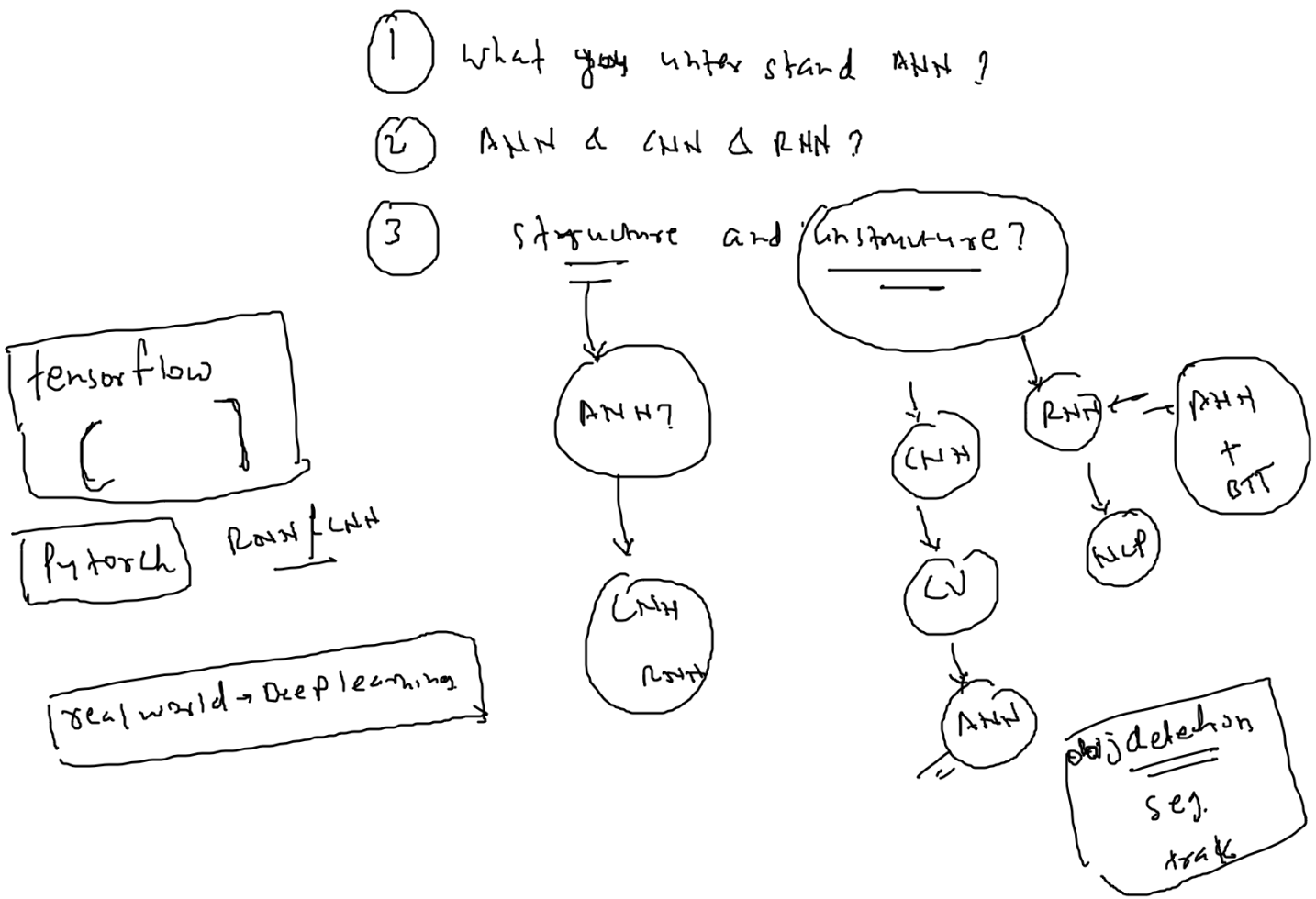
- RNN
- LSTM
- GRU.
- Attn.
- transformer

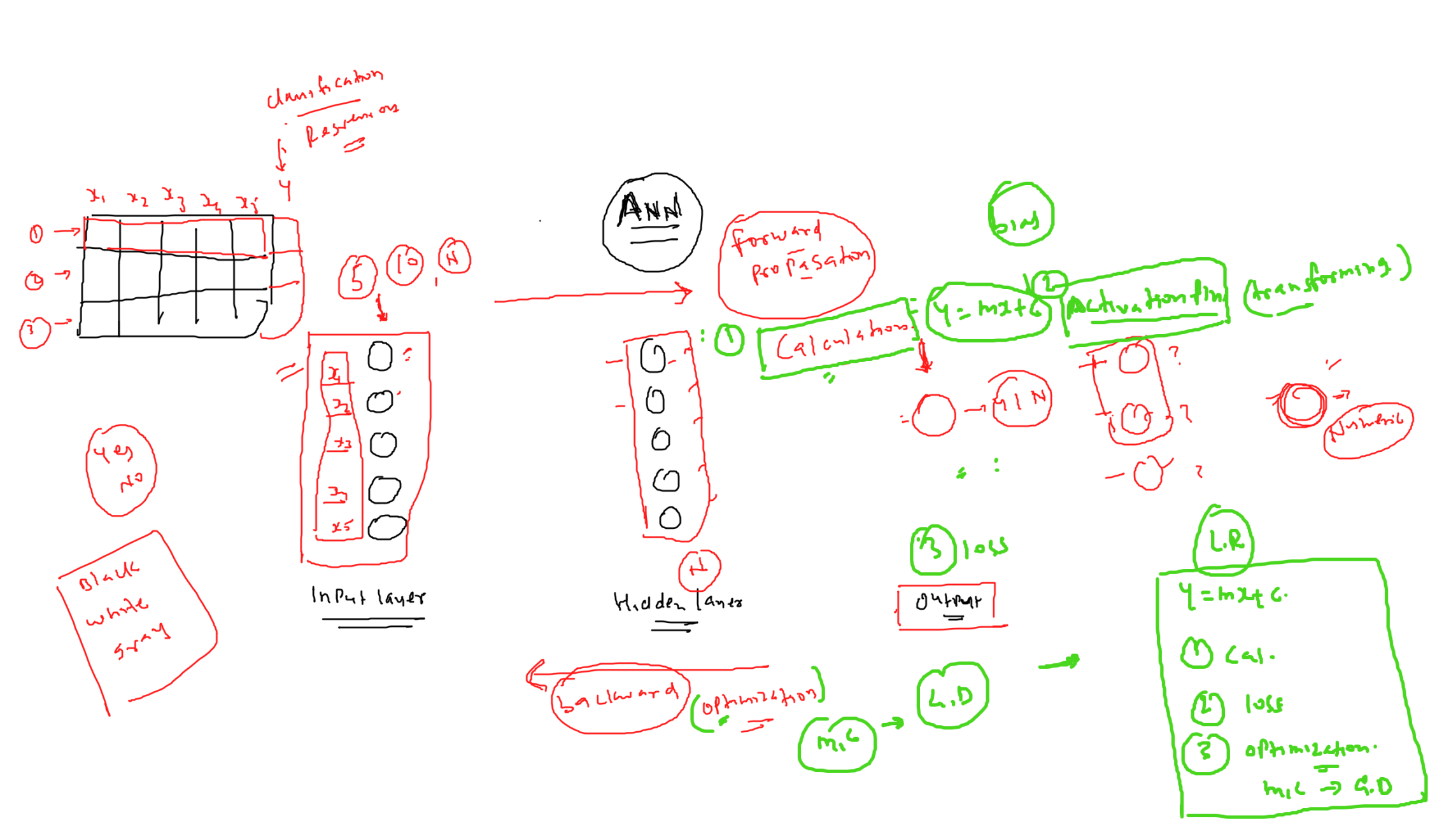


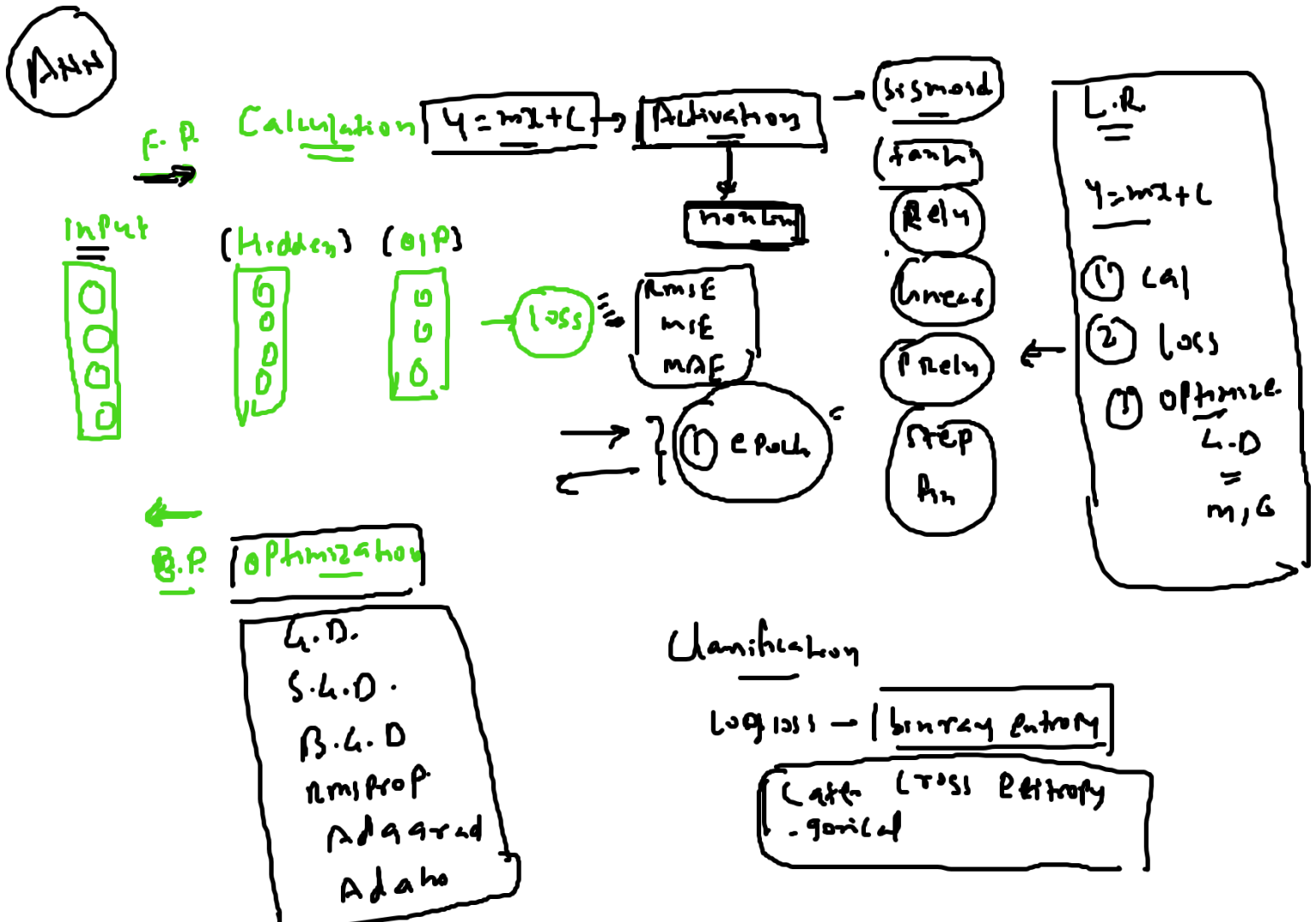

google
Deep learning
1 tensorflow
2 Pytorch
3 keras
facebook
Python
logic
a = 10
b = 20
L = a+b
tensorflow
Python
Scratch
keras ( )
add ( )
Prog lang :-
= Array
= Python
[ ]
= Numpy → [ ]
Array
(tensor) :- [1, 2, 3, 4, 5]
vector
several op
GPU
Preprocessing

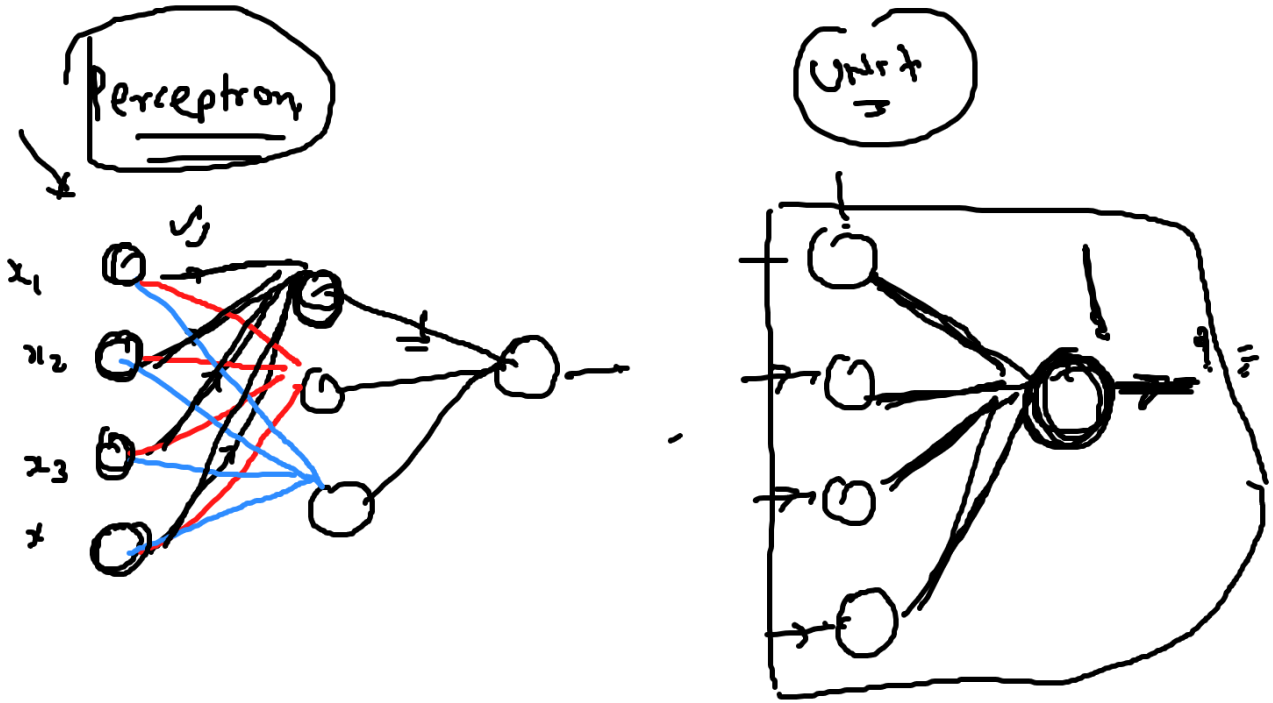

Pytorch
tensorflow
keras.

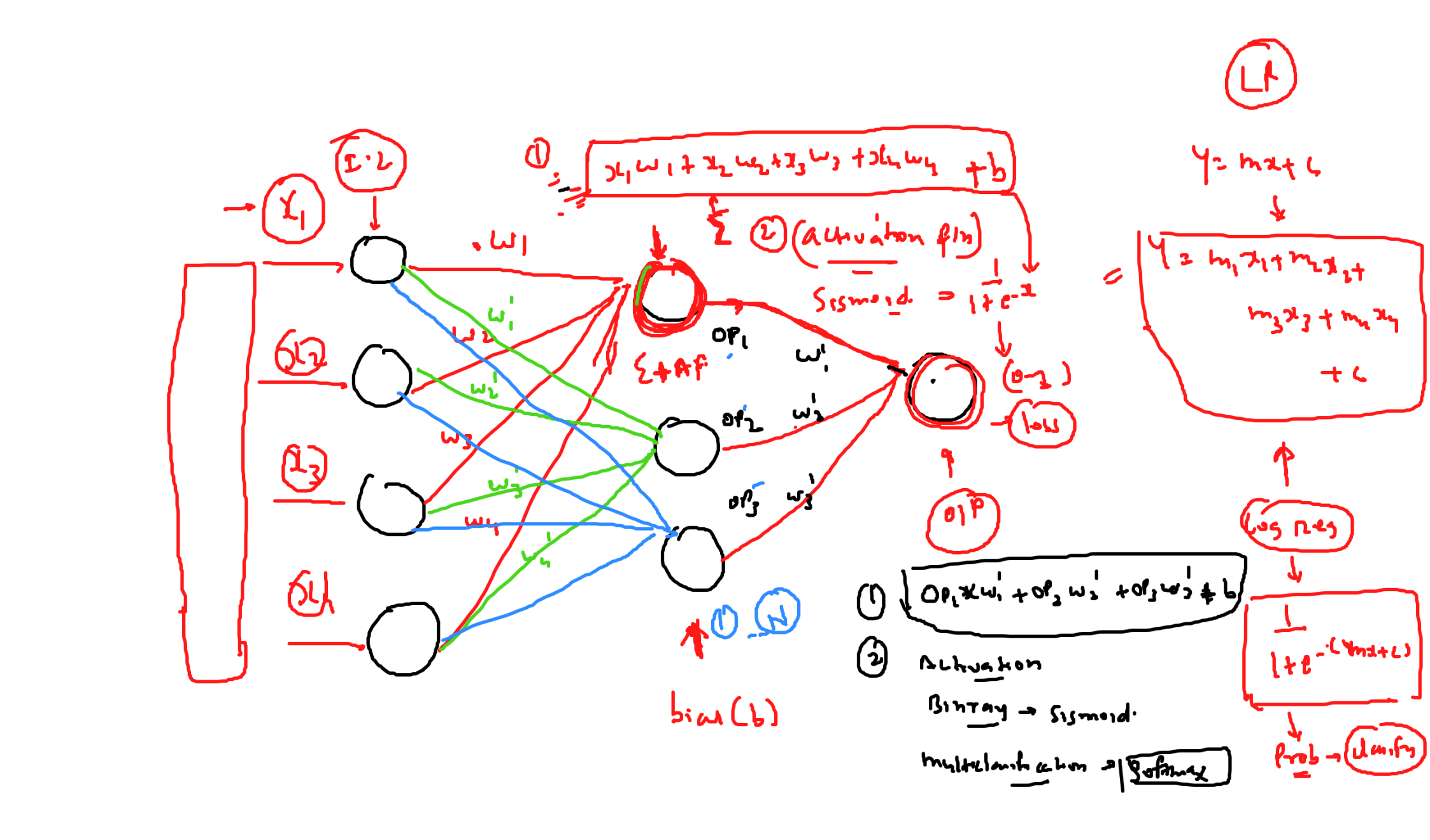

Artificial Neural Network
real
cell
neuron
unit
Neuron
Data
I/P
O/P
input
Hidden
h
hyper parameters
Cnet
Sunny
Nikhil
Shivam


1 what you understand ANN ?
2 ANN & CNN & RNN ?
3 structure and unstructure ?
tensorflow
Pytorch
RNN / CNN
real world → Deep learning
ANN ?
CNN
RNN
CNN
RNN
ANN + BTT
NLP
CV
ANN
obj detection
seg.
track

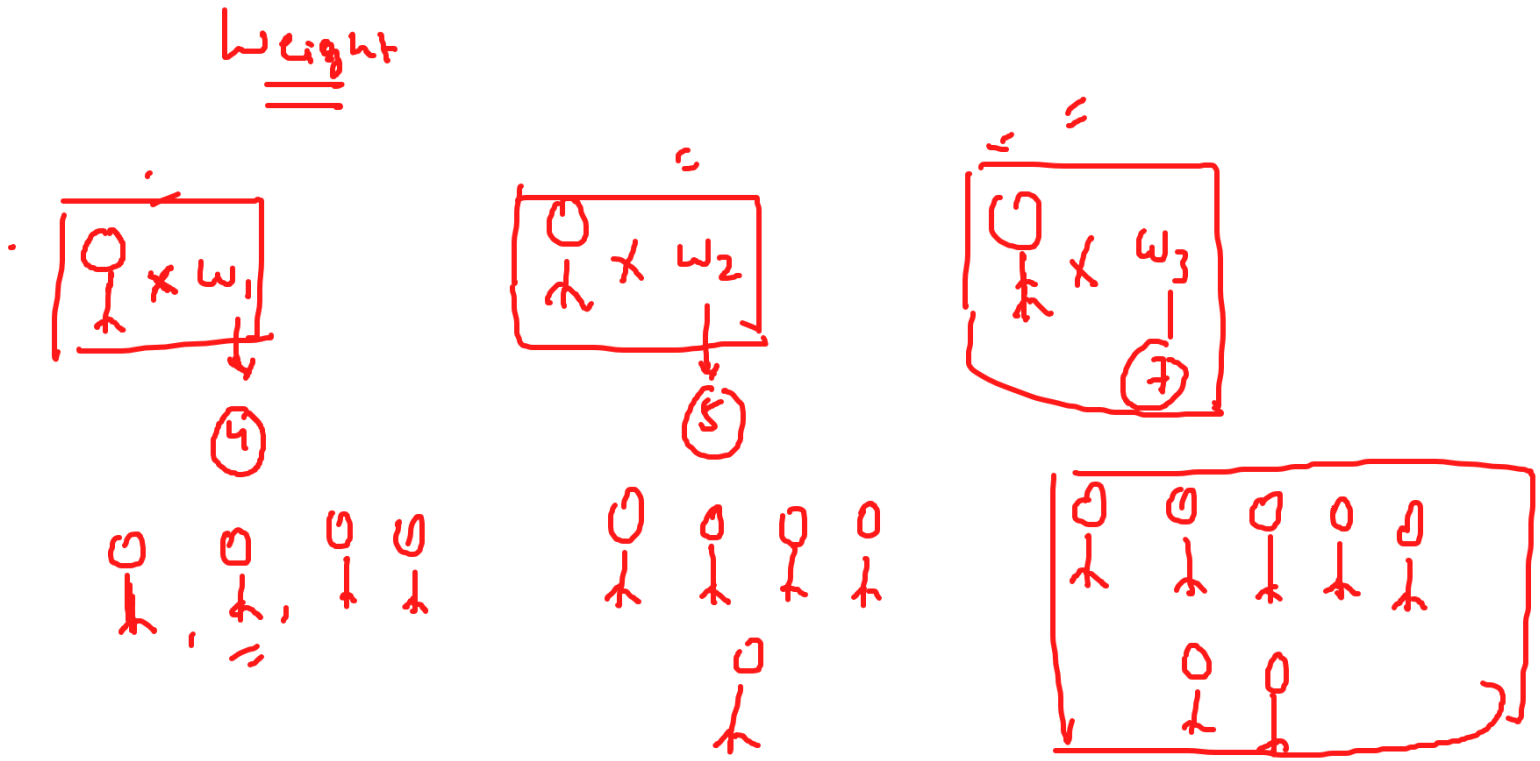

classification
Regression
y
x_1 x_2 x_3 x_4 x_5
1
2
3
5
10
N
yes
no
Black white grey
Input layer
ANN
forward propagation
Calculation
Hidden layer
bias
$y = mx + c$
Activation fun (transforming)
y N
Numerical
loss
Output
backward (optimization)
m, c
L.D
L.R
$y = mx + c$.
Cal.
loss
optimization.
m, c → G.D

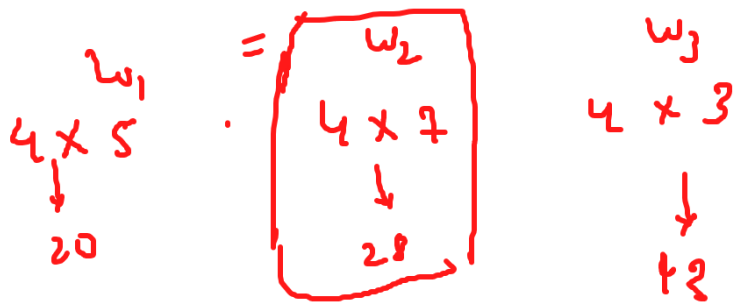

ANN
F.P
Calculation
y = mx + c
Activation
sigmoid
tanh
Relu
Linear
PRelu
Step fn
Monitor
Input
(Hidden)
(O/P)
Loss
RMSE
MSE
MAE
epoch
L.R
y = mx + c
(1) Cal
(2) Loss
(3) Optimize
L.D
=
m, c
B.P
Optimization
G.D.
S.G.D.
B.G.D
RMSprop
Adagrad
Adam
Classification
Logistic
binary entropy
Categ
Logistic
Entropy
-gorical

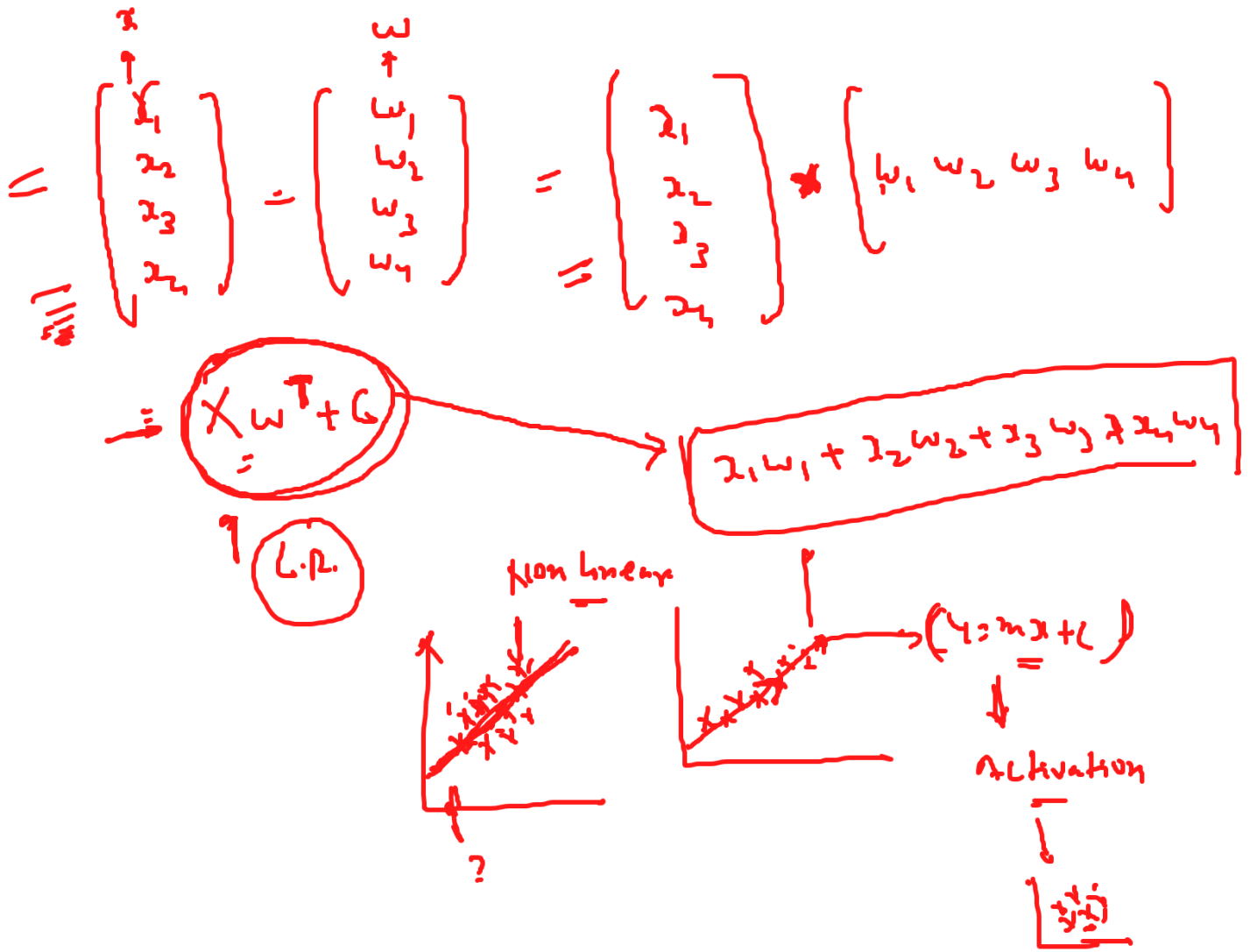

Perceptron
Unit
x_1
x_2
x_3
x

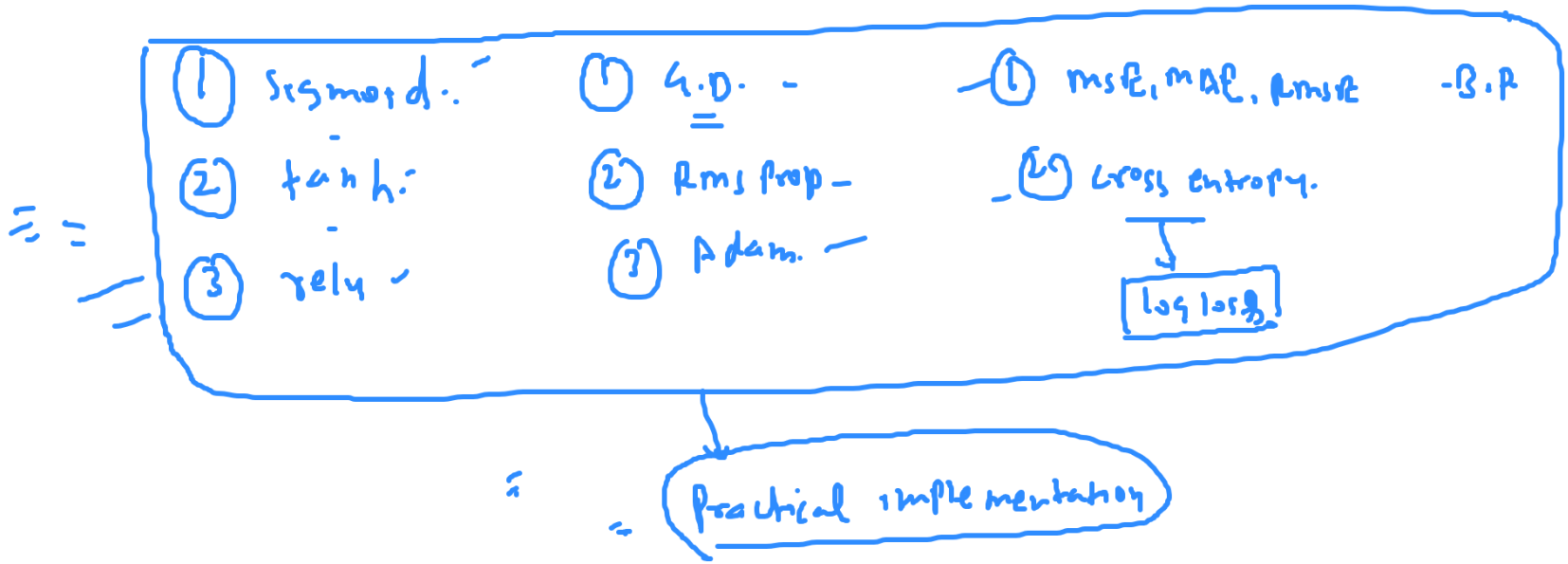

LR
y = mx + c
y = m1x1 + m2x2 + m3x3 + m4x4 + c
Log reg
1 / 1 + e^-(mx+c)
Prob → classify
I1
I2
I3
I4
w1
w2
w3
w4
Σ + Af
OP1
OP2
OP3
w1'
w2'
w3'
bias (b)
(1) OP1*w1' + OP2*w2' + OP3*w3' + b
(2) Activation
Binary → Sigmoid
multiclassification → Softmax
Sigmoid = 1 / 1 + e^-z
(0-1)
loss
o/p

Weight




W_1
4 x 5
20
=
W_2
4 x 7
28
W_3
4 x 3
12


x
w
L.R.
Non Linear
?
Activation


1. Sigmoid
2. tanh
3. relu
1. G.D.
2. Rms prop
3. Adam.
1. mse, mae, Rmse -B.P
2. Cross entropy.
log loss
Practical implementation