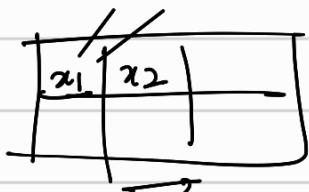
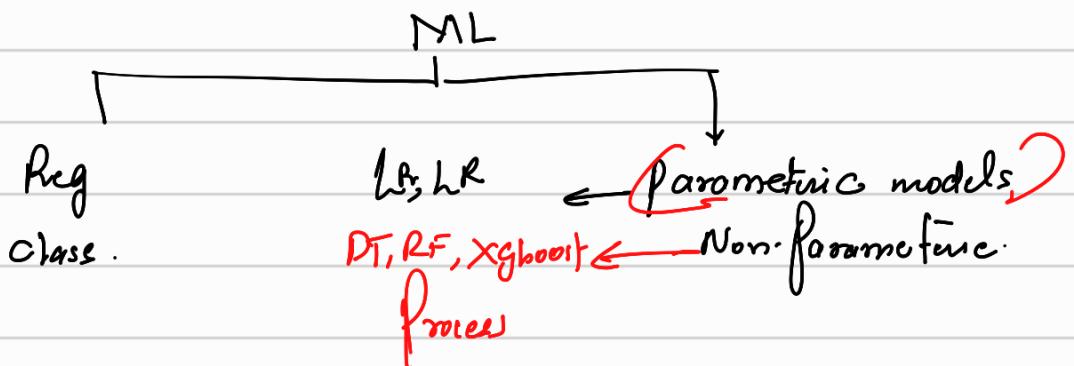


Deep learning

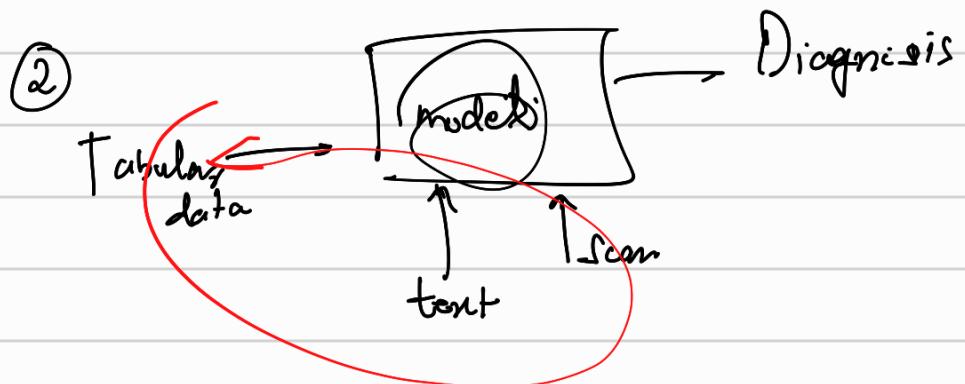
$$y = f(x) + \epsilon$$

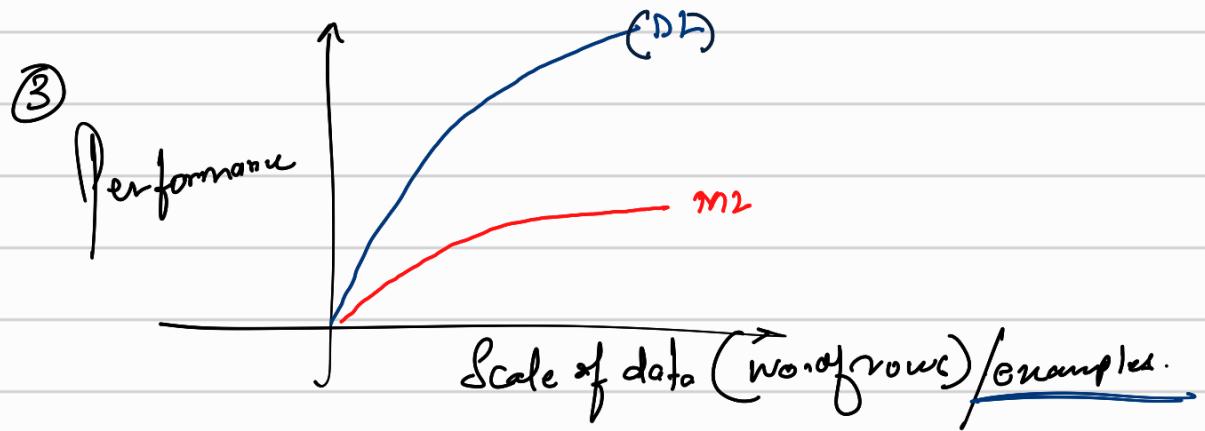


$$y \sim f(x)$$

① $x \rightarrow$ (video, audio, text, tabular data, graph).

multimodality



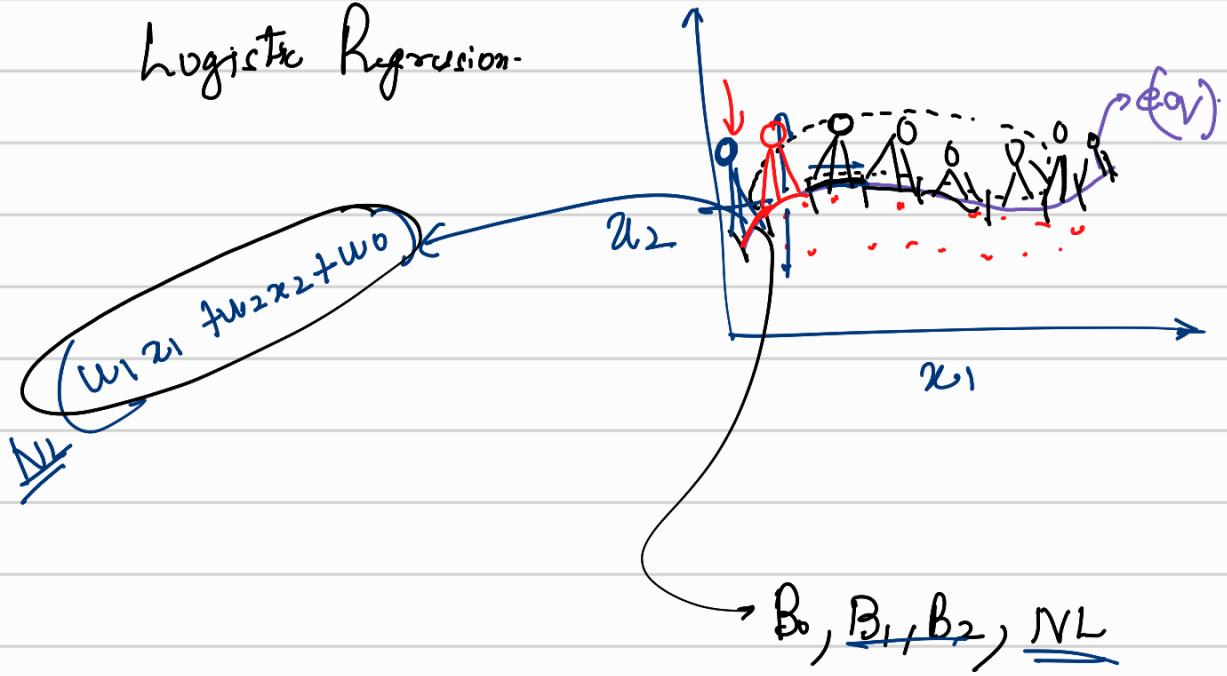


$D_h \rightarrow$ automated feature engineering

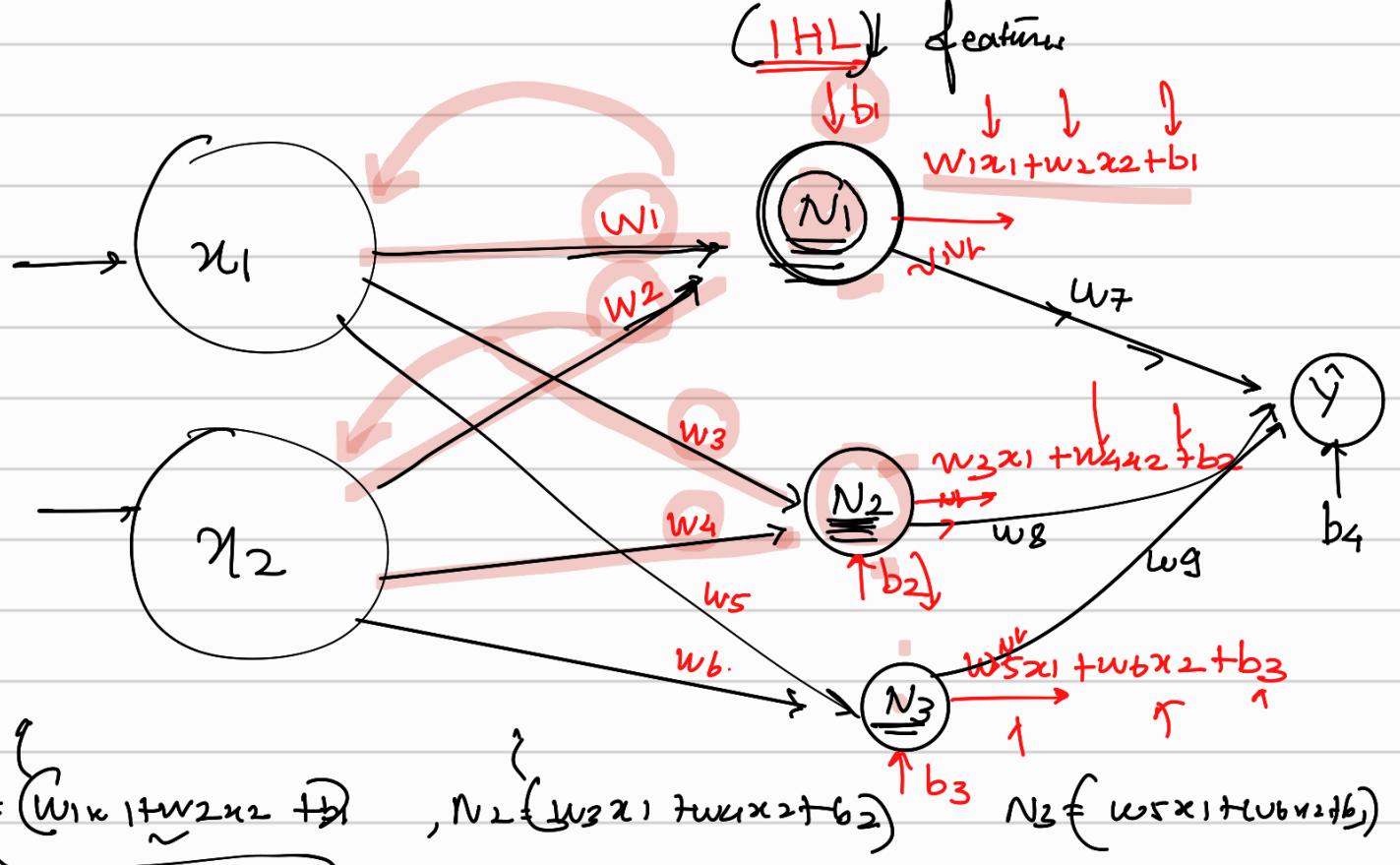


What is a Neuron

logistic Regression-



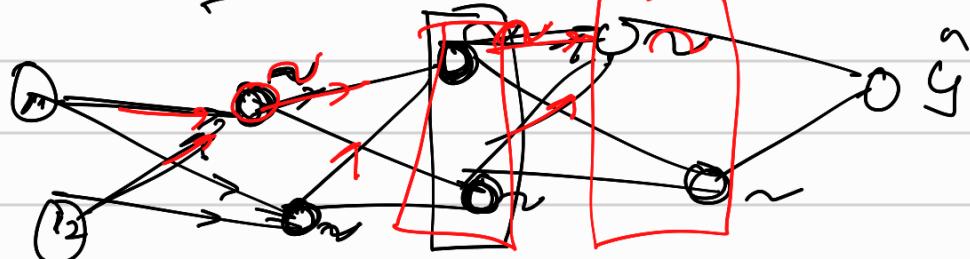
m_1	x_1	y



$$\Rightarrow \hat{y} = (\underline{w_7}) (\underline{w_1 x_1 + w_2 x_2 + b_1}) + \underline{w_8} (\underline{w_3 x_1 + w_4 x_2 + b_2}) + \underline{w_9} (\underline{w_5 x_1 + w_6 x_2 + b_3}) + b_4$$

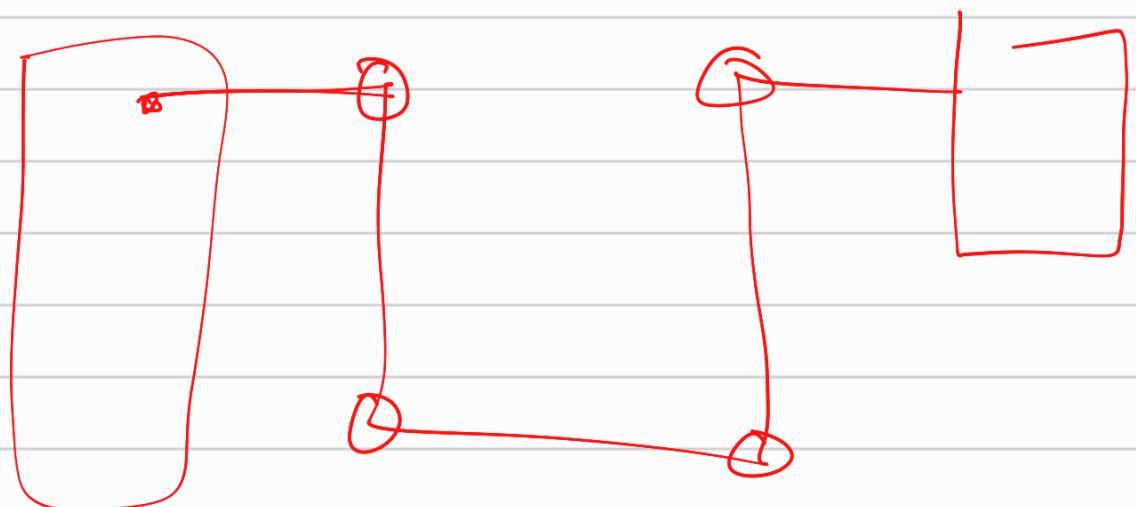
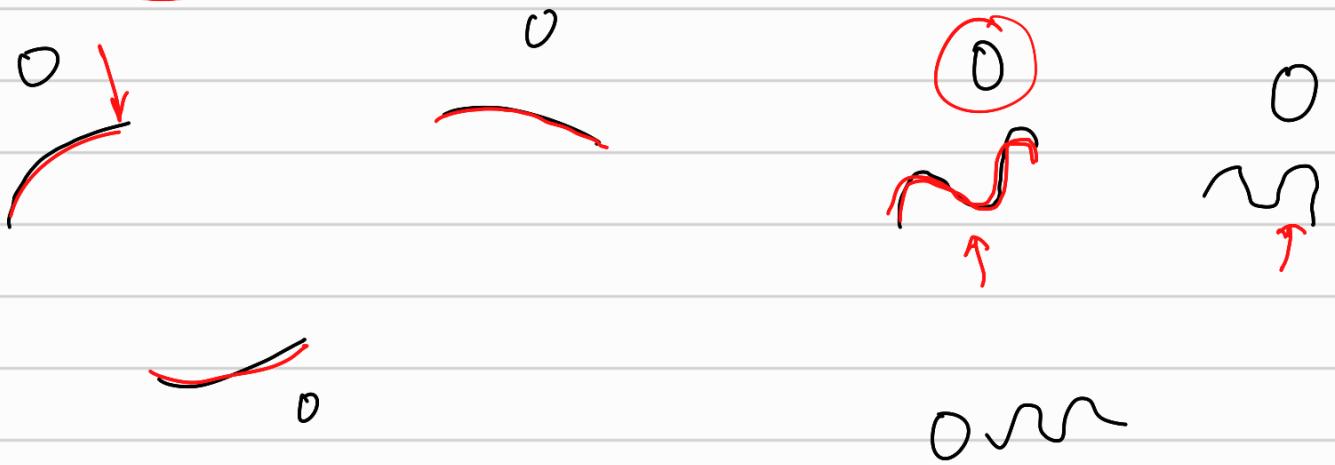
$$\Rightarrow \hat{y} = x_1 (w_7 w_1 + w_8 w_3 + w_9 w_5) + x_2 (w_7 w_2 + w_8 w_4 + w_9 w_6) + w_7 b_1 + w_8 b_2 + w_9 b_3 + b_4$$

$$\Rightarrow \hat{y} = A x_1 + B x_2 + C$$



1 Hc

→ n^{th} Hc

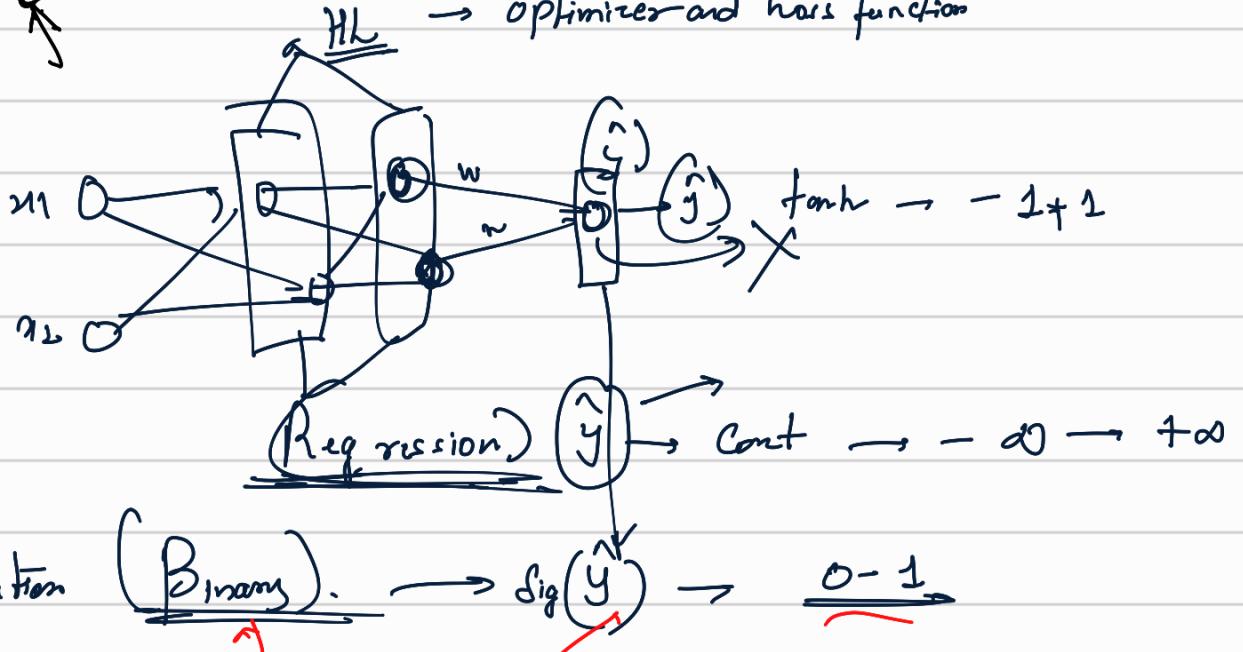
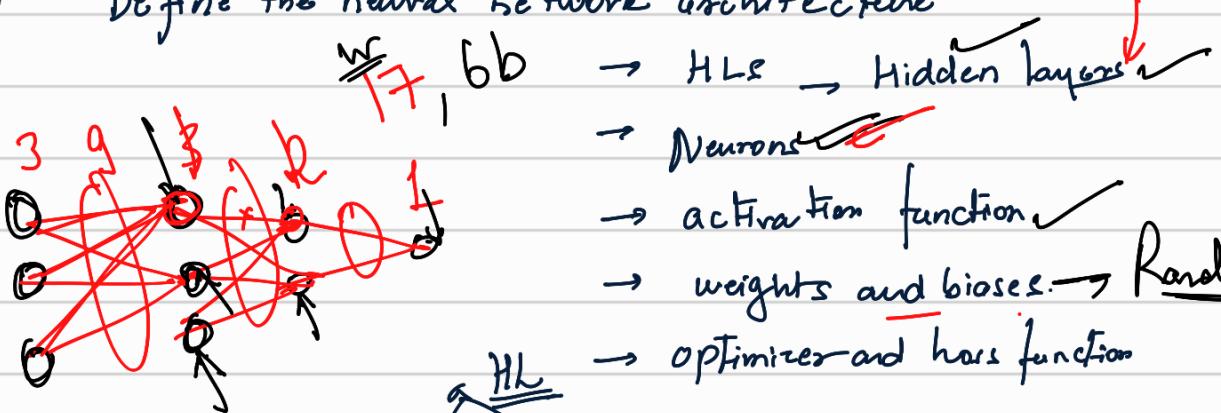


Gradient Descent

: minimise the loss by updating the W, B .
for all neurons.

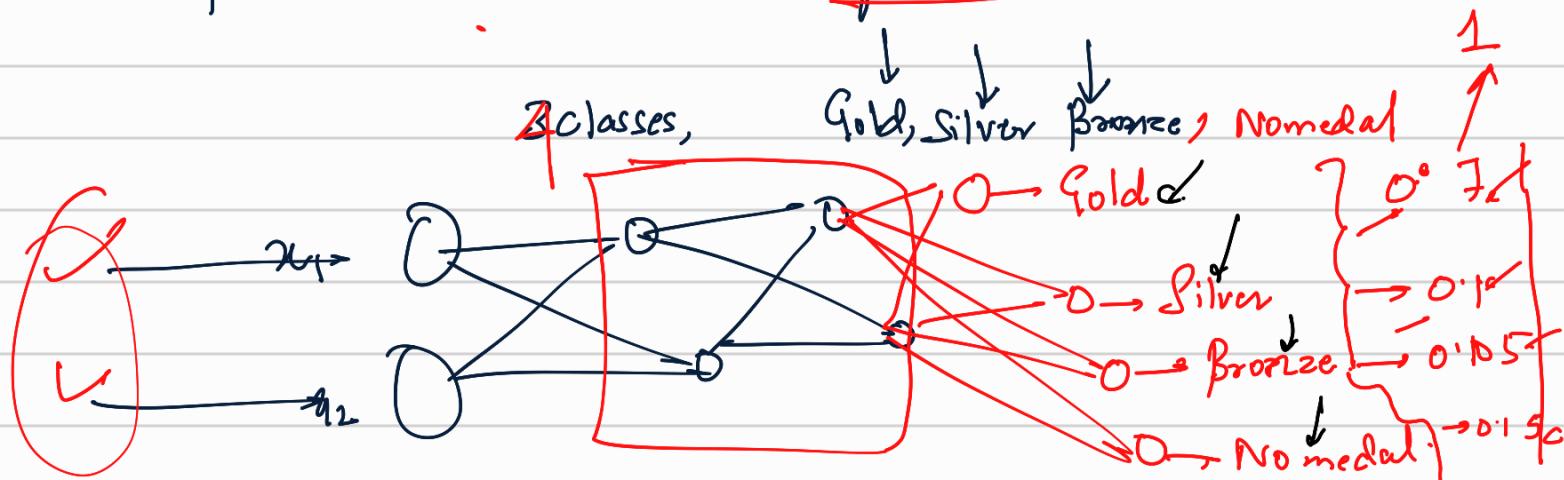
Rancharan

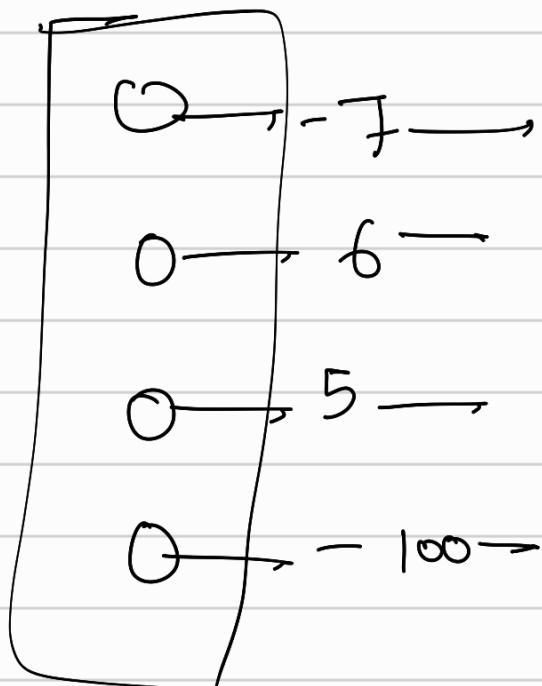
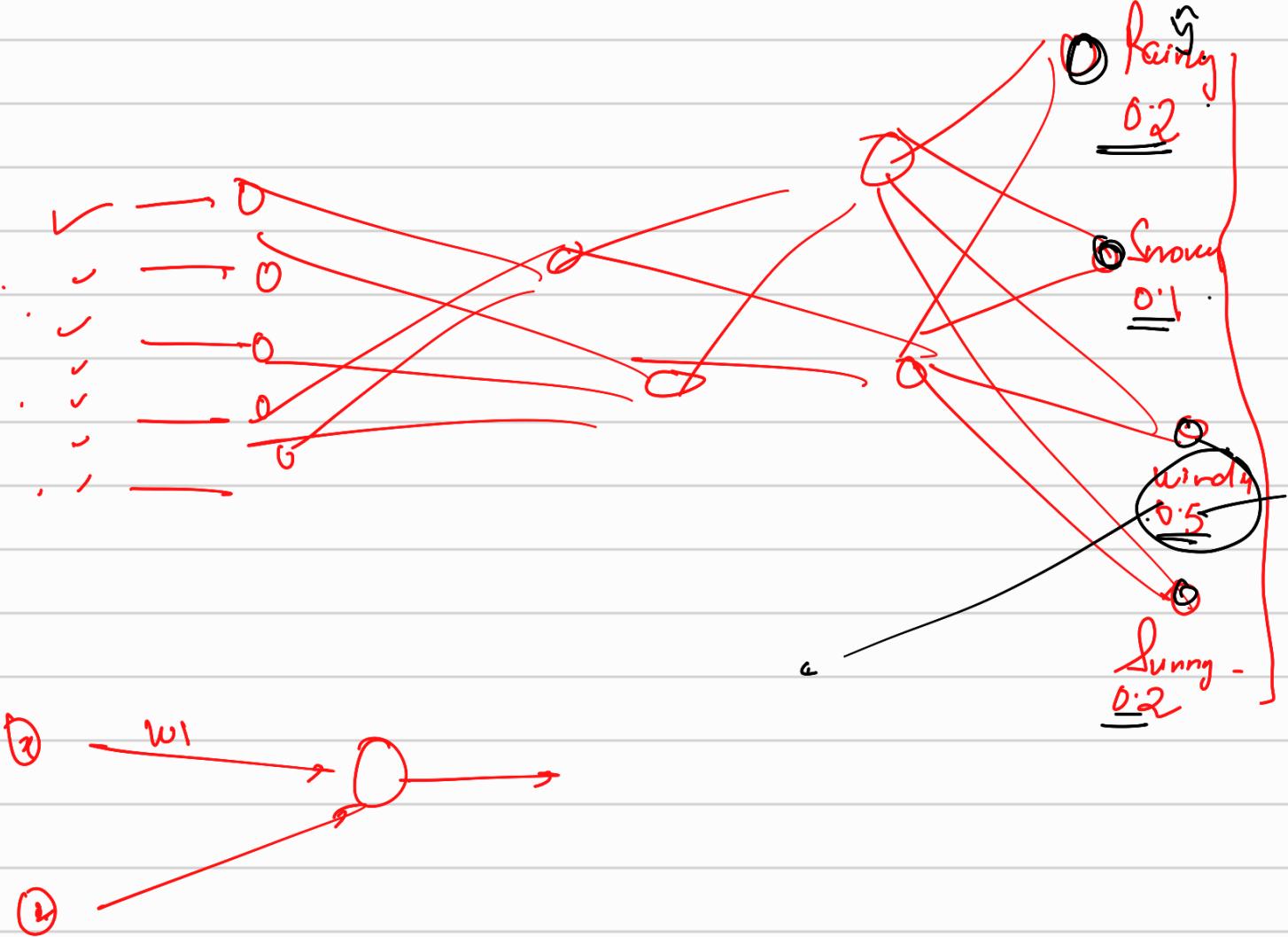
1) Define the neural network architecture



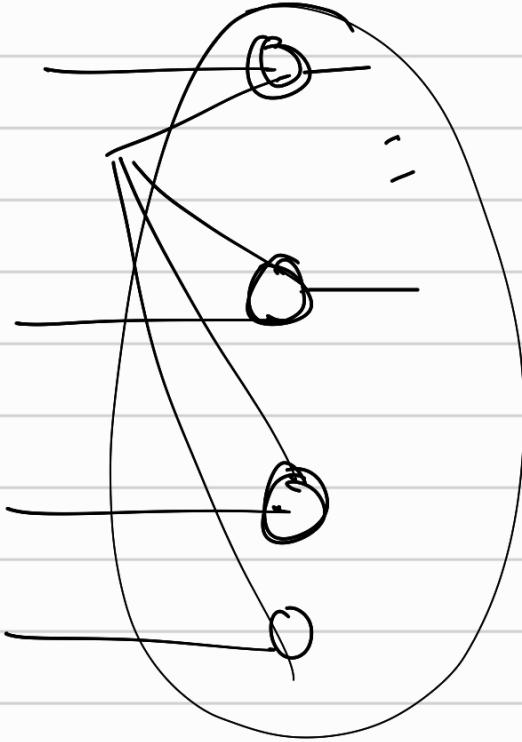
Classification (multiclass)

Softmax ($y\hat{}$)





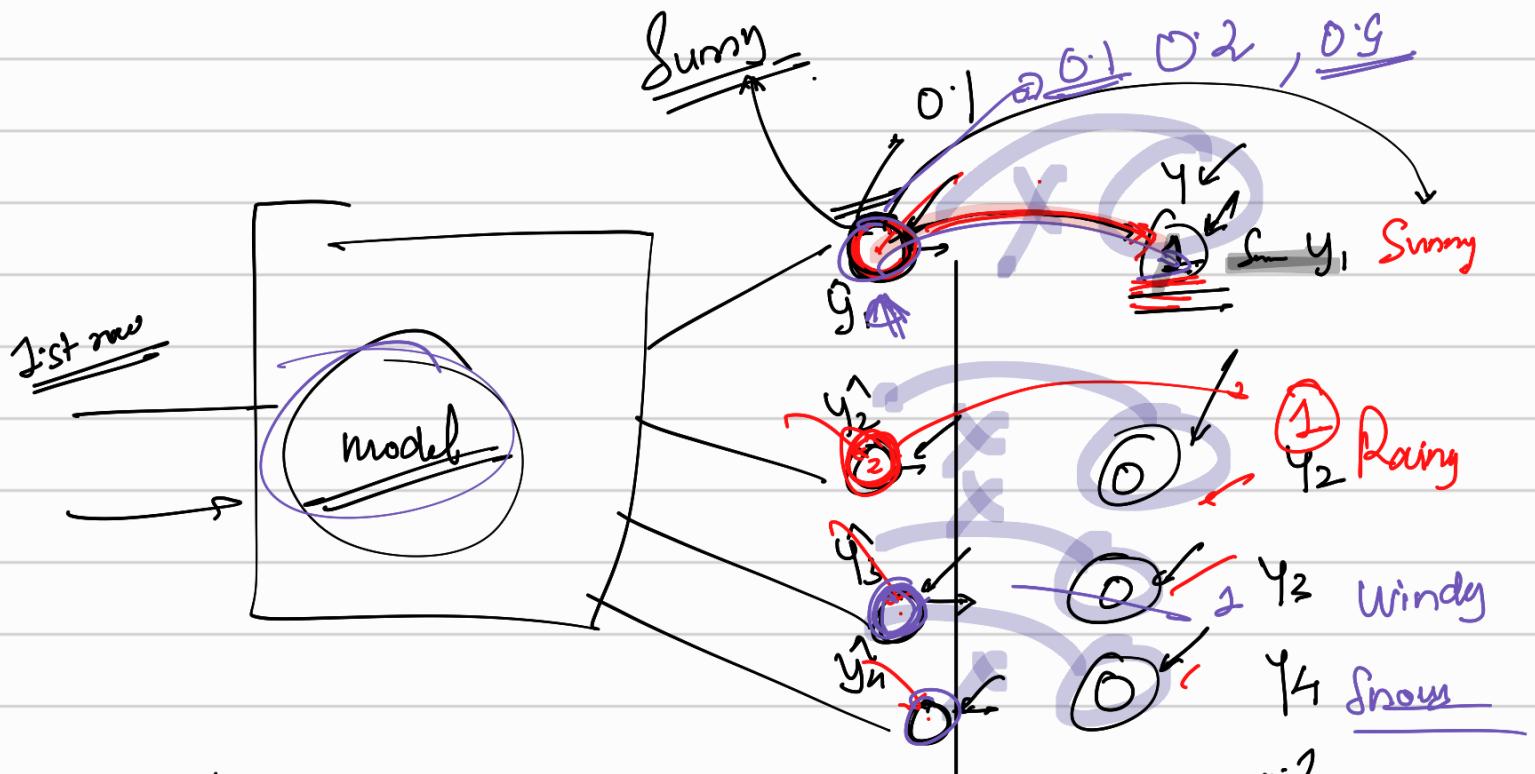
x_1	x_2	Weather (y)	$y-S_n$	$y-W_n$	$y-M_n$	$y-S_m$
		Sunny	1	0	0	0
		Rainy	0	1	0	0
		Windy	0	0	1	0
		Sunny	1	0	0	0
		Cloudy	0	0	0	1



0

0

0
0



$$\begin{aligned}
 \text{log loss} &= \text{for all rows of data sum} - (\hat{y}_1 \log(\underline{y}_1) + \hat{y}_2 \log(\underline{y}_2) \\
 &\quad + \hat{y}_3 \log(\underline{y}_3) + \hat{y}_4 \log(\underline{y}_4)) \\
 &- (\log 0.2) \\
 &= \underline{0.69}
 \end{aligned}$$

$$\begin{aligned}
 &- (\log 0.2) \\
 &= \underline{0.09}
 \end{aligned}$$

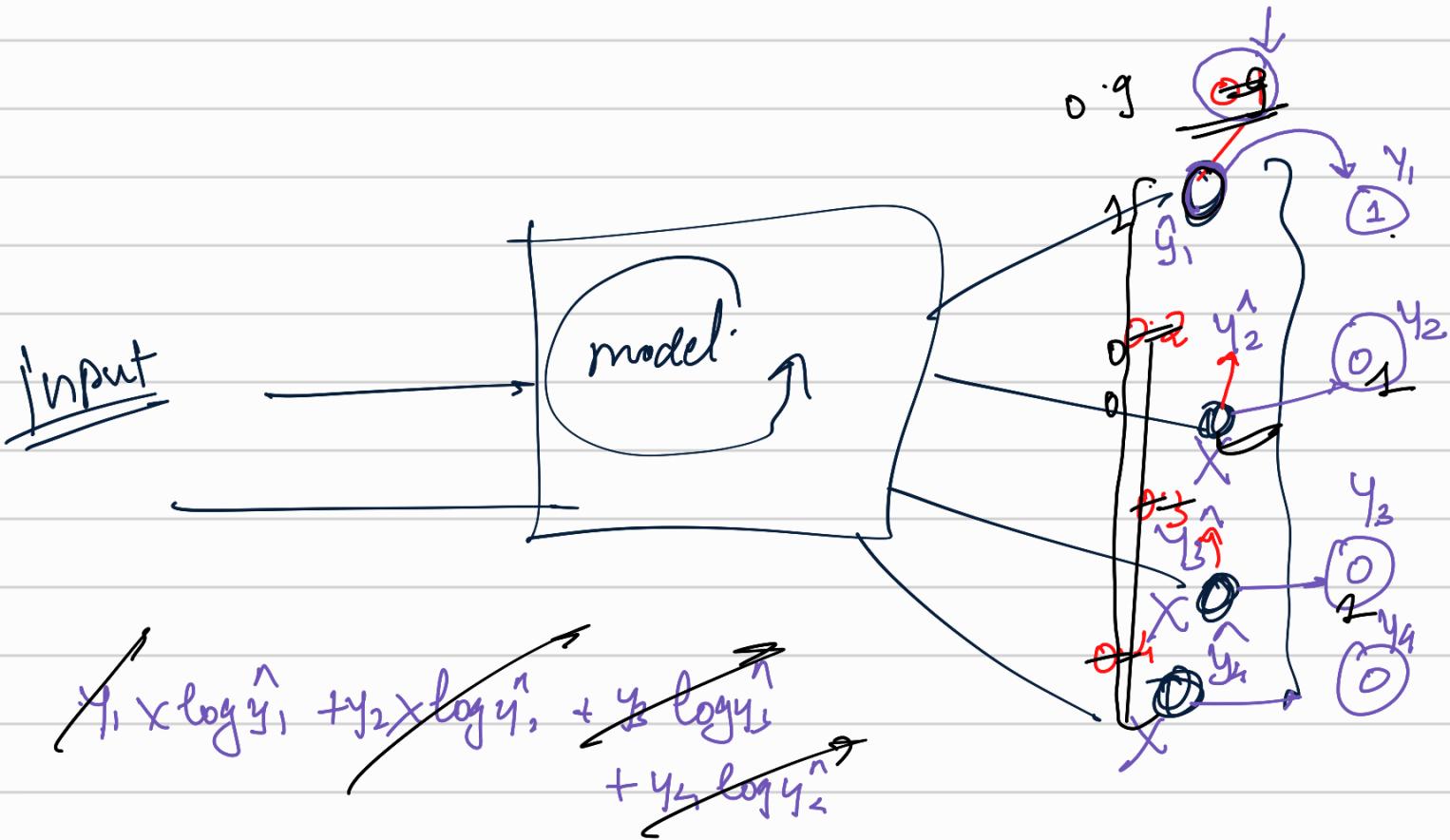
$$- \underline{\log(0.95)} = 0.02$$

$$\log 1 = 0$$

$$y_1 \log \hat{y}_1 + y_2 \log \hat{y}_2 + y_3 \log \hat{y}_3 + \dots$$

~~$y_1 \log y_1 + y_2 \log y_2 + y_3 \log y_3 + \dots$~~

$$\log(0.1) = -1$$



$$-\log \hat{y}_i$$

$$\log 1 = 0$$

$$-\log 0.1$$

VS

$$-\log 0.9$$

$$-\log 0.1$$

VS

$$-\log 0.9$$

0.045

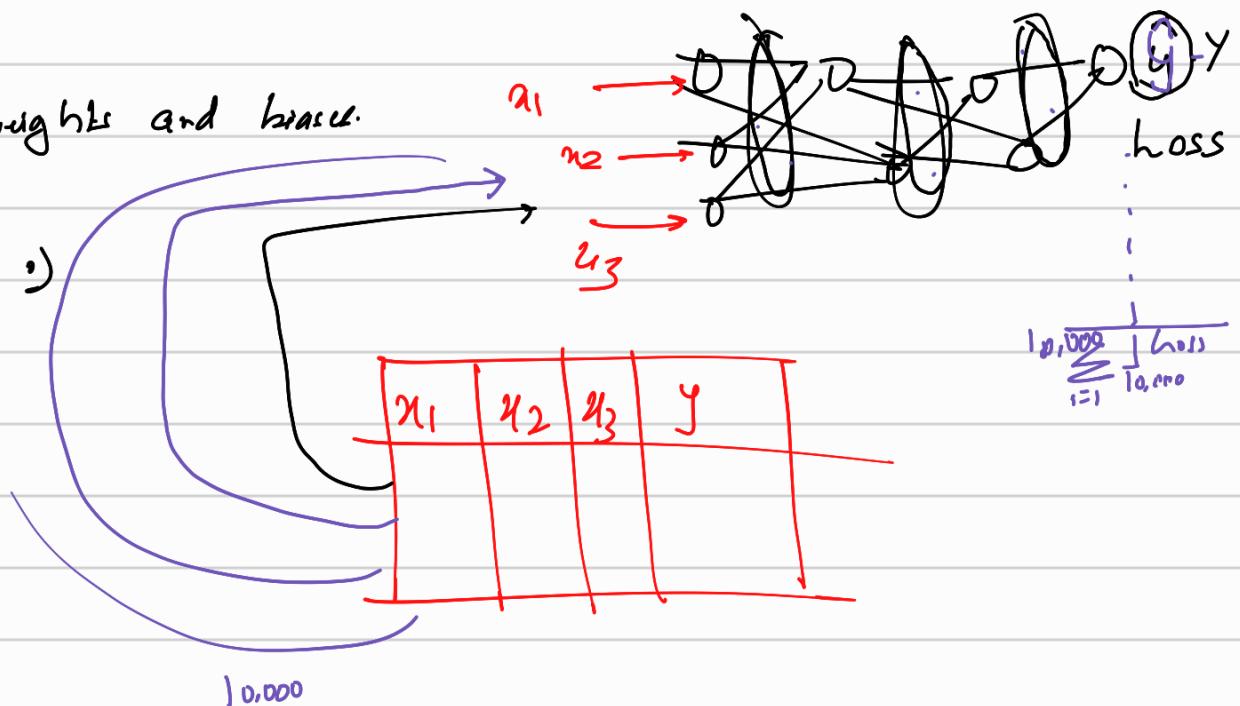
0
0
0

$$(0)$$

softmax, ohe

log loss

Random weights and biases.

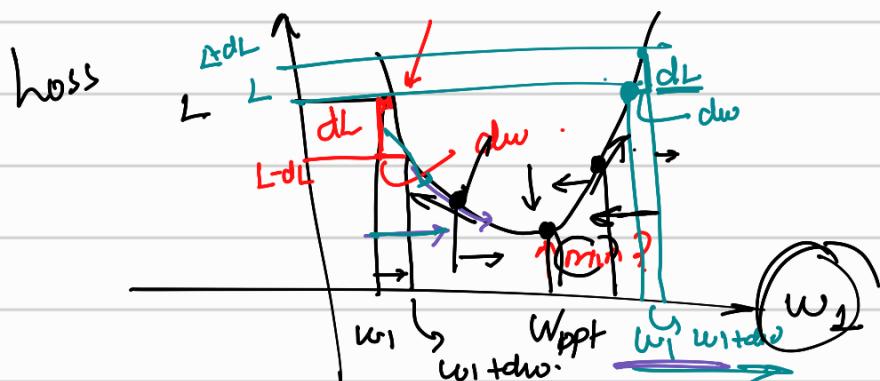
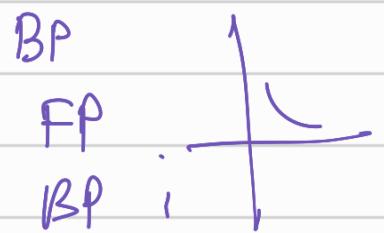




$$L = f(y_{predicted}, y_{actual})$$

$$L = f(w, b, y_{predicted}, y_{actual})$$

$$L = f(w, b)$$



$$\frac{dL}{dw} = -ve.$$

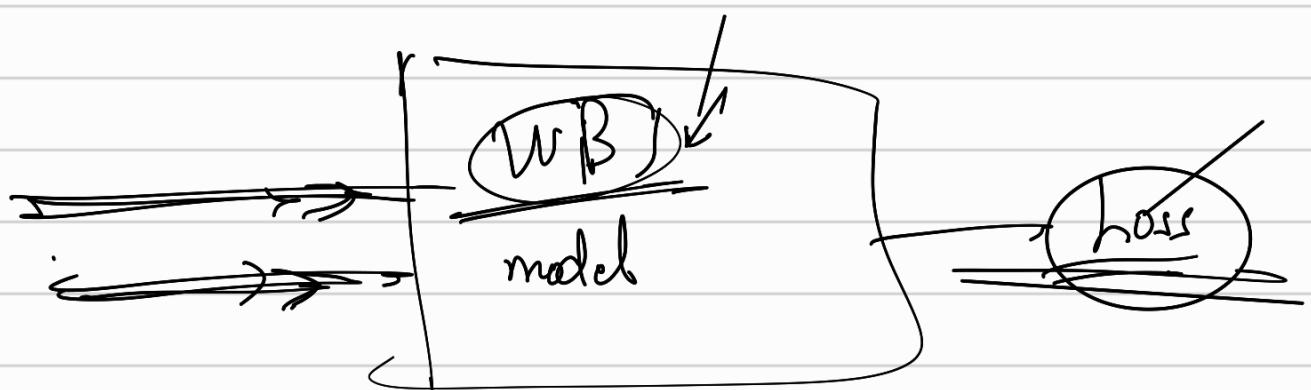
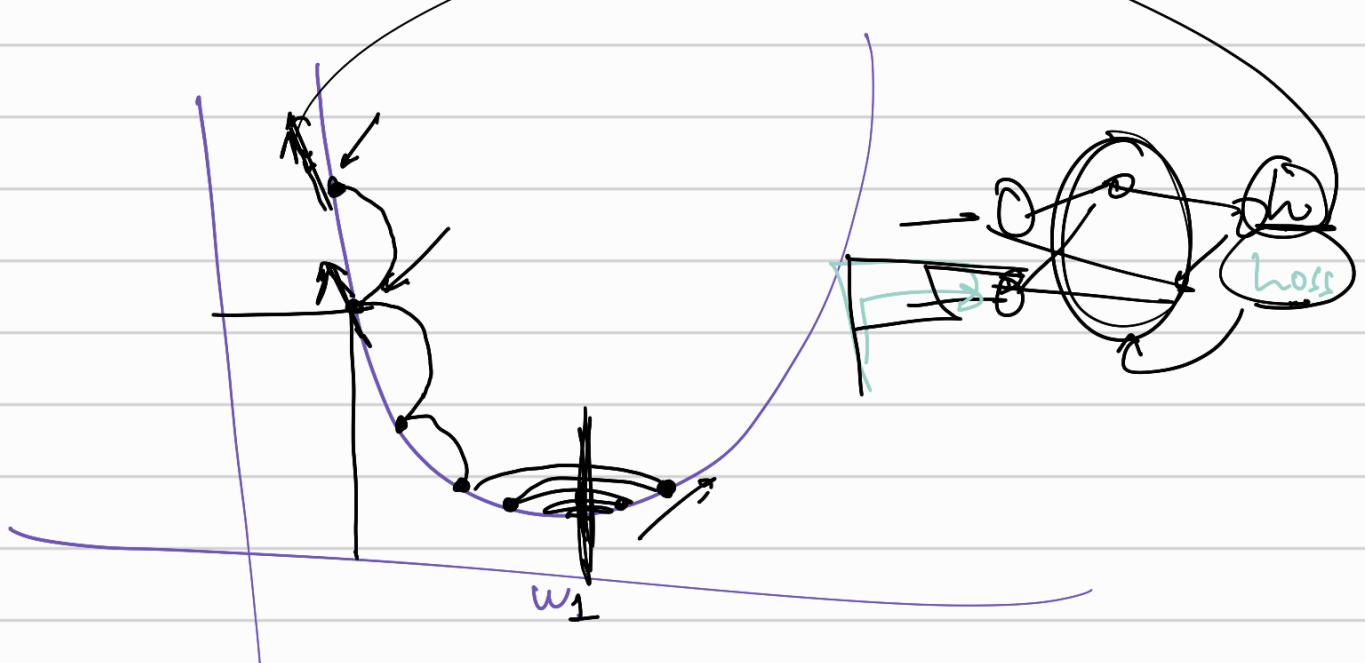
which side would +ve move

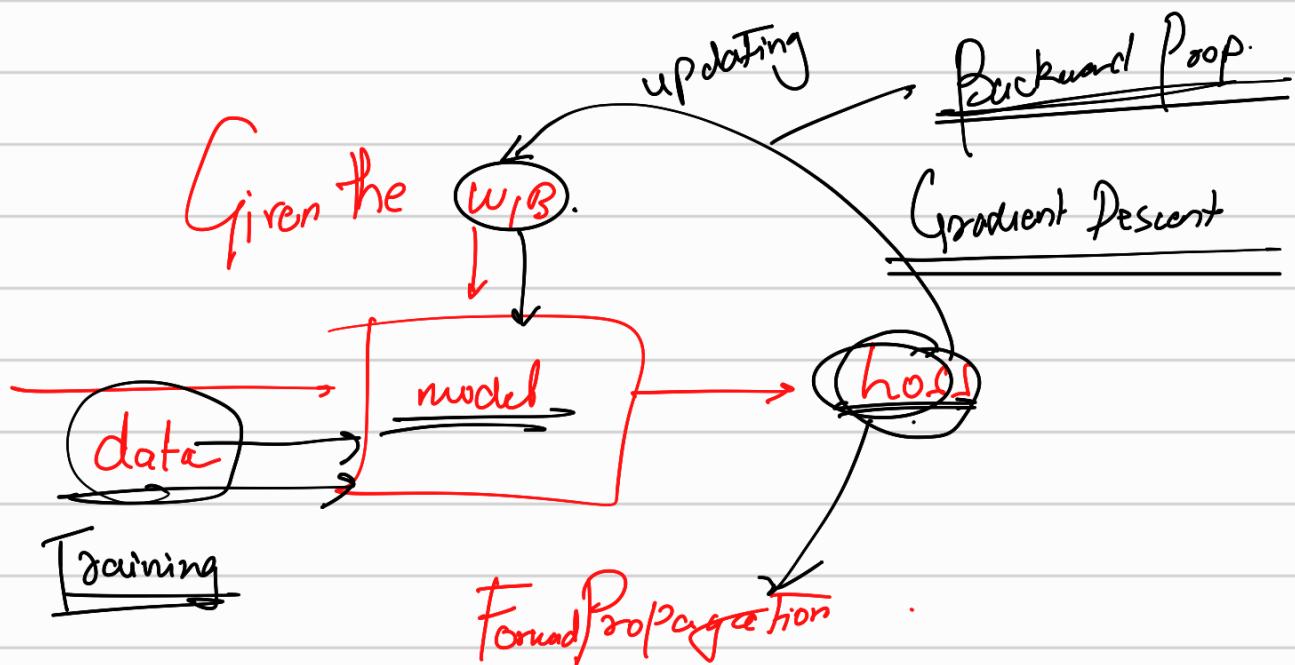
$$\frac{dL}{dw} = +ve$$

$$(w = -ve.)$$

$$w_i(\text{new}) = w_i(\text{old}) \left(-\frac{dL}{dw_i} \right)$$

Applicable for all weights and biases.





$(\text{IFP} + \text{IBP}) \rightarrow \underline{\text{epoch}}$

$(\text{IFP}, \text{IBP}) \rightarrow \text{epoch}$ → $\overset{\text{188}}{\text{epoch}}$

$\boxed{\text{IFP} + \text{IBP}}$

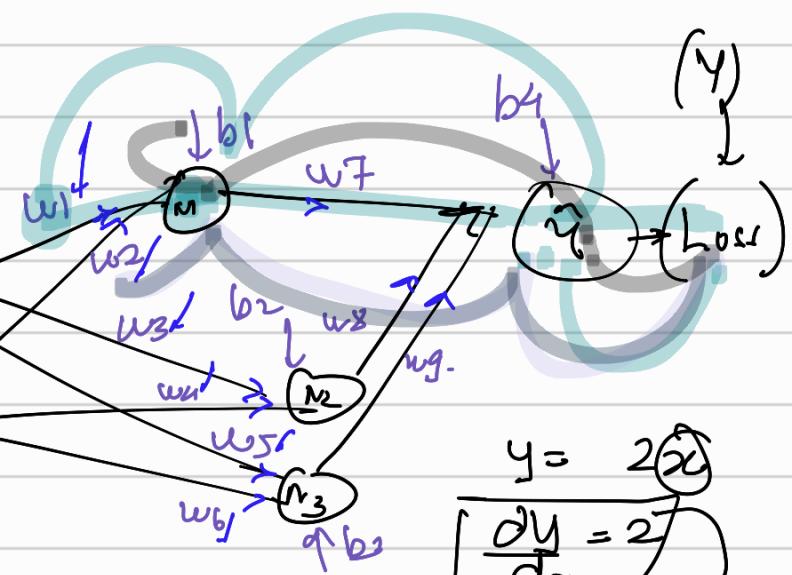
Lecture-3

$$N_1 = w_1 x_1 + w_2 x_2 + b_1 \quad (1)$$

$$N_2 = w_3 x_1 + w_4 x_2 + b_2$$

$$N_3 = w_5 x_1 + w_6 x_2 + b_3$$

$$\hat{y} = w_7 N_1 + w_8 N_2 + w_9 N_3 + b_4$$



$$y = 2x$$

$$\frac{\partial y}{\partial x} = 2$$

$$\frac{\partial L}{\partial \hat{y}} = \frac{1}{n} (y_i - \hat{y}_i)^2 \rightarrow \text{MSE}$$

$$\frac{\partial L}{\partial w_7} = \frac{\partial L}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial w_7} \Rightarrow \sum_{i=1}^n \frac{1}{n} \times 2 \times \frac{(y_i - \hat{y}_i)}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial w_7} \times N_1$$

$$\frac{\partial h}{\partial w_9}, \frac{\partial h}{\partial w_1}$$

6 mins

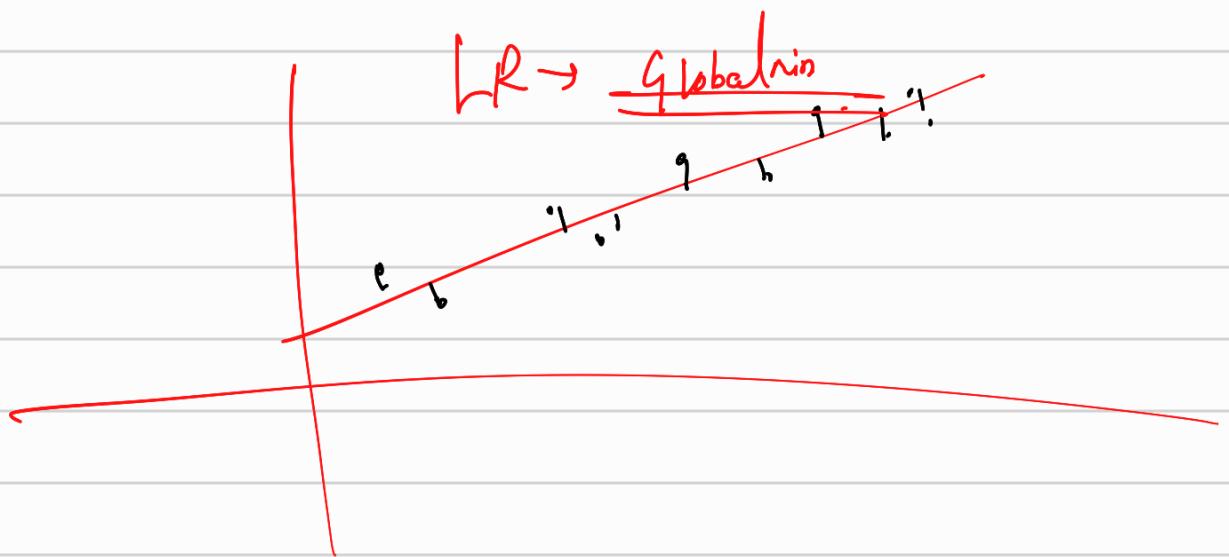
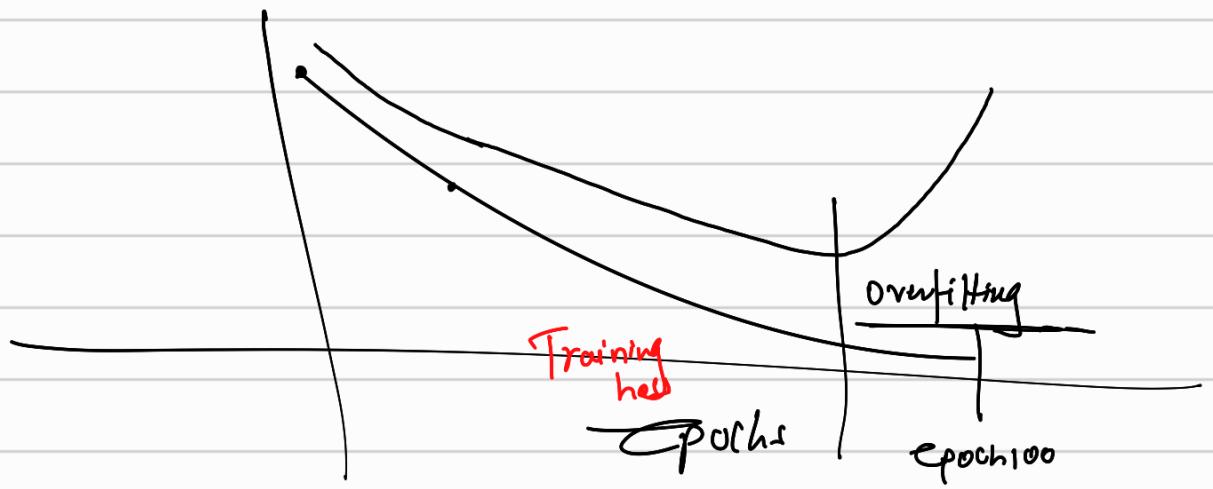
$$\frac{\partial L}{\partial w_9} = \left(\frac{\partial h}{\partial \hat{y}} \right) \times \frac{\partial \hat{y}}{\partial w_9} \rightarrow N_3$$

$$\frac{\partial h}{\partial w_1} = \frac{\partial h}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial N_1} \times \frac{\partial N_1}{\partial w_1}$$

$$\hat{y} = w_7 N_1 + w_8 N_2 + w_9 N_3 + b_4$$

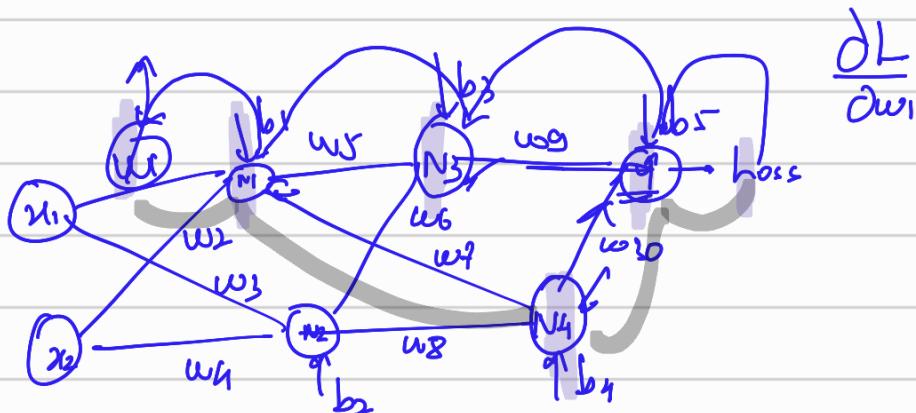
$$\frac{\partial \hat{y}}{\partial w_7} = \frac{\partial (\textcircled{w_7} \textcircled{N_1})}{\partial w_7} = N_1$$

$$\frac{\partial (x_1 x_2)}{\partial x} = 2$$



Activation functions

$$w_1x_1 + w_2x_2 + b_1$$



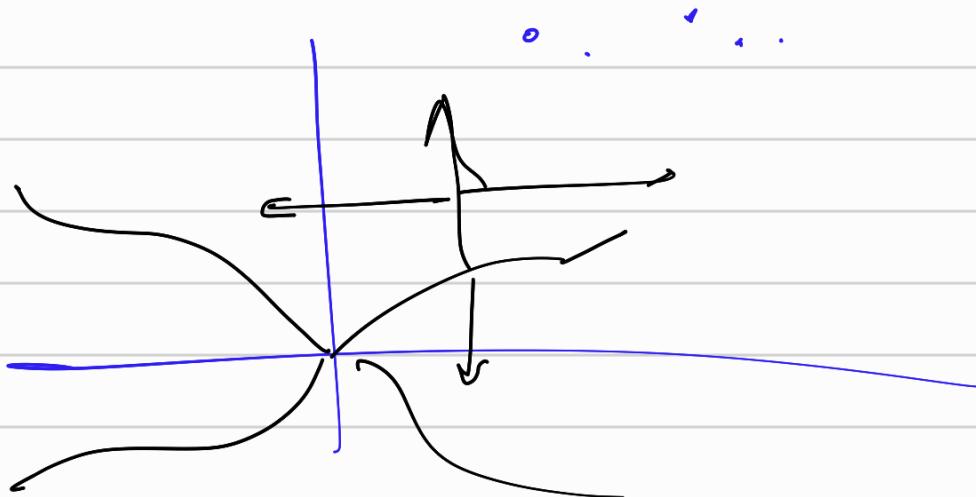
$$\frac{\partial L}{\partial w_1}$$

$\frac{\partial h}{\partial w_1} = \left(\frac{\partial L}{\partial y} \times \frac{\partial y}{\partial N_3} \times \frac{\partial N_3}{\partial w_1} \times \frac{\partial w_1}{\partial w_1} \right) + \left(\frac{\partial h}{\partial y} \times \frac{\partial y}{\partial N_4} \times \frac{\partial N_4}{\partial w_1} \times \frac{\partial w_1}{\partial w_1} \right)$.

$\frac{\partial L}{\partial w} = 1000000$
~~Terms = $N(N_L)^2$~~

$$\frac{\partial h}{\partial w_1} = 0.00000^2$$

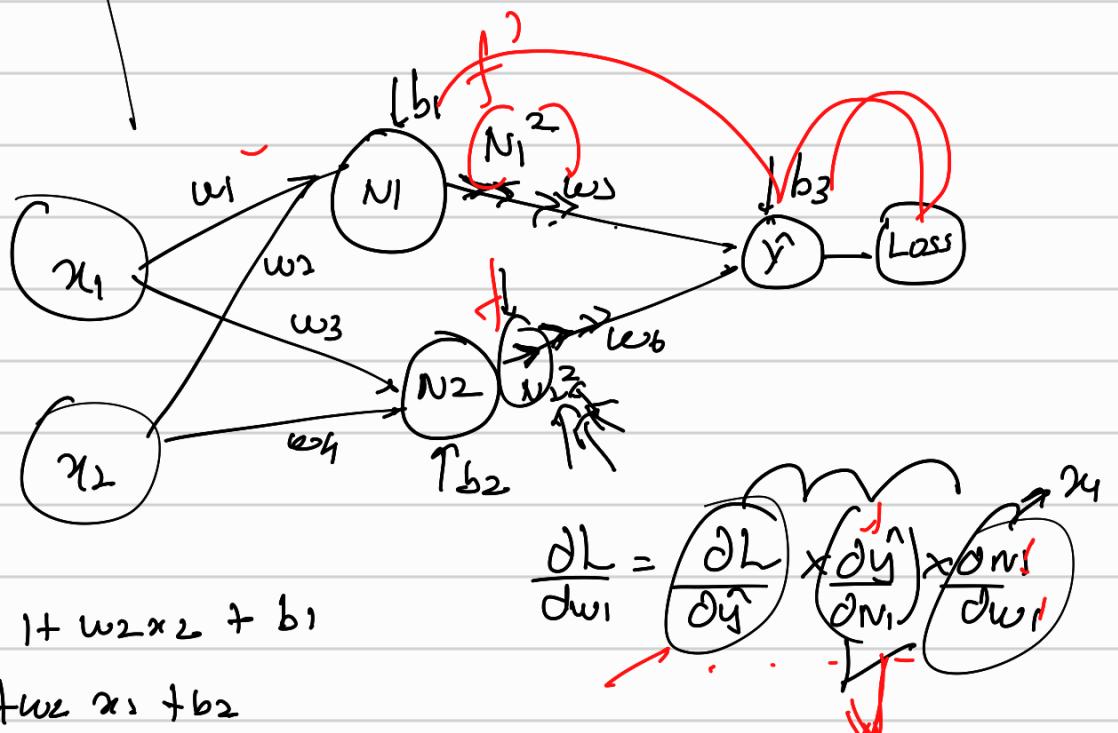
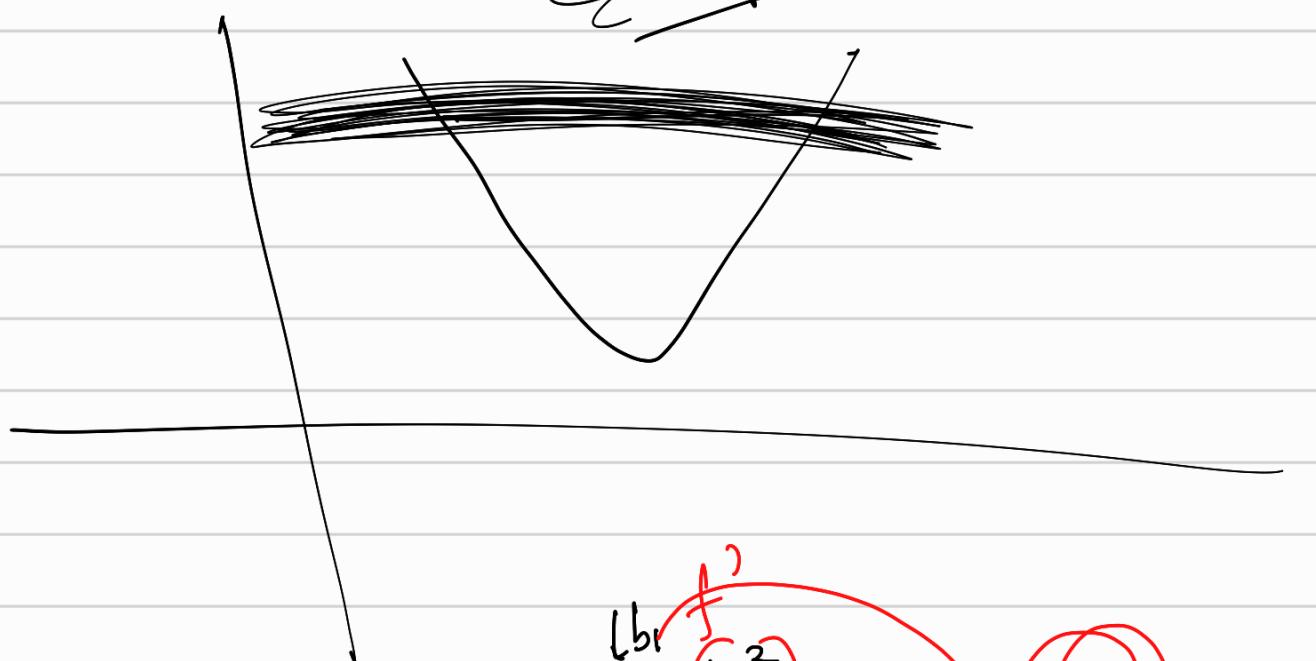
$w_1 -$



$$\left(\frac{0.99}{1.01}\right)^{365} \approx \underline{\downarrow \downarrow \downarrow}$$

$$\left(\frac{1.01}{1.00}\right)^{365} = 1.01 \underline{\uparrow \uparrow \uparrow}$$

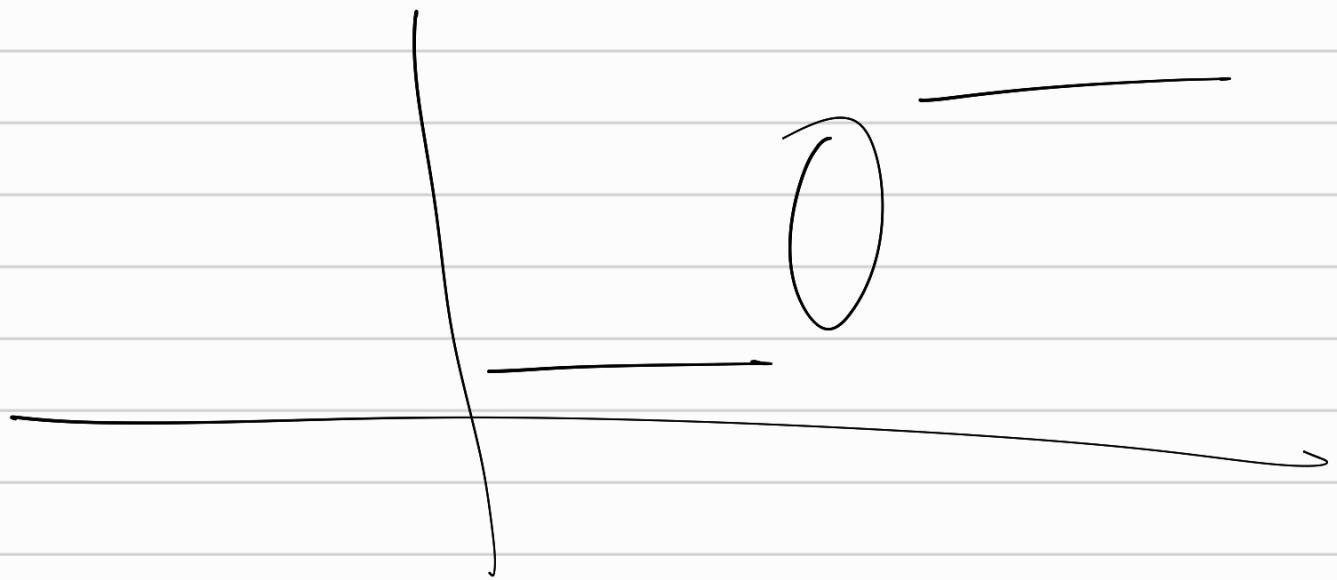
Exploding gradient



$$\hat{y} = w_5 \underline{N_1^3} + w_6 \underline{N_2^2} + b_3$$

$$\frac{\partial \hat{y}}{\partial w_1} = w_5 \times \underline{2 \times N_1}$$

~~(Activation function)~~ → Should be cont & diff.



Activation functions are functions which should be continuous & diff that can be applied at the output of any hidden layers to create non-linear feature or at the 0th to create bounded outputs → Probs

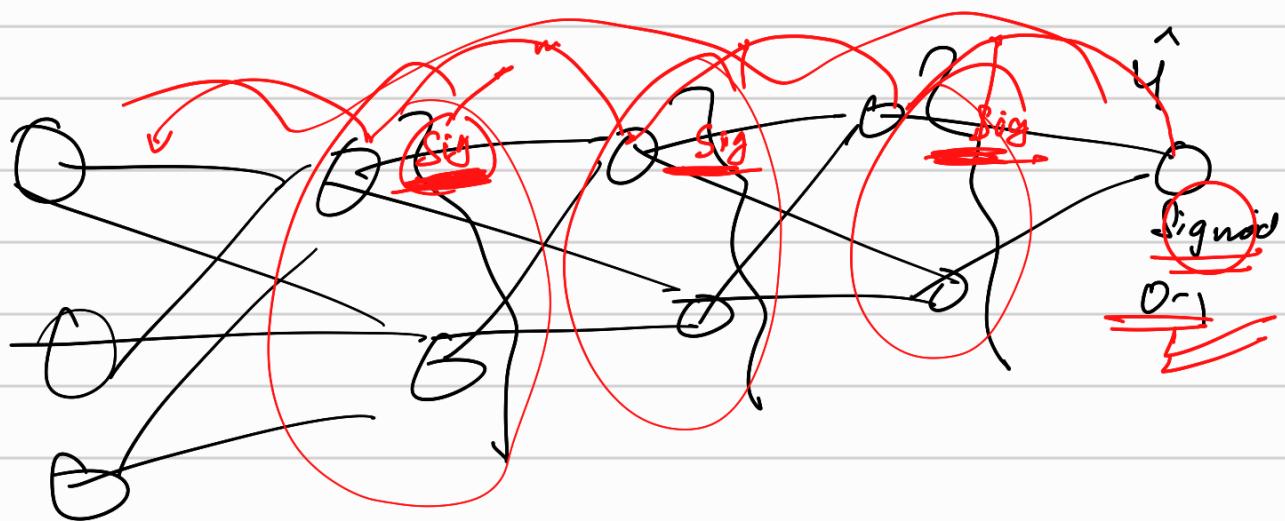
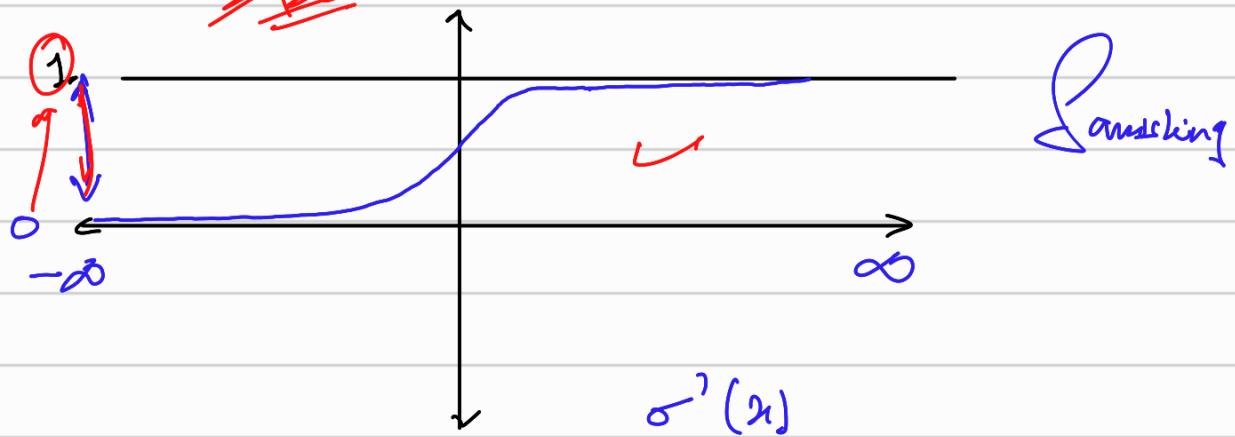
→ output layer

→ Sigmoid

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

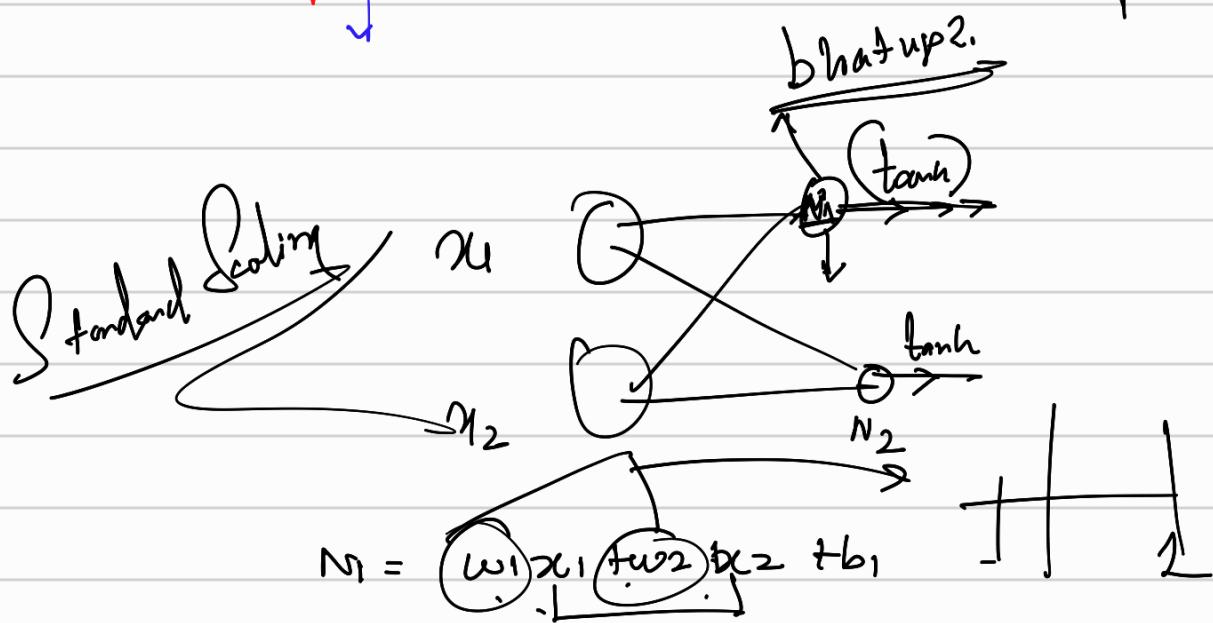
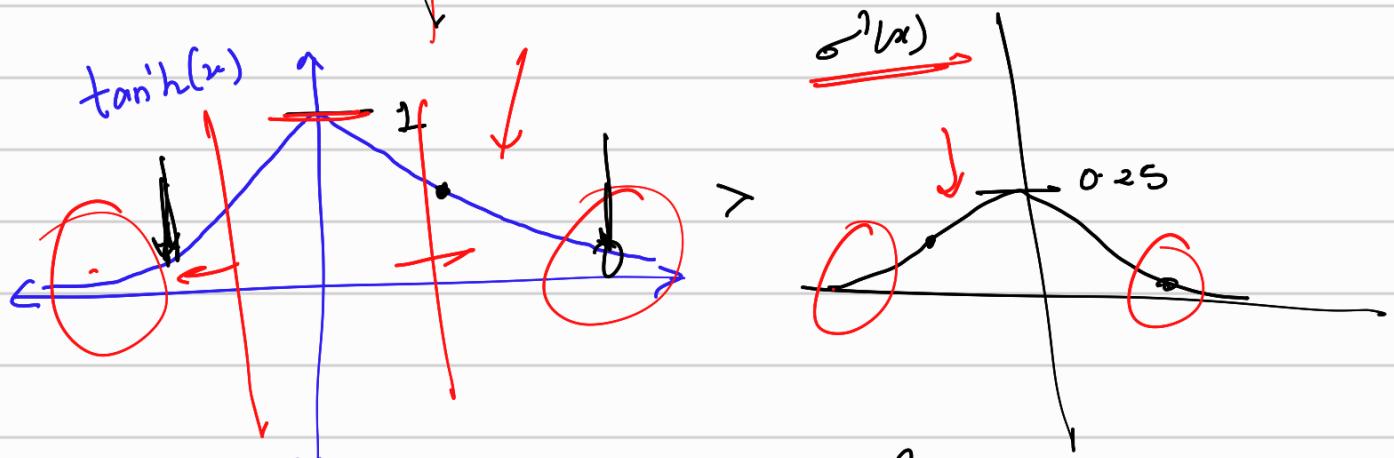
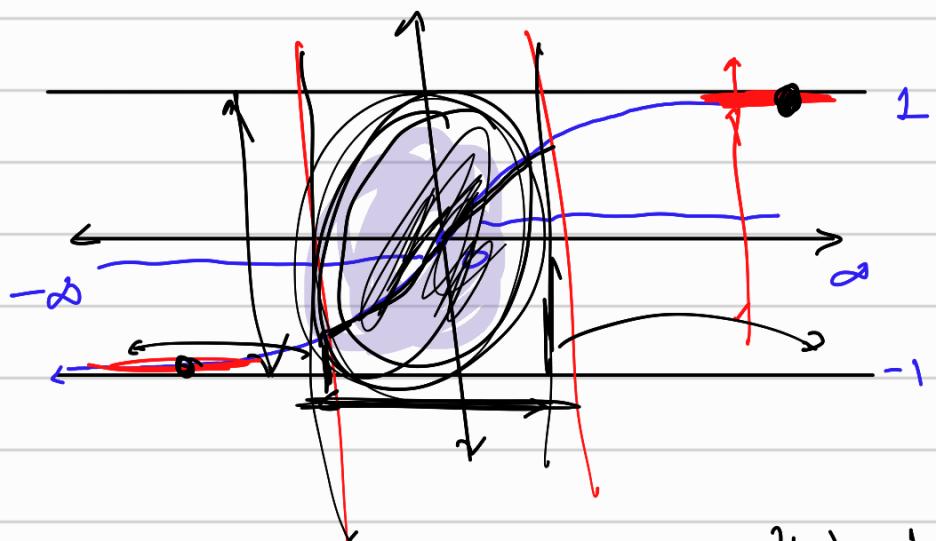
$-\infty < x < \infty$

HLY \rightarrow $0 < \sigma(x) < 1$

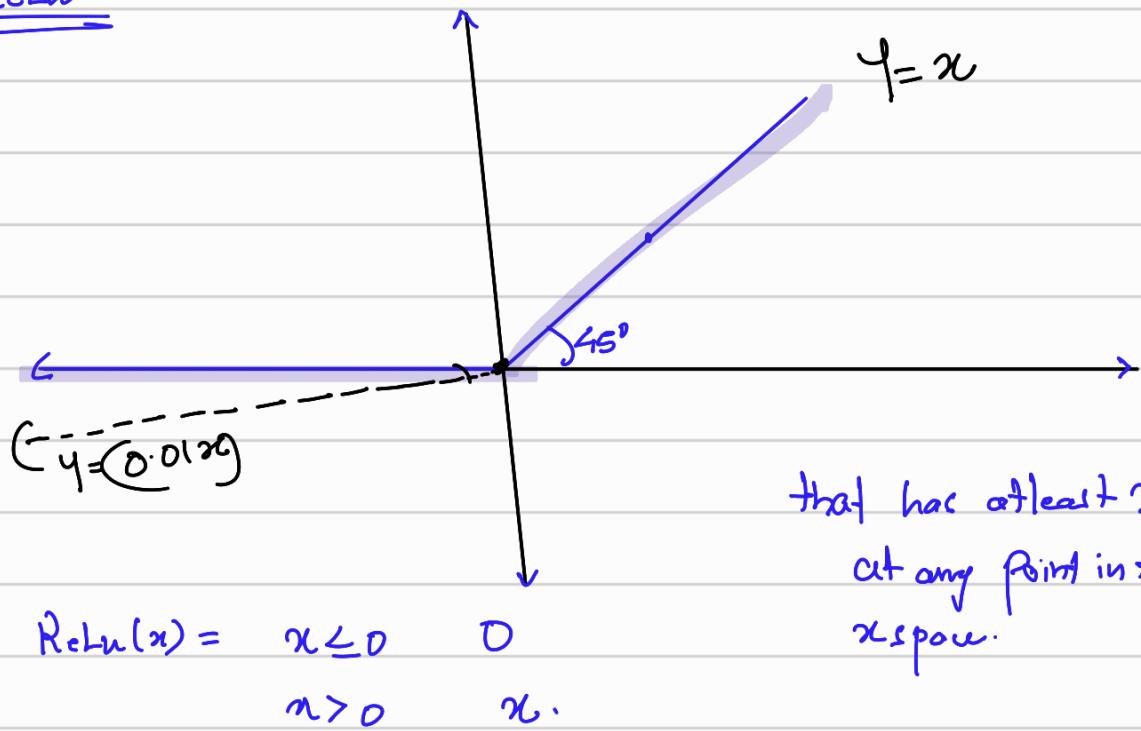


$\tanh \rightarrow$ hyperbolic tangent.

$$\tanh = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$



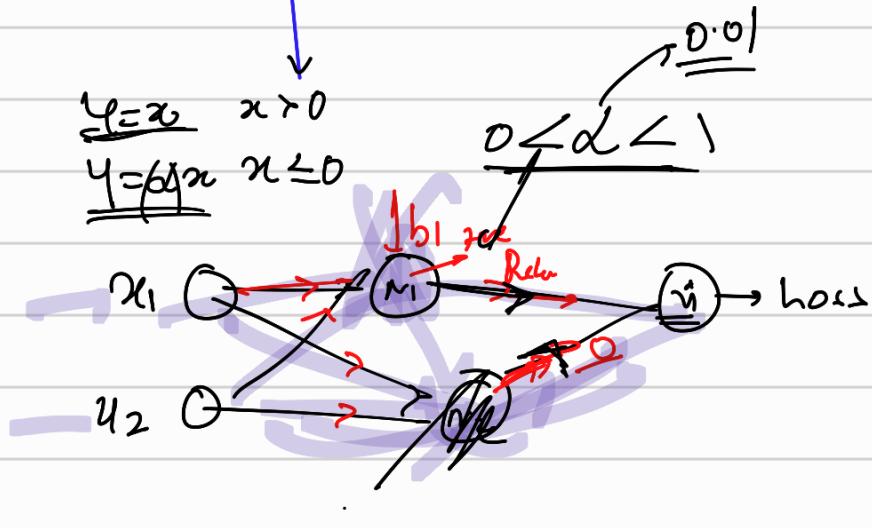
ReLU



that has at least 2 slopes
at any point in the
 x -spac.

↳ $\text{ReLU}(x) \Rightarrow x \leq 0 \oplus \underline{0.01x}$
 $x > 0 \quad 1x$

Parameteric ReLU



Deep Neural network

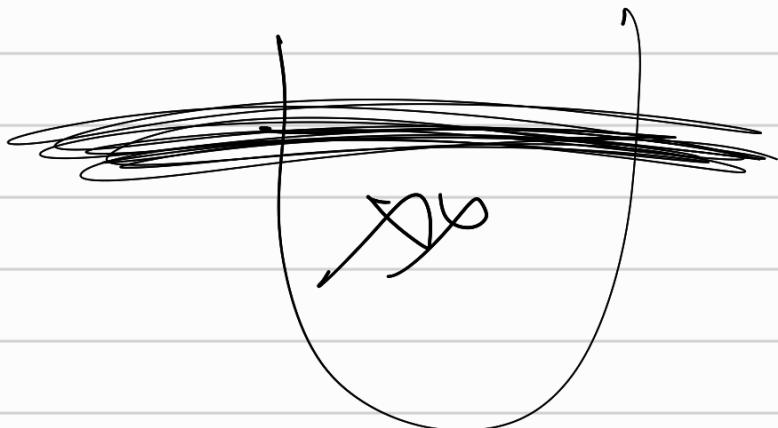
$$(2)^{\cancel{0} + \cancel{10}} = 32$$

a a a a

 2 2 2 2

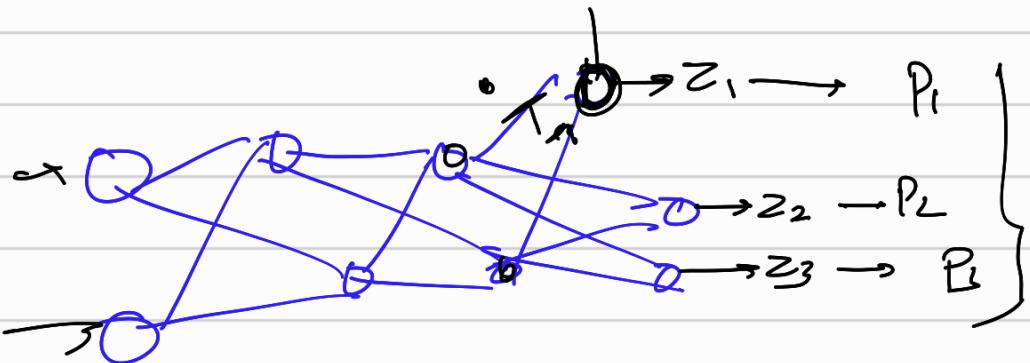
$$(2)^3 \quad \underline{(2)^{100}}$$

$$\frac{dh}{dx_1} = w_r - \alpha \left(\frac{1}{g_m} \right)$$



Softmax

: needs to be applied at the o/u of a multiclass Problem

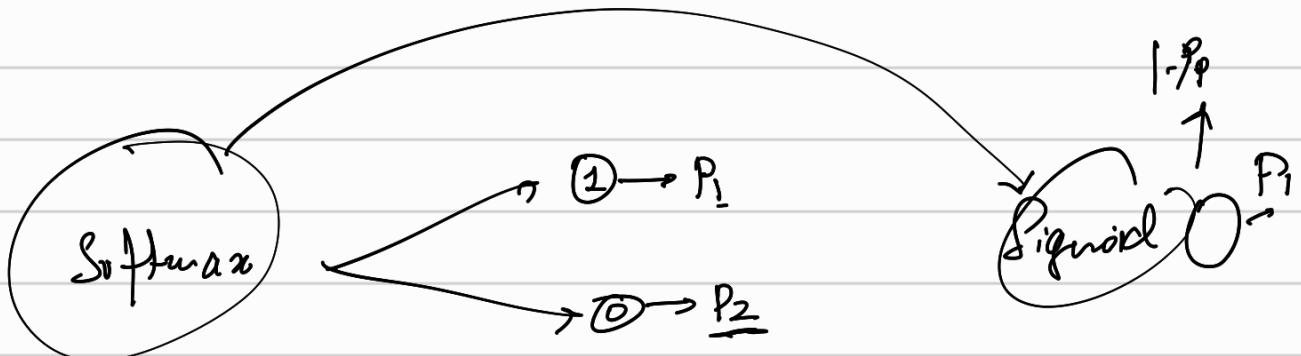


$$p_1 + p_2 + p_3 = 1$$

$$p_1 = \frac{e^{z_1}}{e^{z_1} + e^{z_2} + e^{z_3}}$$

$$p_2 = \frac{e^{z_2}}{e^{z_1} + e^{z_2} + e^{z_3}}$$

$$p_3 = \frac{e^{z_3}}{e^{z_1} + e^{z_2} + e^{z_3}}$$



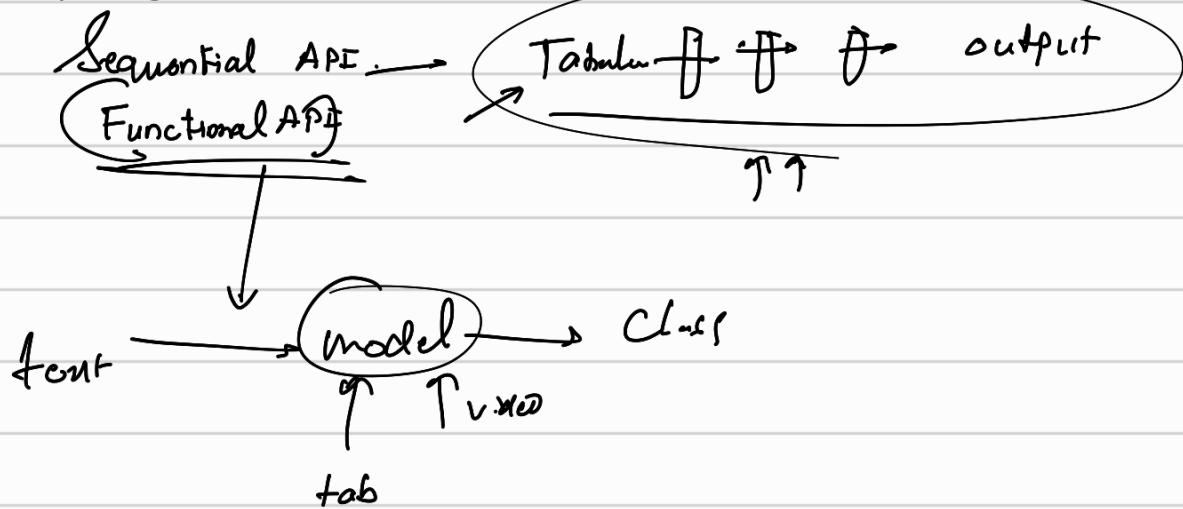
$$P_1 + P_2 = 1$$

$$P_1 = 1 - P_2$$

$$P_2 = 1 - P_1$$

Bocah

lecture -5:



Spec.

$$y \rightarrow LE(0, 1, 2)$$

$$(y=1)$$

$$\hat{y} = [0.1 \quad 0.3 \quad 0.6]$$

$$-\log(\hat{y}[y])$$

$$-\log 0.3$$

$$y \rightarrow DCE.$$

$$y \rightarrow [0, 1, 0].$$

cce.

$$\hat{y} = [0.1 \quad 0.3 \quad 0.6]$$

$$y = [0 \quad 1 \quad 0].$$

$$- [0 \times \log 0.1 + 1 \times \log 0.3 + 0 \times \log 0.6]$$

$$= - \log 0.3$$

$$\frac{\partial f}{\partial x_1} = 0$$

$$\frac{\partial f}{\partial x_2} = 0$$

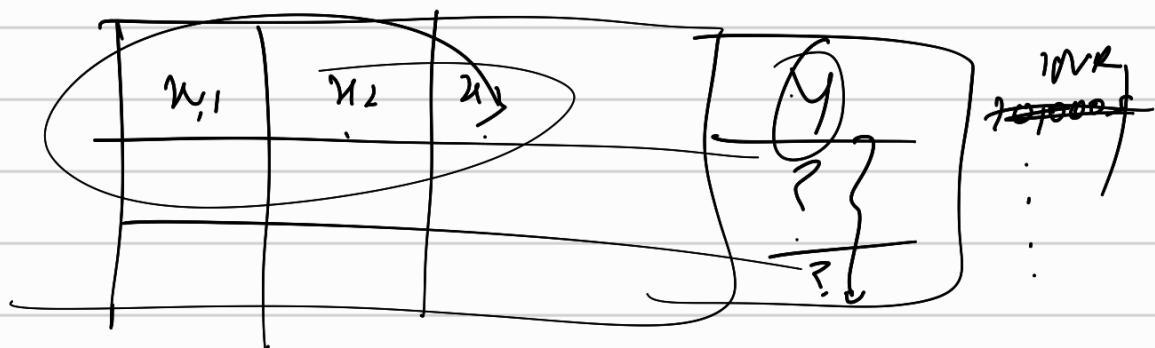
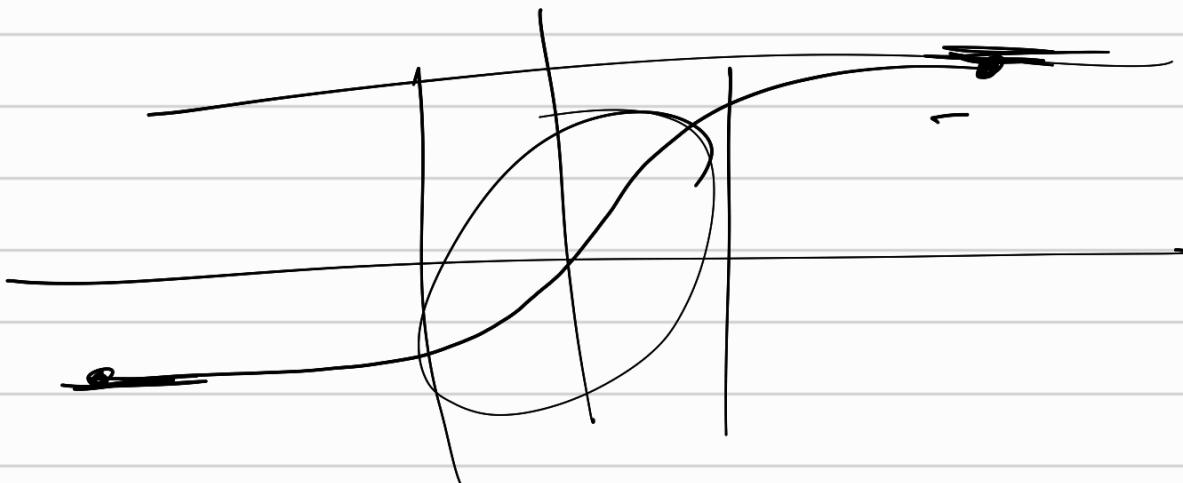
$$\frac{\partial f}{\partial x_3} = 0$$

Owner	Kms	Mileage	y
w_1	w_2	w_3	
$\frac{\partial f}{\partial w_1}$	$\frac{\partial f}{\partial w_2}$	$\frac{\partial f}{\partial w_3}$	
$w_1 \times \text{Owner}$	$w_2 \times \text{Kms}$	$w_3 \times \text{Mileage}$	

$$w_1 \times \text{Owner} + w_2 \times \text{Kms} + w_3 \times \text{Mileage}$$

(1) Honda City
50,000 ↘

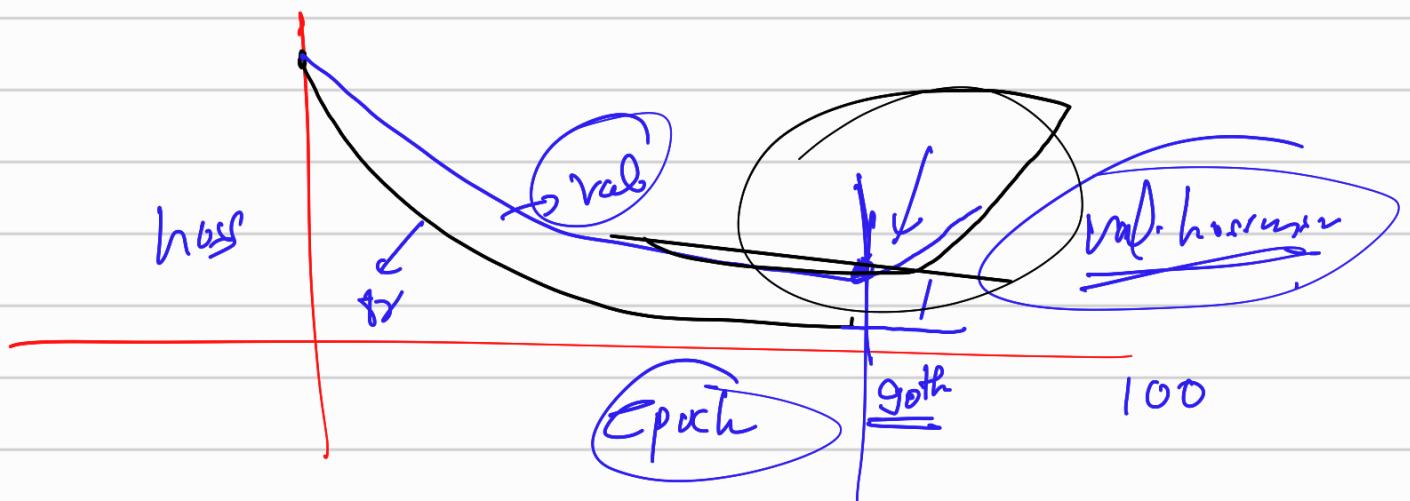
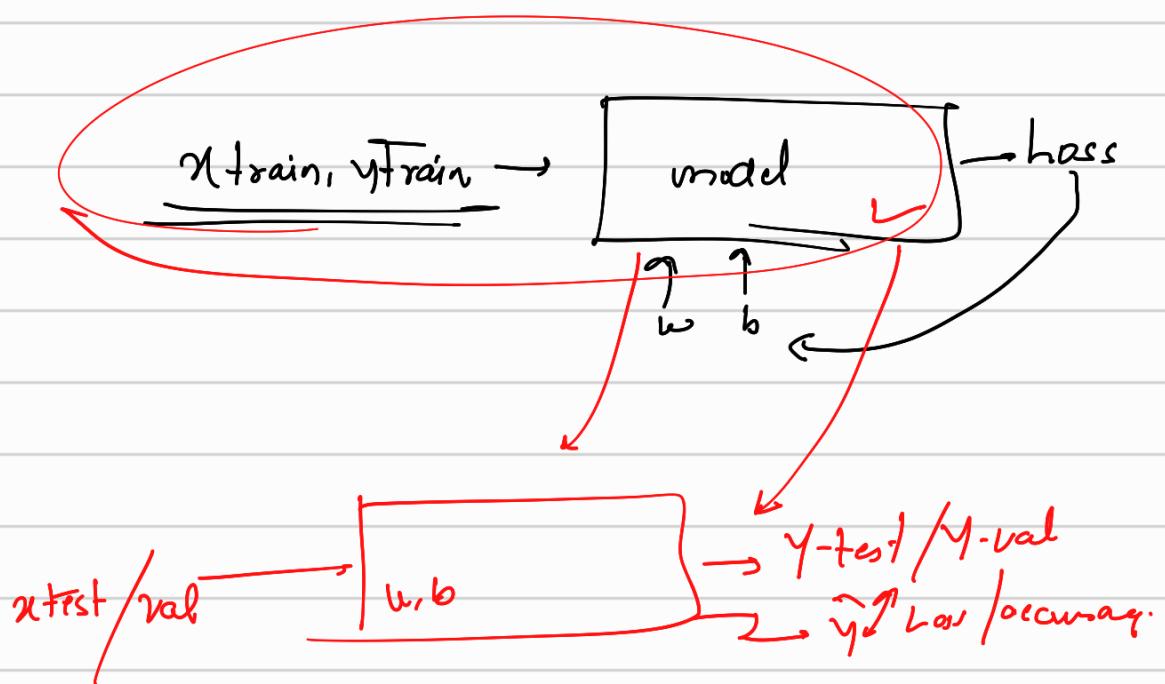
(2) Honda City
40,000



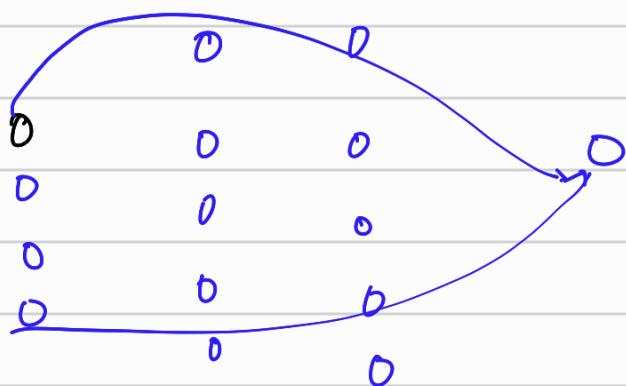
3,-3
($w_1 \sim -1, 1$)

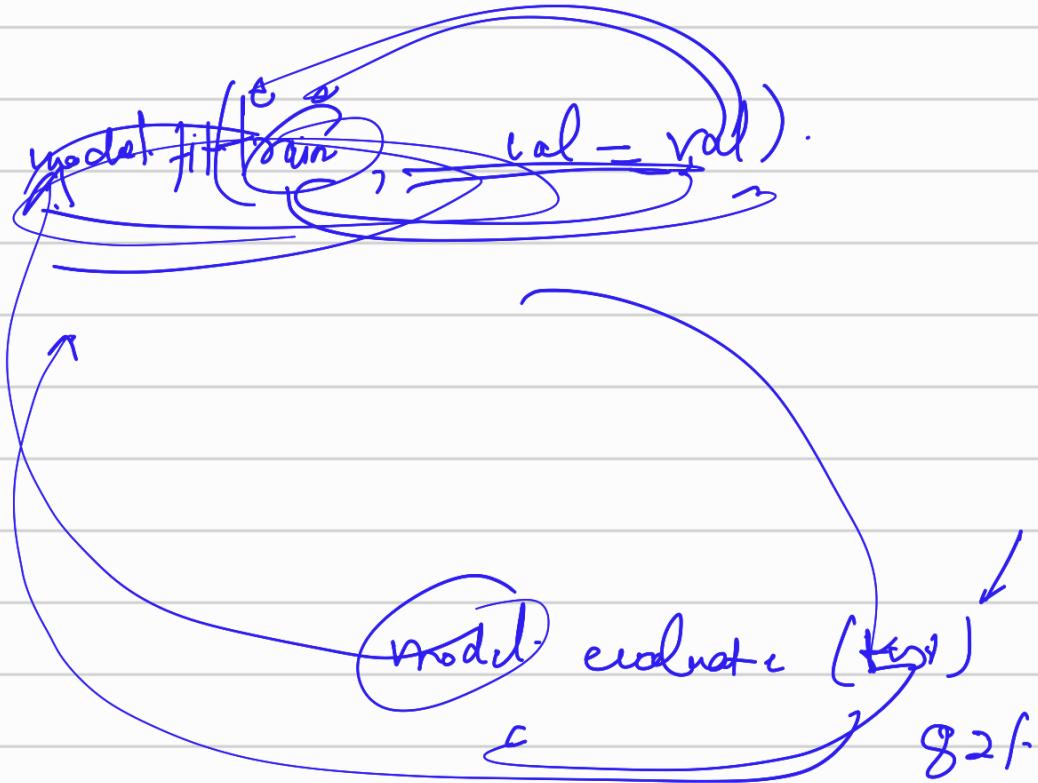
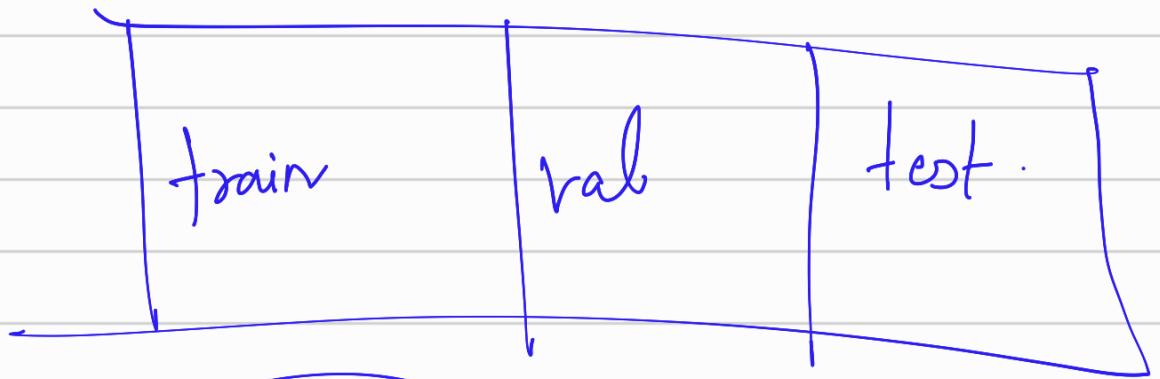
3,-3 → (2-1) → 0 → 5,00,000
3,-1 → 0 → 0 → 5,00,000

$$L = \left(\frac{5,00,000 - 0 \cdot 9}{2} \right)^2$$



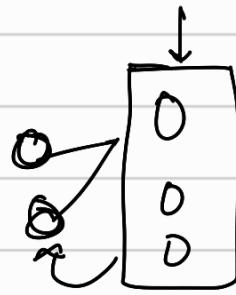
Callbacks



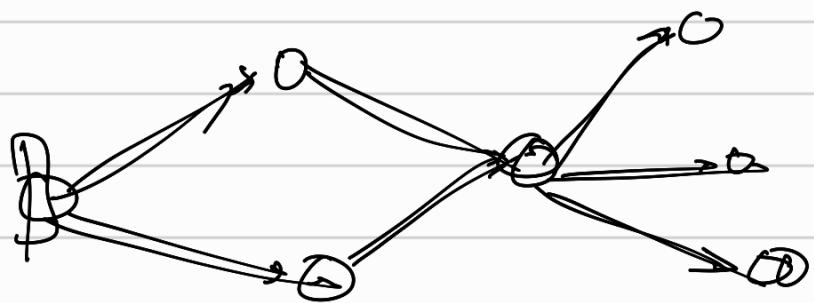


Functional API:

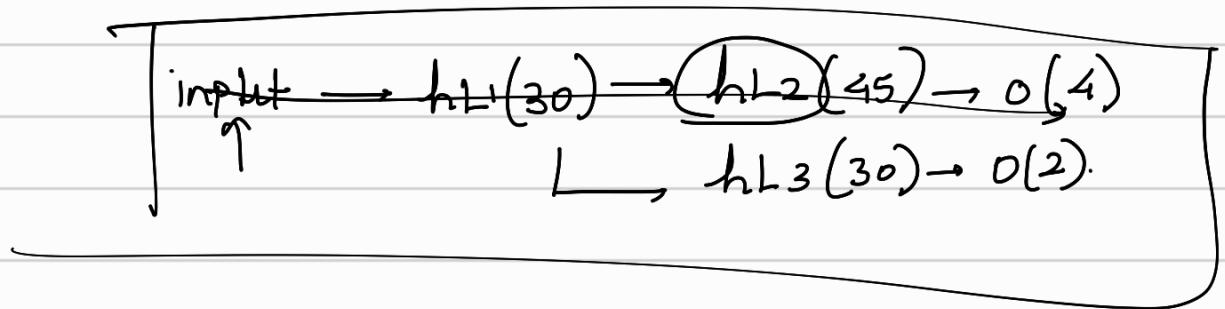
Graph Structure



any layer (ϵ)



$D \rightarrow D$



Input $\xrightarrow{(4)} hL_1(30) \rightarrow hL_2(2)(40)$ Concatenate

Case 1: input $\xrightarrow{(4)} hL_1(20) \rightarrow hL_2(2) \rightarrow 50$

hL3

Case 2:

inp1 = Input(4,)

inp2 = Input(4,)

hL1 = Dense(30)(inp1)

hL2 = Dense(40)(hL1)

hL3 = Dense(20)(inp2)

hL4 = Dense(50)(hL3)

o = Concatenate([hL2, hL4])

Case 1:

inputs = Input(4,)

hL1 = Dense(30)(inputs)

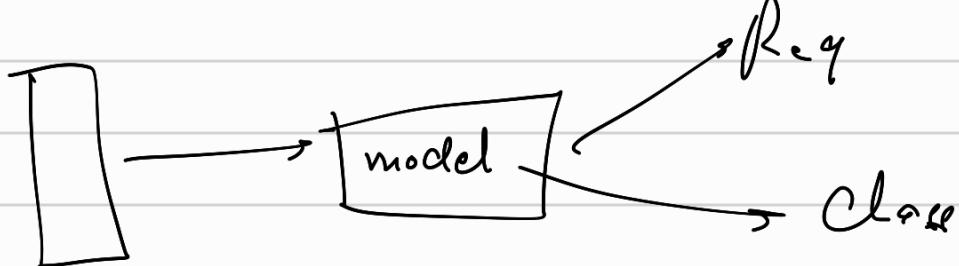
hL2 = Dense(45, activation='relu')(hL1)

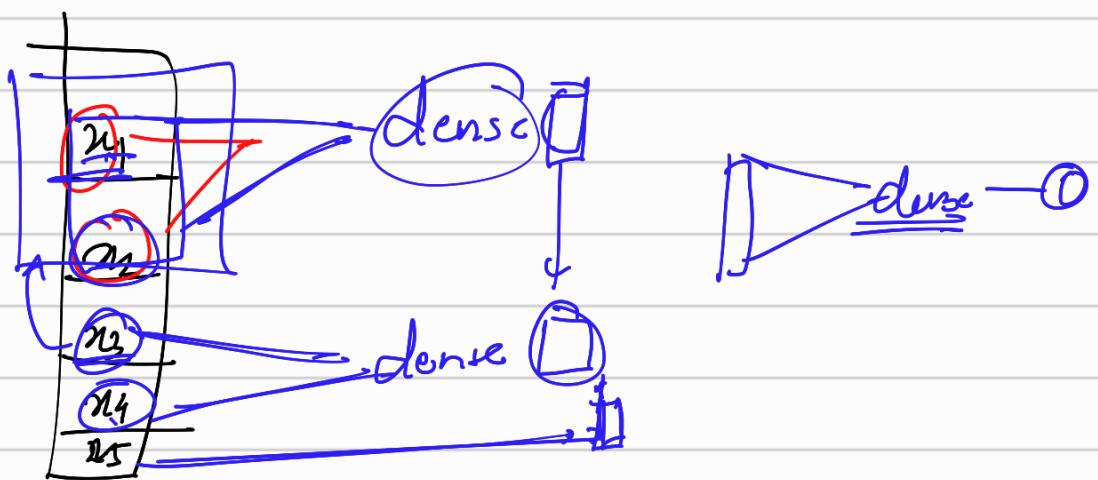
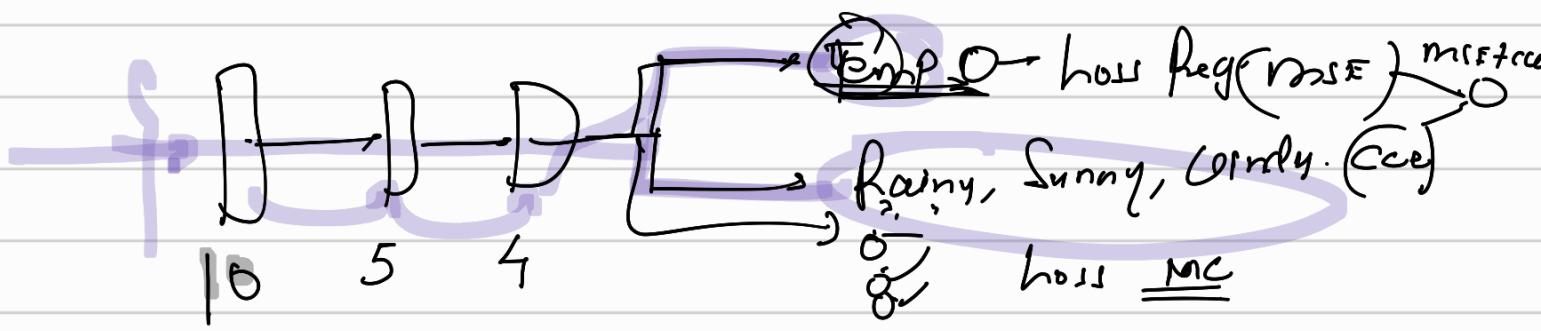
o1 = Dense(4)(hL2)

hL3 = Dense(30)(hL1)

o2 = Dense(2)(hL3).

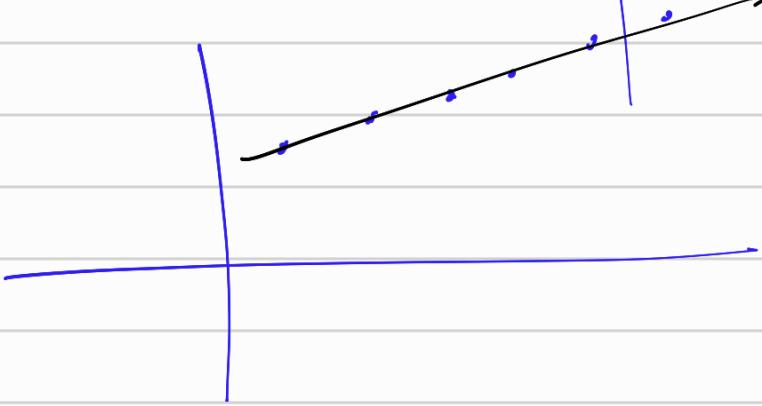
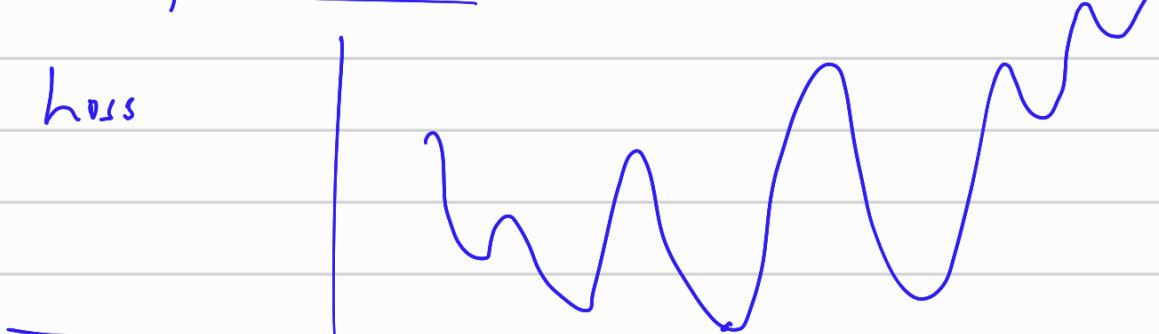
T

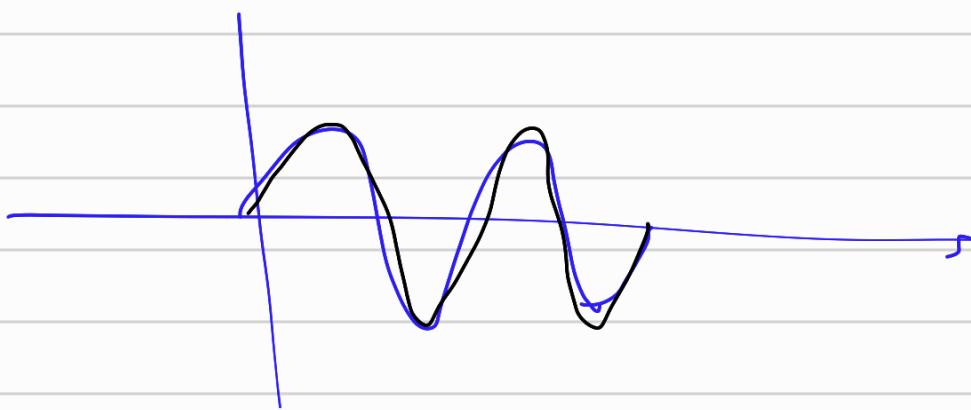
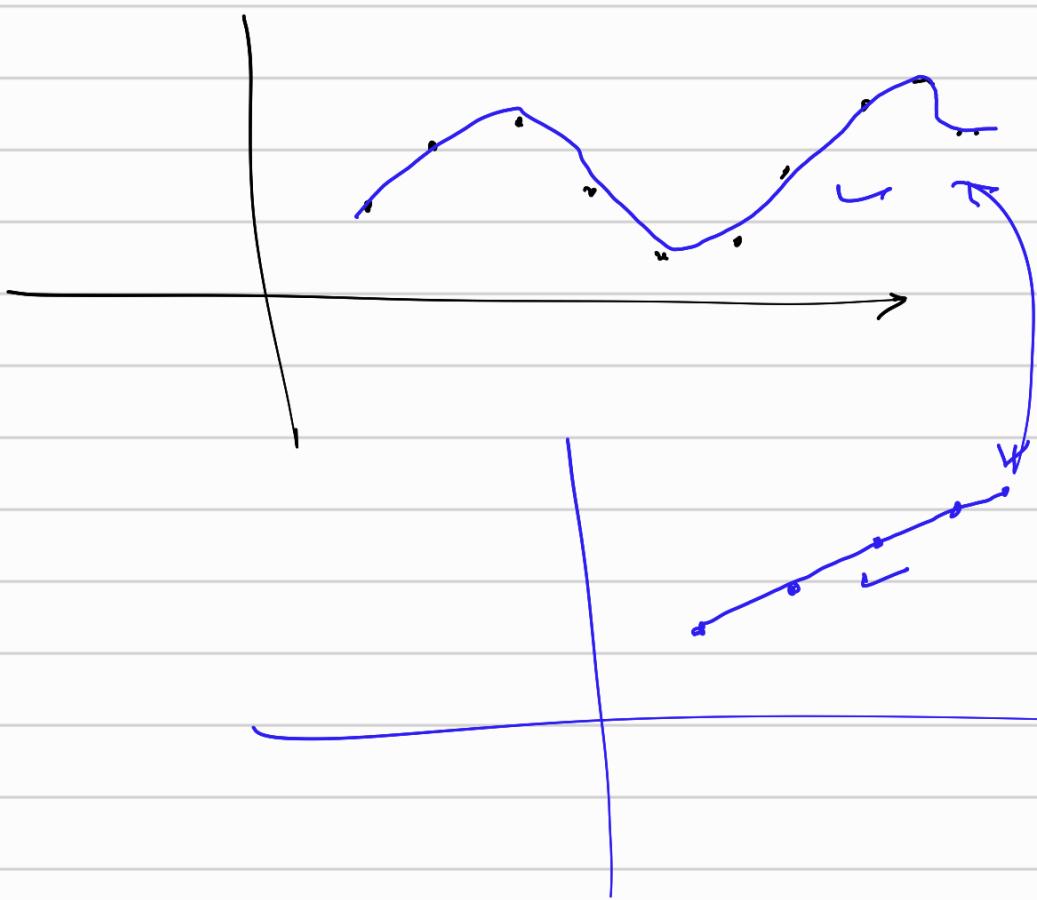
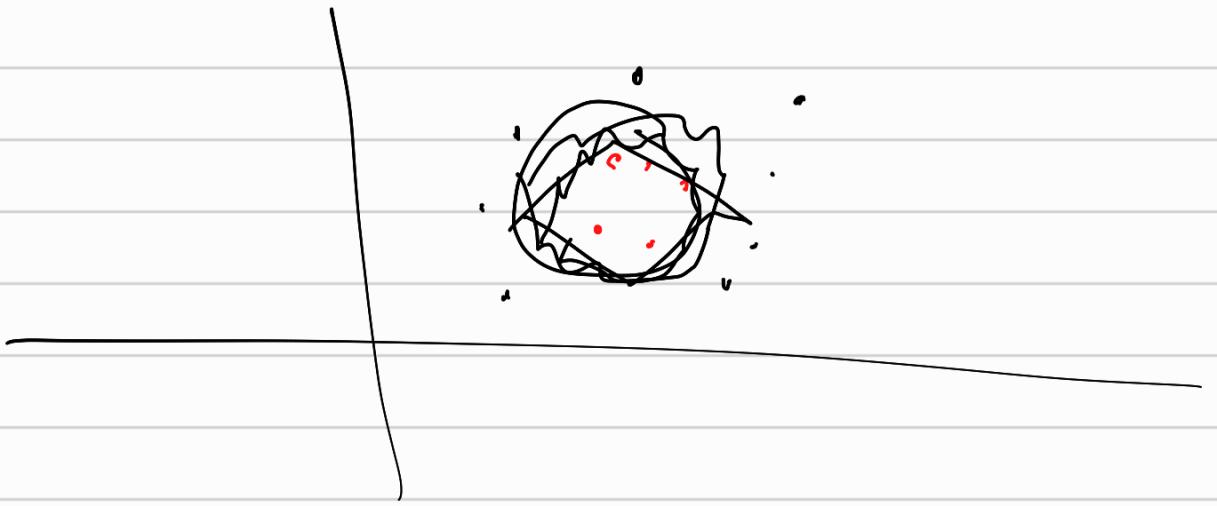




Optimizers

loss





gg.999).



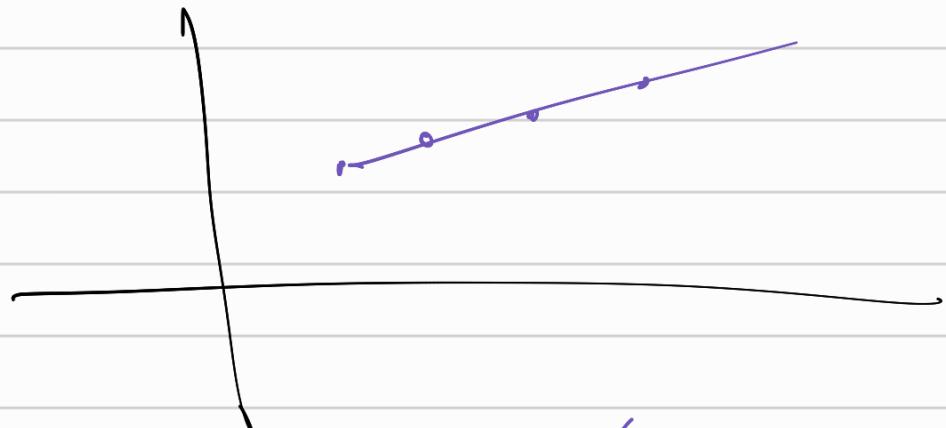
(Regularization.)

- Regularization
- ① Dropout.
- ① Optimizers.
- ① Callbacks
- ① Batch Normalization
- ① weight initialization

Hyperparameter Tuning.

Regularization:

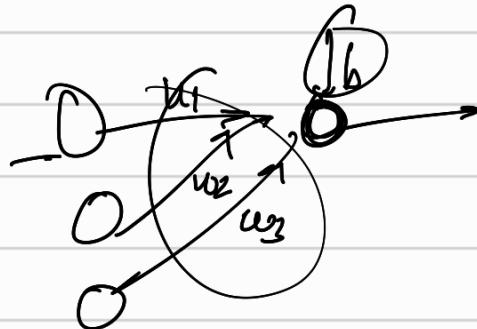
↓↓
(Ridge and Lasso.) ?



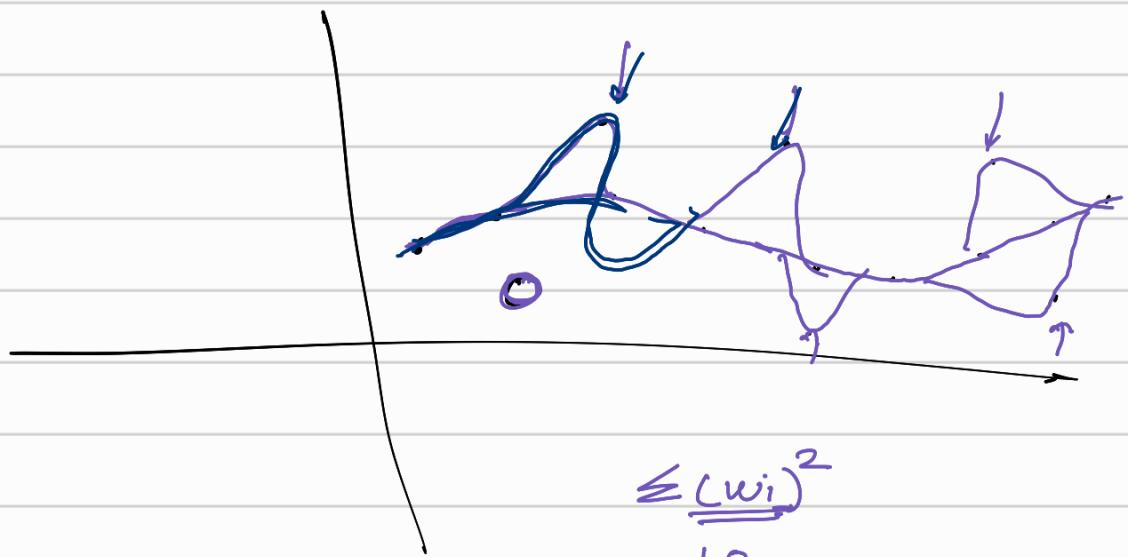
Overfit → has high variance (model is extremely sensitive to changes in n dict.).

$$\textcircled{1} \quad y = \underline{100x_1} + \underline{500x_2} \quad y=1000$$

$$\textcircled{2} \quad y = 2x_1 + x_2 = \textcircled{2}$$



$$N1 = \underline{w_1 x_1} + \underline{w_2 x_2} + \underline{w_3 x_3} + b_1$$



$$\begin{aligned}
 TL & \quad mSE \\
 510 & = 500 \\
 425 & = 400 \\
 375 & = 320 \\
 355 & = 260 \\
 320 & = 200 \\
 305 & = 150 \\
 \hline
 345 & = 145
 \end{aligned}$$

$$\leq \underline{(w_i)^2}$$

10

25

55

95

= 120

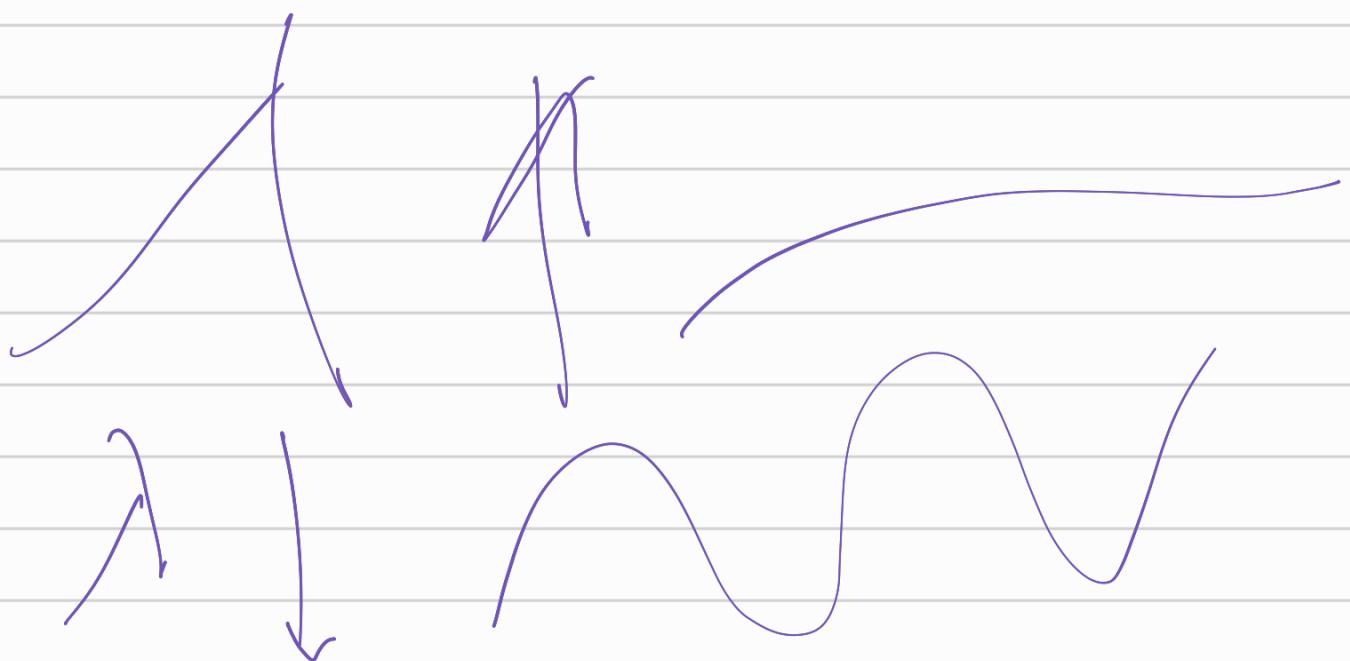
$$\begin{aligned}
 &= \frac{155}{200} \\
 &= \underline{\underline{200}}
 \end{aligned}$$

$$TL = \text{mse} + \boxed{\frac{1}{g}} \sum_{j=1}^p w_j^2$$

Lagrangean multiplier

↓
Hyperparameter (ο - ∞)

↓
~~Val-loss~~



$\lambda =$

$\lambda =$ $\boxed{0.9}$

$\lambda =$ $\boxed{0.95}$

$\lambda =$ ~~1.0~~

$\rightarrow 92\%$

$\rightarrow 95\%$

$\rightarrow 95\%$

Reg.

$$T_L = \text{mse} + \lambda \sum w_j^2 \quad \text{Ridge } (L^2).$$

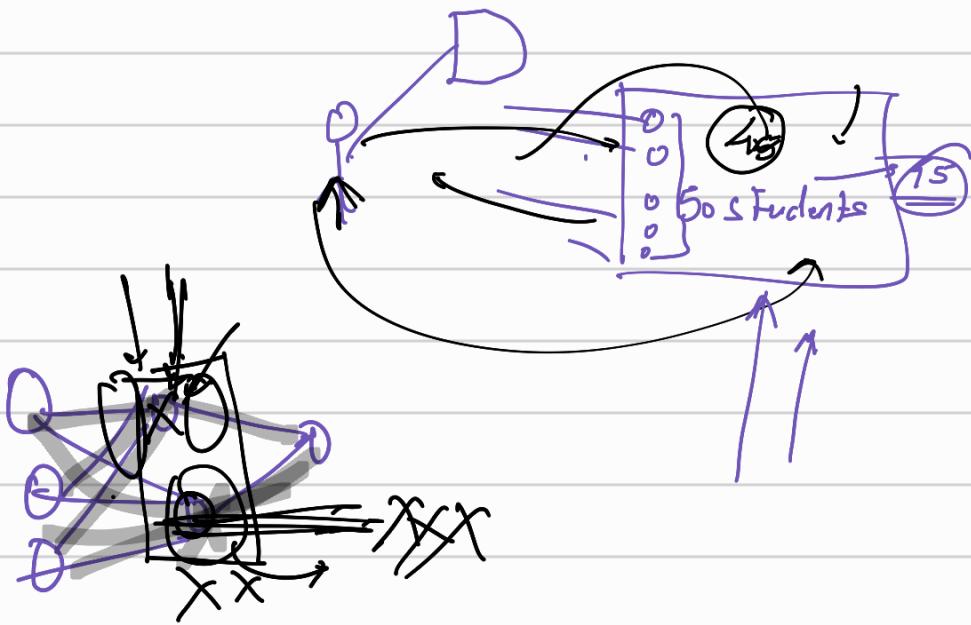
$$T_h = \text{mse} + \lambda \sum |w_j| \quad \text{Lasso } (L_1).$$

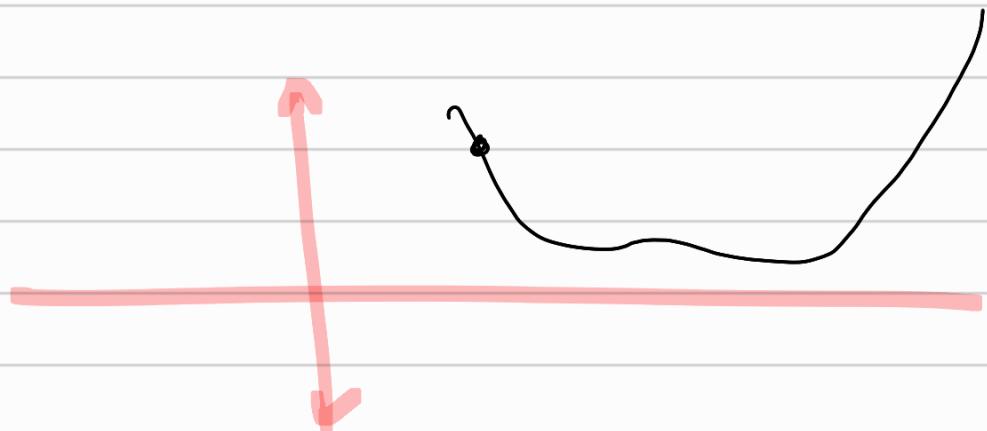
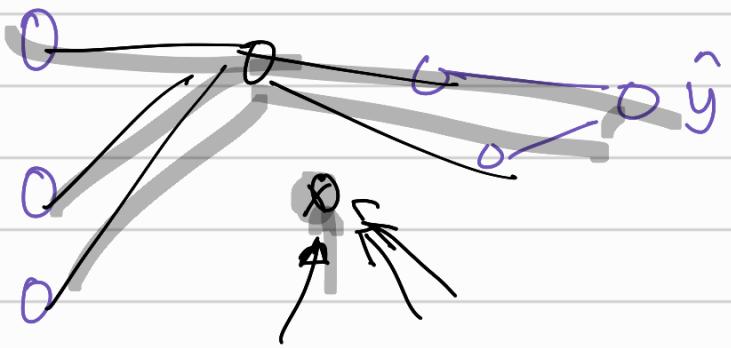
$$T_L = \text{mse} + \lambda_1 (\sum w_j^2) + \lambda_2 \sum |w_j| \quad L_1 L_2$$

Elastic Net

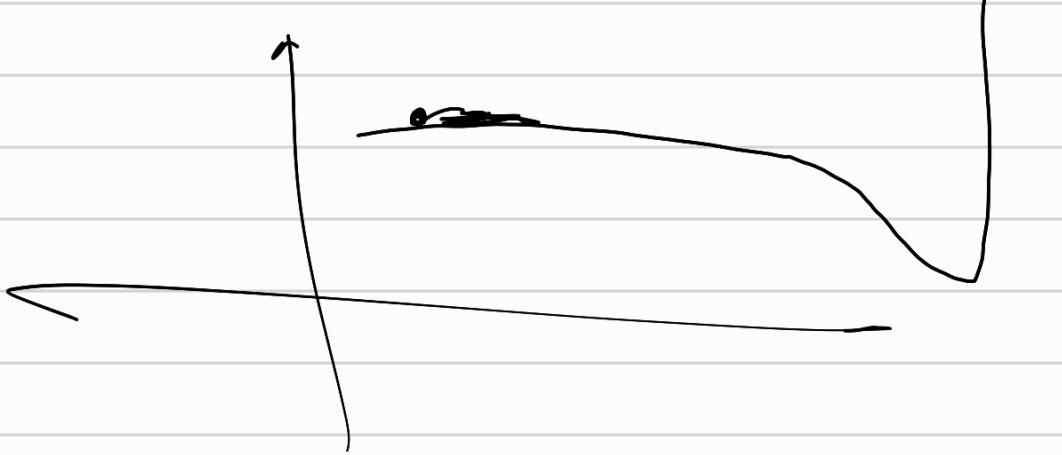
Break!

Dropout:



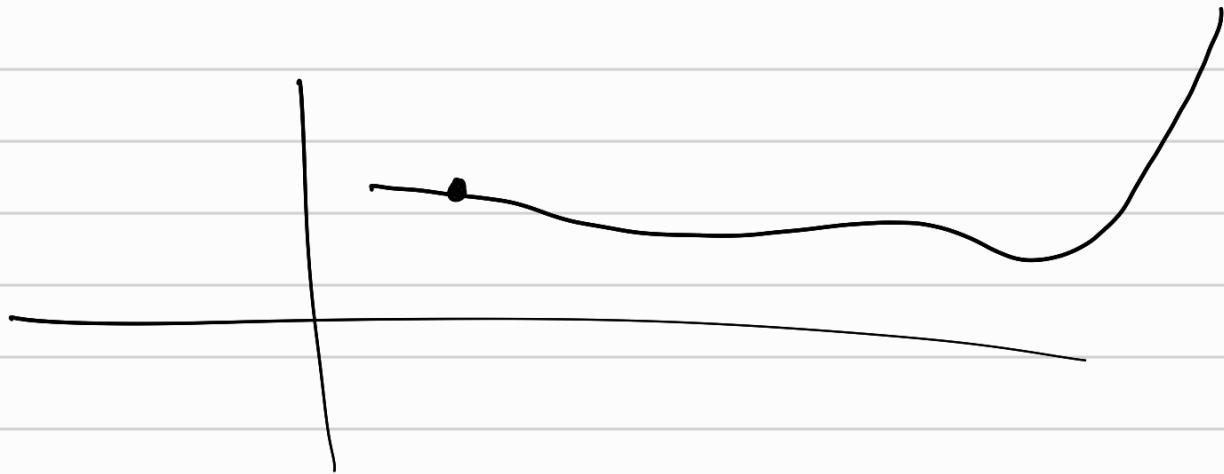
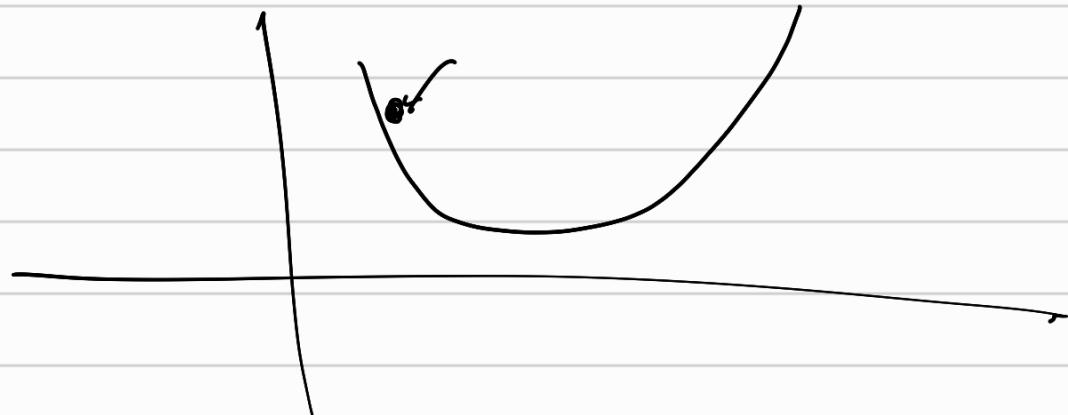
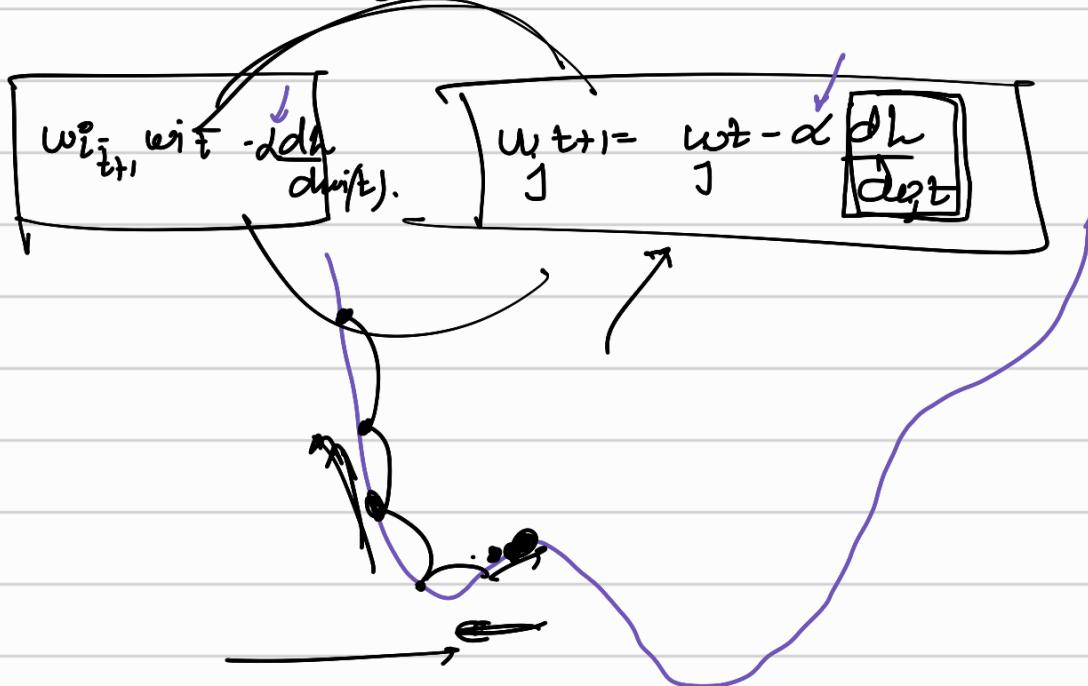


10

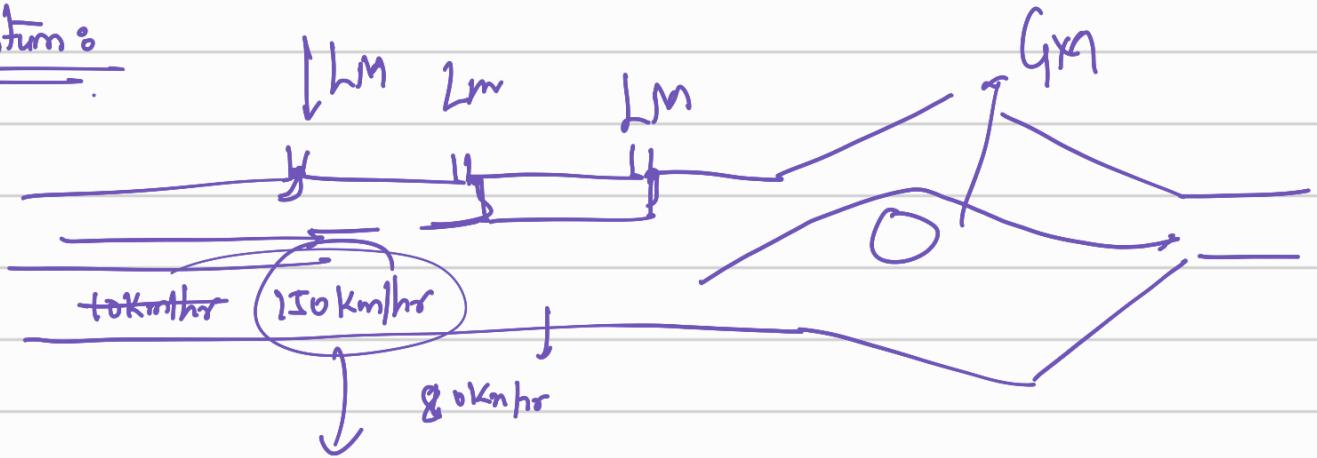


optimizers:

minimizing the loss by changing the weights and biases.



Momentum:



Momentum

GD

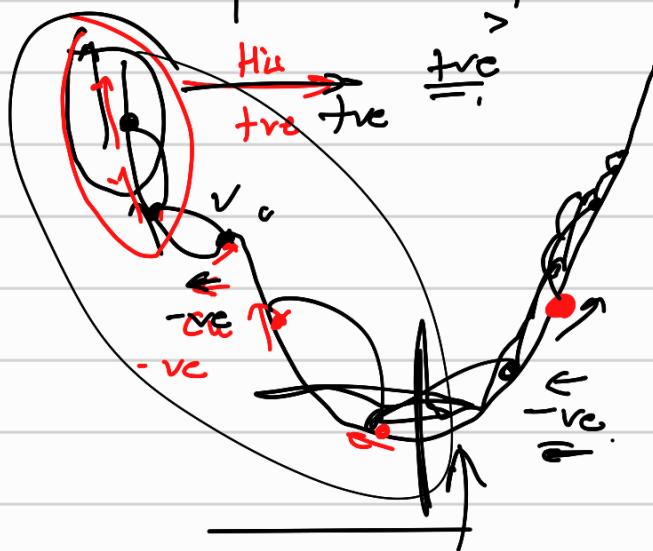
$$w_{t+1} = w_t - \alpha \frac{\partial L}{\partial w_t}$$



GD with momentum

$$w_{t+1} = w_t - \text{update}_t$$

$$\text{update}_t = \beta \text{update}_{t-1} + (1-\beta) \frac{\partial L}{\partial w_t}$$



$t=2$ $t=1$ $t=2$. $t=3$

$$\text{update}_1 = (1-\beta) \alpha \frac{\partial L}{\partial w_2}$$

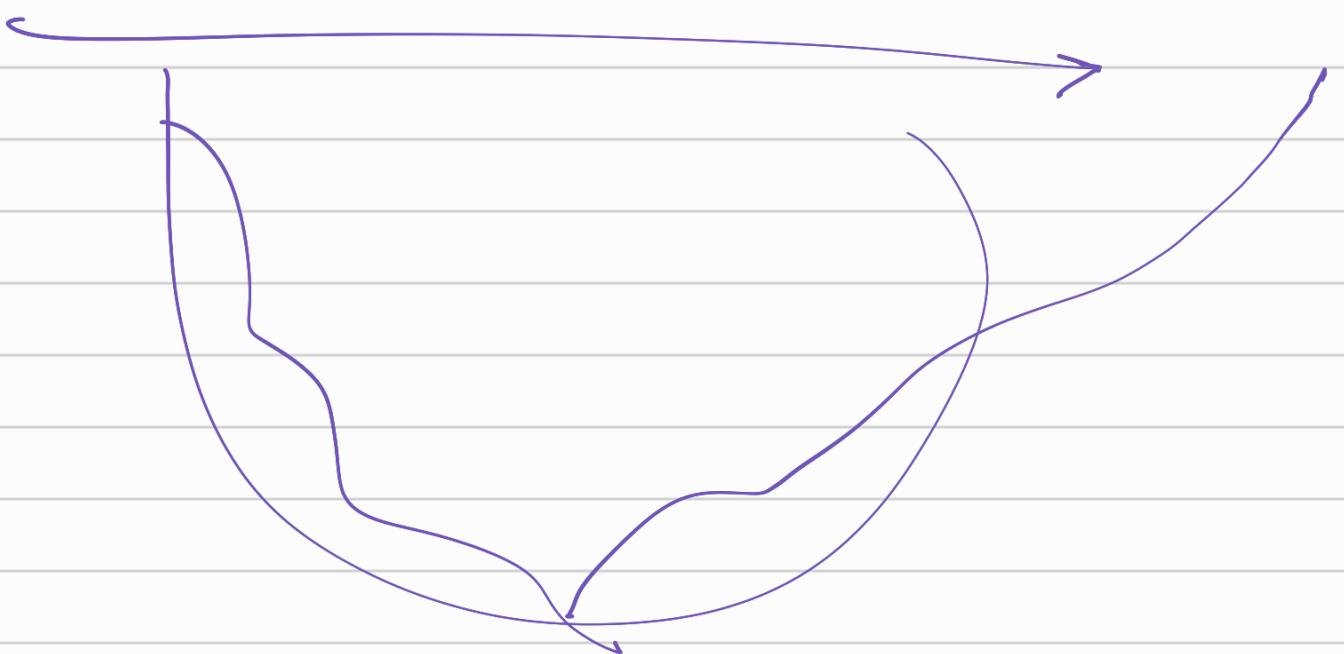
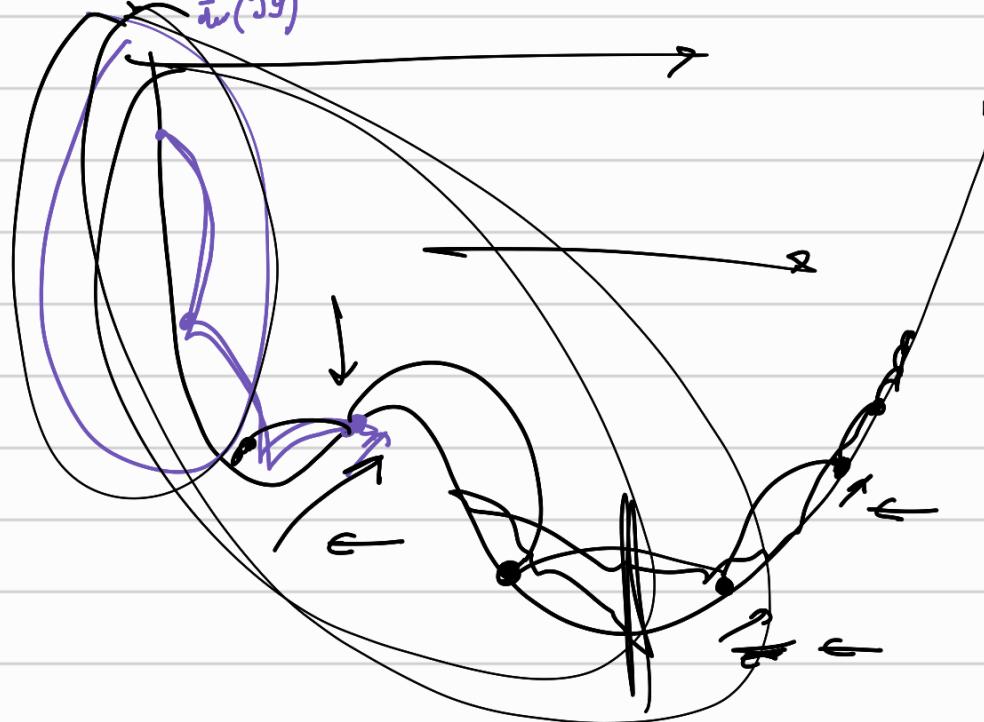
$$\text{update}_2 = \beta \text{update}_1 + (1-\beta) \alpha \frac{\partial L}{\partial w_2}$$

$$\text{update}_2 = \beta ((1-\beta) \alpha \frac{\partial L}{\partial w_1}) + (1-\beta) \alpha \frac{\partial L}{\partial w_2}$$

$$\text{update}_3 = \beta \text{ update}_2 + (1-\beta) \alpha \frac{\partial h}{\partial w_3}$$

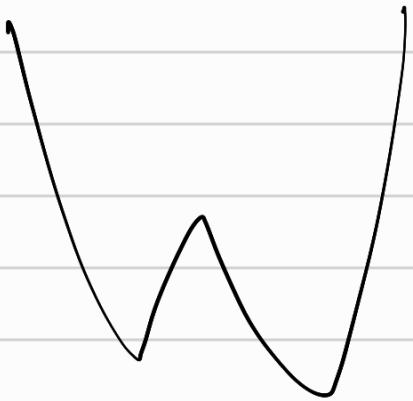
$$\text{update}_3 = \beta \left(\beta (1-\beta) \alpha \frac{\partial h}{\partial w_1} + (1-\beta) \alpha \frac{\partial h}{\partial w_2} \right) + (1-\beta) \alpha \frac{\partial h}{\partial w_3}$$

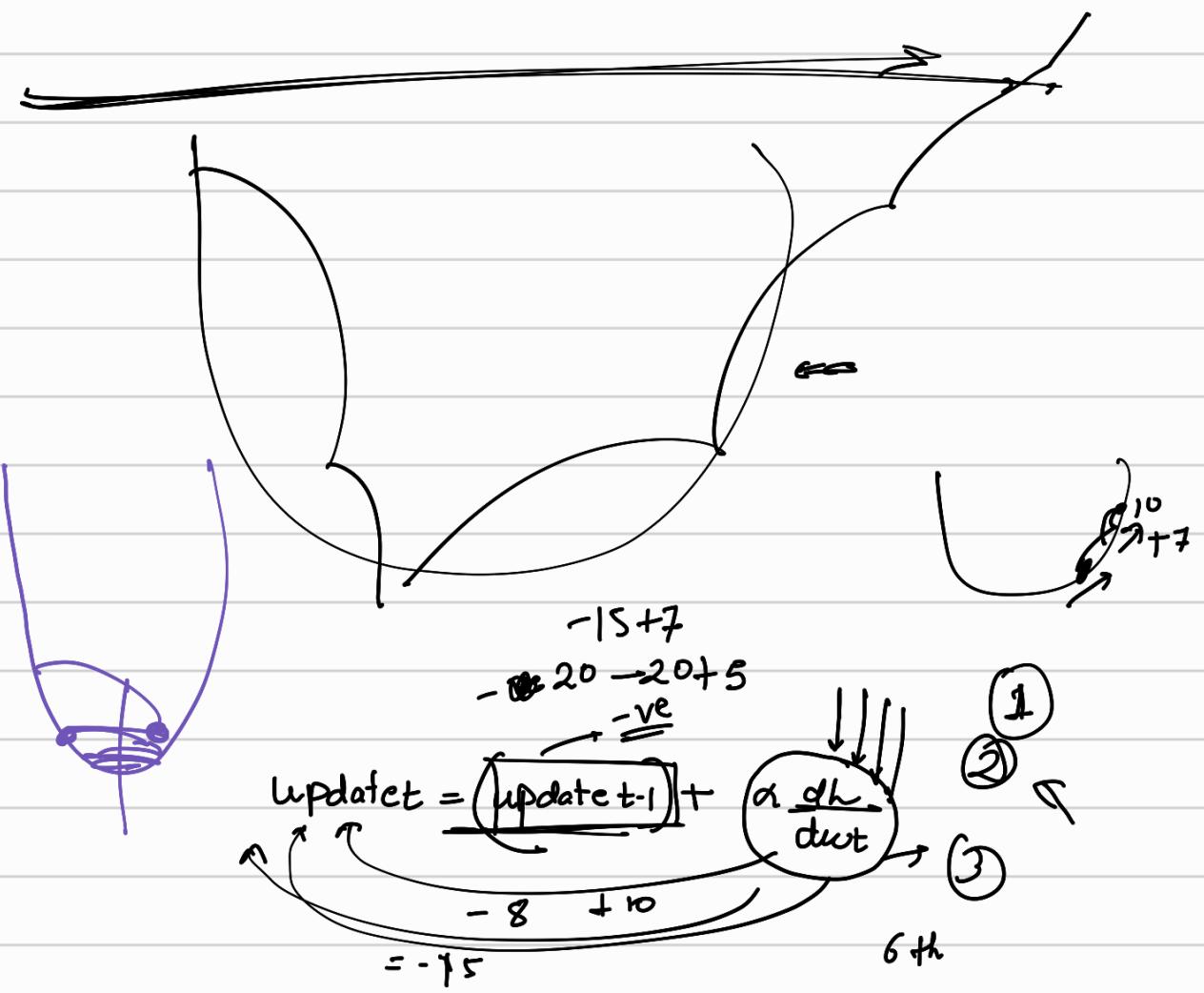
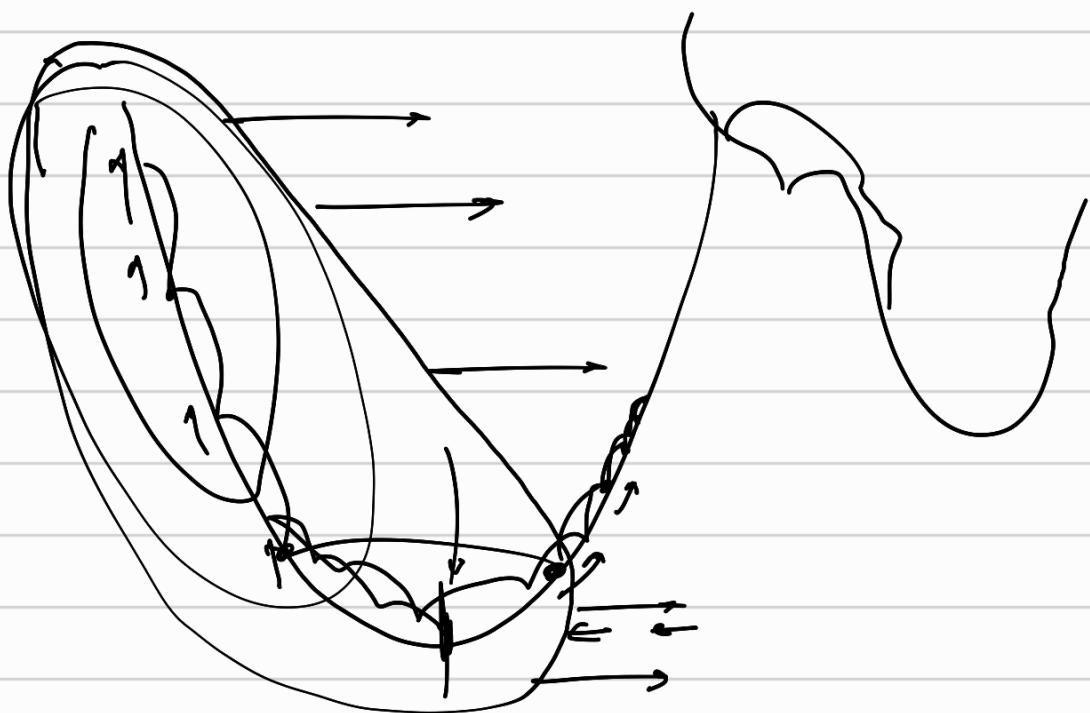
~~$\text{update}_3 = \beta \left(\beta (1-\beta) \alpha \frac{\partial h}{\partial w_1} + (1-\beta) \alpha \frac{\partial h}{\partial w_2} \right) + (1-\beta) \alpha \frac{\partial h}{\partial w_3}$~~

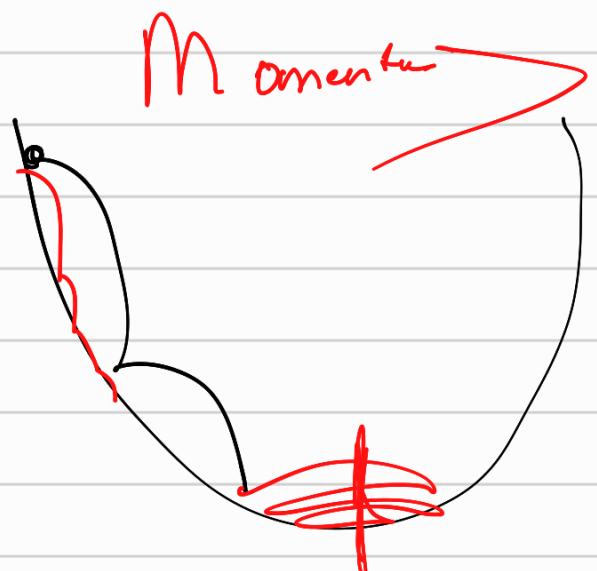
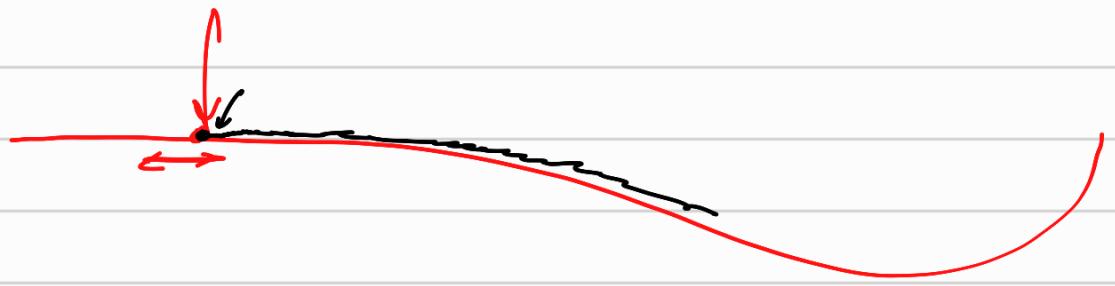
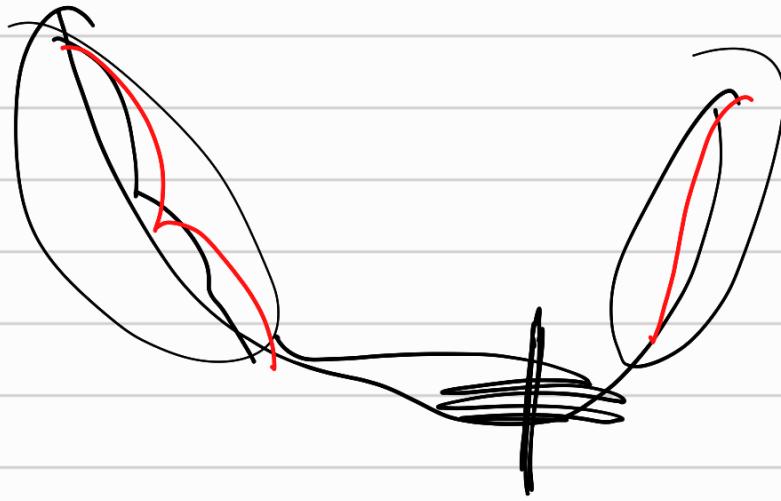


$i = i + \underline{\text{S}}$

EF







$$\alpha_0 = 0$$

$$\alpha_t = \alpha_{t-1} + s$$

$$\alpha_1 = 0 + s$$

$$\alpha_1 = s$$

$$\alpha_2 = \alpha_1 + s$$

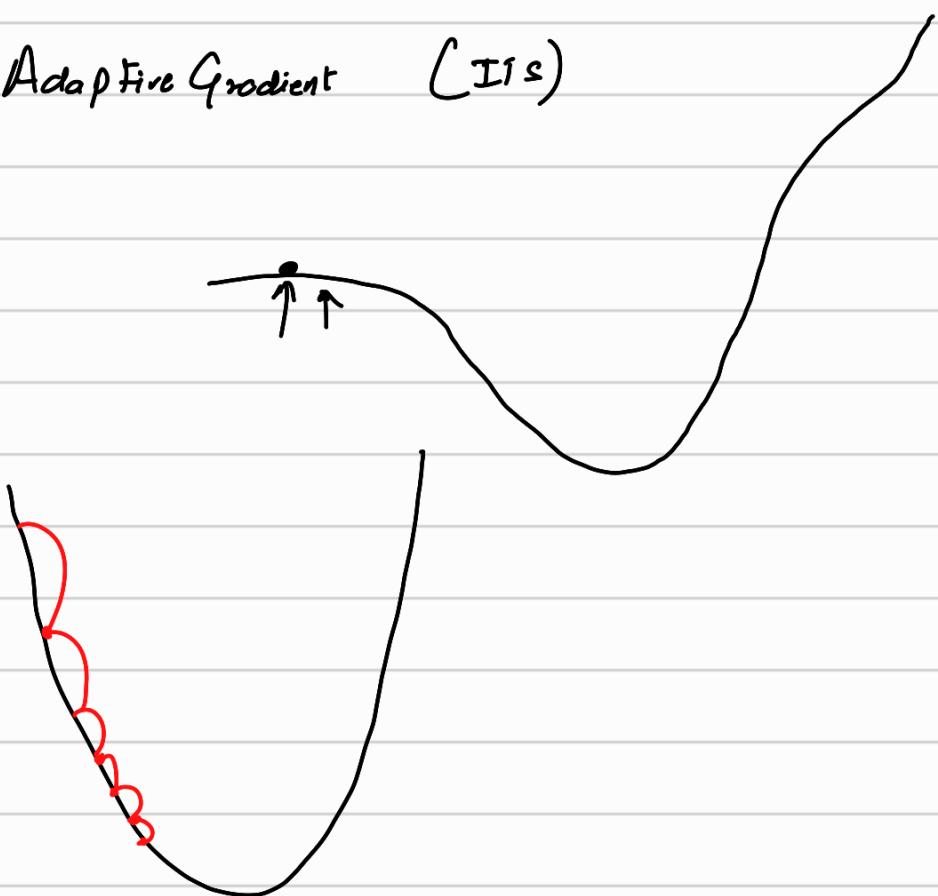
$$\alpha_2 = 10$$

$$\alpha_3 = \alpha_2 + s$$

$$\alpha_3 = (\alpha_1 + s) + s$$

Break!

Adagrad: Adaptive Gradient (It's)



$$w_{t+1} = w_t - \frac{\alpha}{v_{t+1}} \frac{dh}{d w_t}$$
$$v_t = v_{t-1} + \left(\frac{dh}{d w_t} \right)^2$$

Rms Prop

$$w_{t+1} = w_t - \frac{\alpha}{\sqrt{v_{t+\epsilon}}} \frac{\partial h}{\partial w_t}$$

v_t $\beta = 0.99$ $\epsilon = 0.01$ $\frac{\partial h}{\partial w_t}$

$$v_{t+1} = \beta v_t + \frac{(1-\beta)}{\tau} \left(\frac{\partial h}{\partial w_t} \right)^2$$

Adam:

Adaptive moments

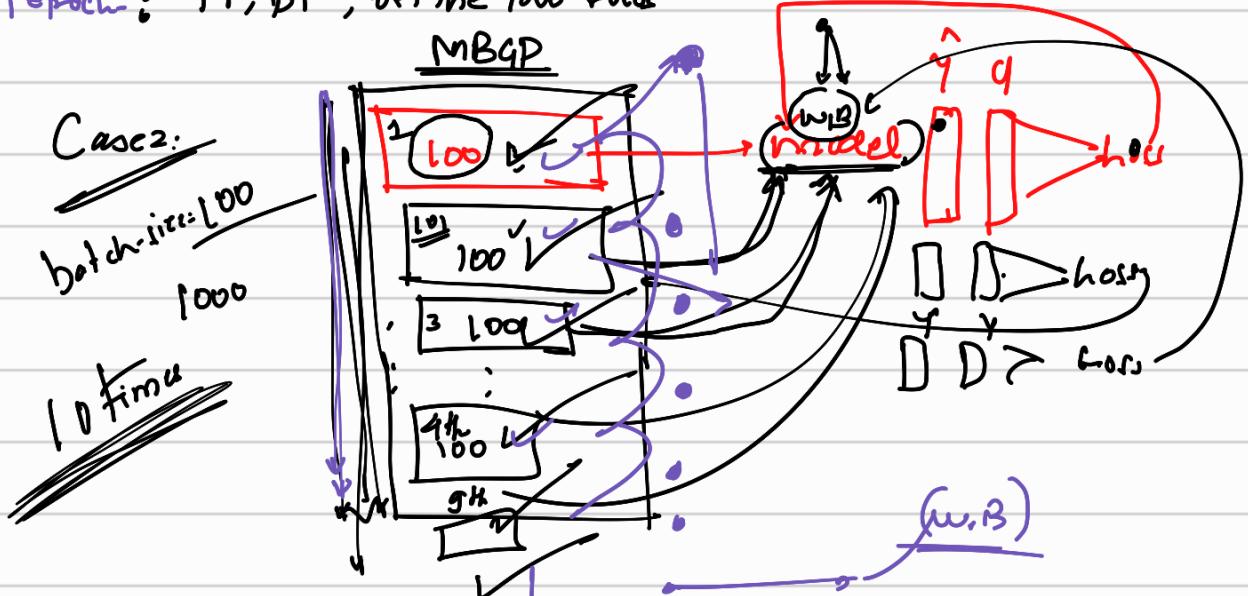
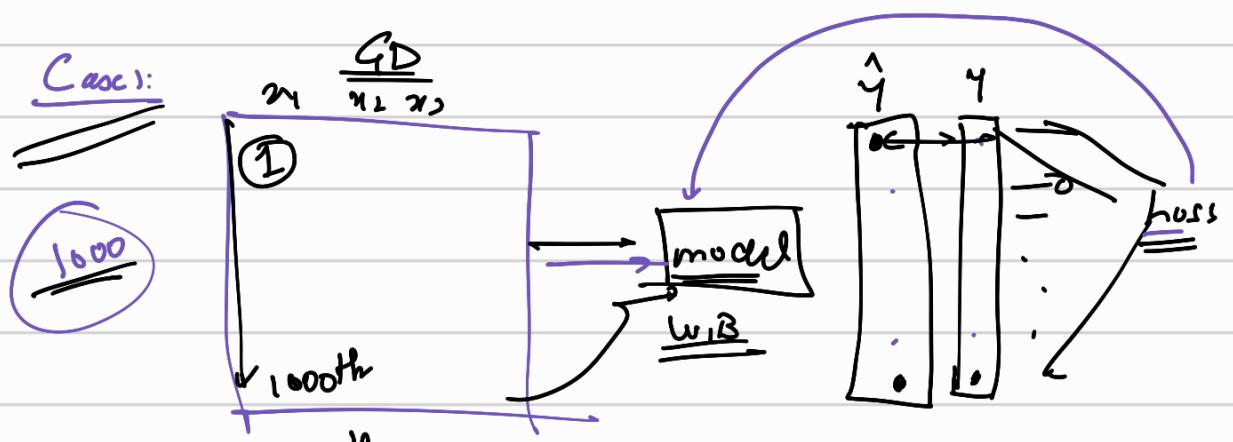
$$\hat{m}_t = \beta_1 m_{t-1} + (1-\beta_1) \frac{\partial h}{\partial w_t}$$

$$\hat{v}_t = \beta_2 \left(\hat{v}_{t-1} + (1-\beta_2) \left(\frac{\partial h}{\partial w_t} \right)^2 \right)$$

$$w_{t+1} = w_t - \frac{\alpha}{\sqrt{\hat{v}_{t+\epsilon}}} \times \hat{m}_t$$

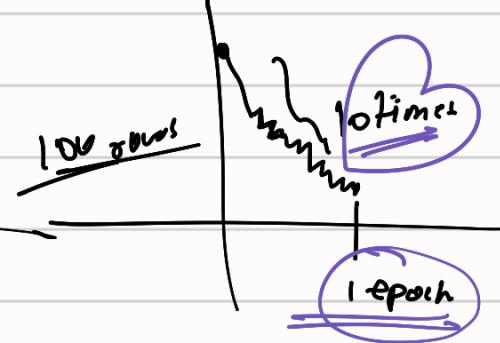
Mini Batch Gradient Descent \rightarrow Batch Normalization

Calibration



Case 1 randomly without replacement

Case 2



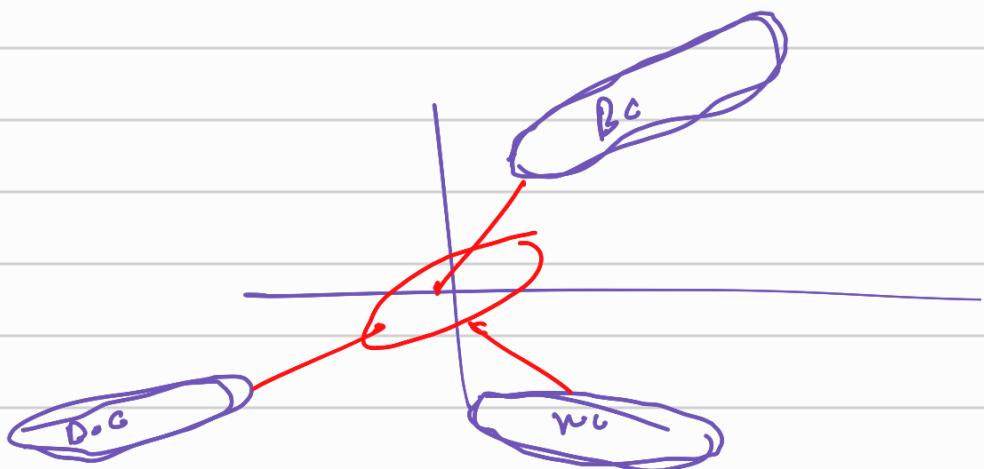
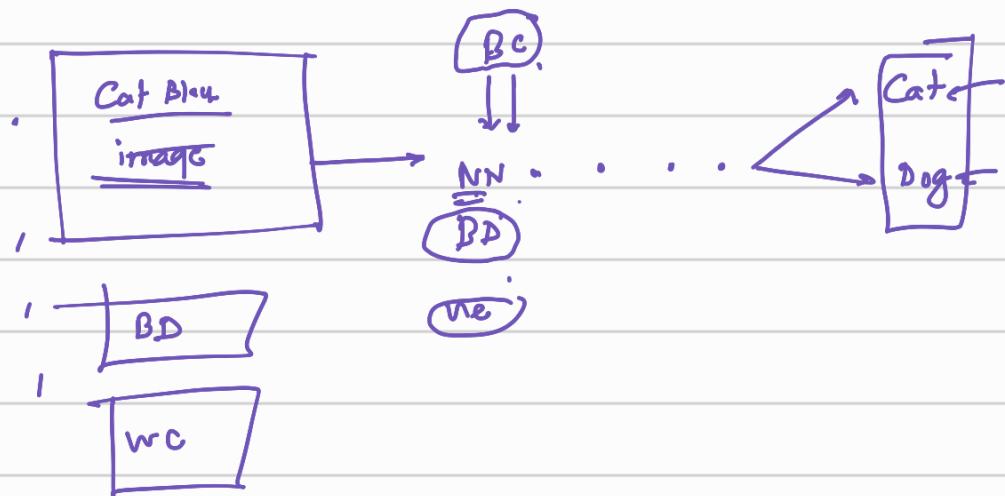
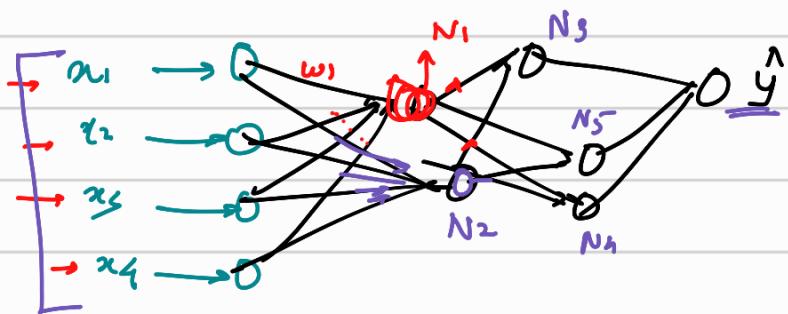
model-fit (< 0)

model-fit (20)

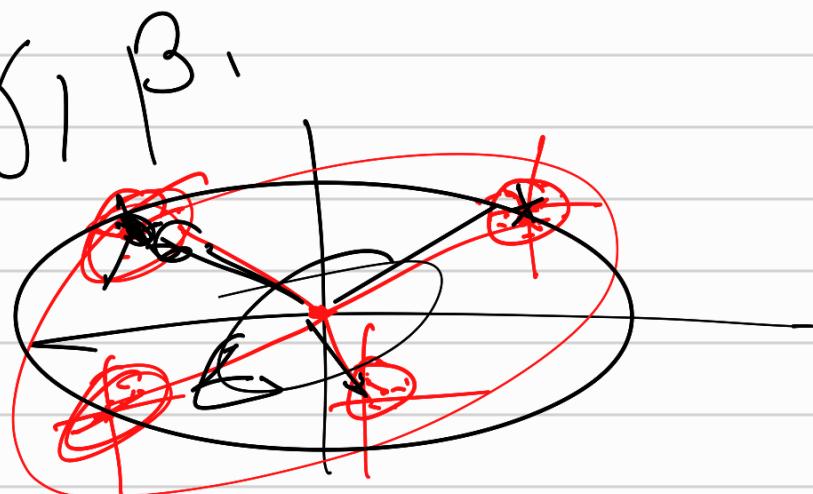
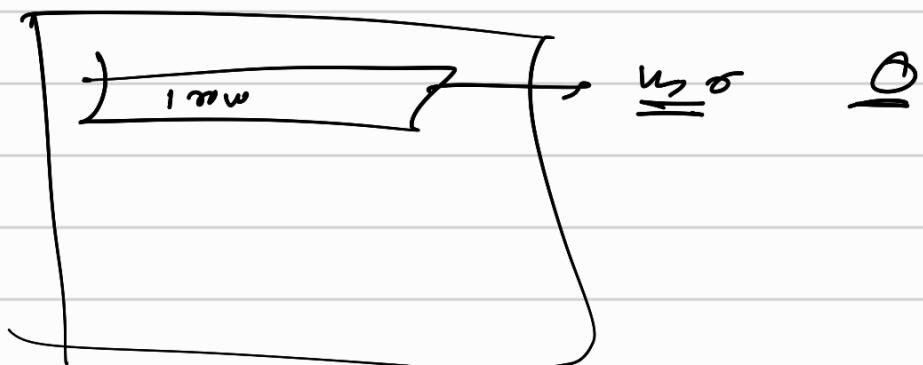
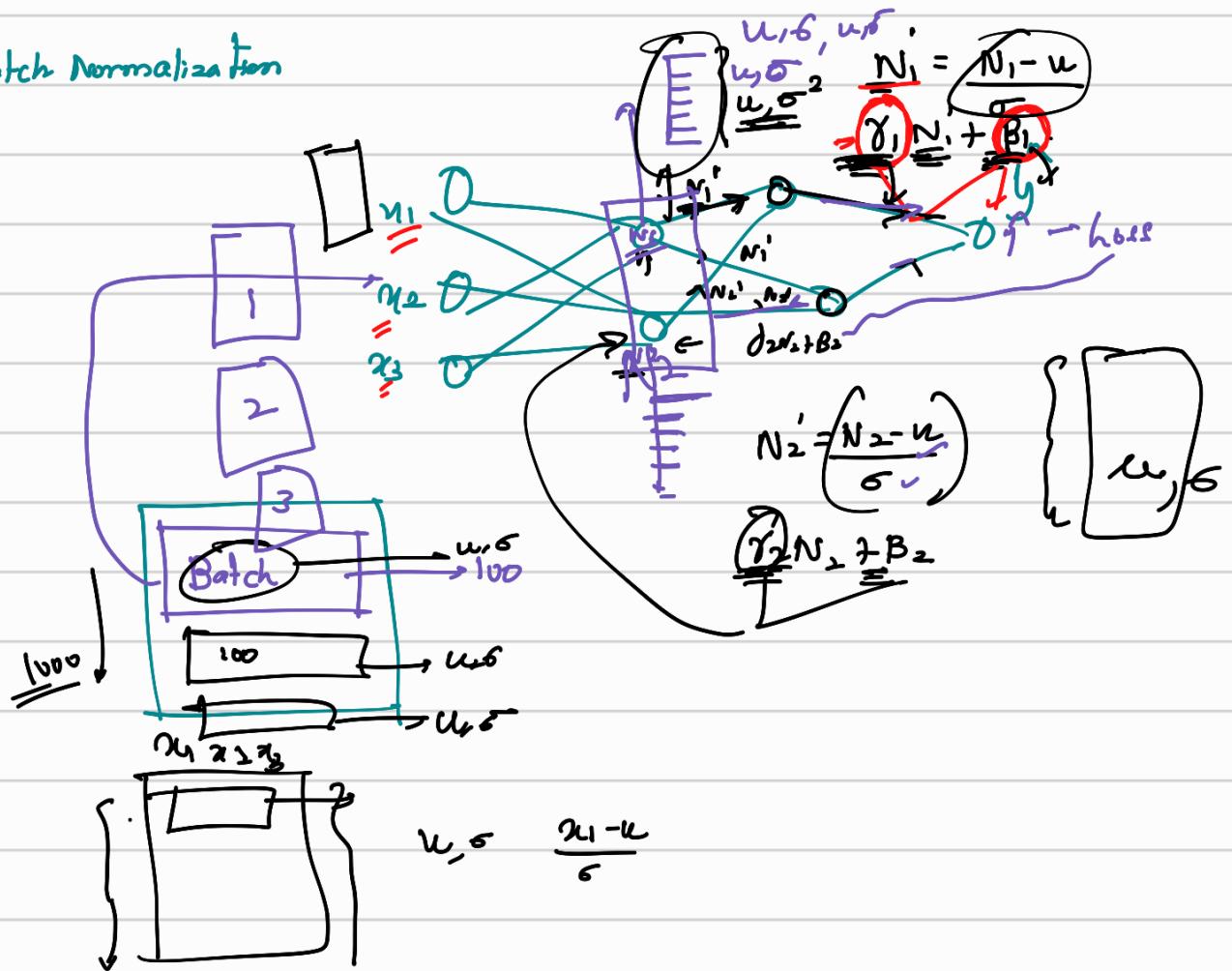
Batch Normalization

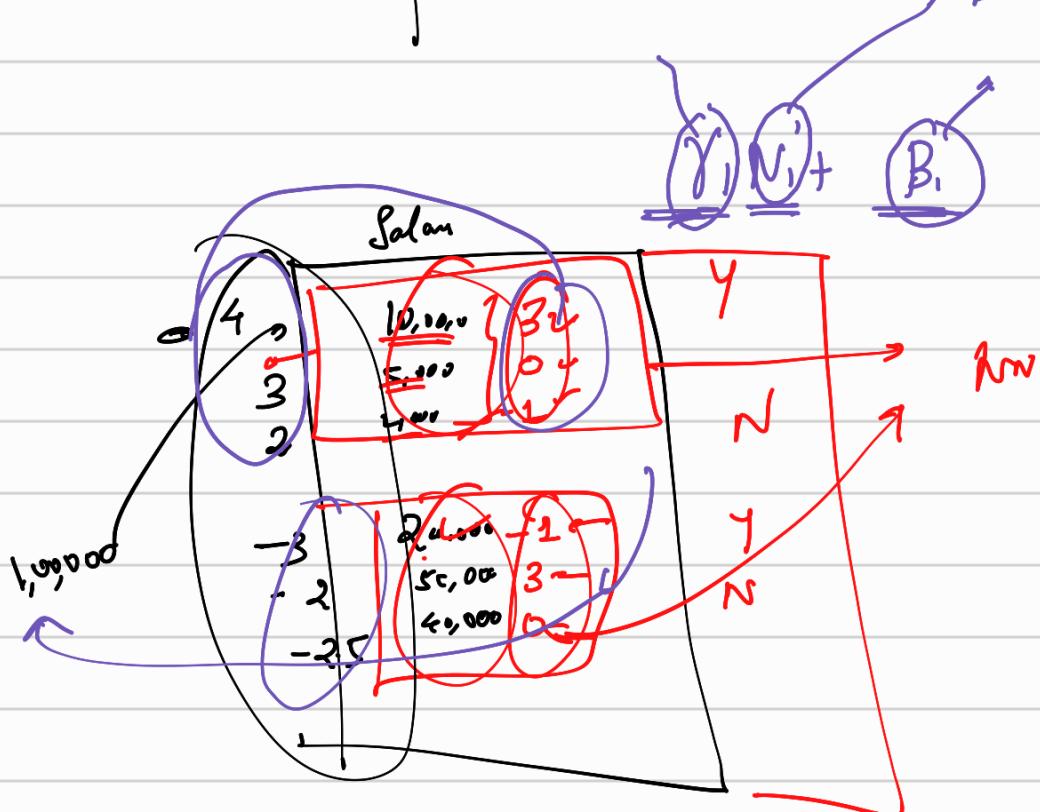
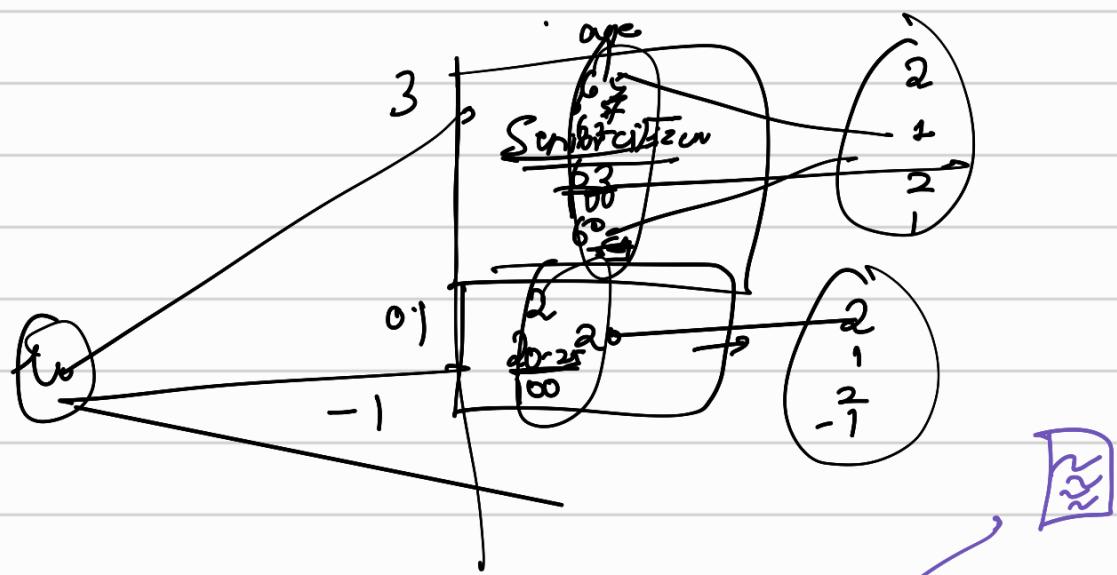
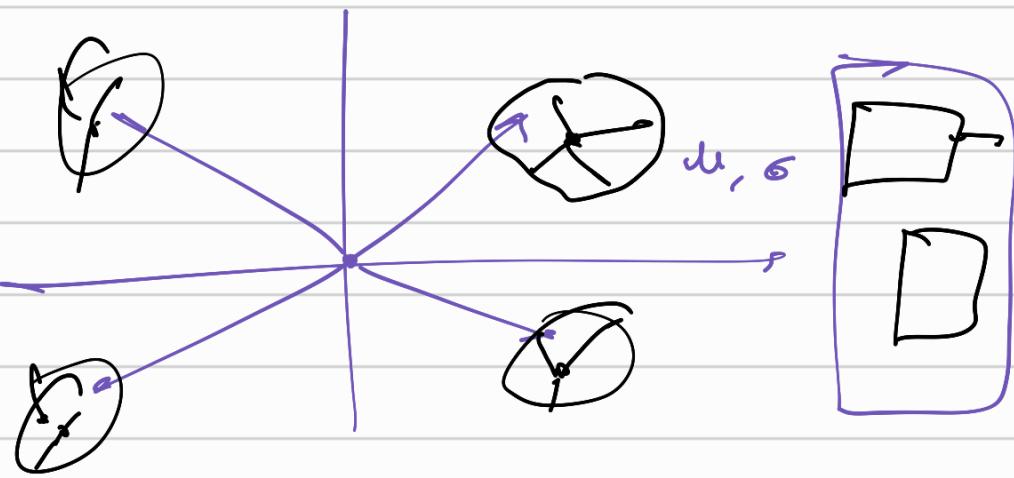
Callbacks

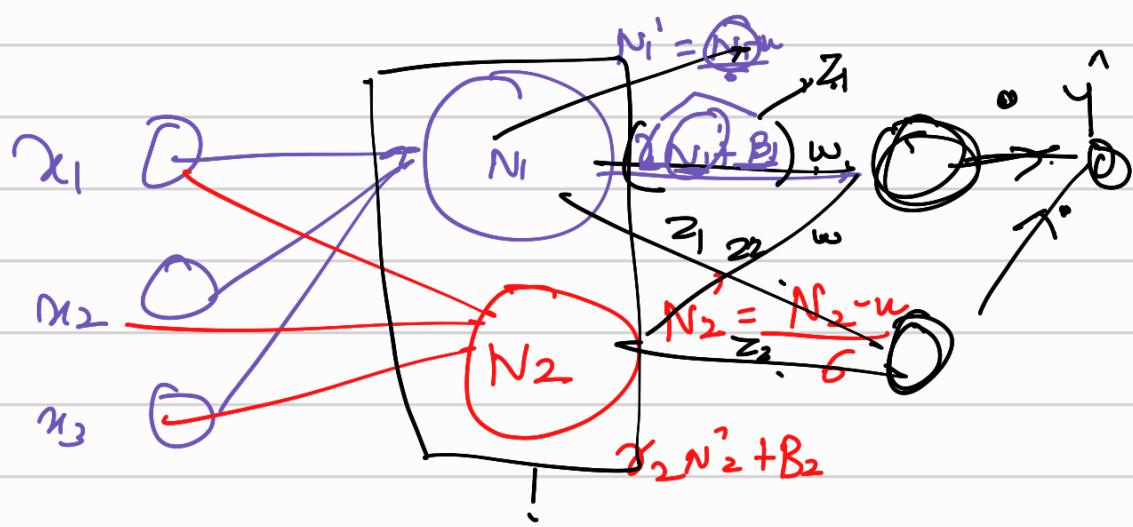
Batch Normalization



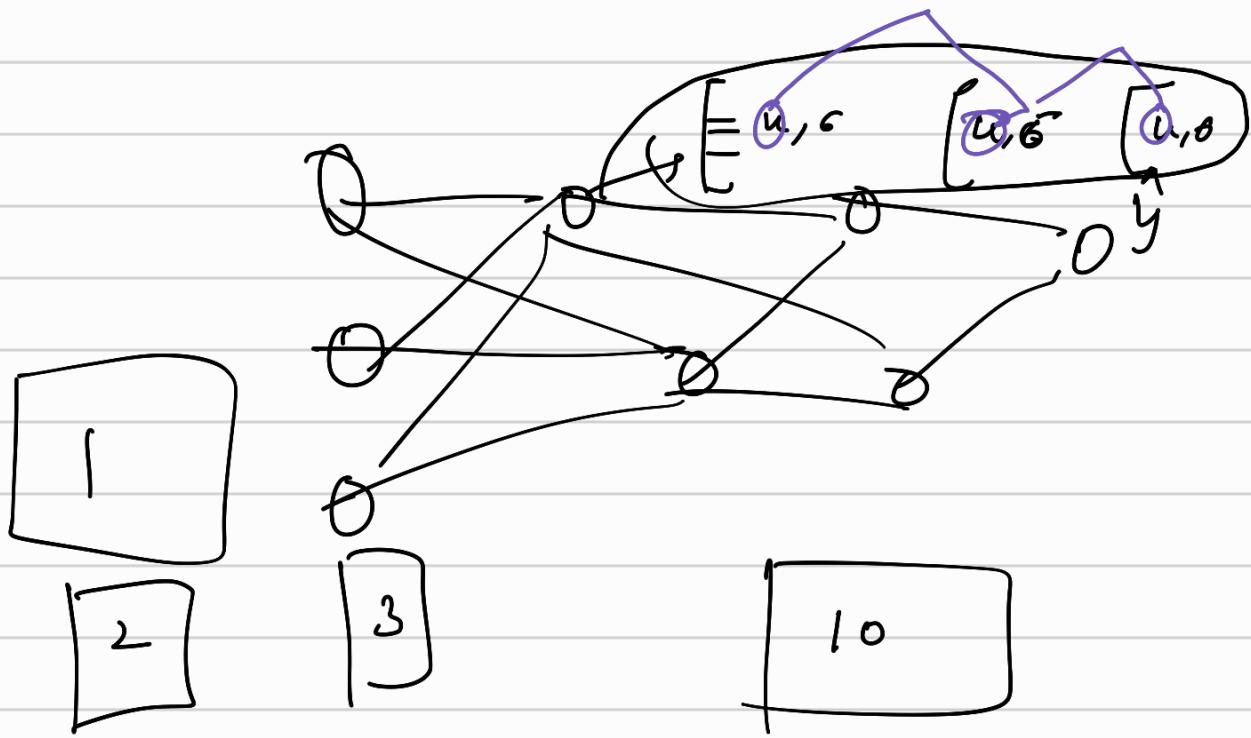
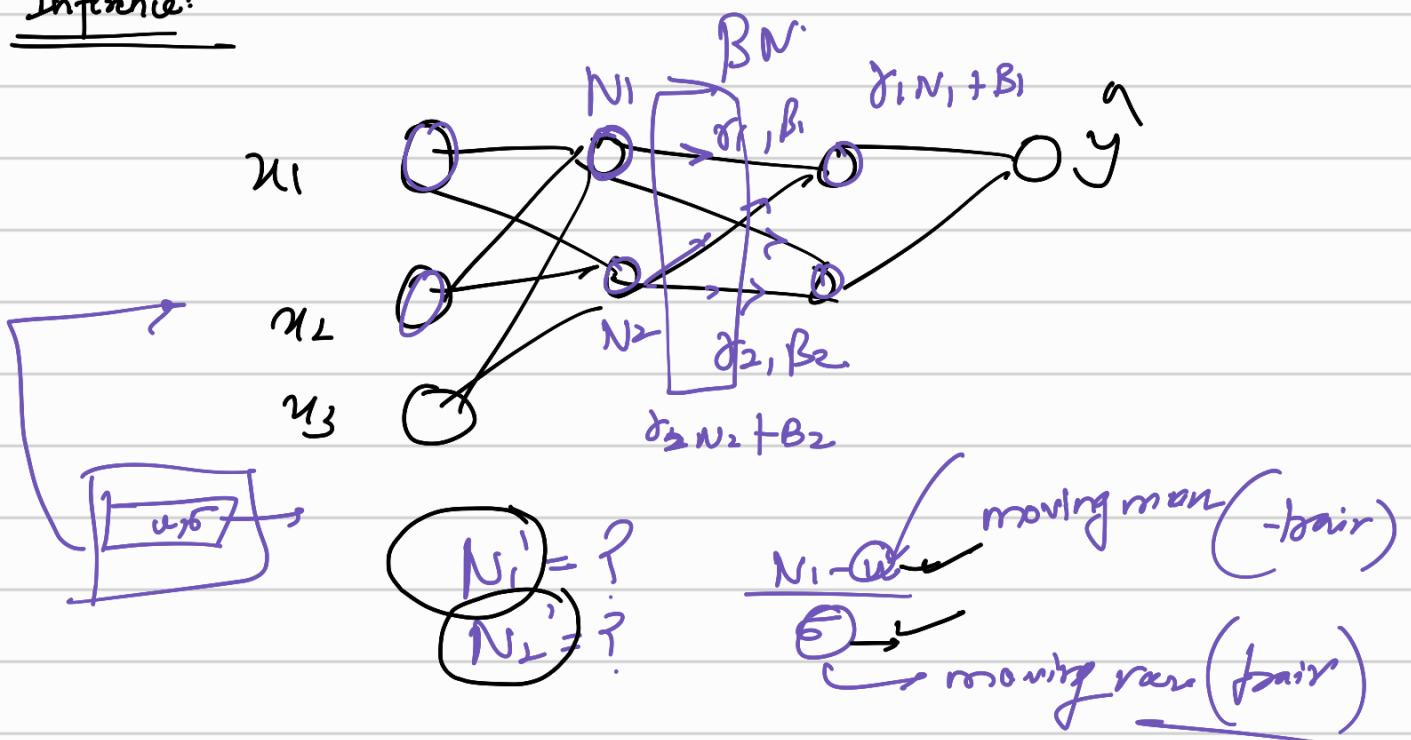
Batch Normalization







Inference:

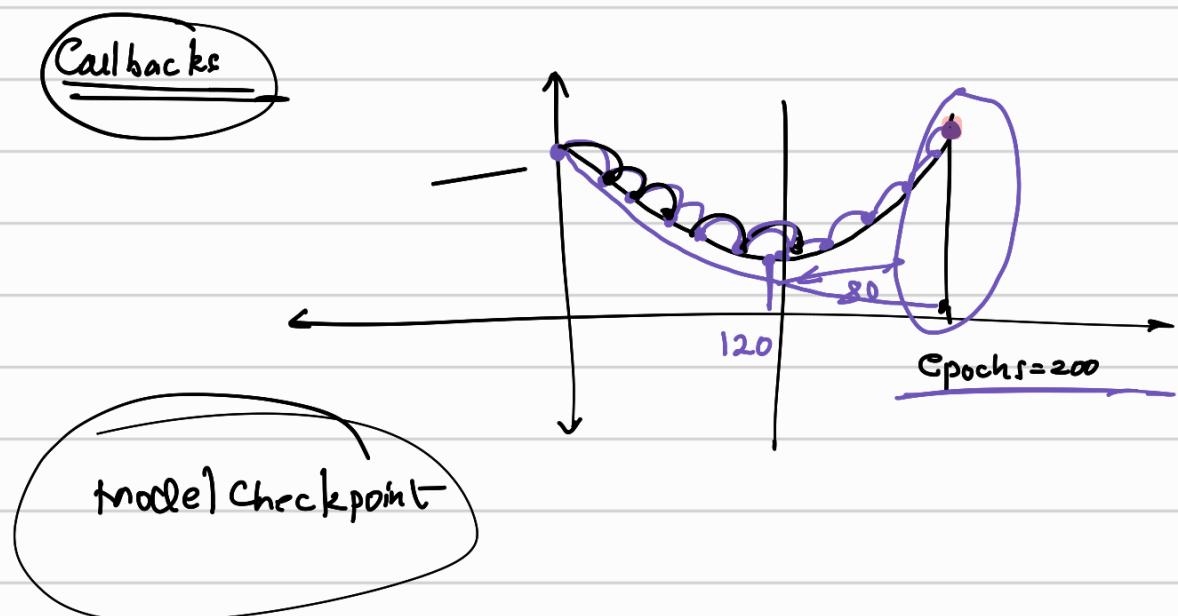


N1

~~moving mean~~ $\rightarrow \beta \times \text{moving mean} + (1-\beta) \times \text{mean(batch)}$

N2

~~moving var~~ $= \beta \times \text{moving var} + (1-\beta) \times \text{var(batch)}$



Early Stopping

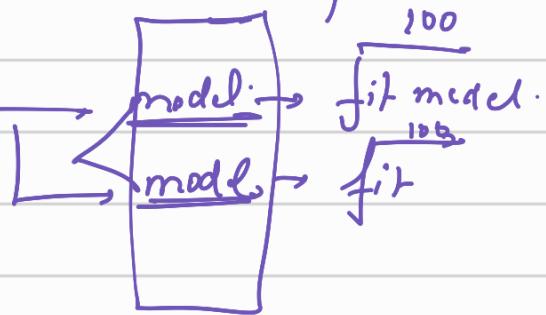




monotonic, = 3

$\{3, 32, \text{relu}, d=T, h=10^{-2}\}$

Execution parallel: 2



$[4, 32, \text{relu}, d=N, h=10^{-4}]$

$[4, 64, \tanh, d=T, h=10^{-3}]$

11:33 AM

Input layer (784) ^{1c}

Hidden layer (200, 400, activation (tanh, relu)).

Hidden layer (100, 200, activation (tanh, relu)).

2,4 { [Hidden (50, 100, activation, (tanh, relu)) $\lambda = 0.1, 0.5$]

Batch Norm () ← maybe add / not

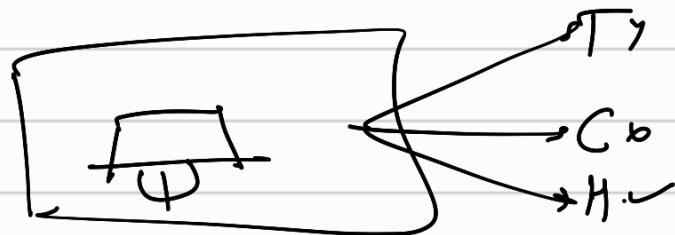
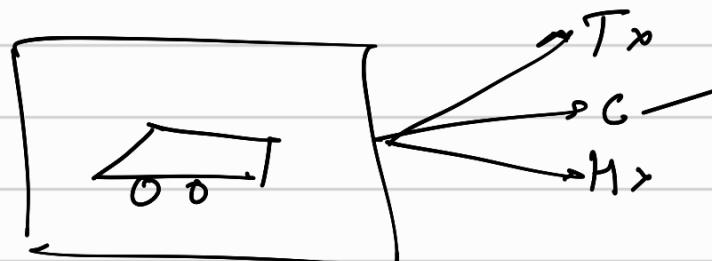
model. layers (Dense (10, 50) final).

log sampling

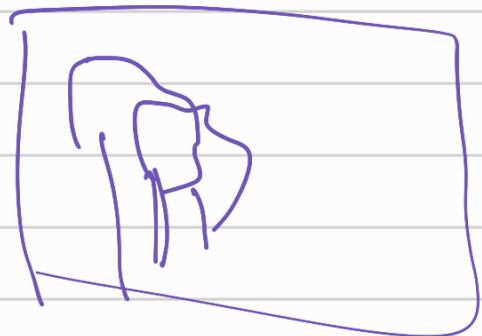
Introduction to Computer Vision

①

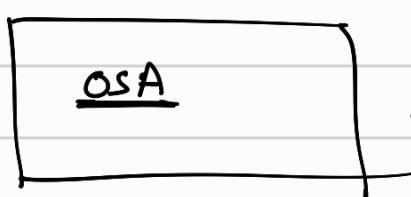
Image Classification



Object detection & localization



Bounding box



Asm Delhi



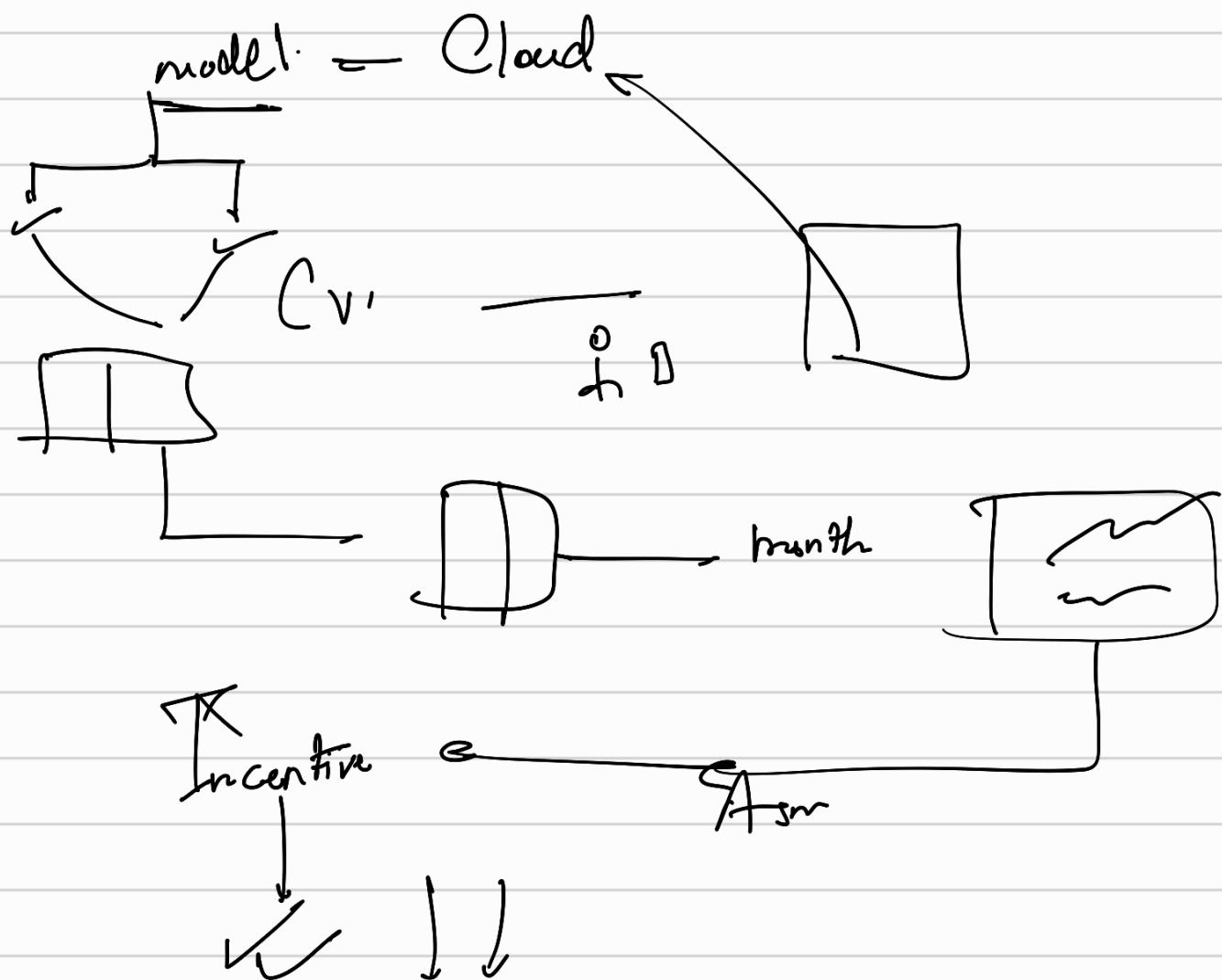
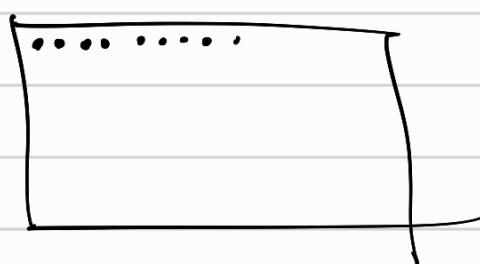
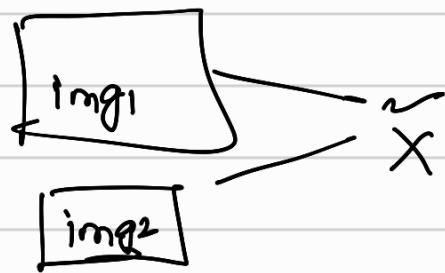


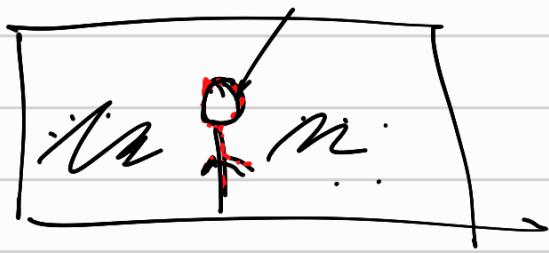
Image Segmentation



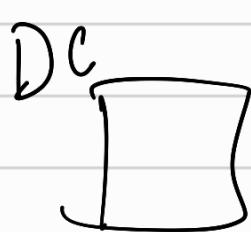
④ Frame networks

FR



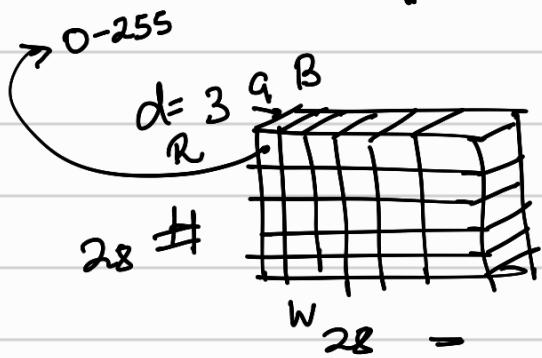


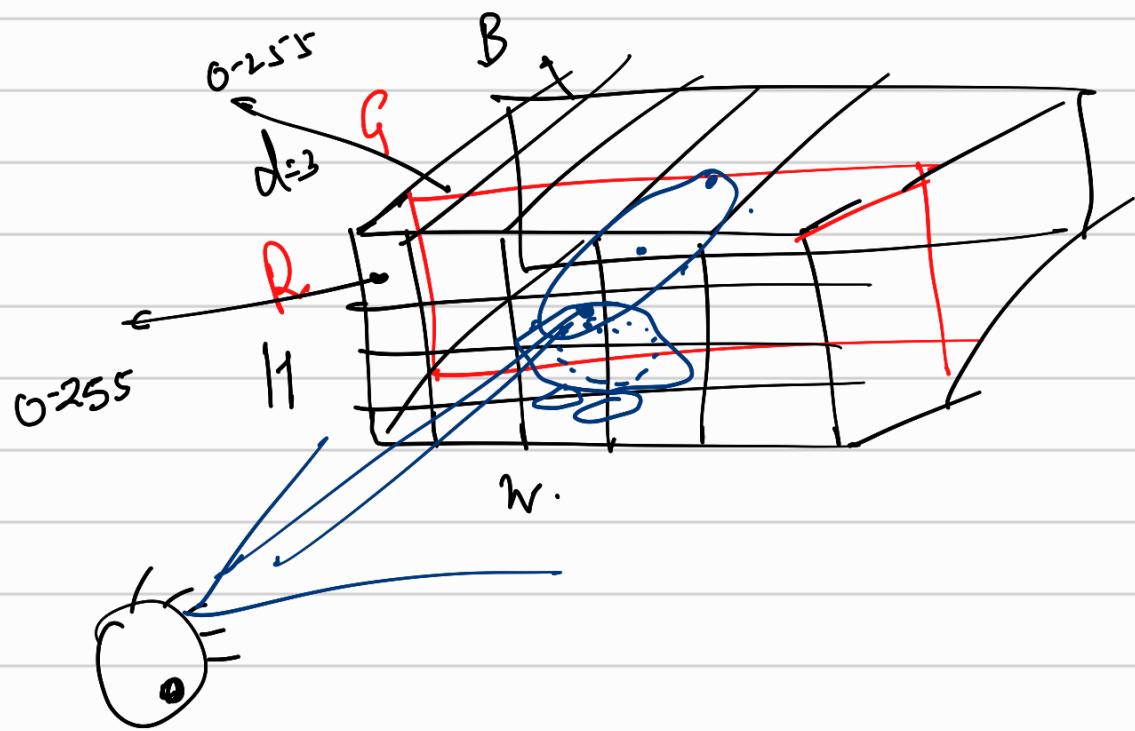
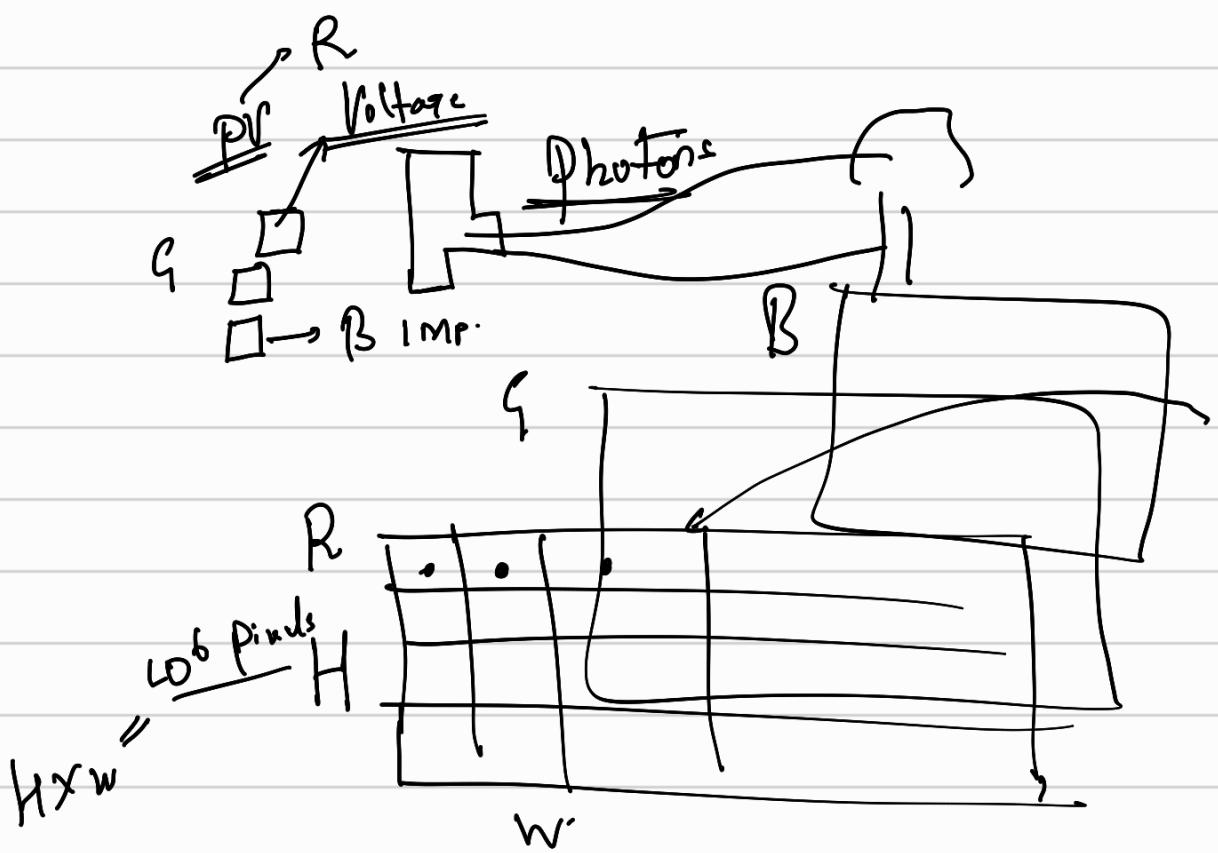
⑤ GAN → Gen Adver Networks.

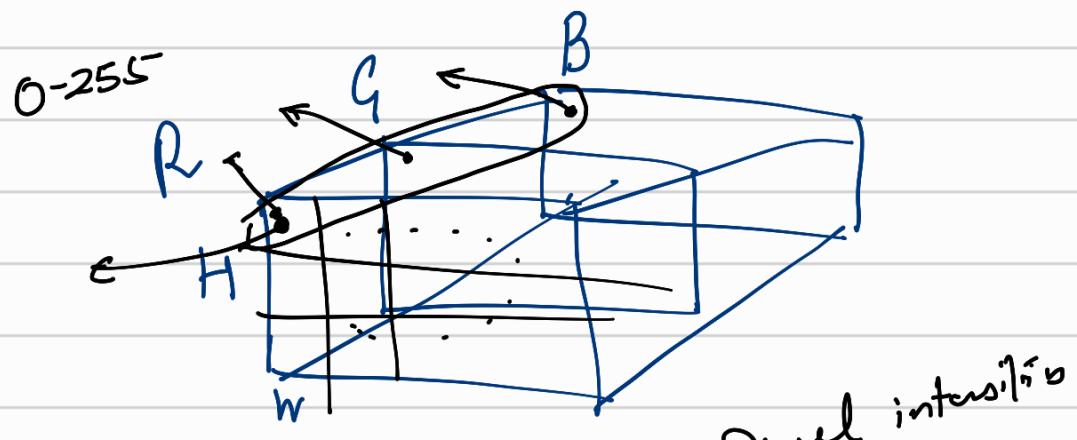


What is an image?

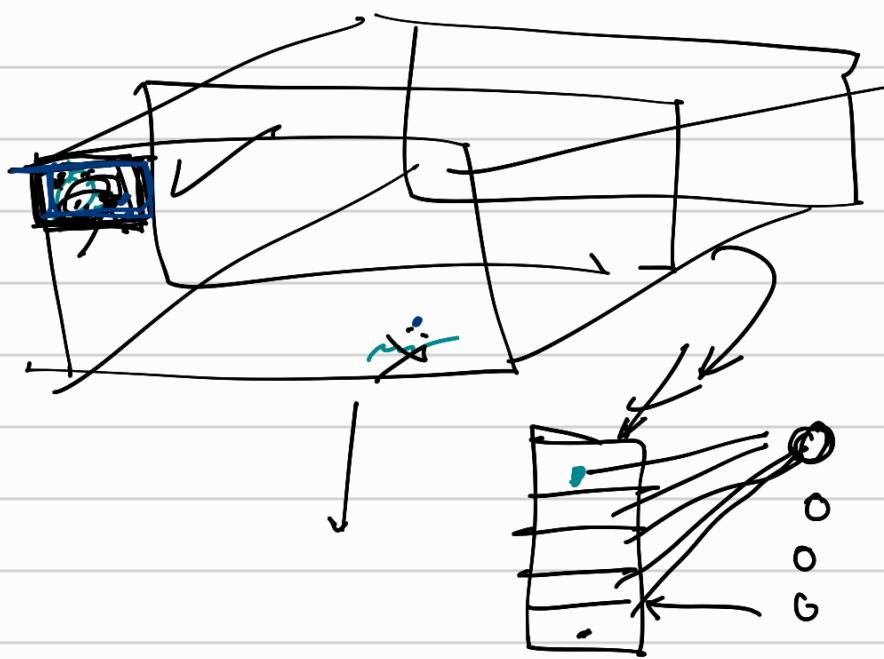
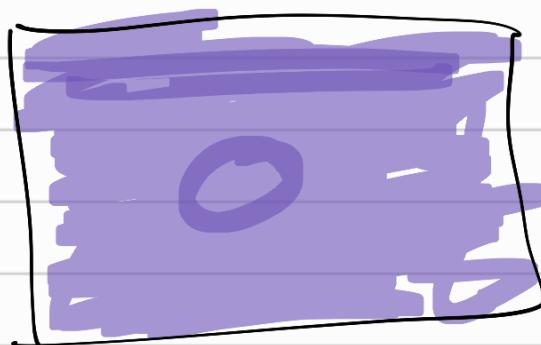
R, G, B

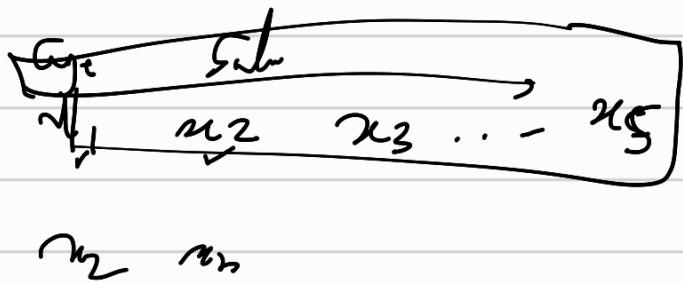




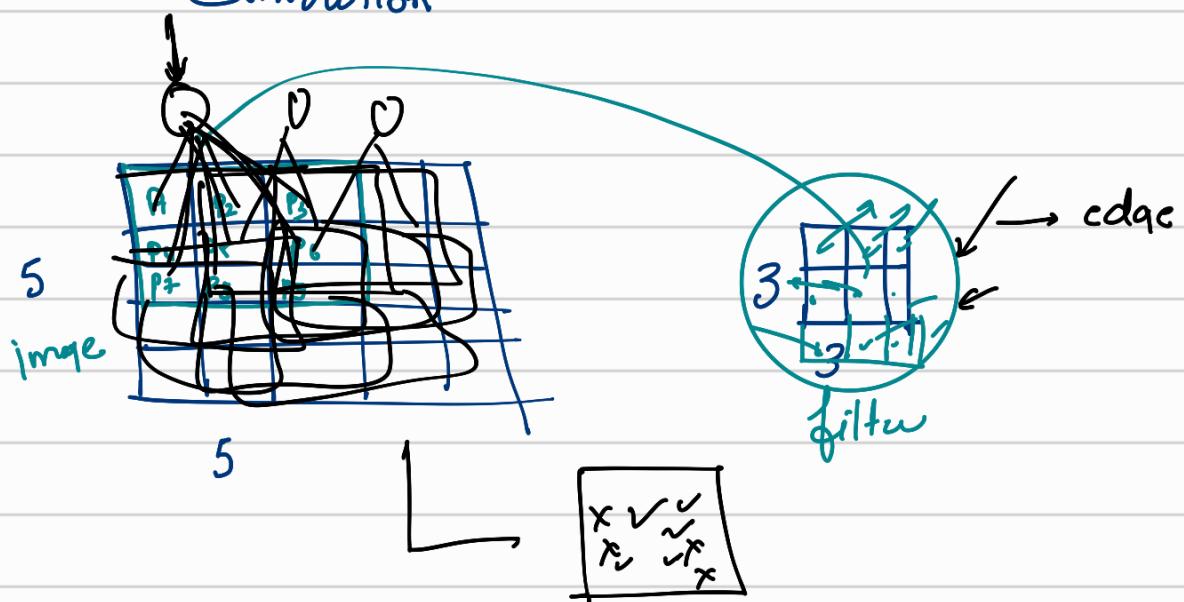


Information → Variation in the parallel intensity

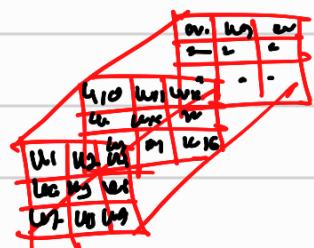
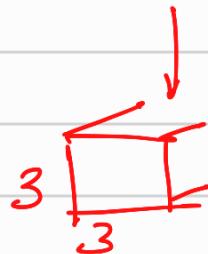
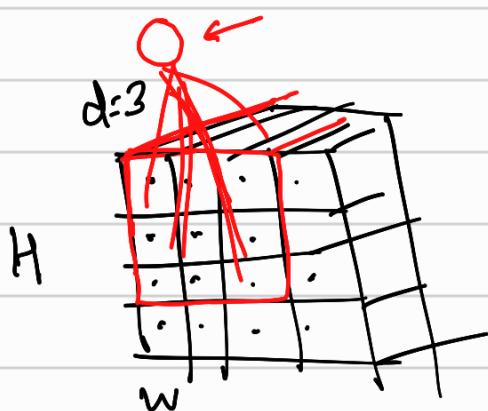




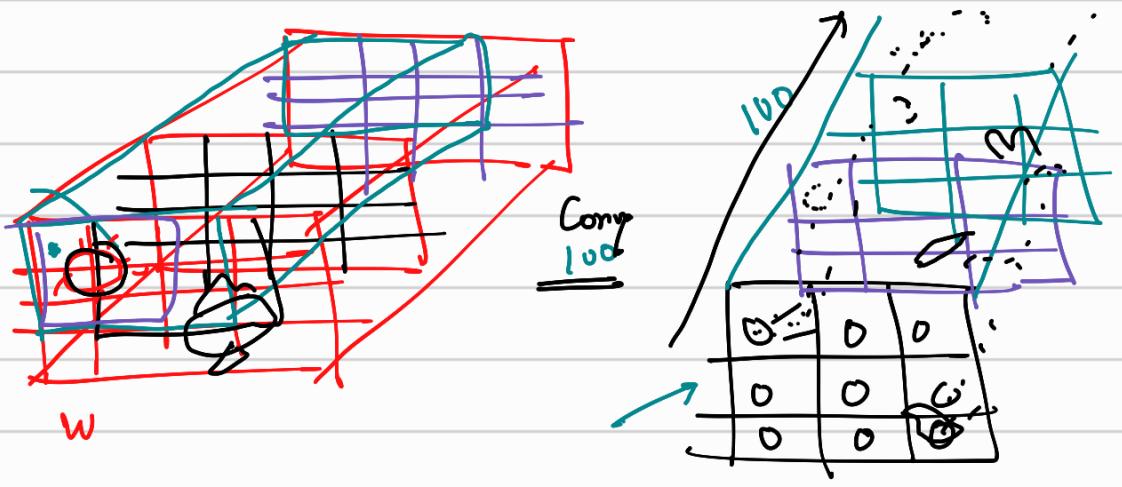
Convolution



Convolutional Neural Network



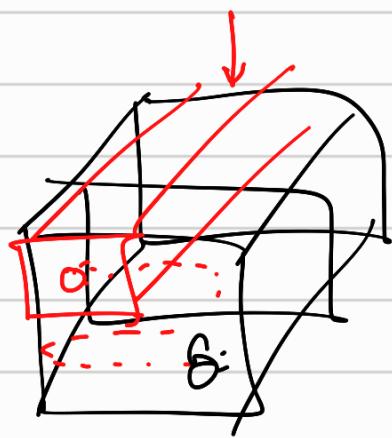
$$\text{tanh}(p_1 w_1 + p_2 w_2 + p_3 w_3 + \dots + p_{27} w_{27})$$



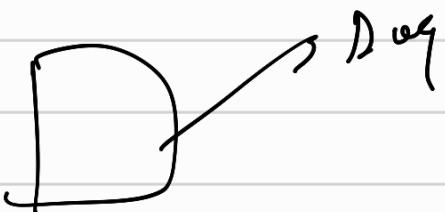
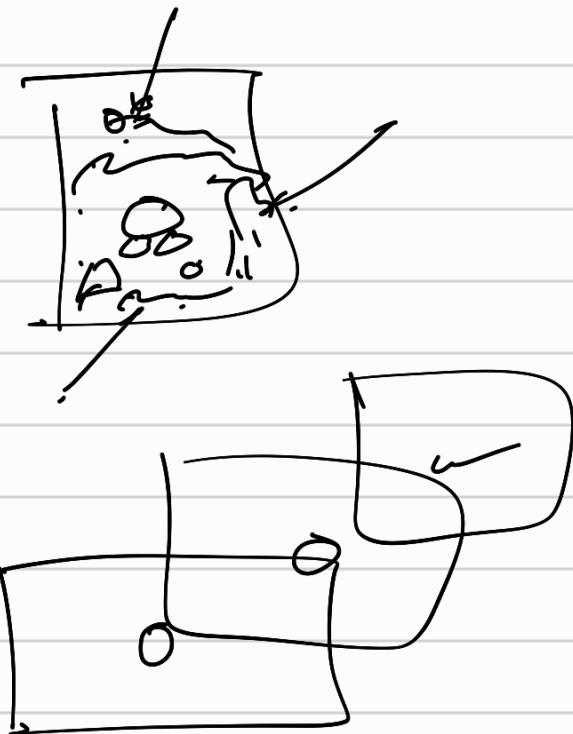
$3 f$ f_3



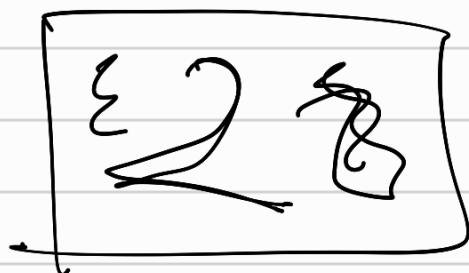
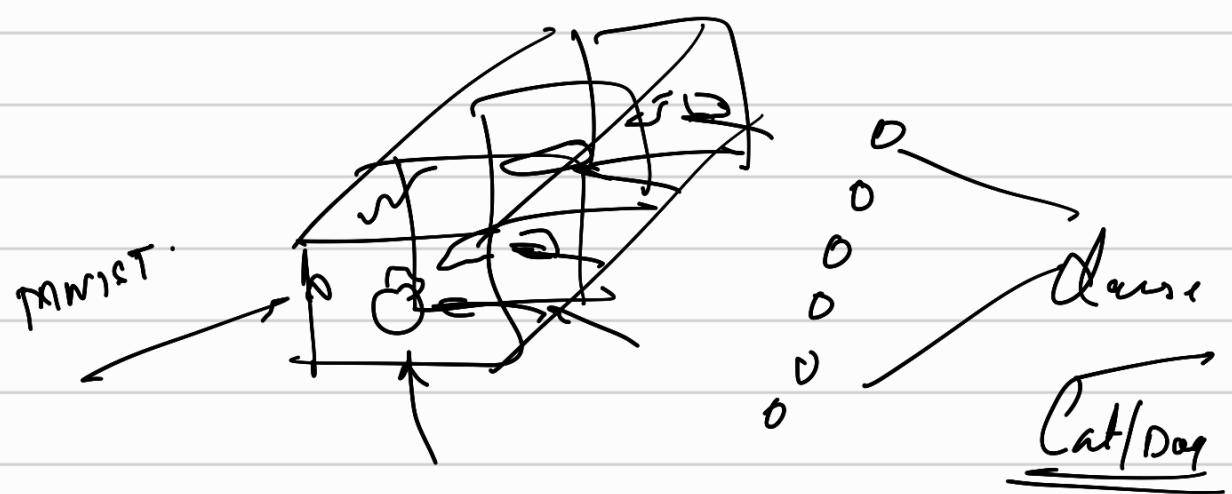
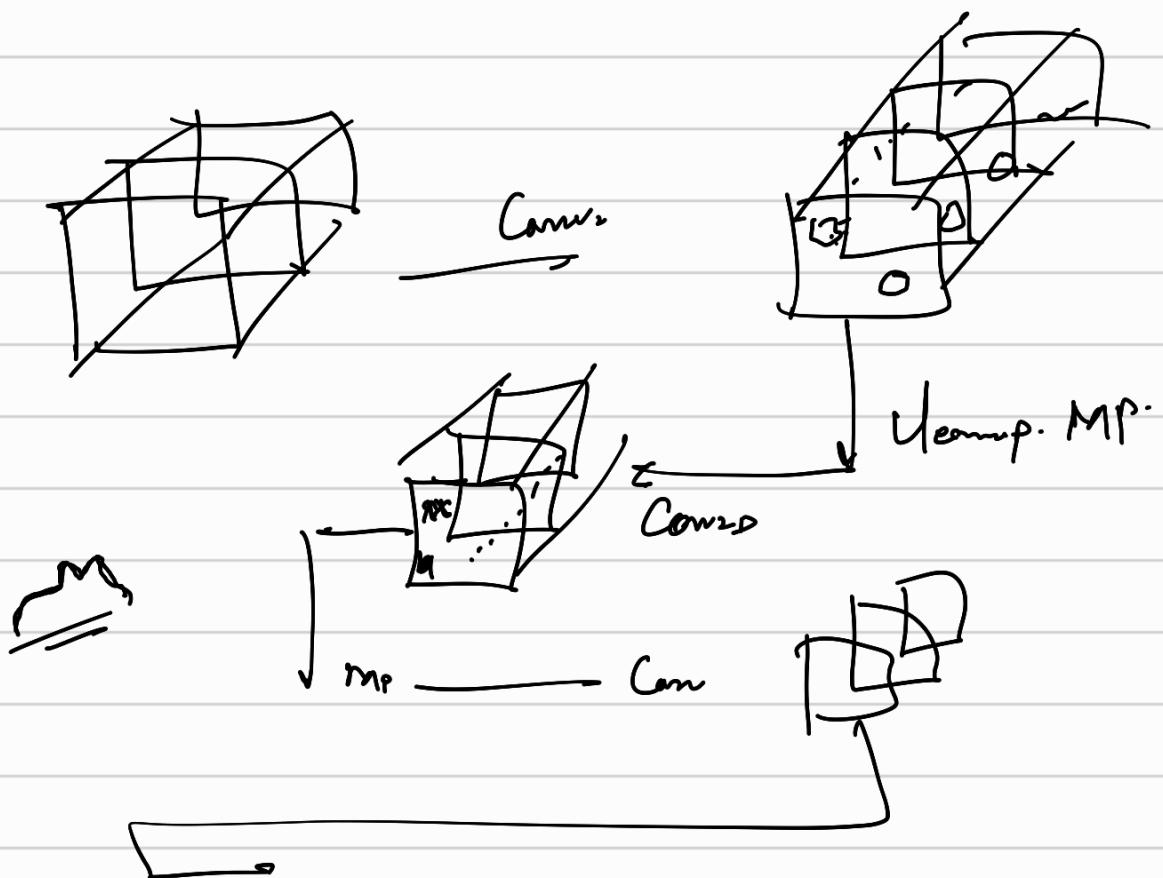
- ① If the filters (100) learnt the weights to extract the dog's features, then the feature map will be containing mainly the features of the dog and very slight features of other objects like tree/bird/sun.

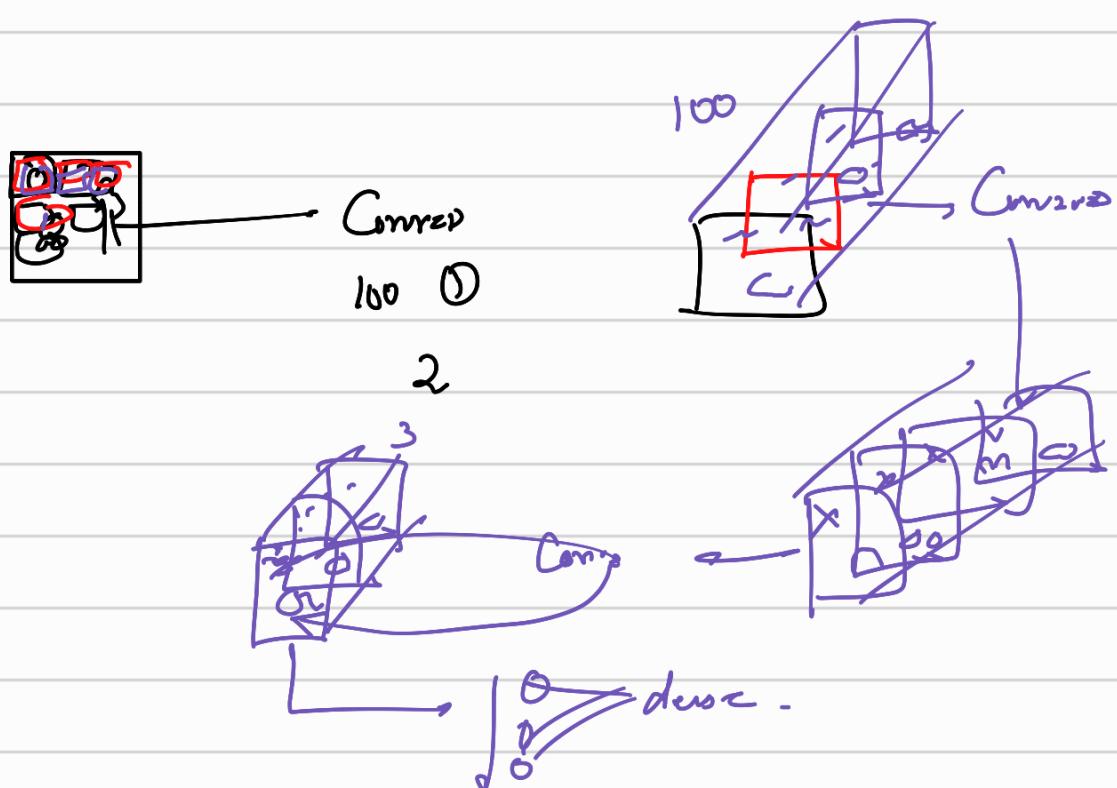
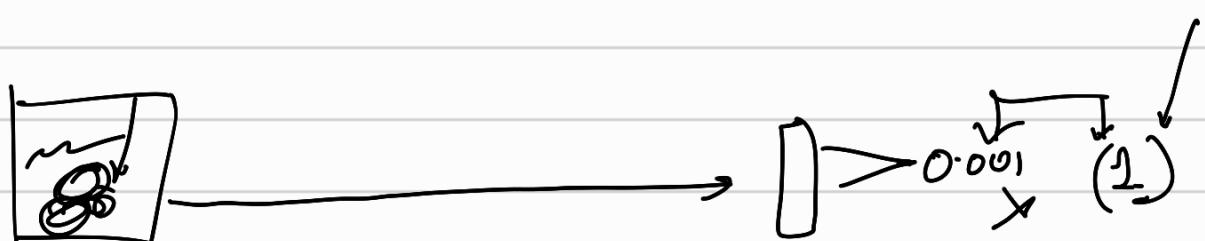
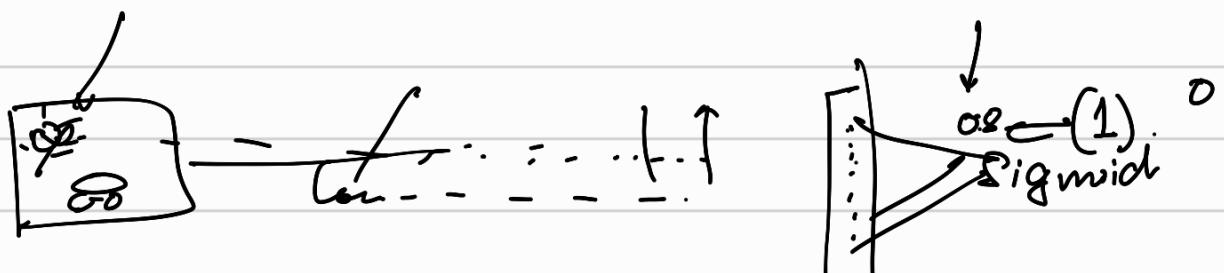


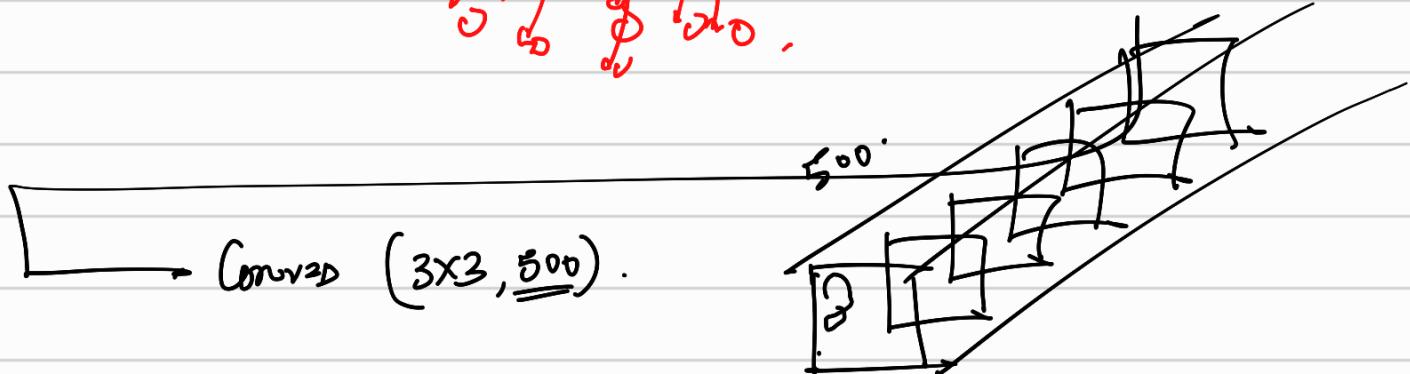
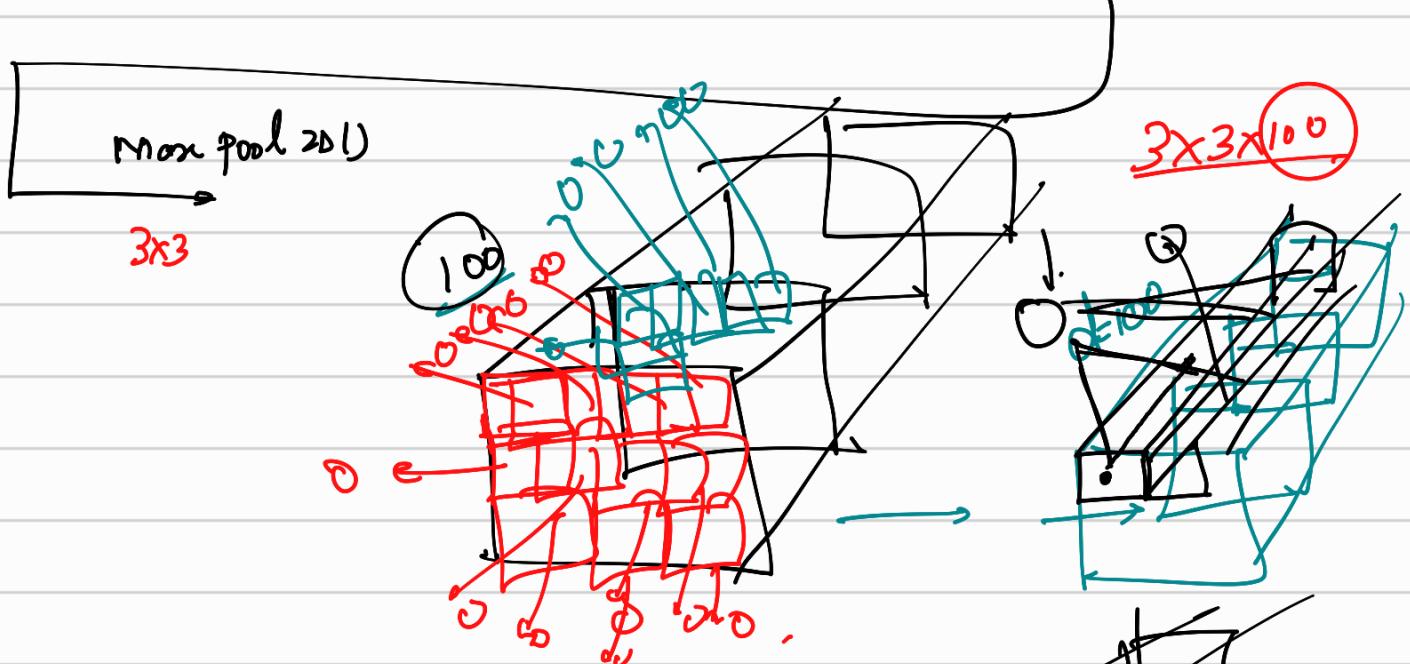
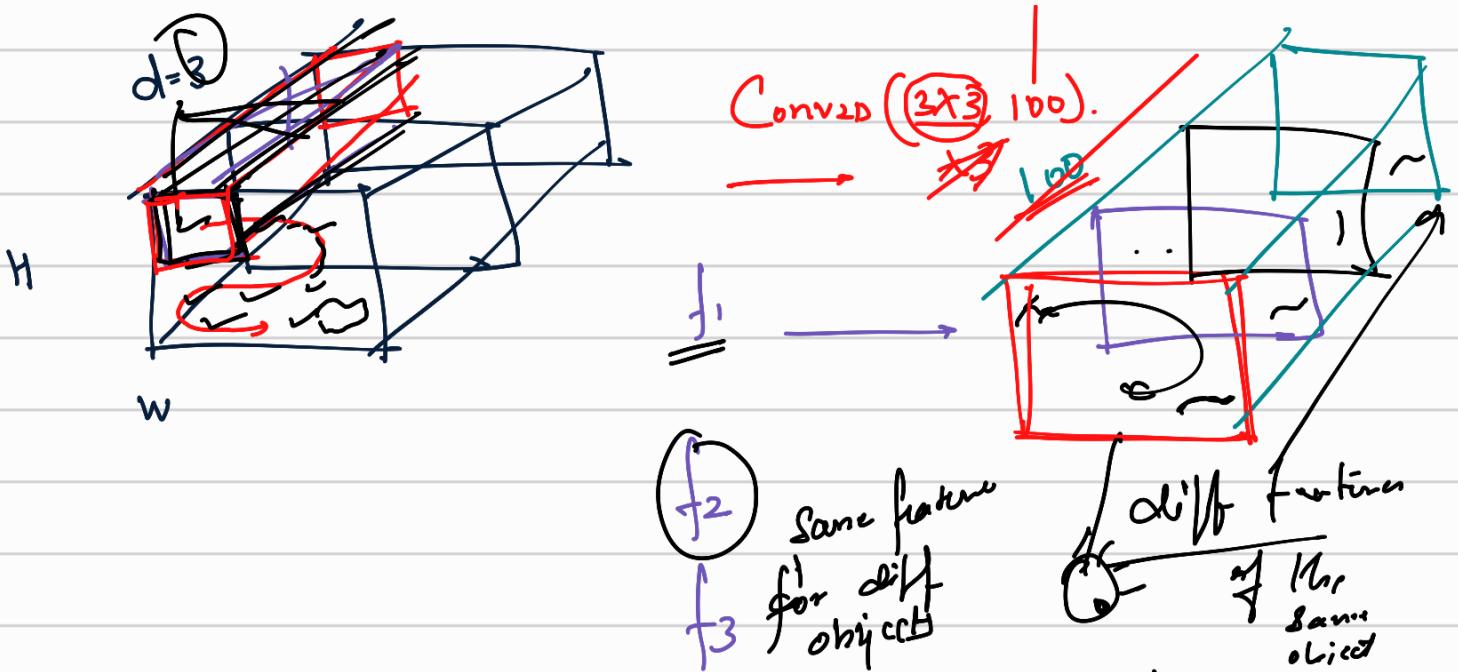
↗	↘	○
○	→	-
↑	○	↖

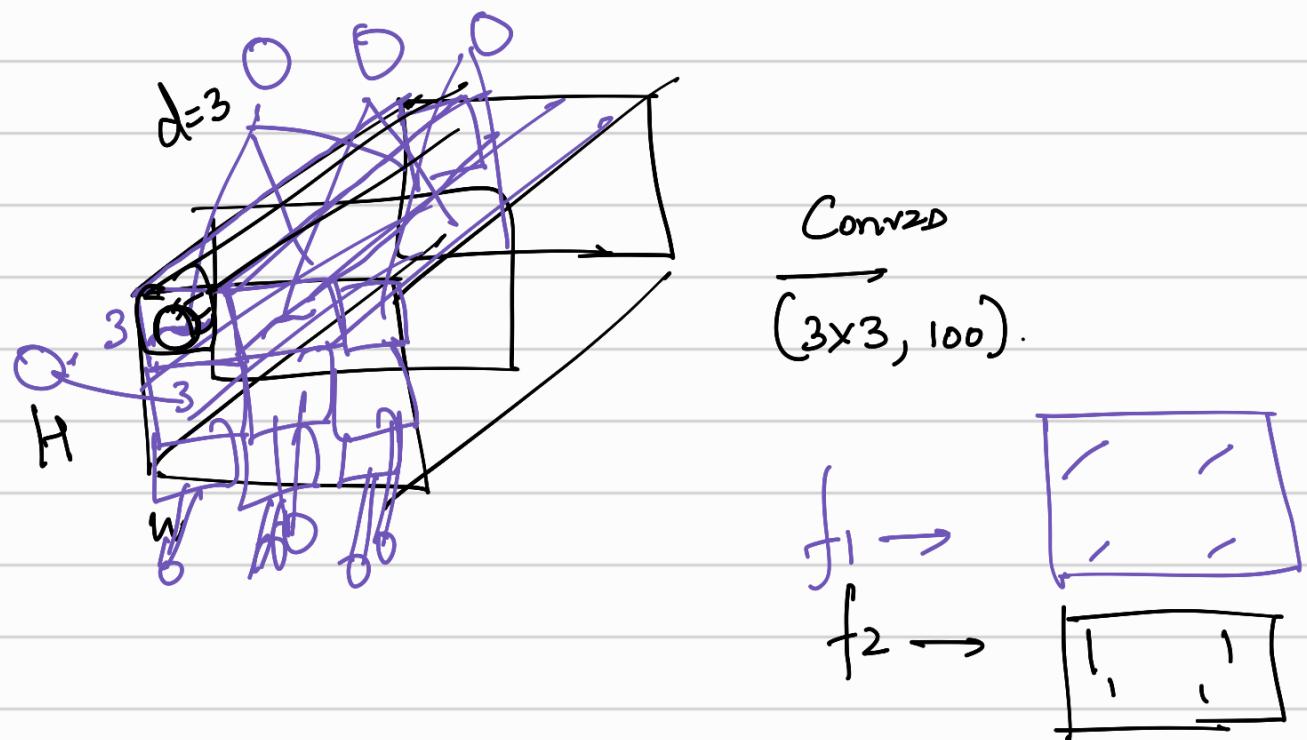
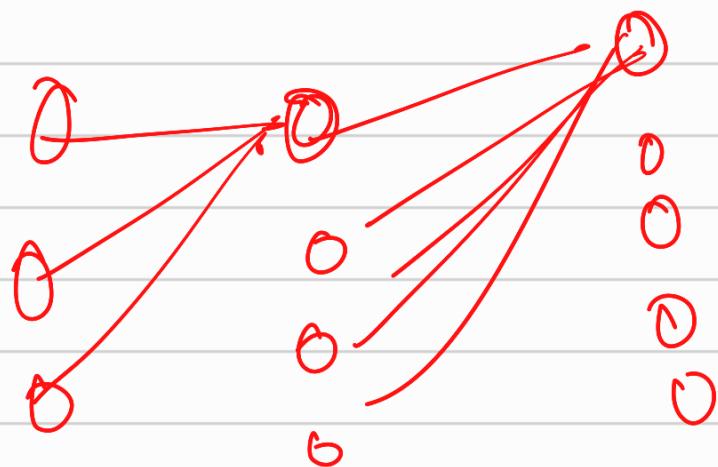
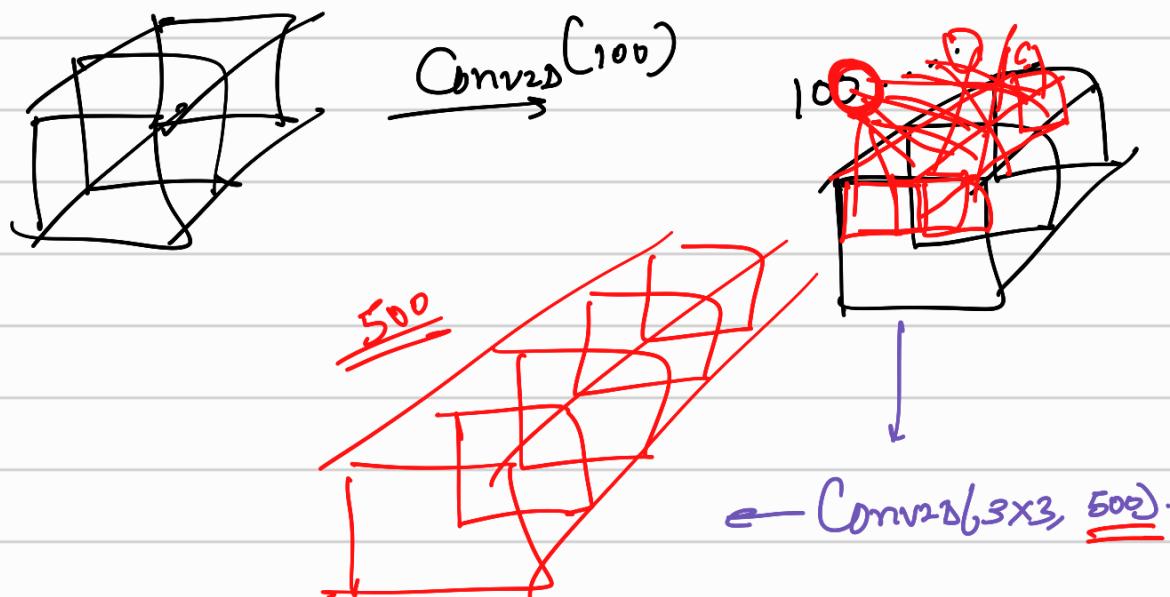


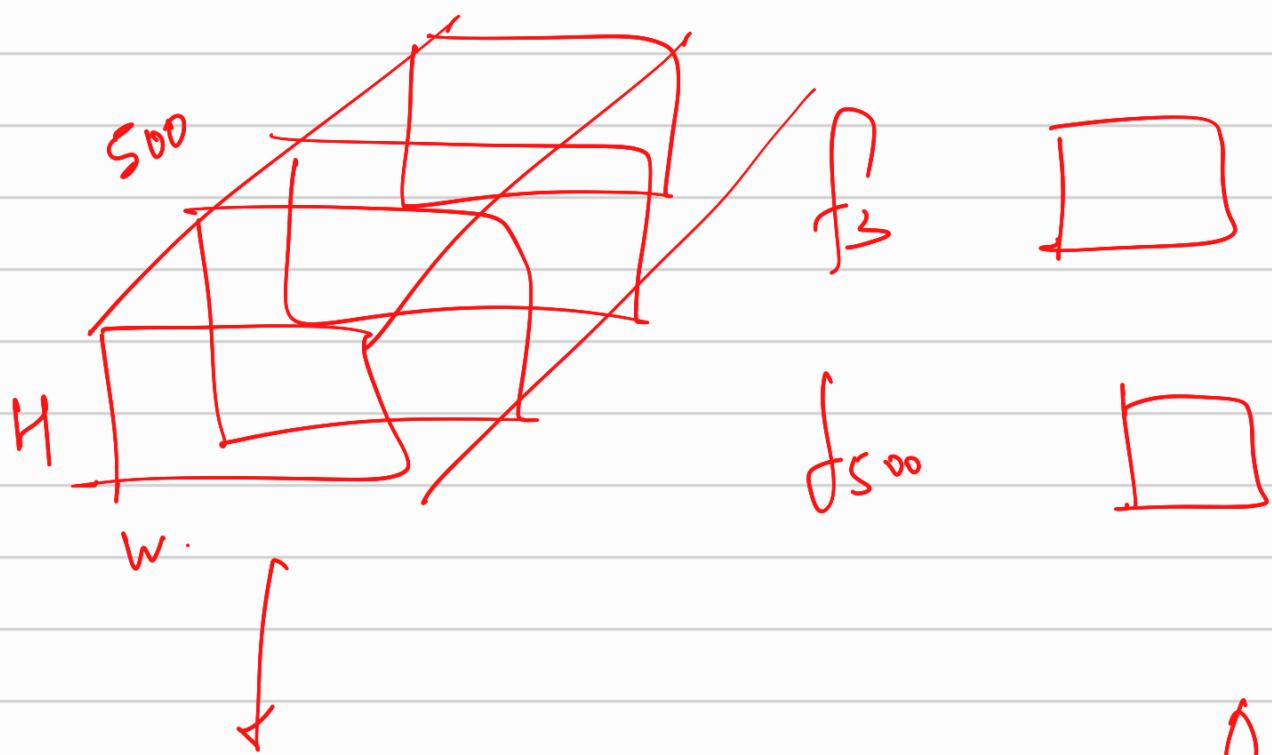
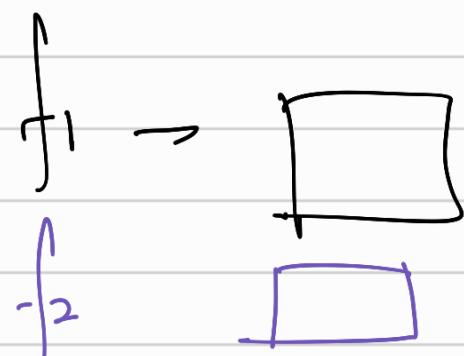
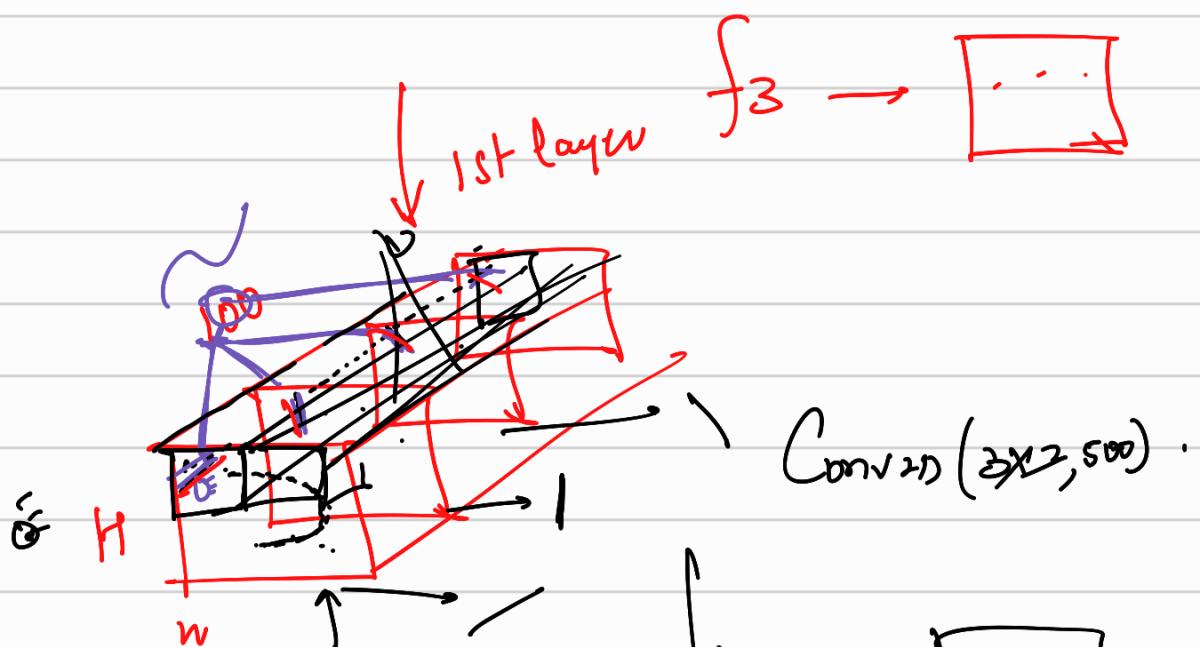
Cat





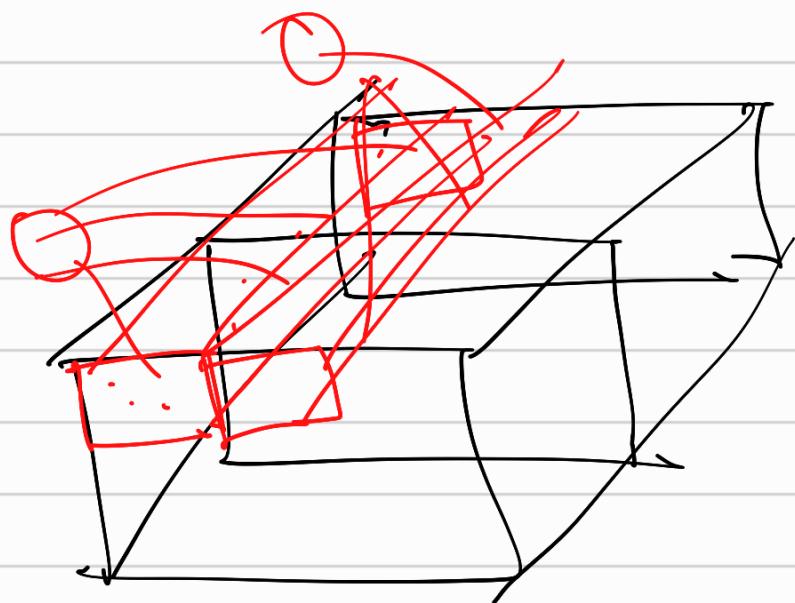
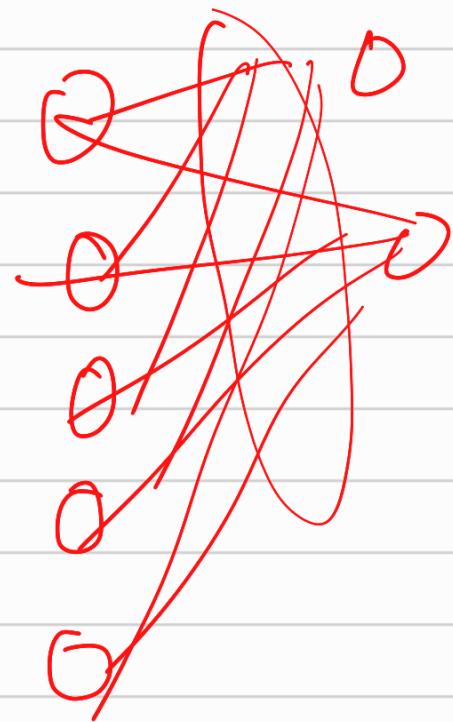






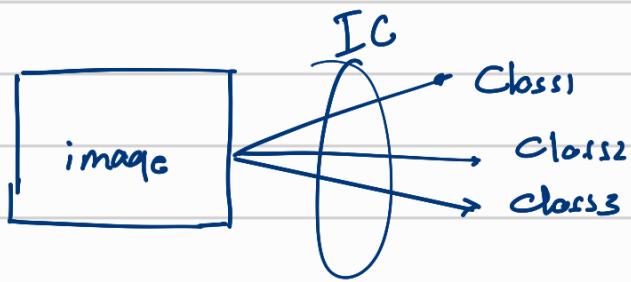
Conv2D



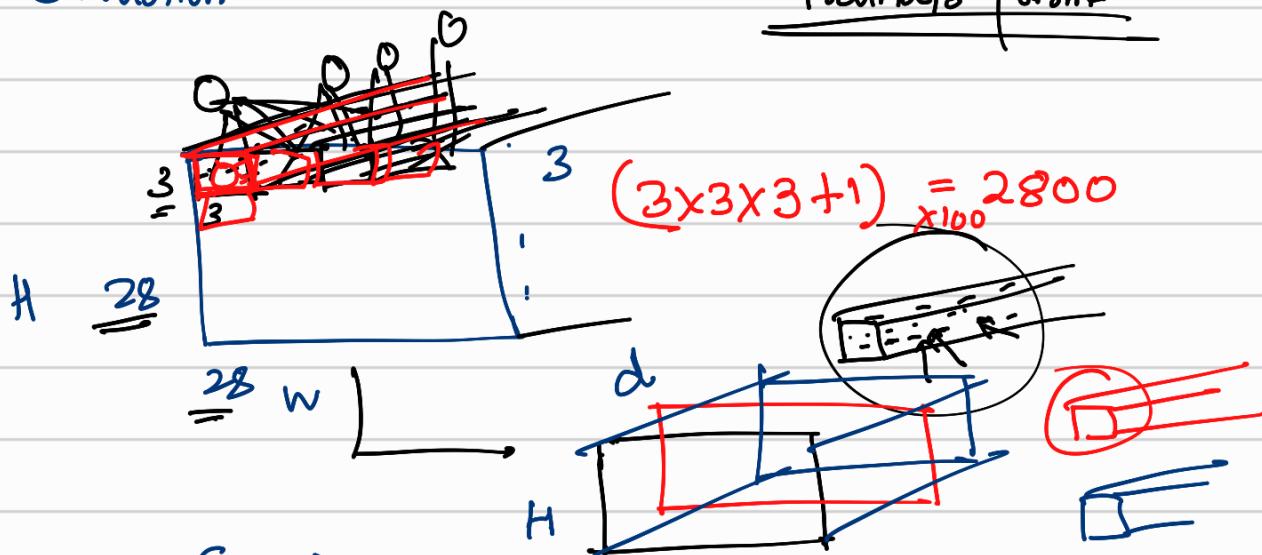


model.add (Conv2D(3x3, 100, s=1, p='valid')
(int))

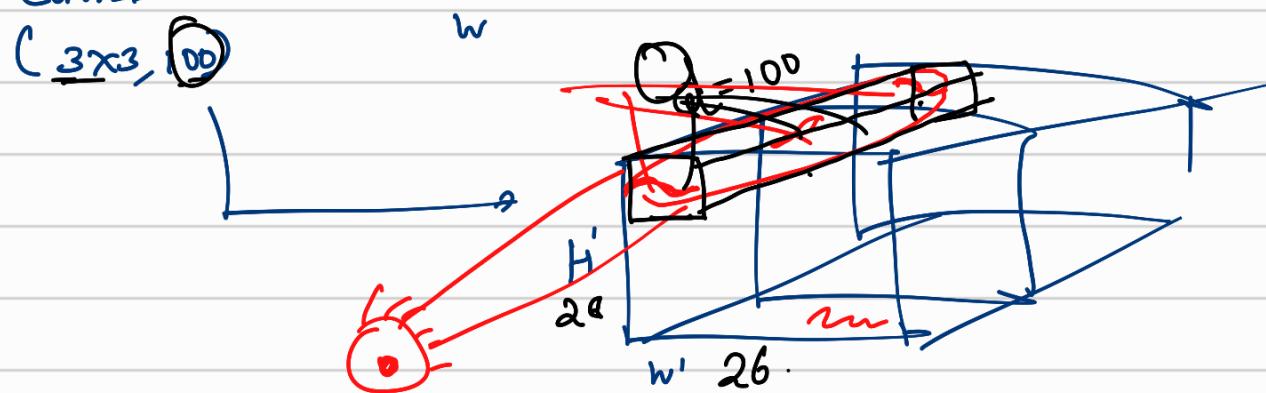
Convolution Neural Network



Convolution



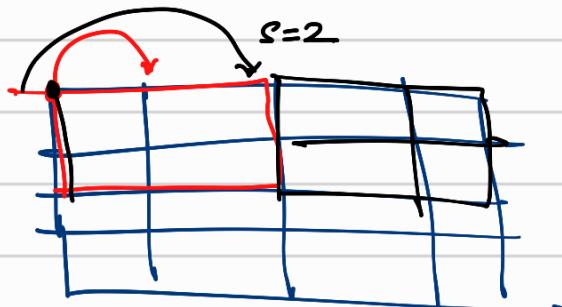
Conv2D



$$H' = \frac{H - f}{S} + 1$$

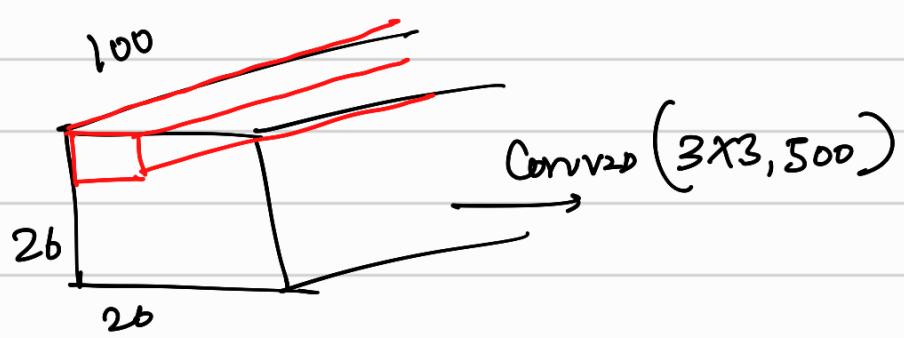
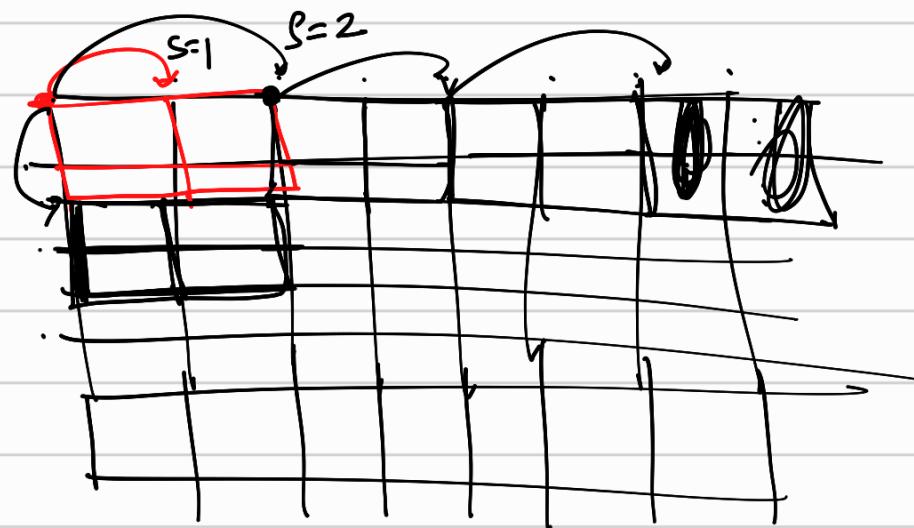
$$\textcircled{S} = 1$$

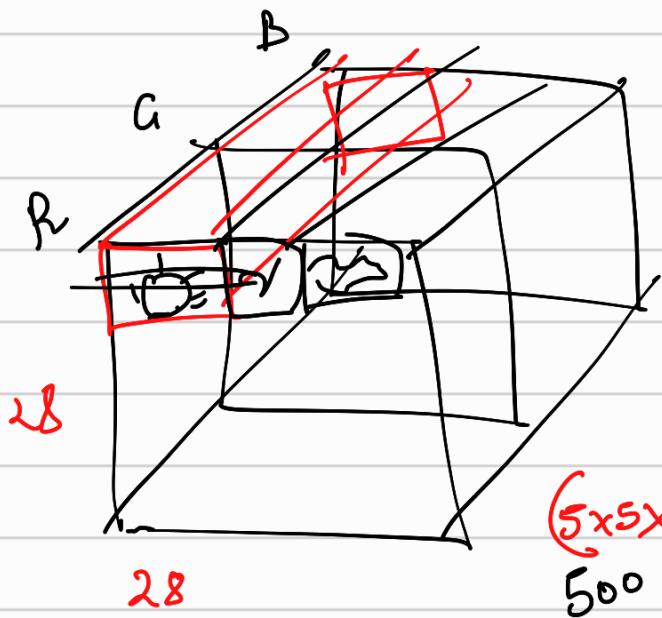
$$W' = \frac{W - f}{S} + 1$$



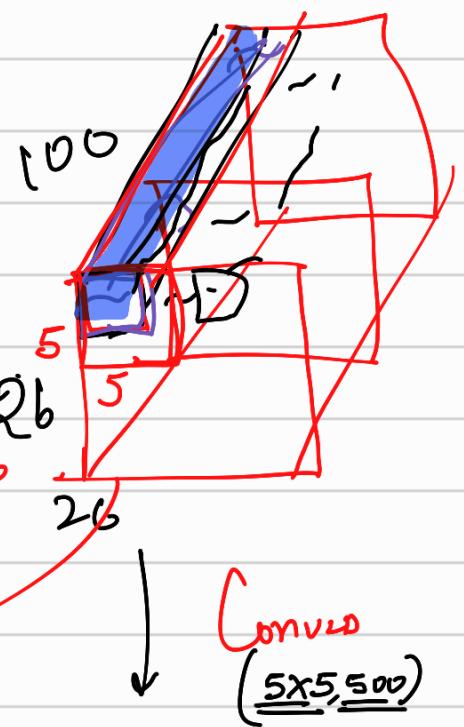
$$H' = \frac{28 - 3}{2} + 1$$

$$= 14 \quad 14$$

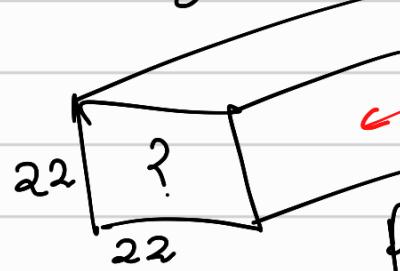




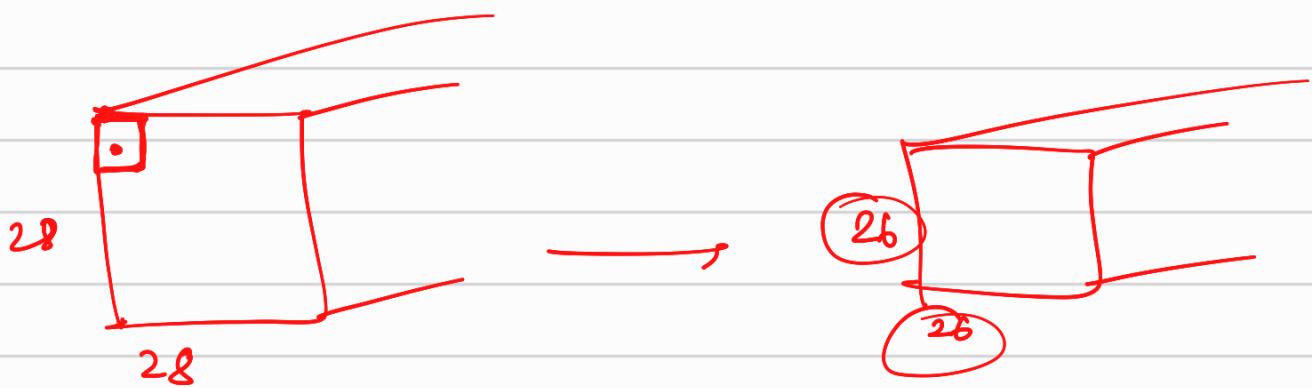
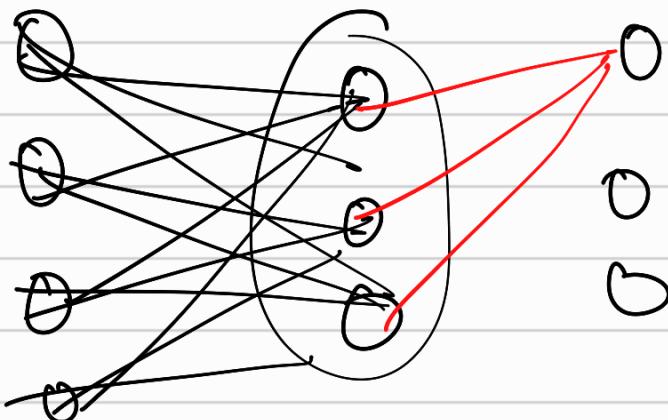
$$(5 \times 5 \times 100 + 1) \times 500$$

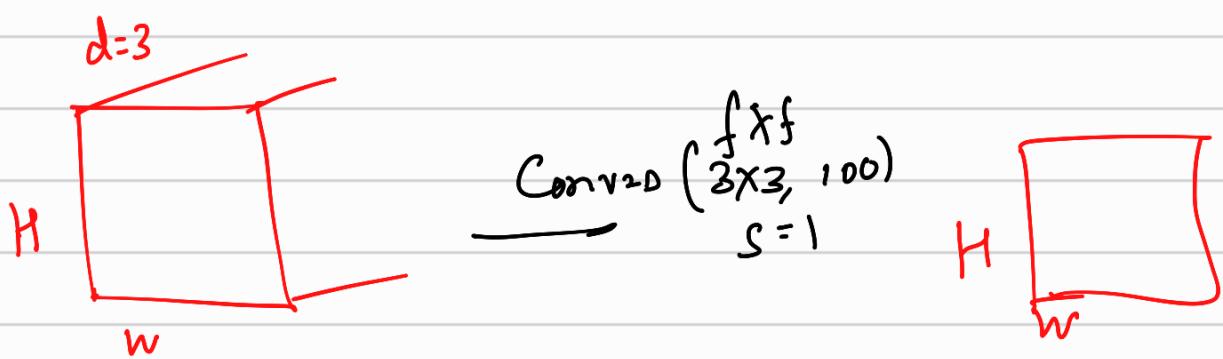
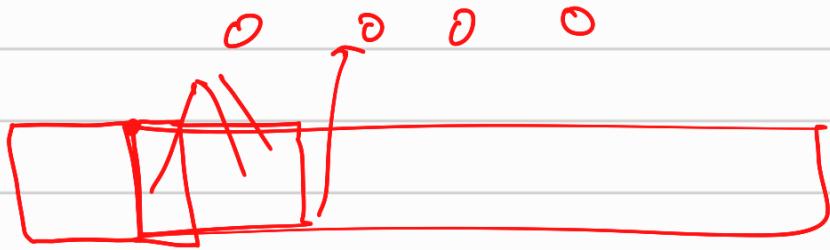


$$\frac{26-5}{1} + 1 = 22$$



param?





$$H' = \frac{H - f + 1}{s}$$

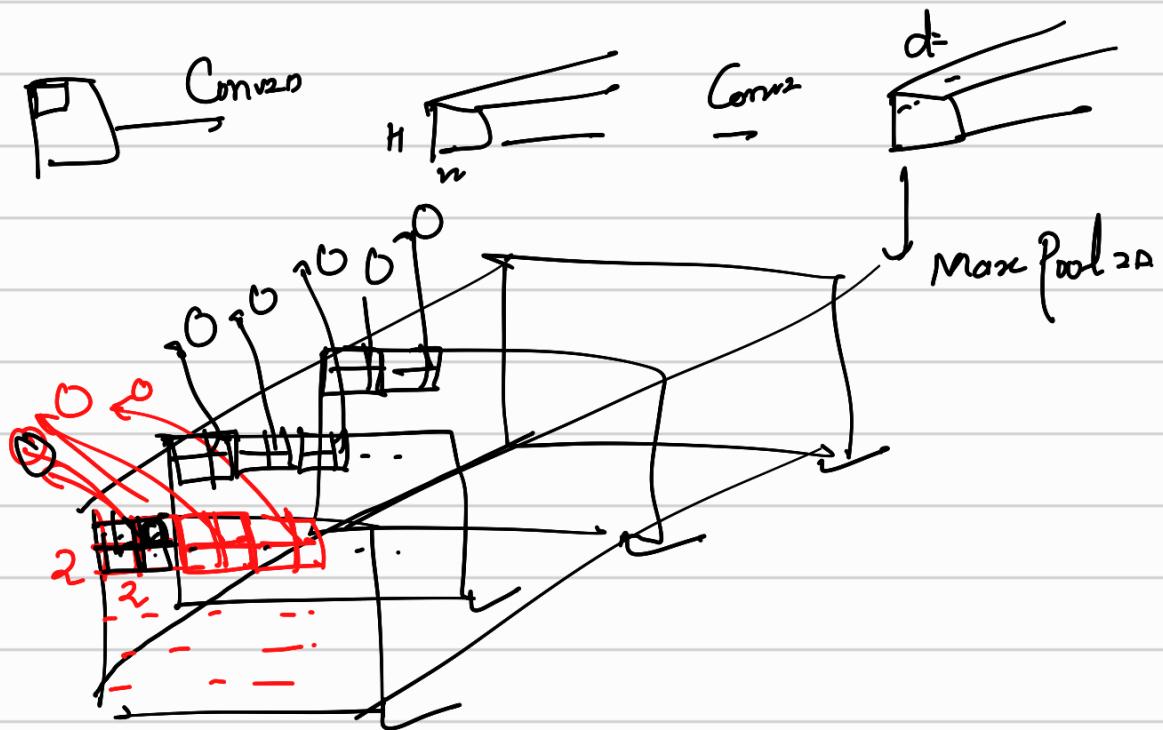
$$W' = \frac{W - f + 1}{s}$$

With padding

$$\underline{H'} = \frac{\underline{H} + 2P - f}{s} + 1$$

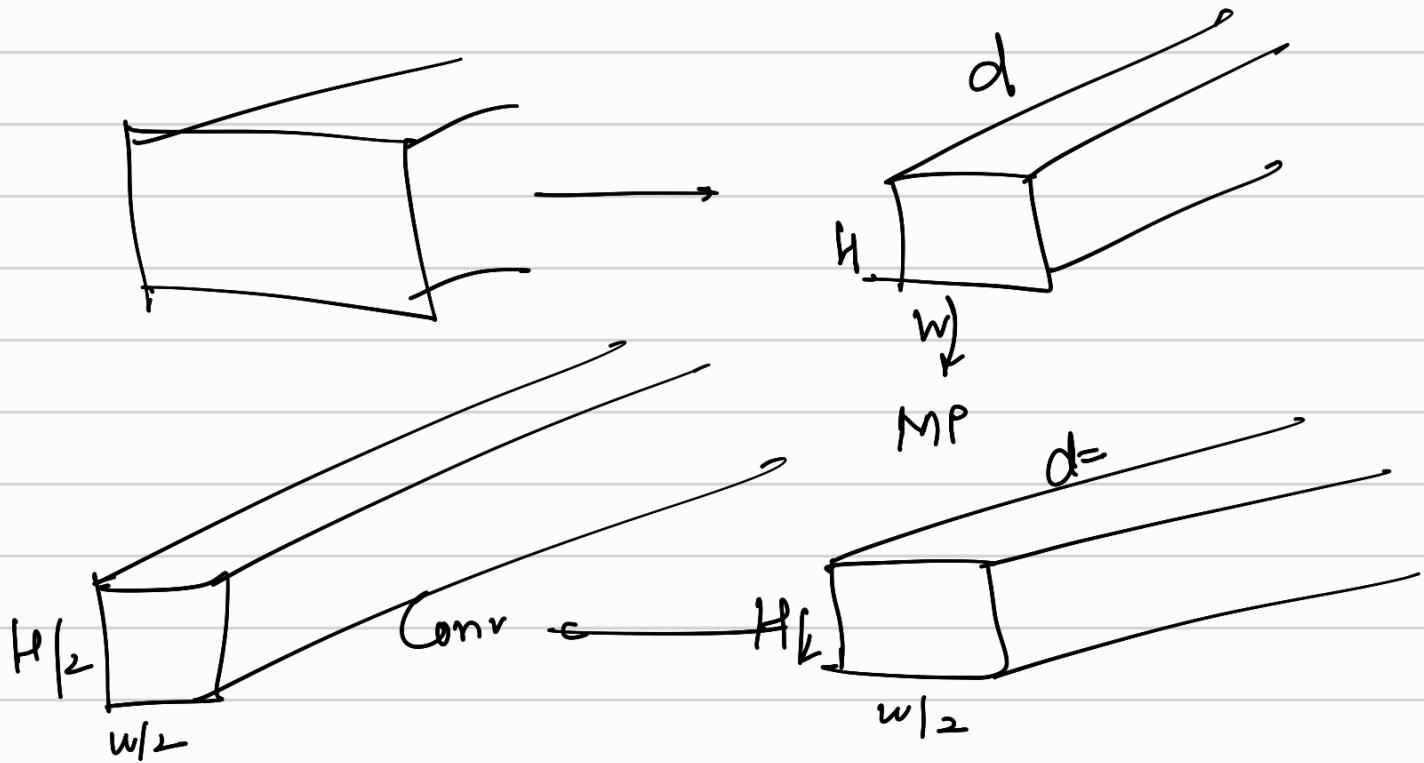
$$W' = \frac{W + 2P - f}{s} + 1$$

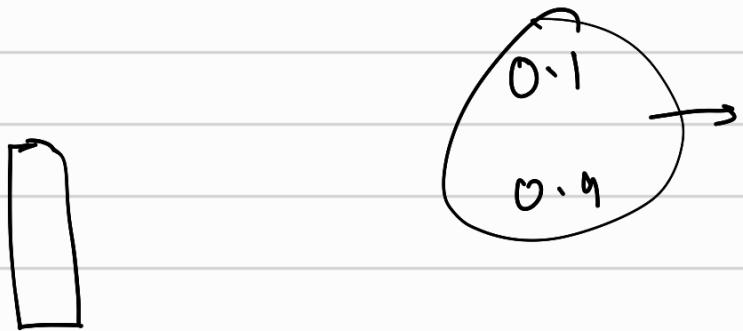
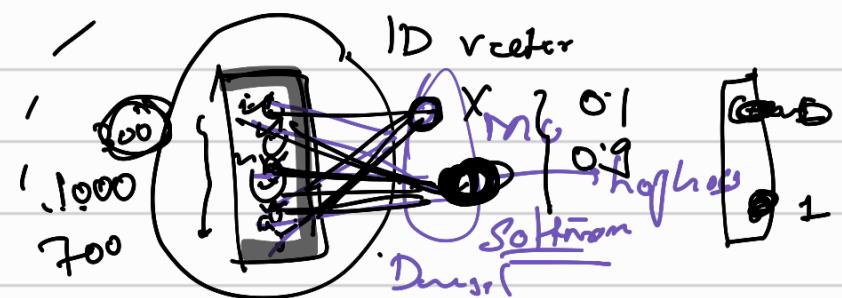
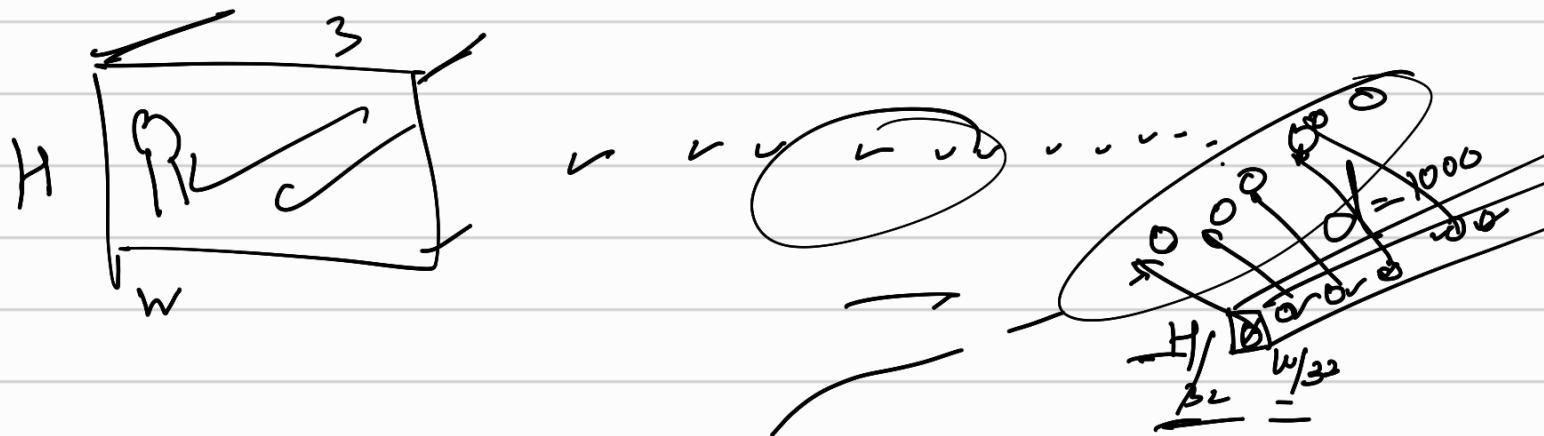
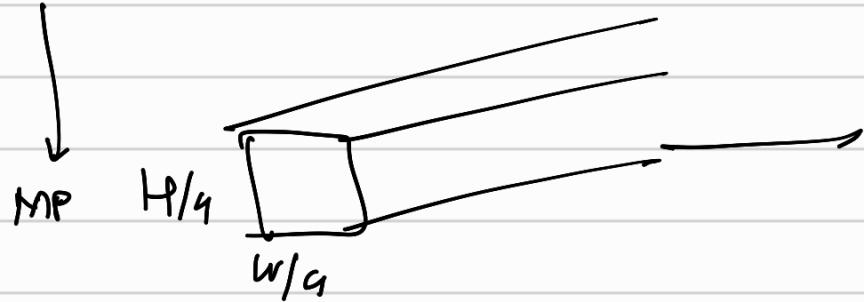
Conv2D (3x3, S=1, Padding="Same")

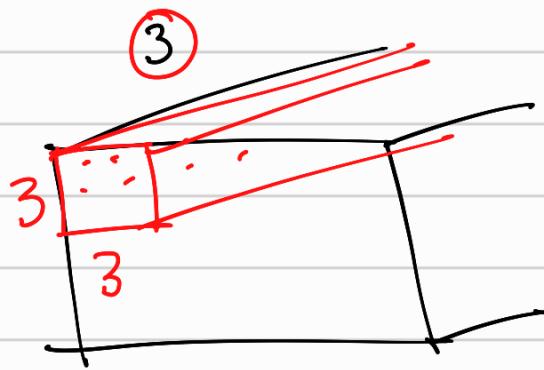


$$H' = \frac{H - f + 1}{s}$$

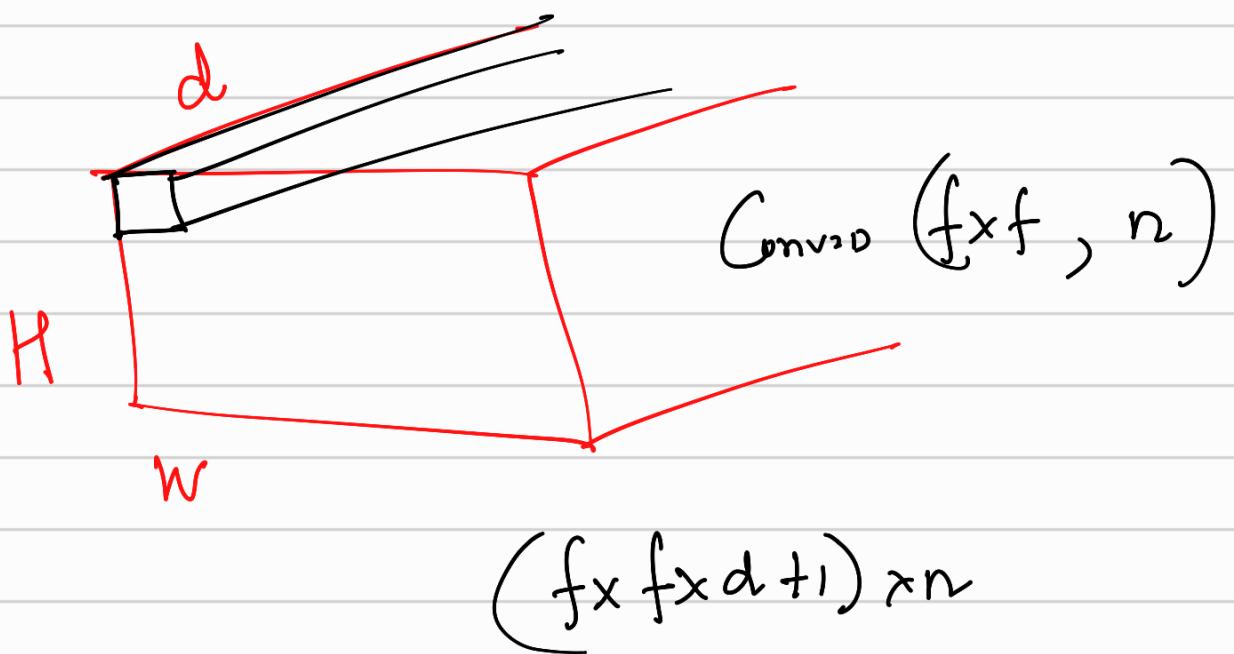
$$H' = \frac{H - 2 + 1}{2}$$







$$(7+1) \times 32$$



$$\text{Conv2D } (f \times f, n)$$

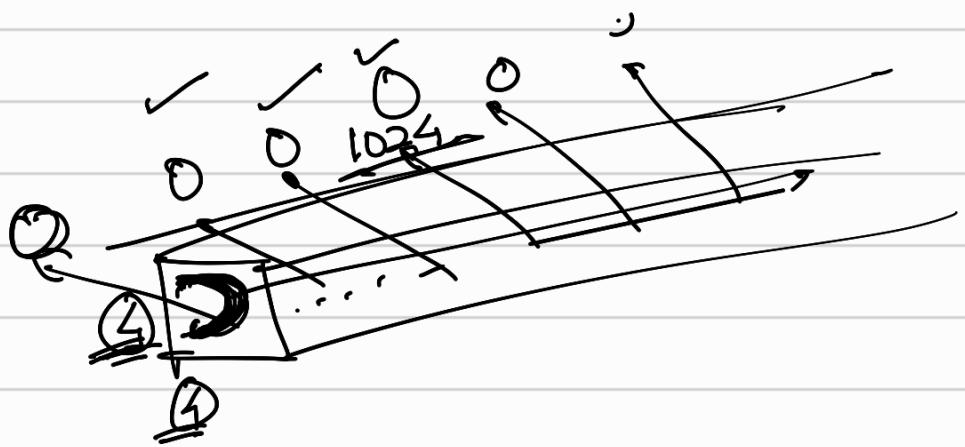
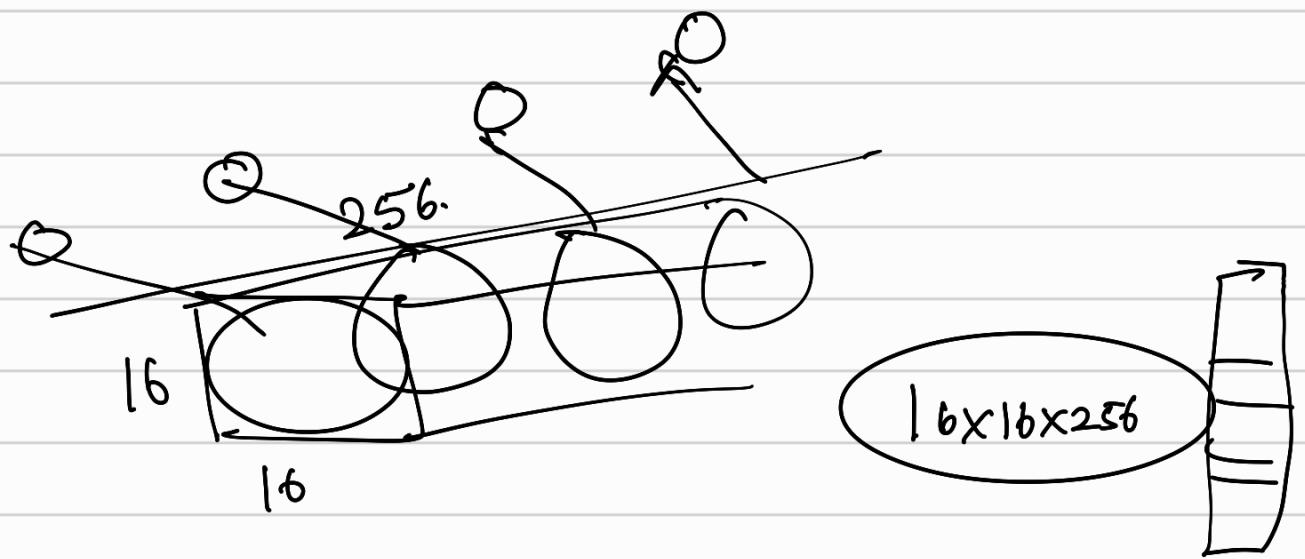
$$(f \times f \times d + 1) \times n$$

$$\frac{H-f}{s} + 1$$

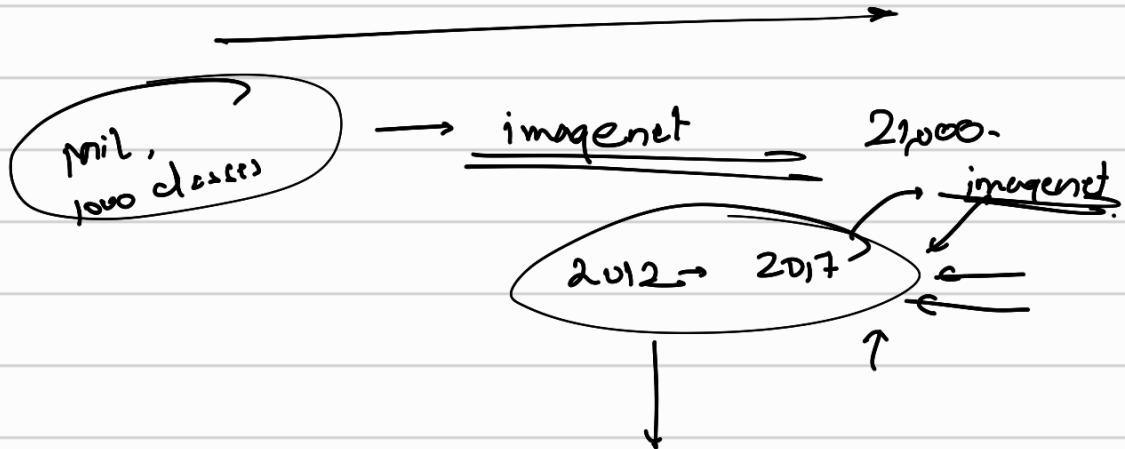
$$= \frac{256-2}{2} + 1$$

$$= 127 + 1$$

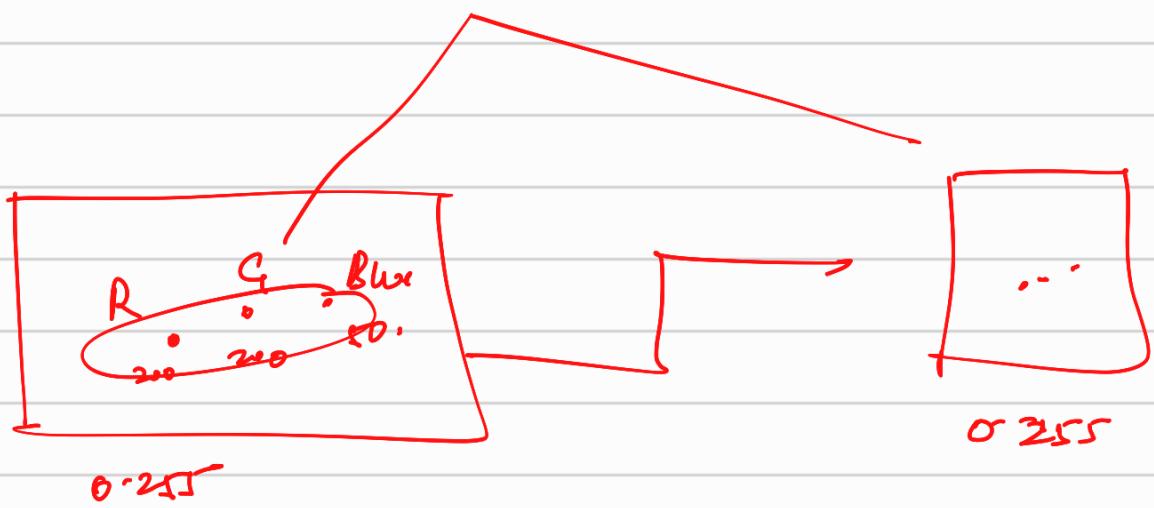
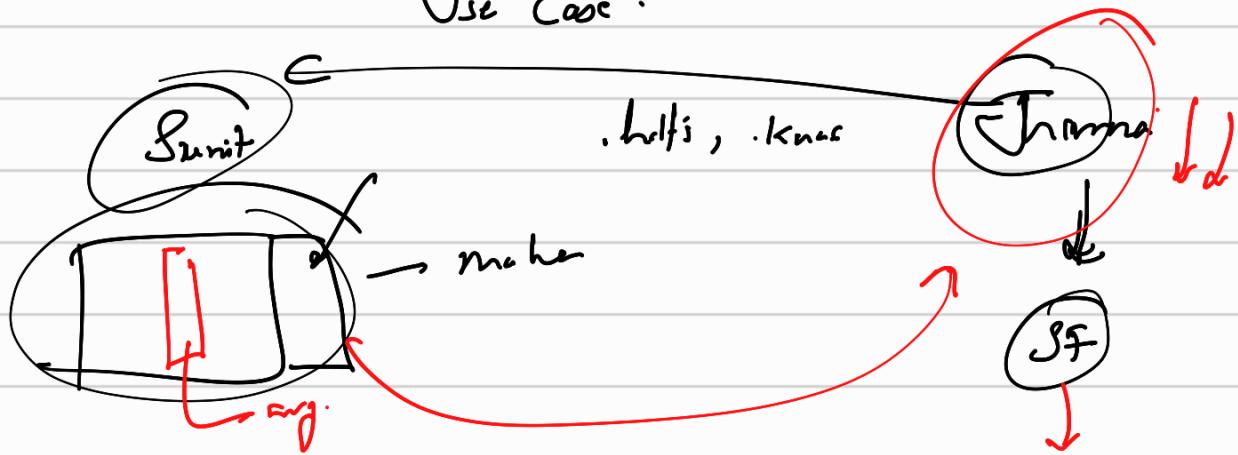
$$= 128$$

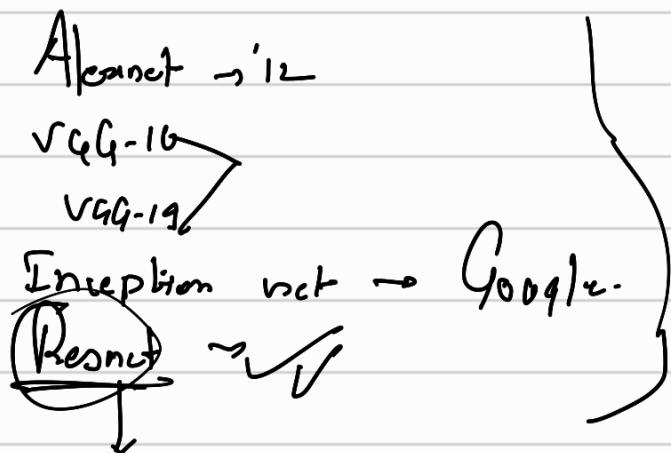
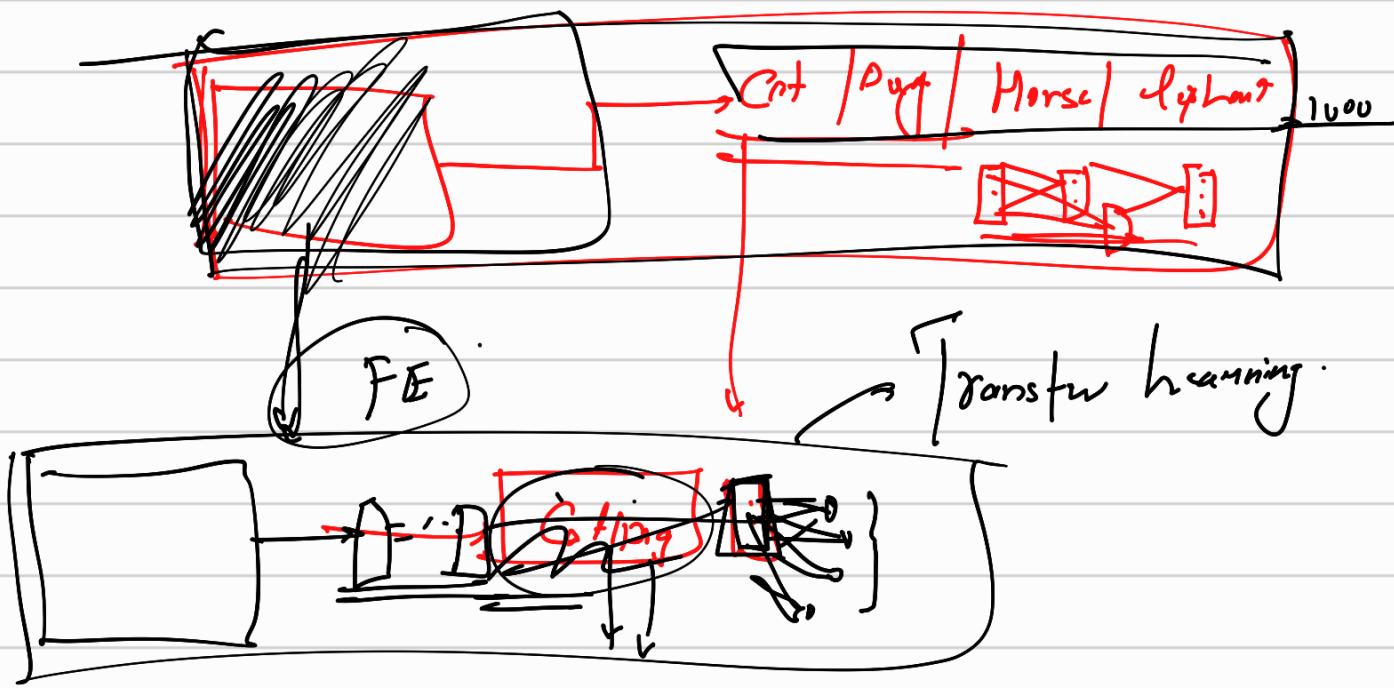


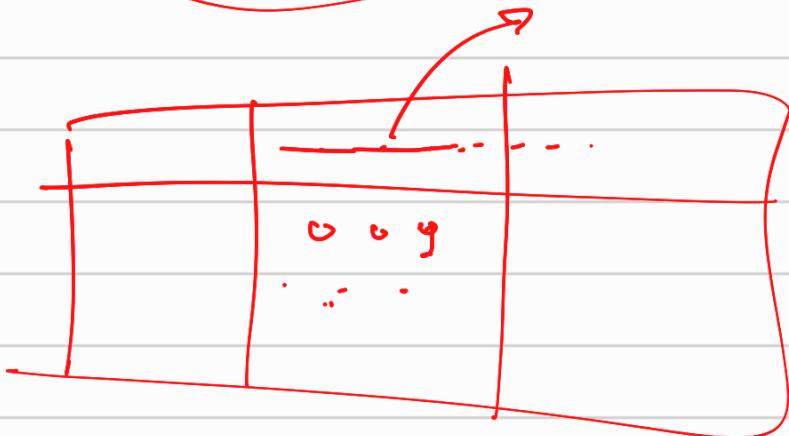
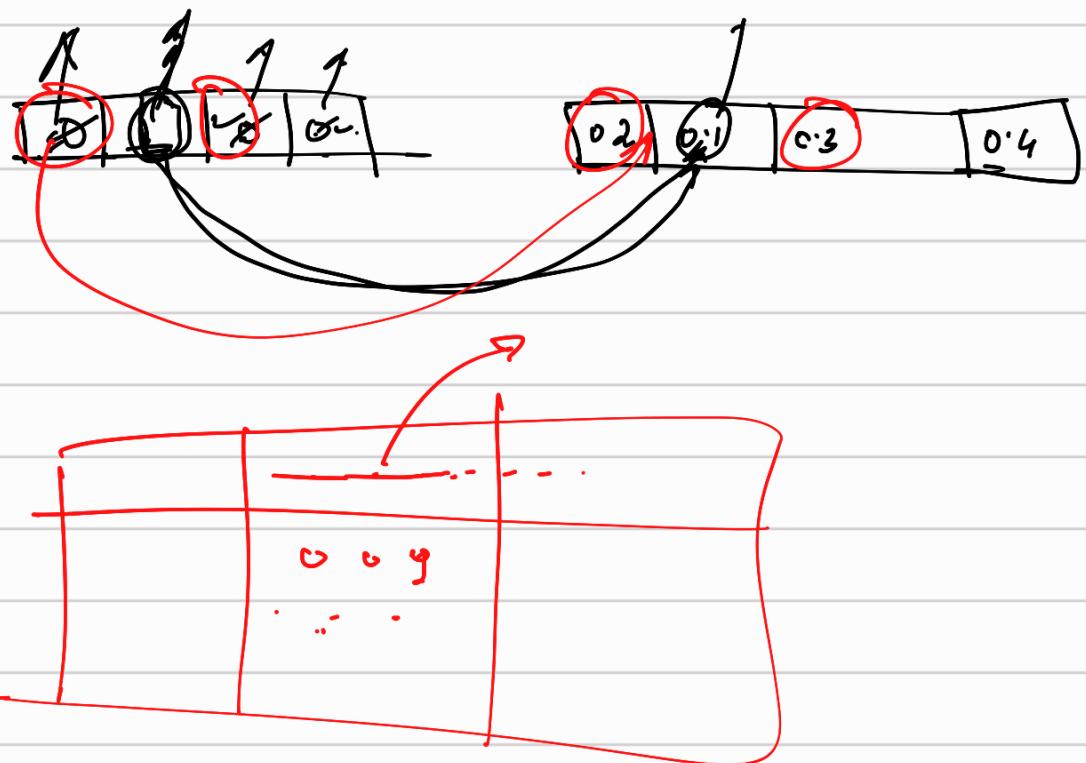
Transfer learning



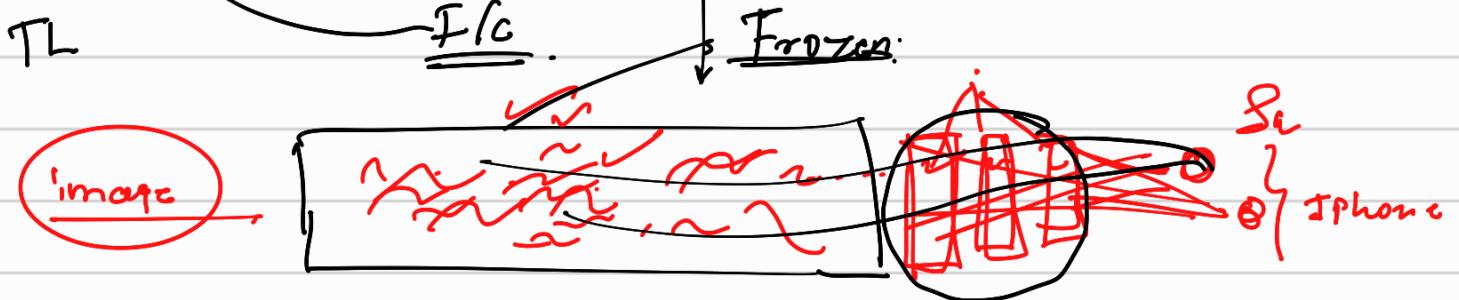
Use Case :





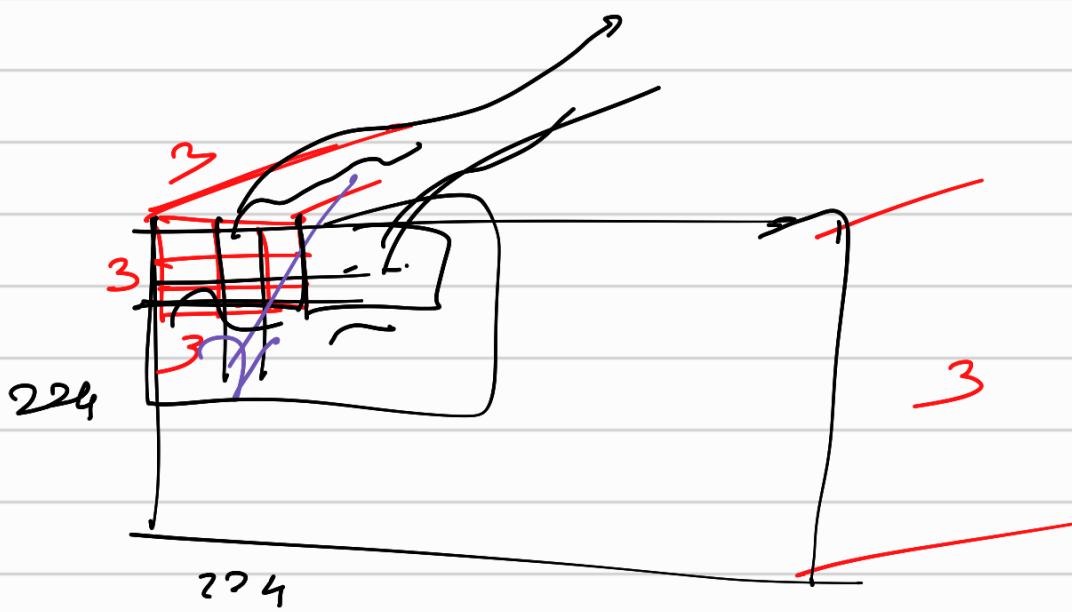
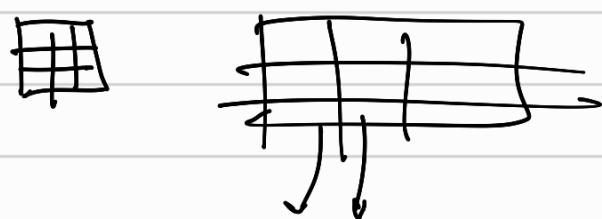


TL (Imagenet dataset).



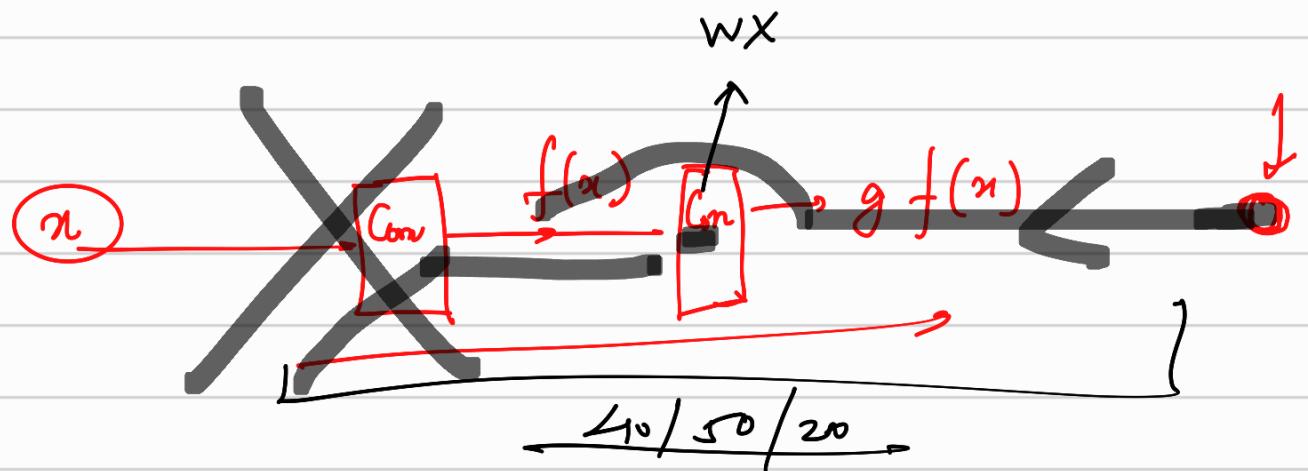
VGG-16 VGG-19

→ Conv conv MP → Conv conv MP → FC FC FC

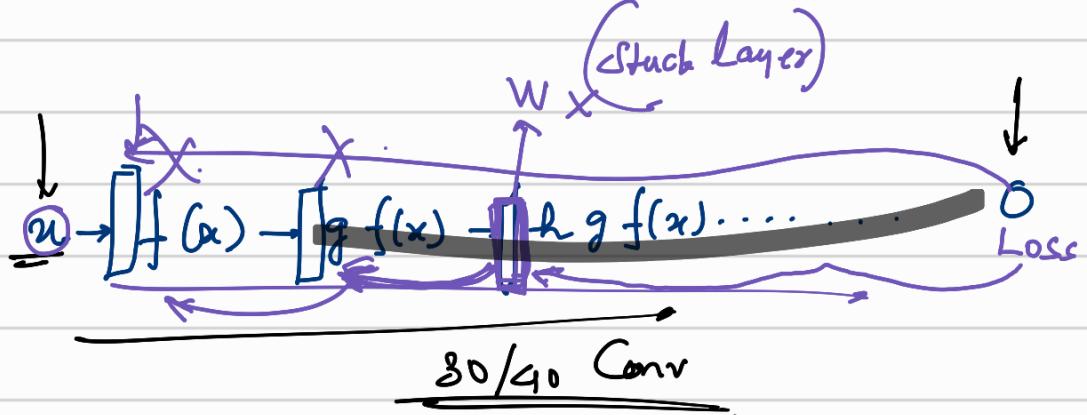




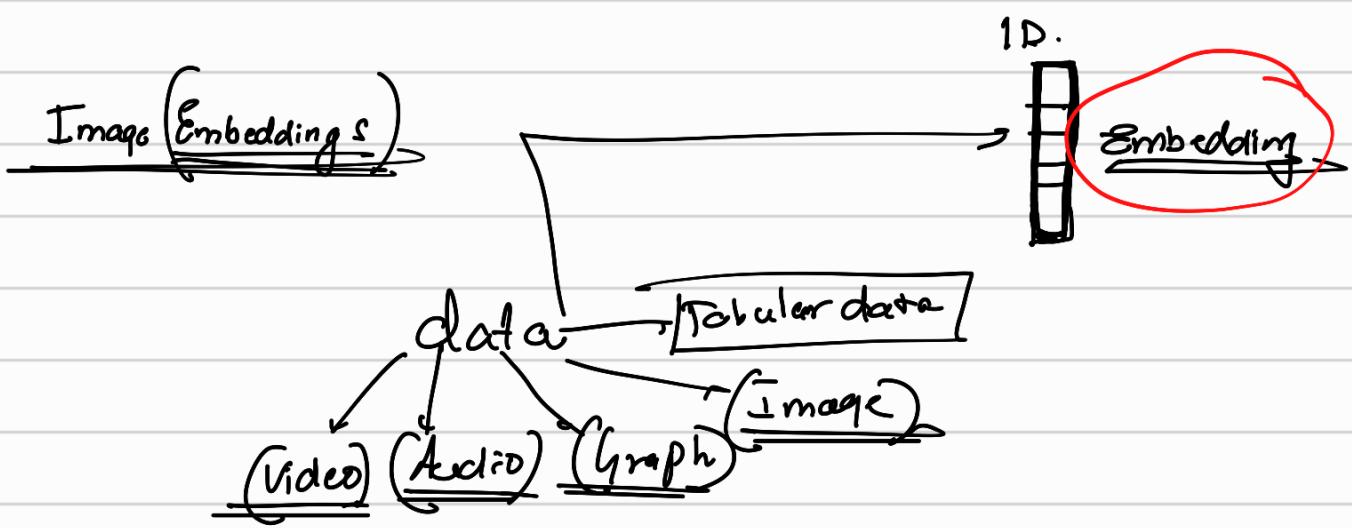
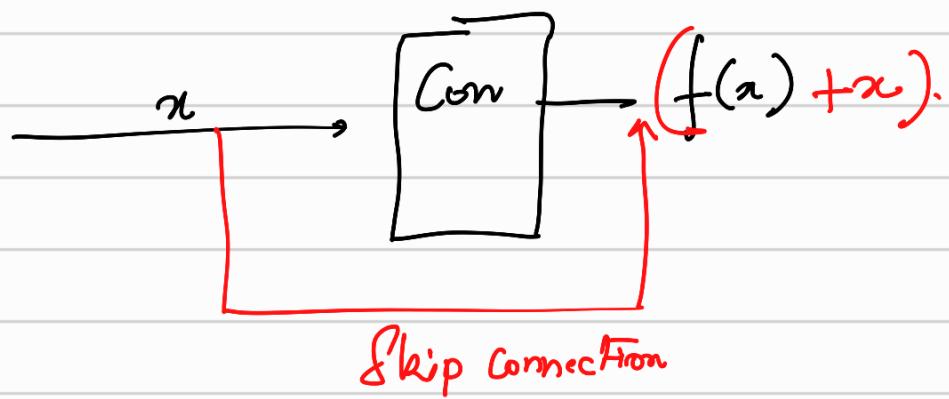
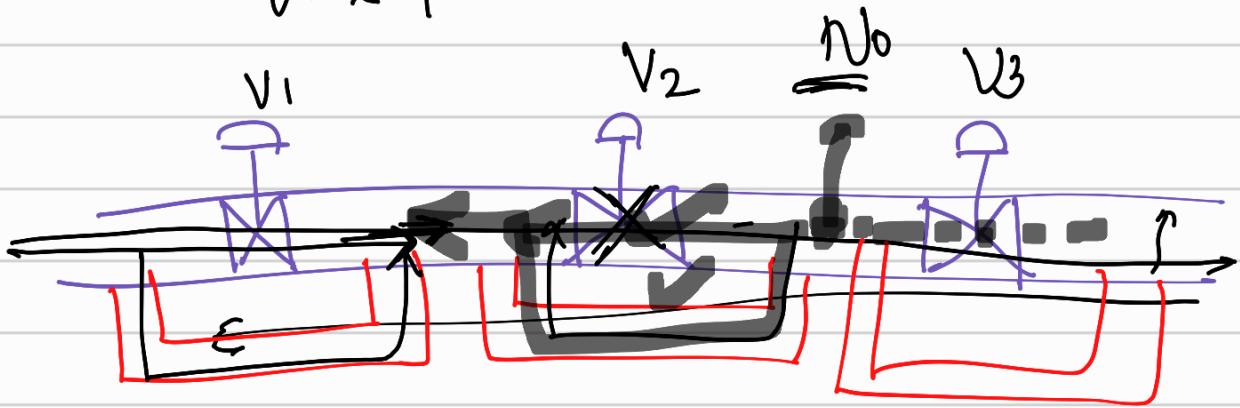
Resnet



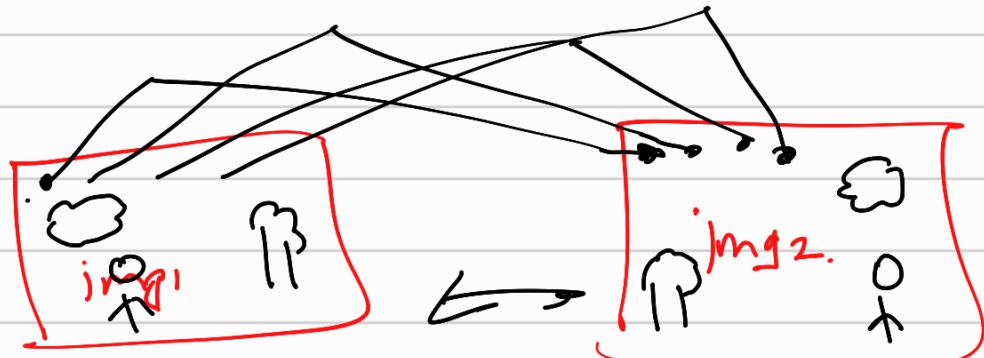
Resnet



$V = \text{Layers}$

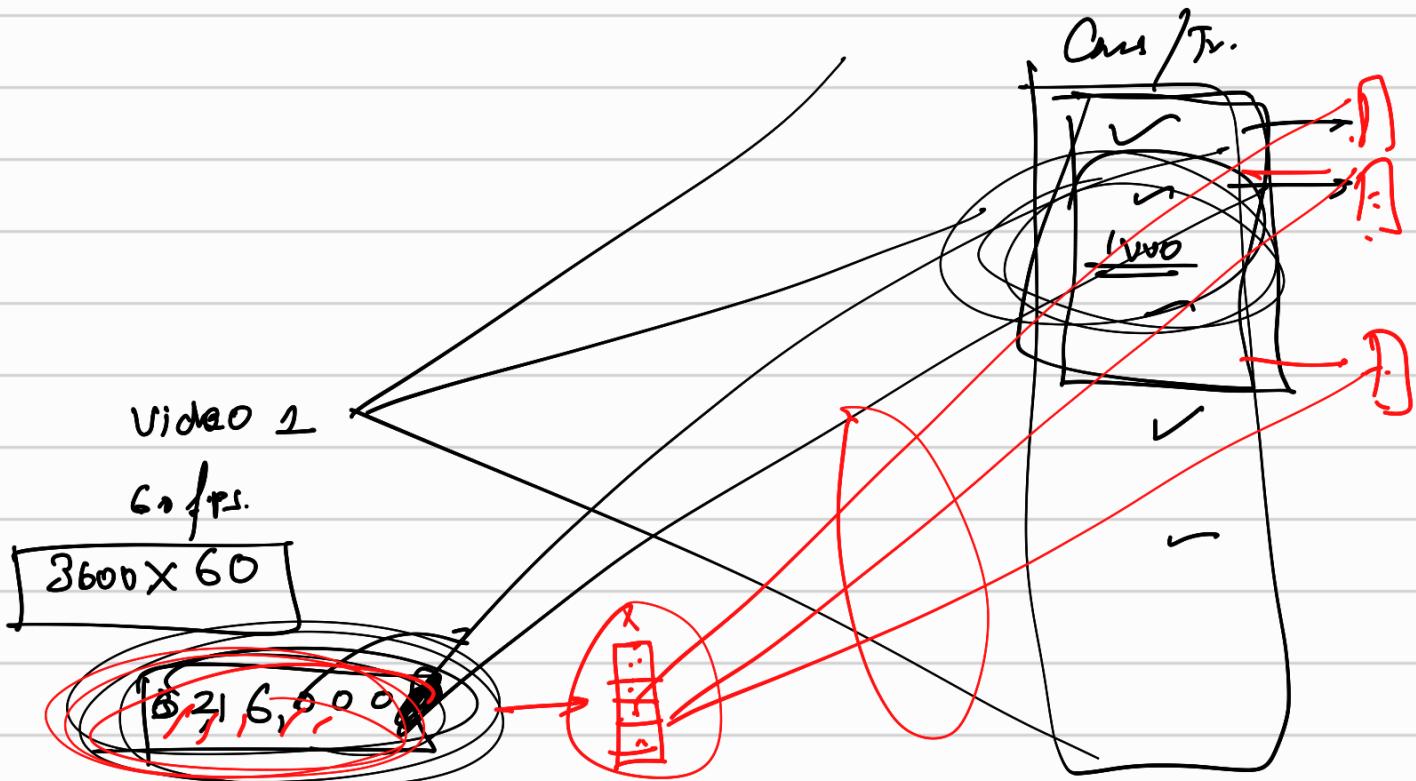


Youtube Video \rightarrow 500 MB 20%



EV

Frobenius Norm.



Video 1

60fps.

3600×60

18,216,000

100

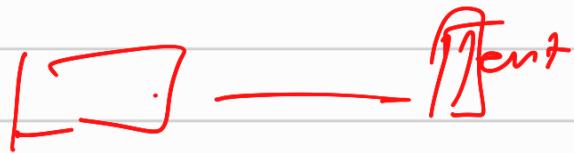
BMW
USA

Review

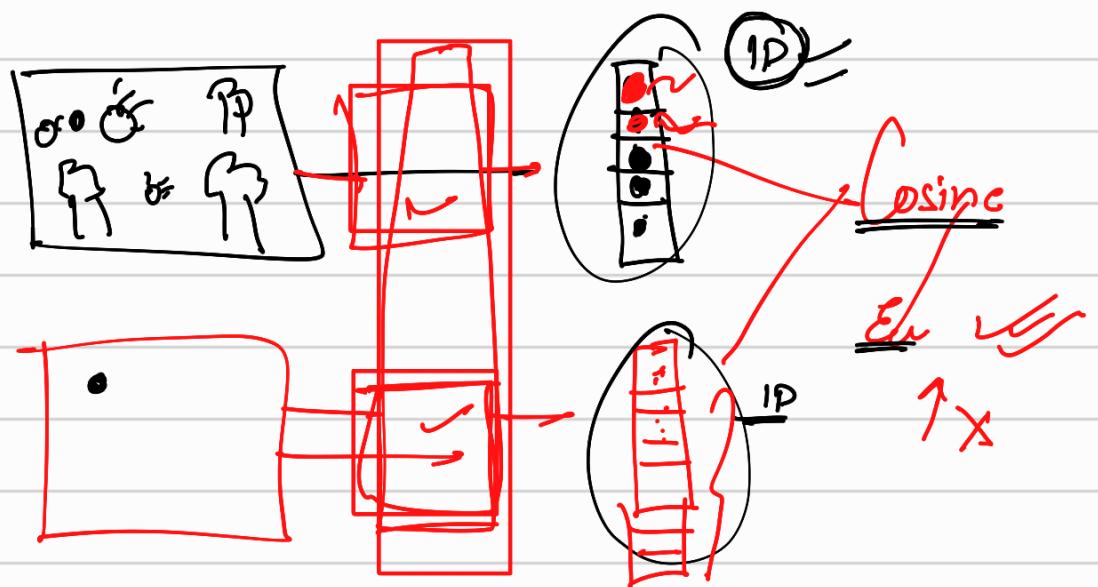
Mercedes Review

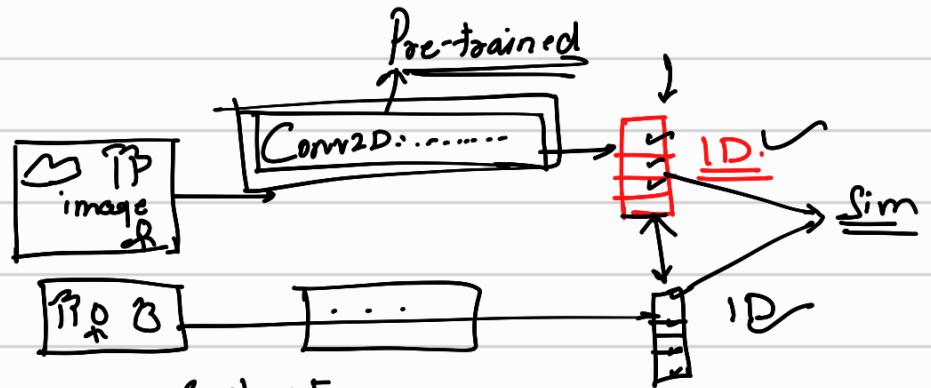
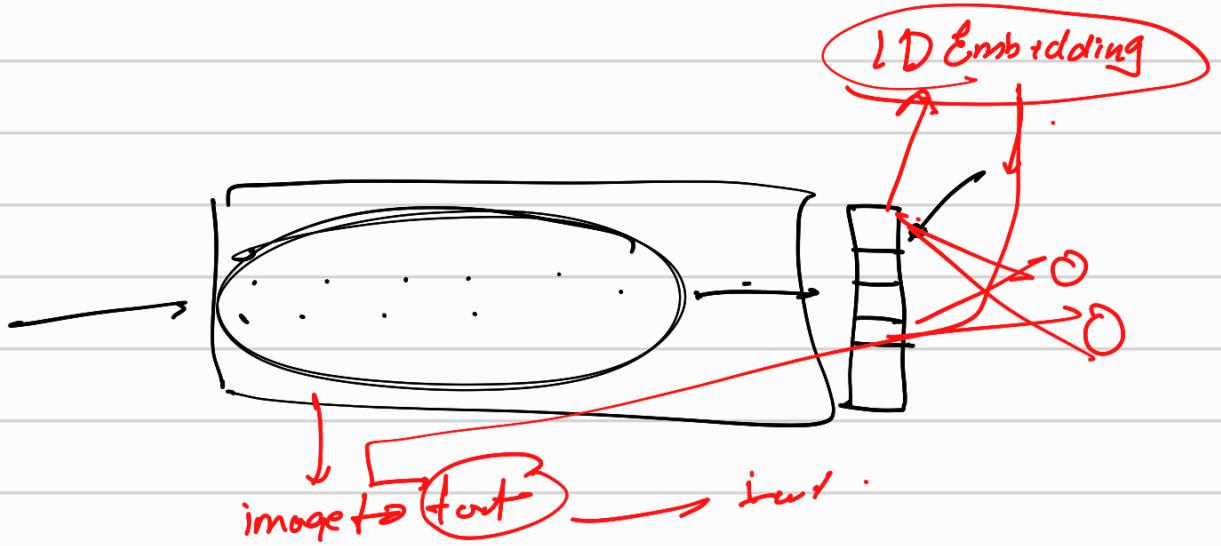
Bmw India

Tent → image



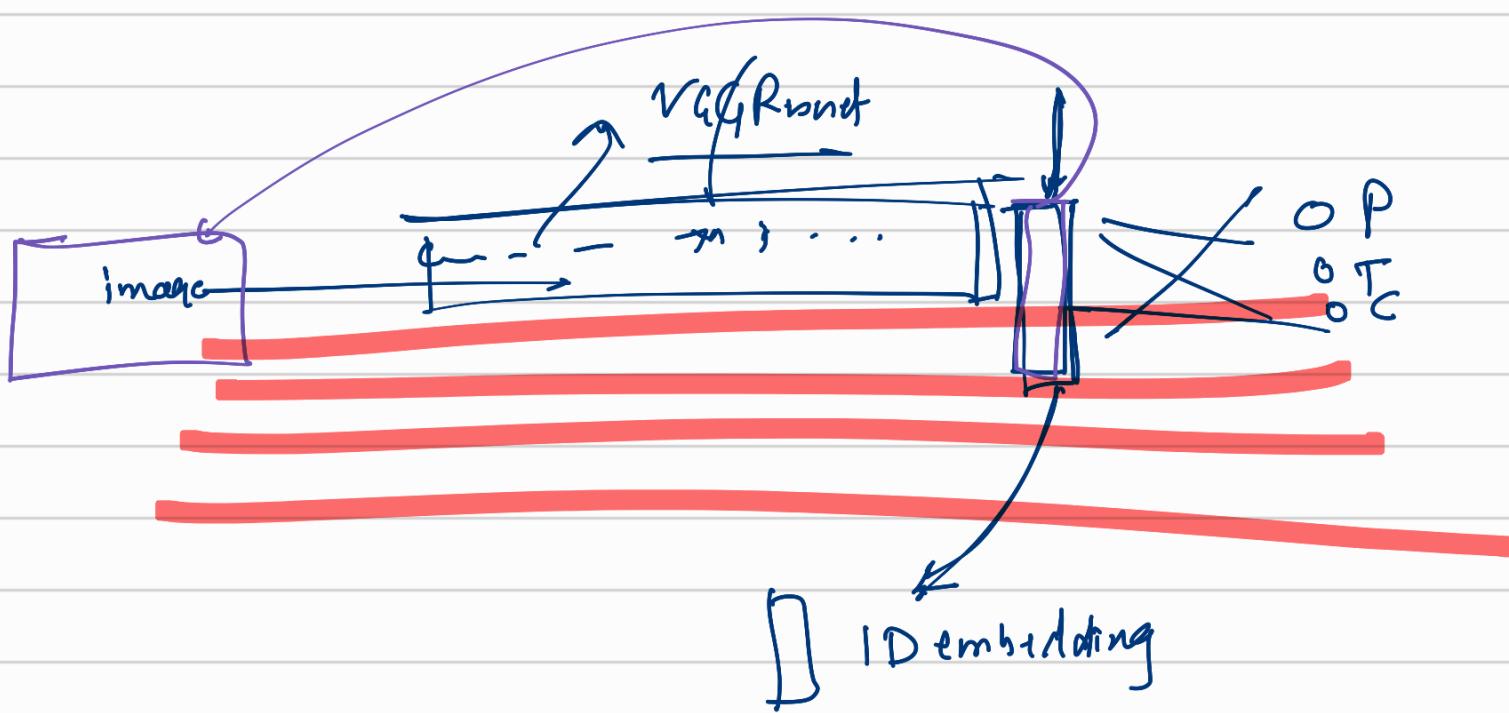
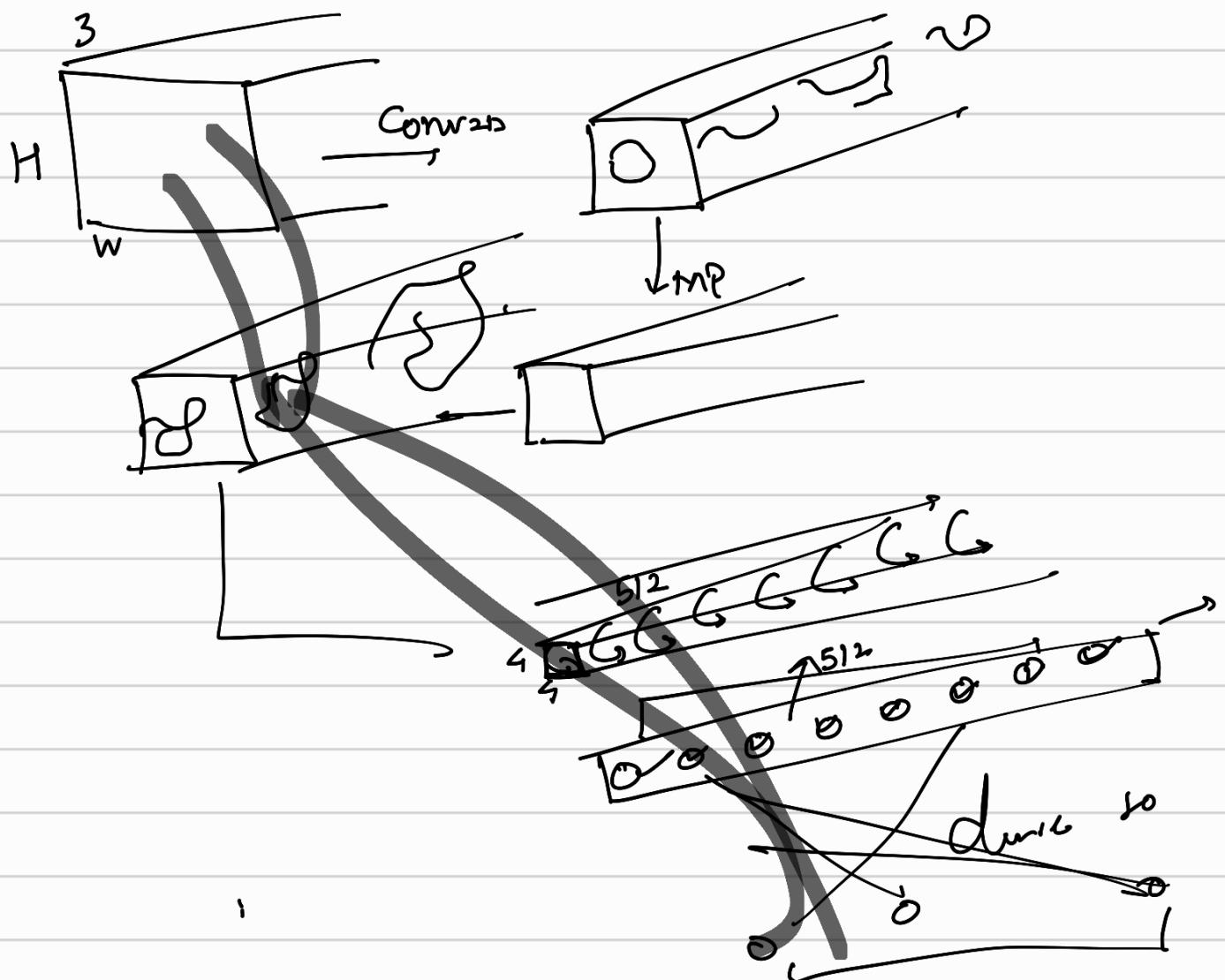
Tent → video.



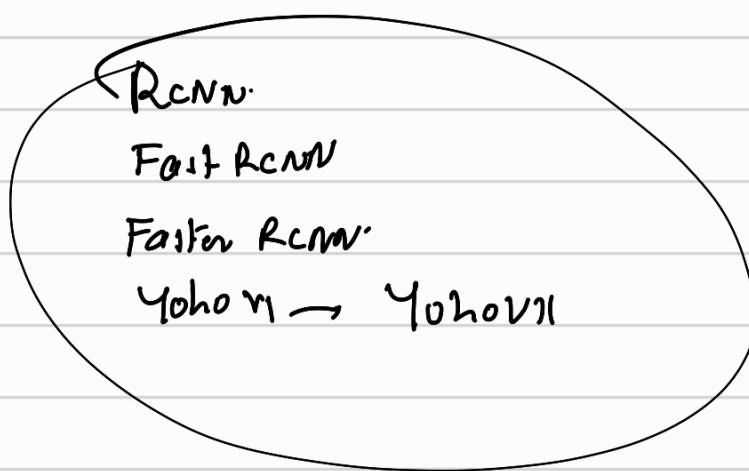
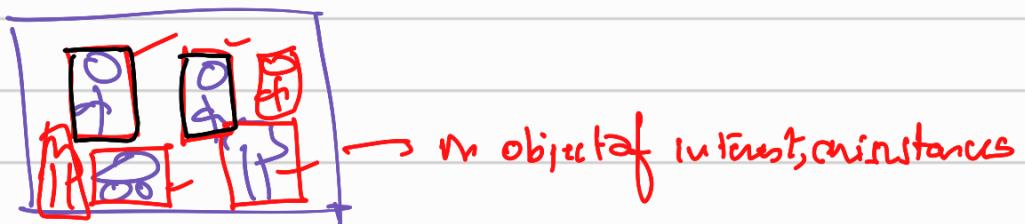
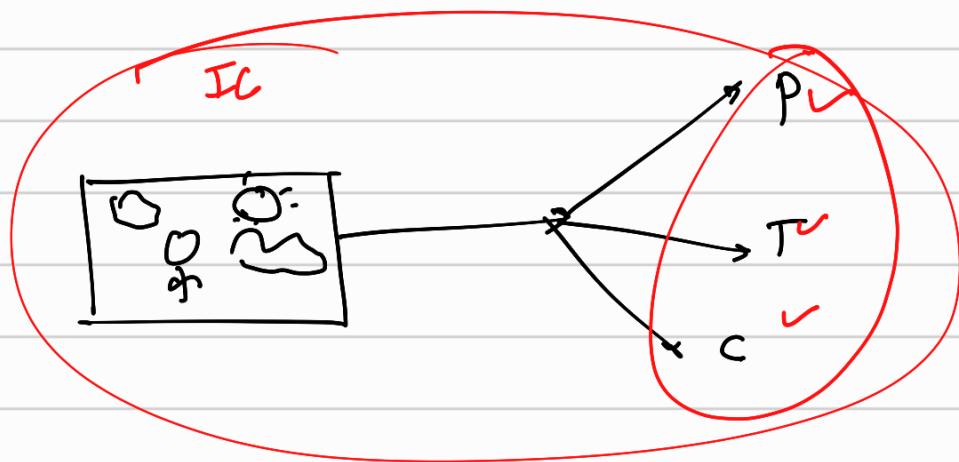


① Calculating image similarity.

- Recommendation systems ✓
- Clustering images ✓
- Facial Recognition Systems ✓
 - ↳ Siamese Network -
- image → text
- ↳ text → image



Object detection & localization :

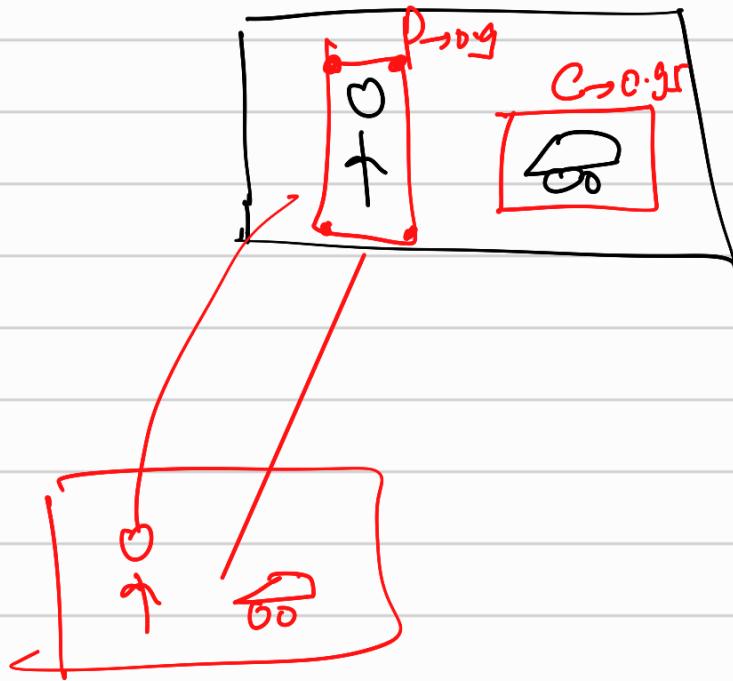


Annotation

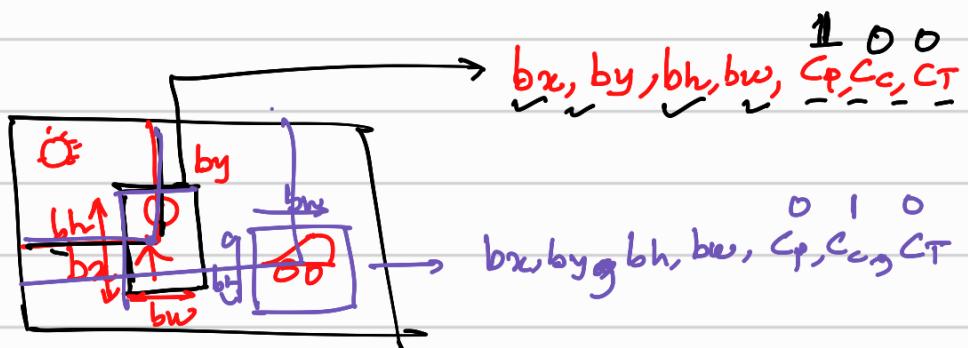
.

,

.



Annotation of bounding boxes to solve for the supervision of obj det and localization problem.

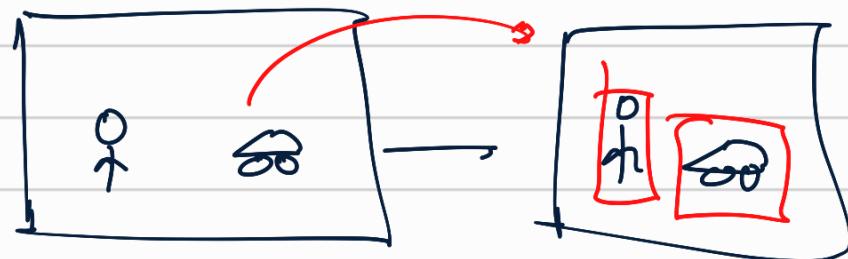
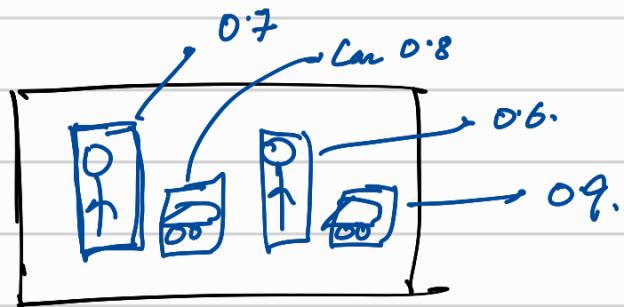


Annotation tools

Robotflow



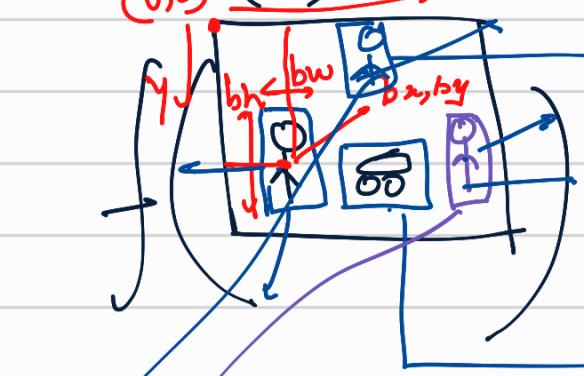
Obj Data and Io



Training time is:

$$(0,0) \quad (\alpha) \quad \alpha_s$$

$$\rightarrow y$$



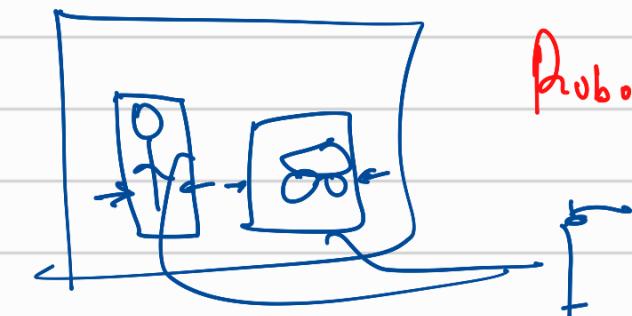
bbox 1: PTC
 $[bx, by, bh, bw, 1, 0, 0]$.

bbox 2:
 $[bx, by, bh, bw, 1, 0, 0]$.

bbox 3:
 $[bx, by, bh, bw, 1, 0, 0]$.

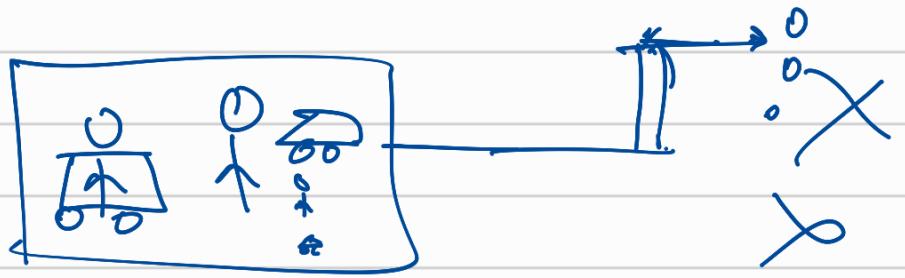
bbox 4 $[bx, by, bh, bw, 1, 0, 0]$.

Roboflow:

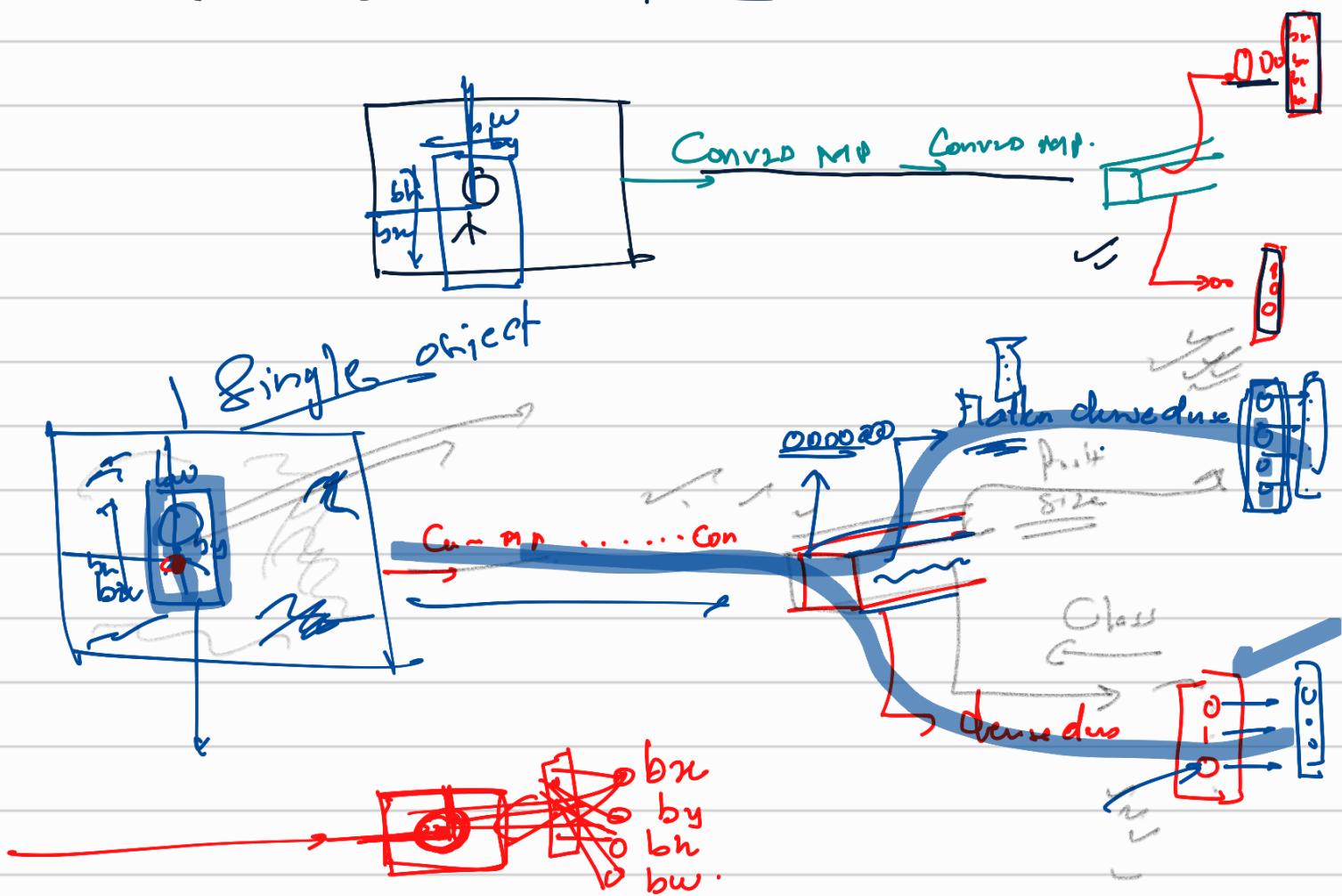


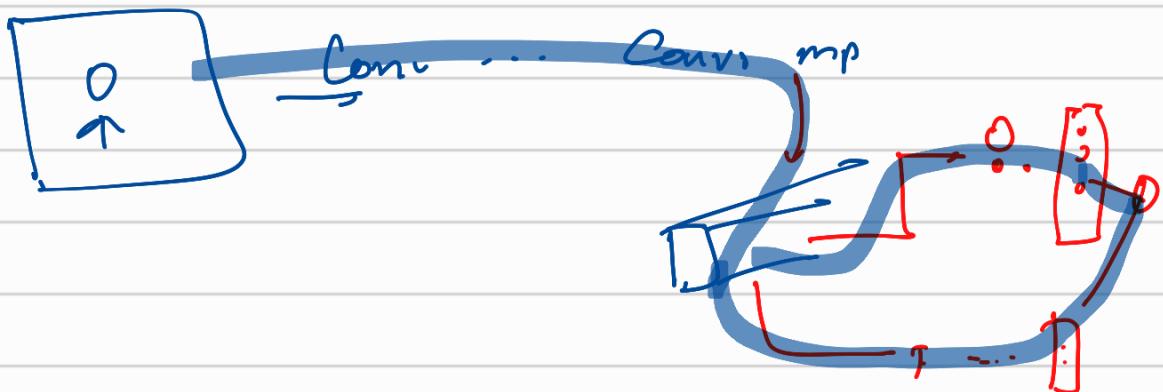
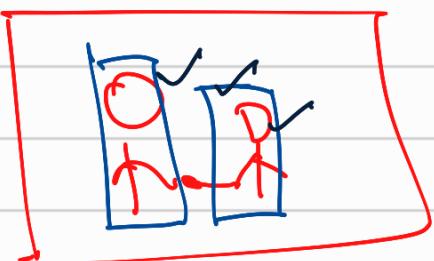
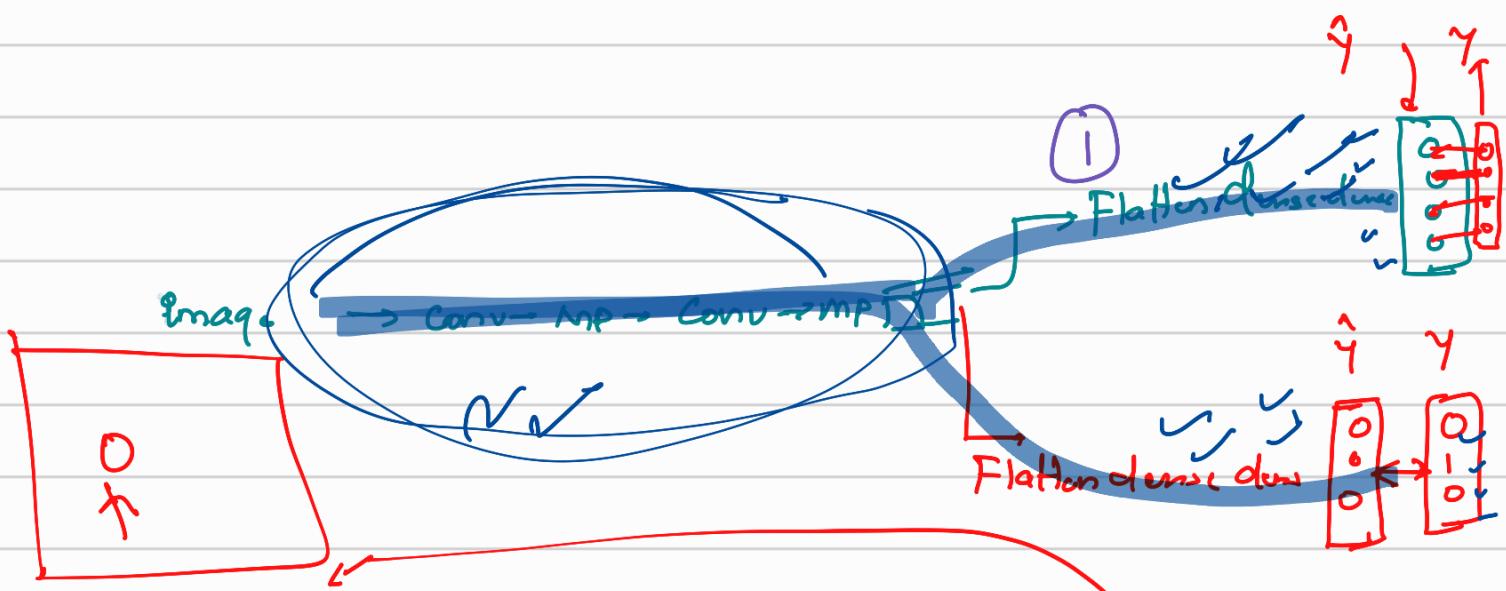
Roboflow.

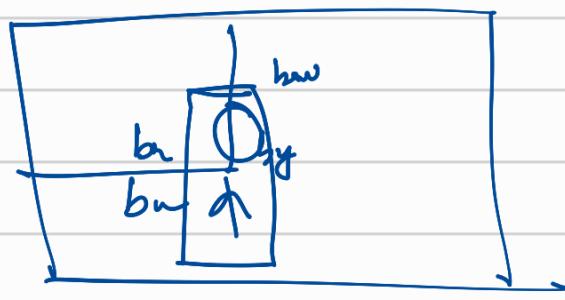
]



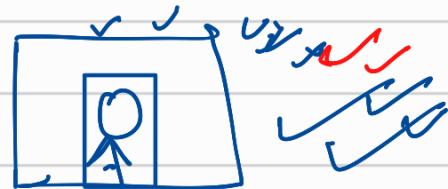
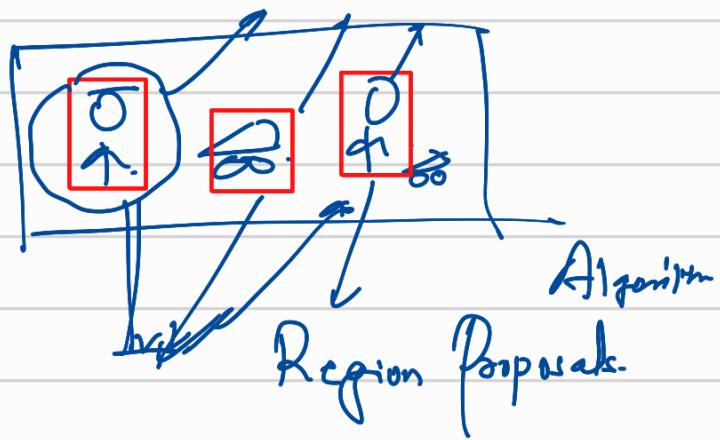
Lets solve the easier problems.



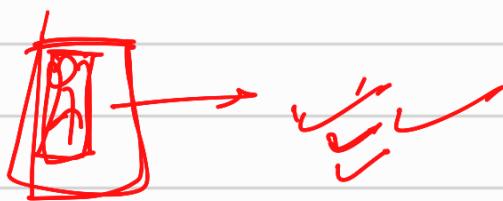
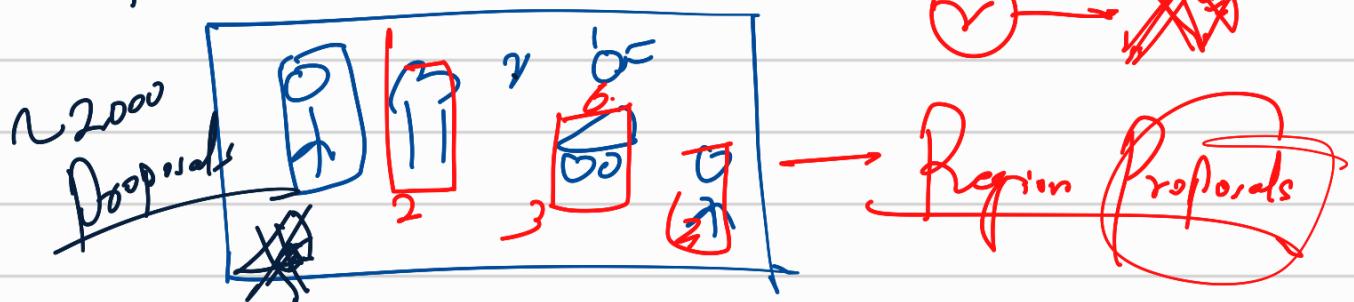




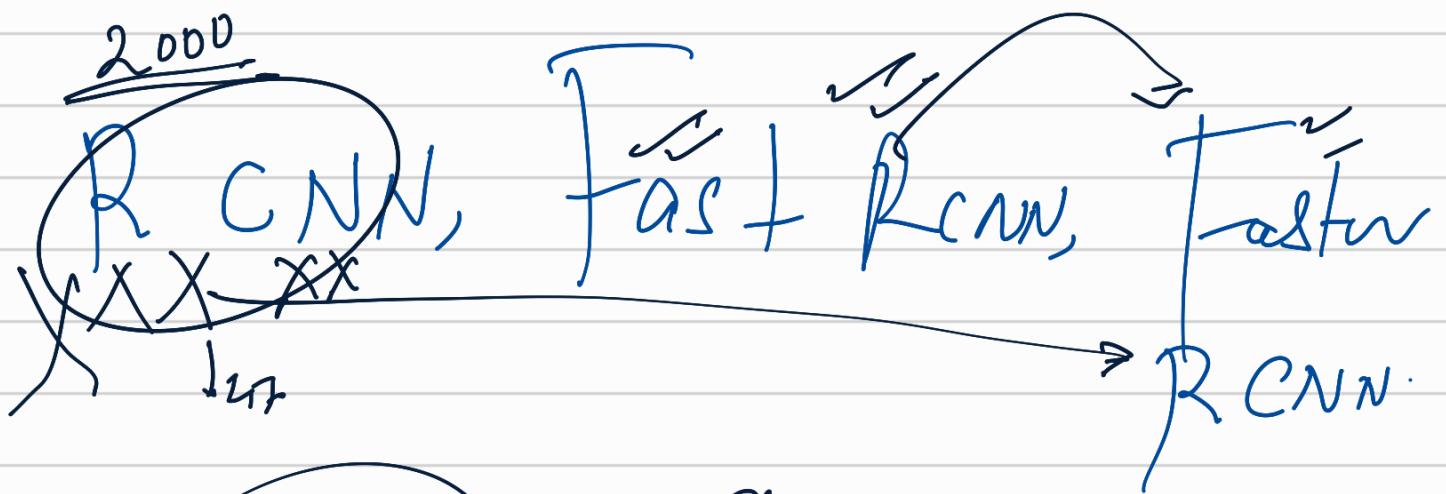
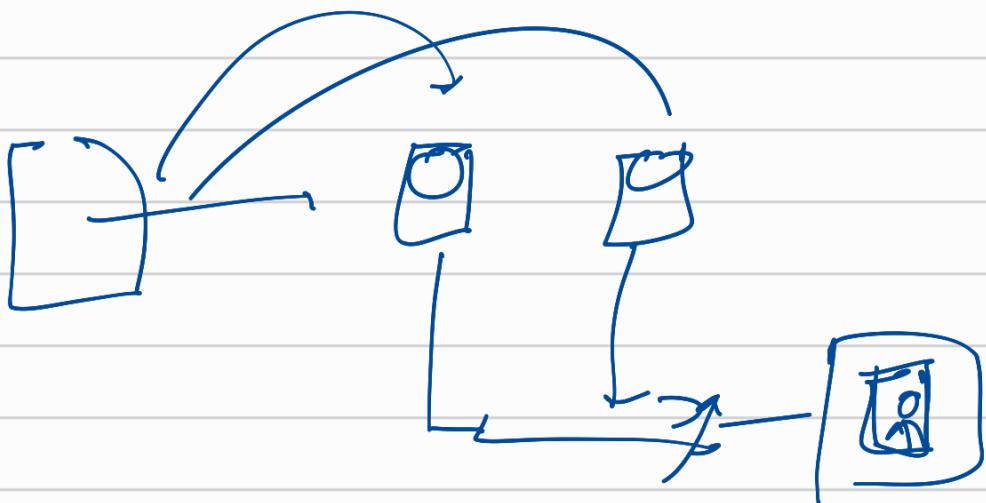
Architecture



Rcnn

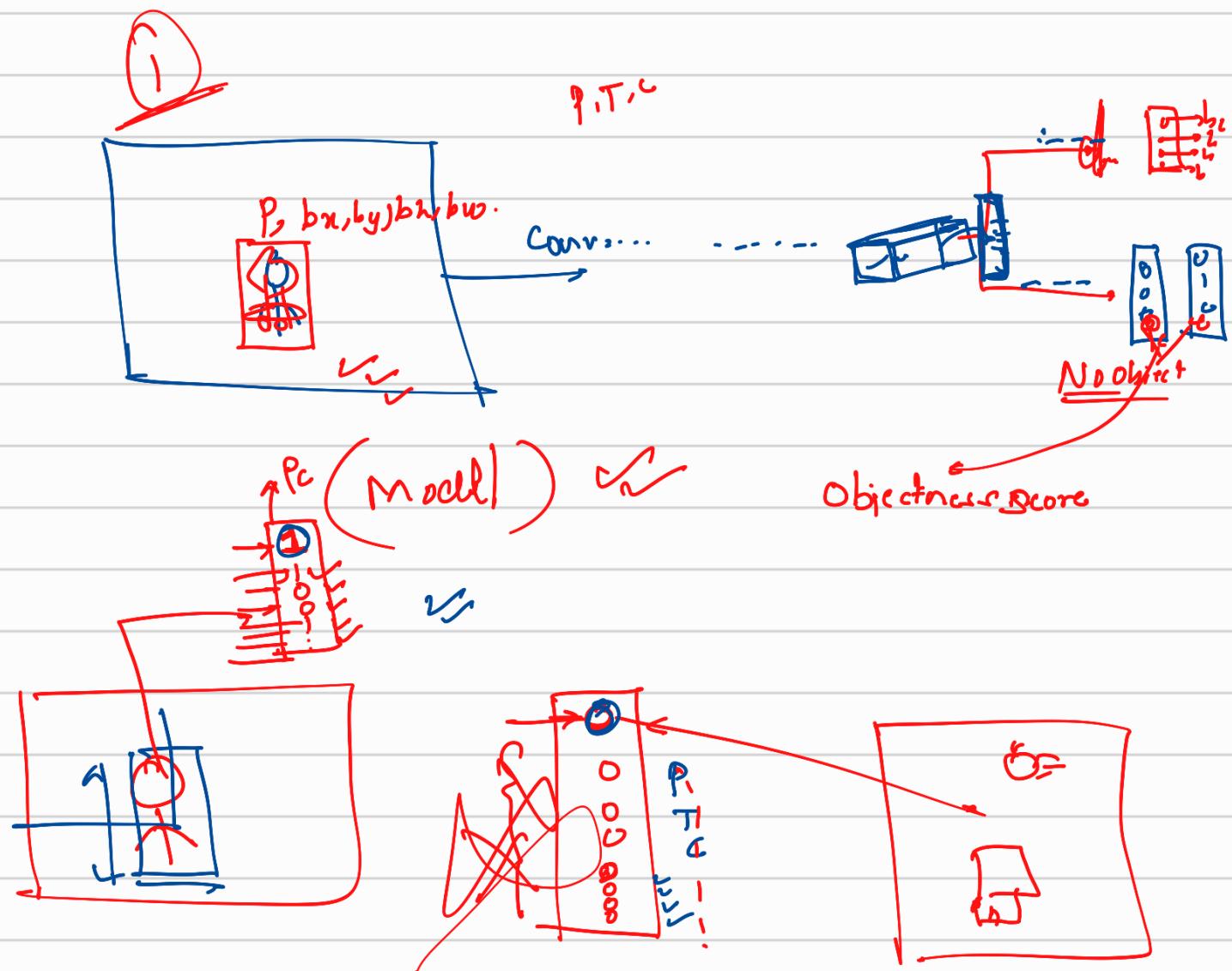
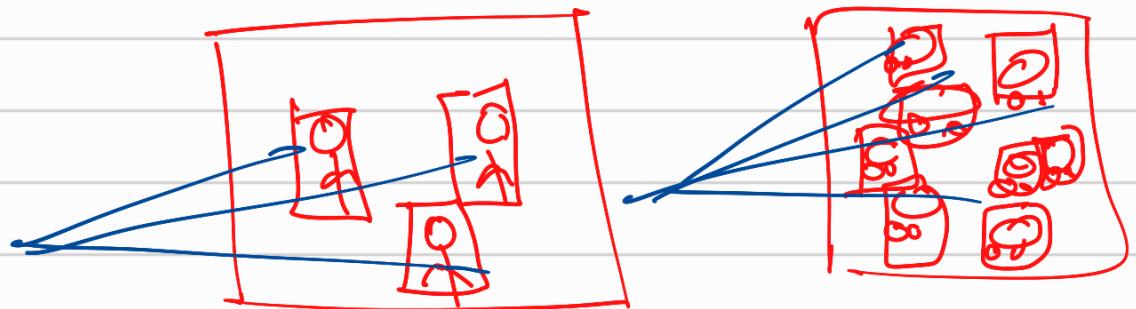
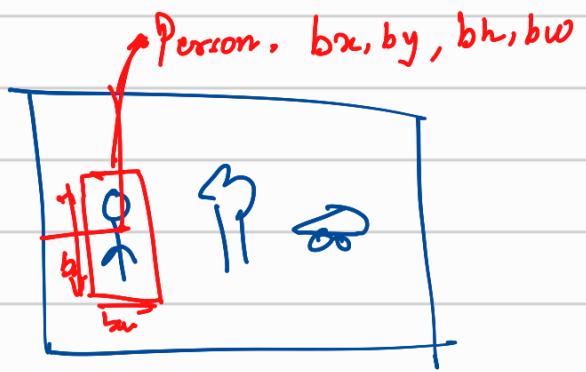


OS 2 Stage



~~YOLO~~ = 1 Stage

~~VGG~~ ~~ResNet~~

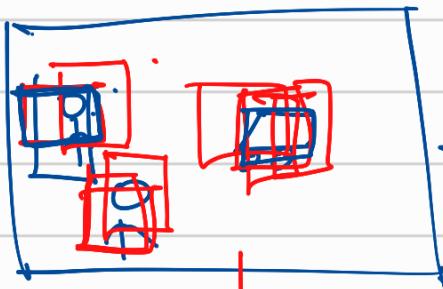


Two Stage Detection Pipeline

① Region Proposals

Fast R-CNN

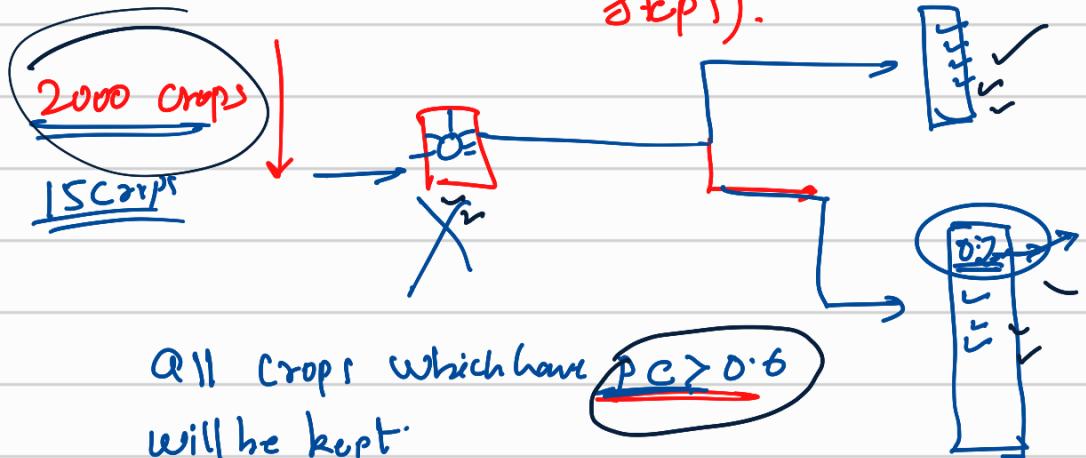
Selective Search



→ Suggest "crop" wherein you may find an object

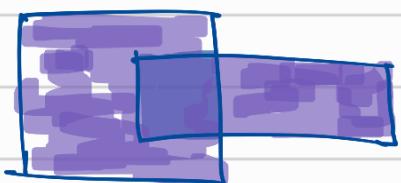
~500 / 2000 crops

model (which has been found by step 1).

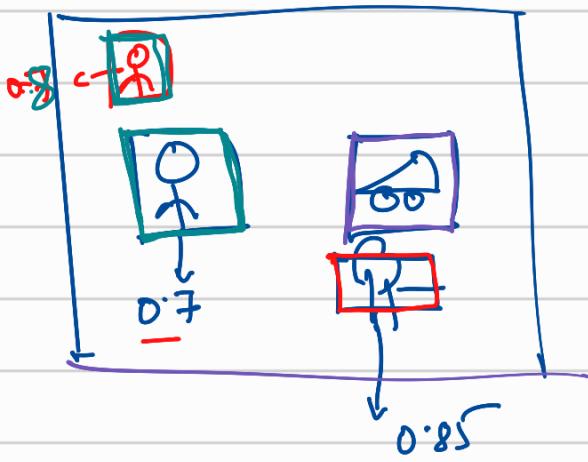


Post Processing : Non-Max Suppression

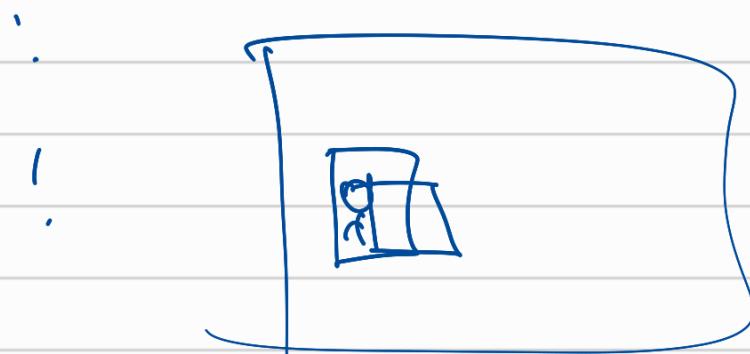
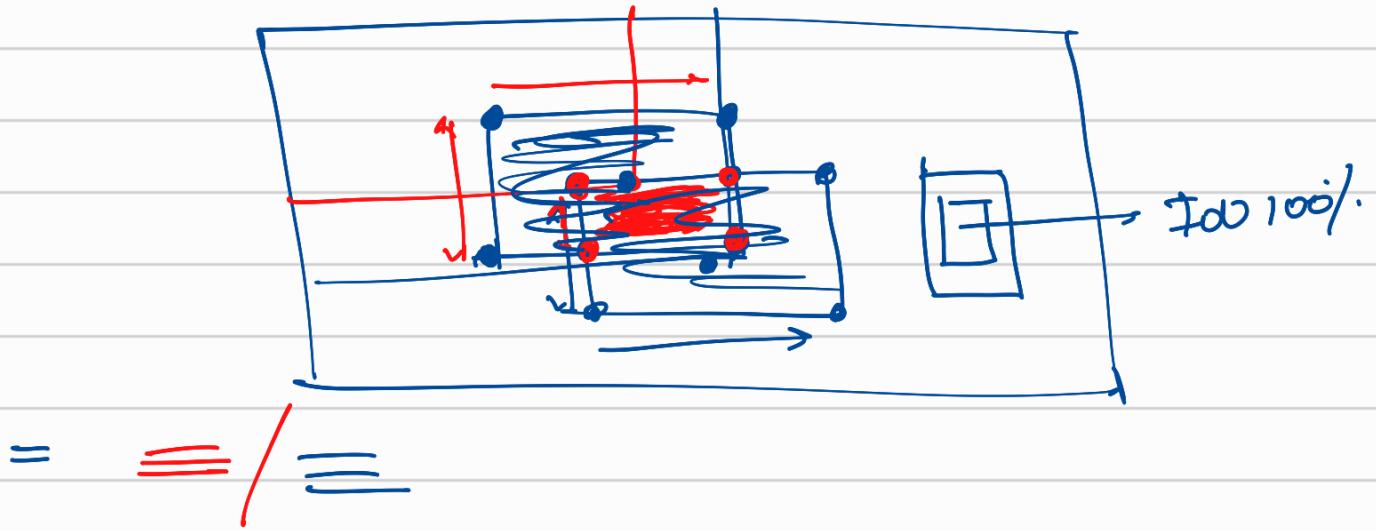
IoU → Intersection over Union



$$IoU = \frac{\text{Intersection Area}}{\text{Union Area}}$$

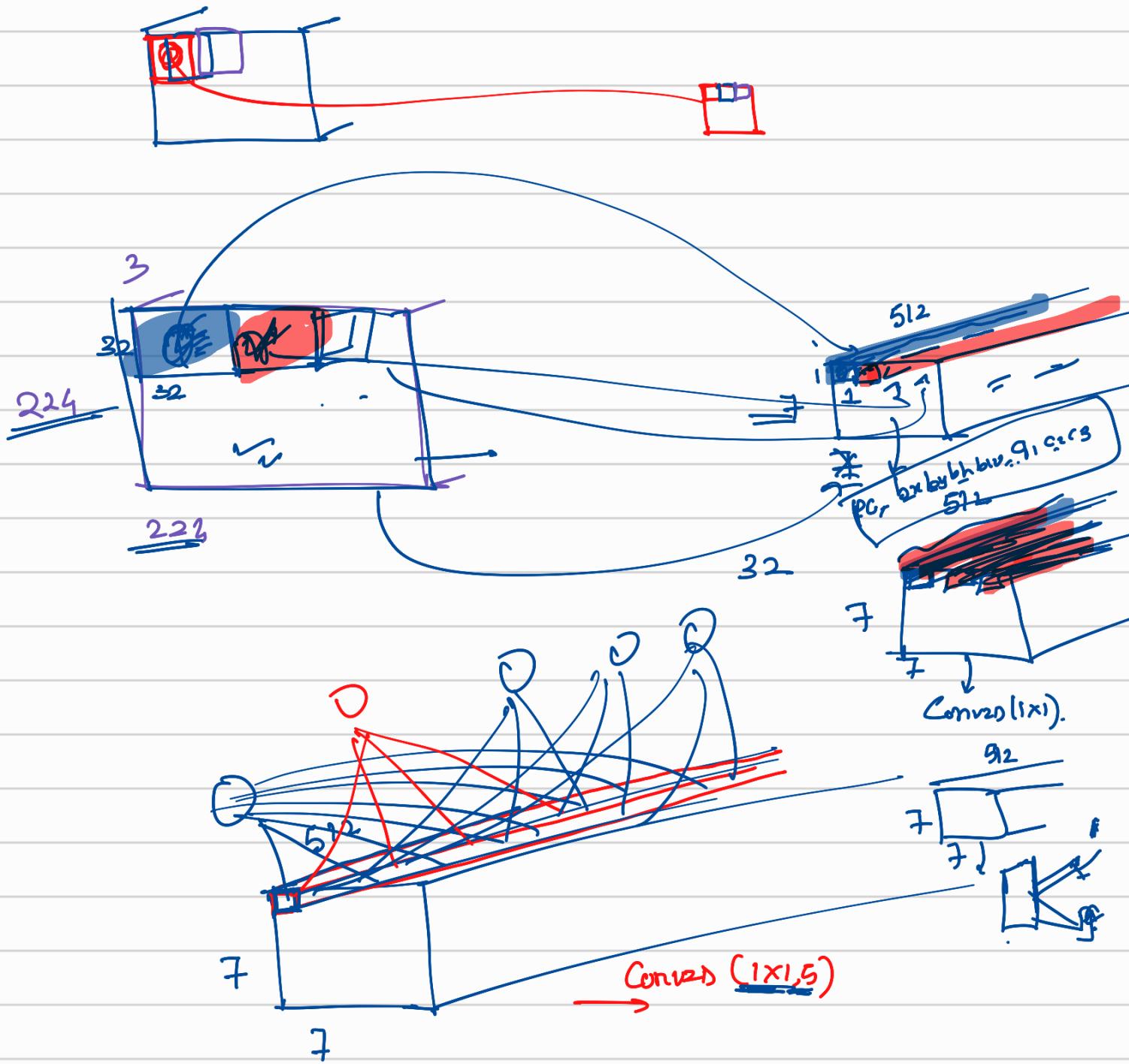


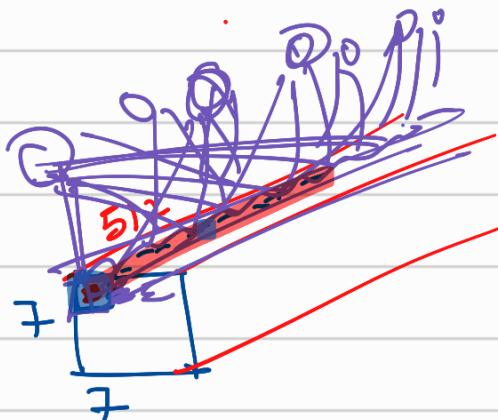
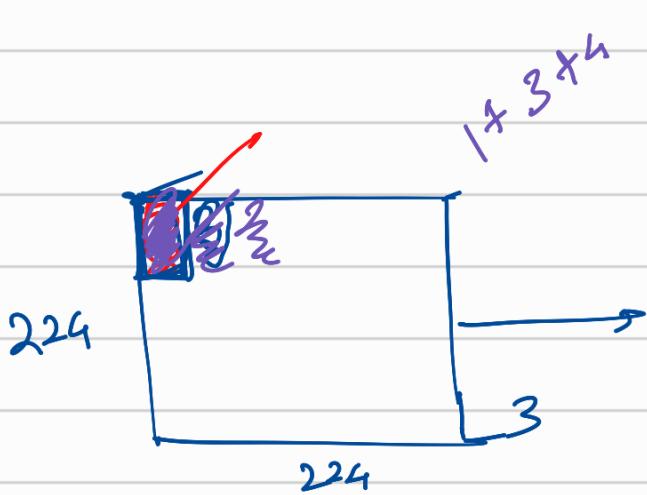
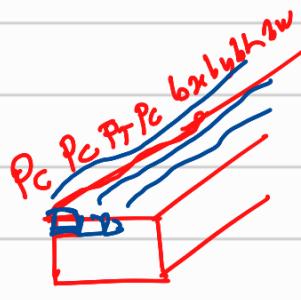
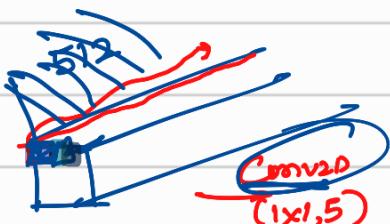
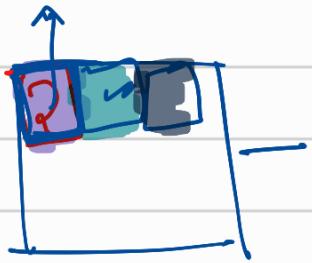
for Person class:
 tree
 Find the box with highest P.C.
 → all boxes having $T_{obj} > 0.7$
 with the above box, eliminate



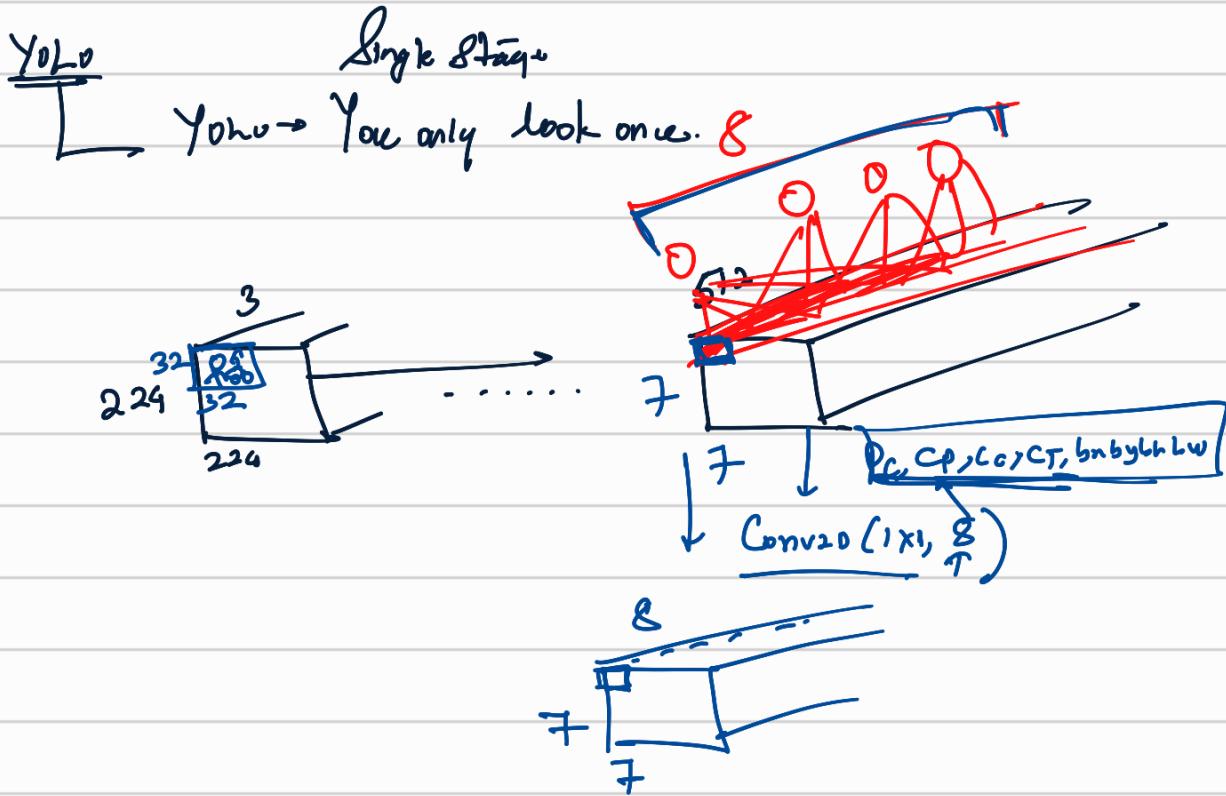
FASTER RCNN

Convolution implementation of Sliding window

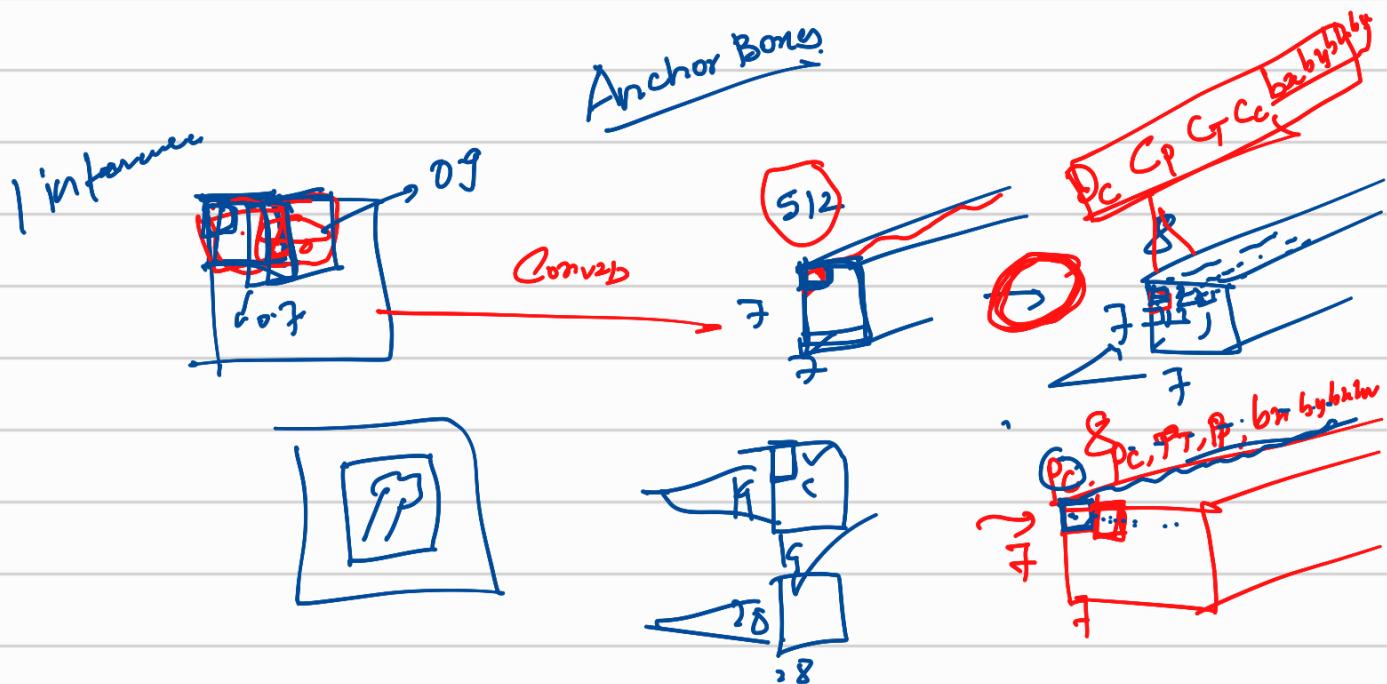


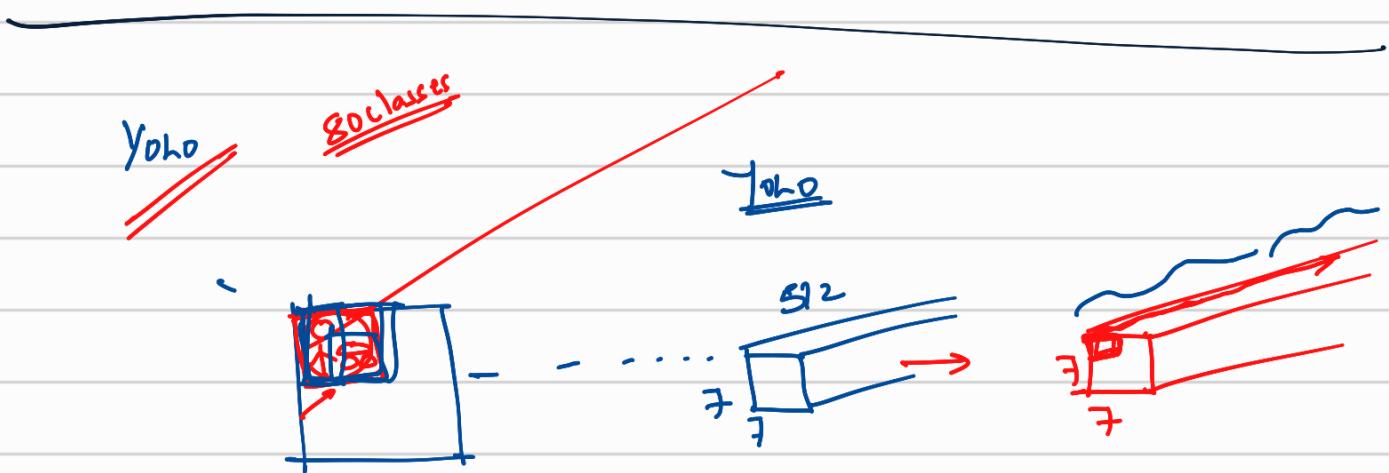
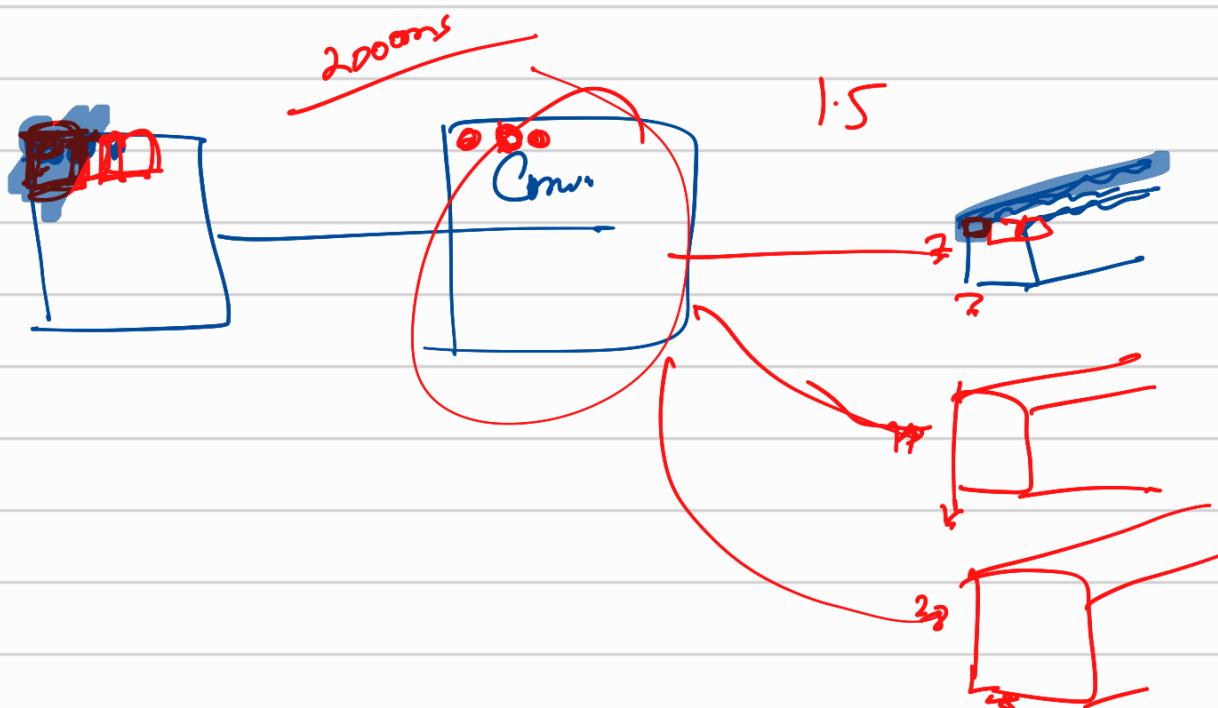
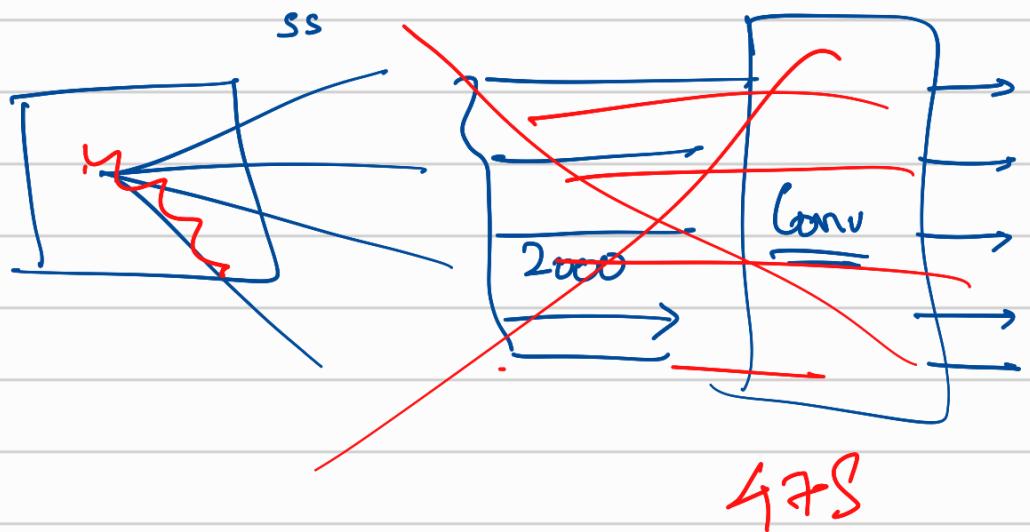


Faster RCNN



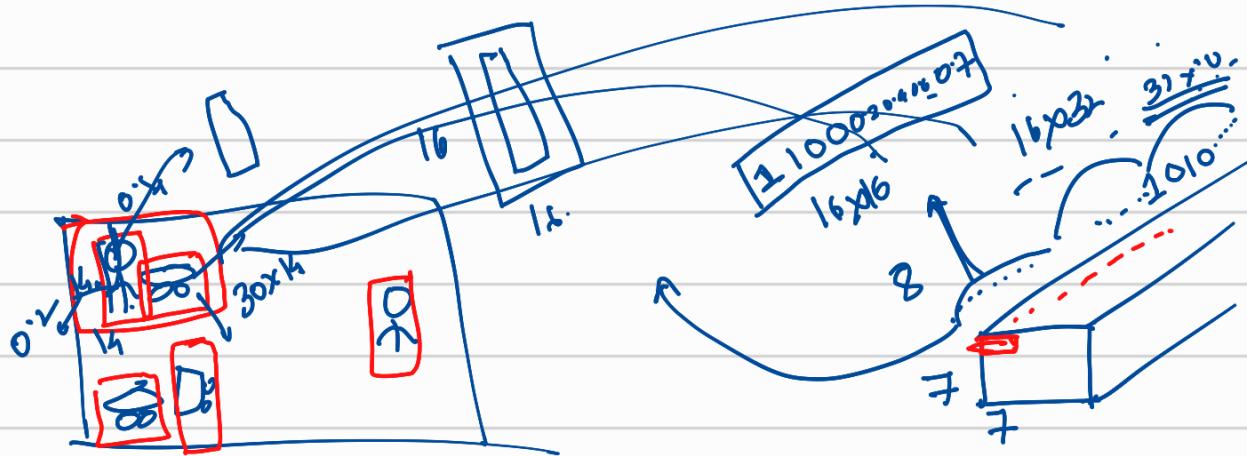
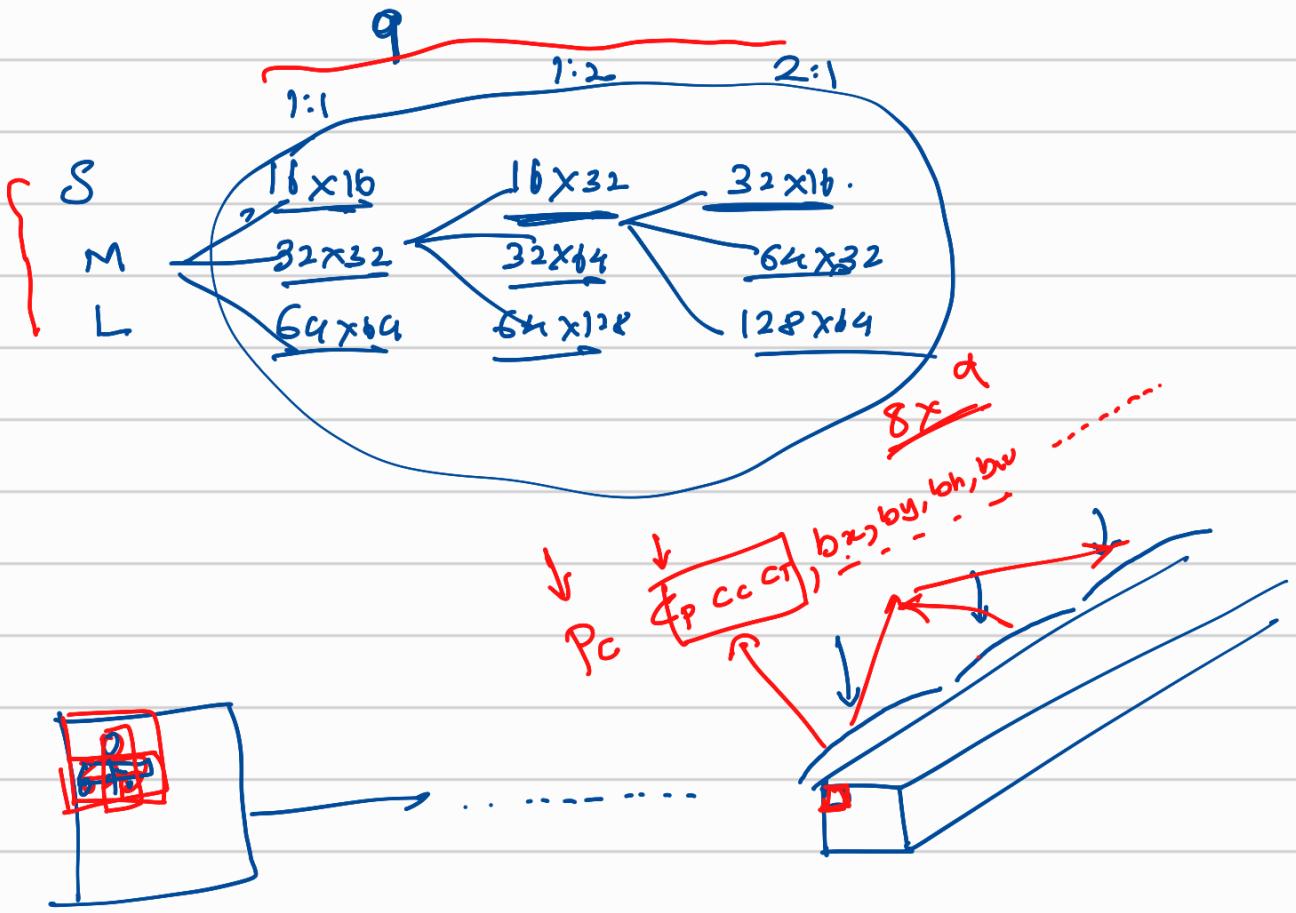
Anchor Boxes



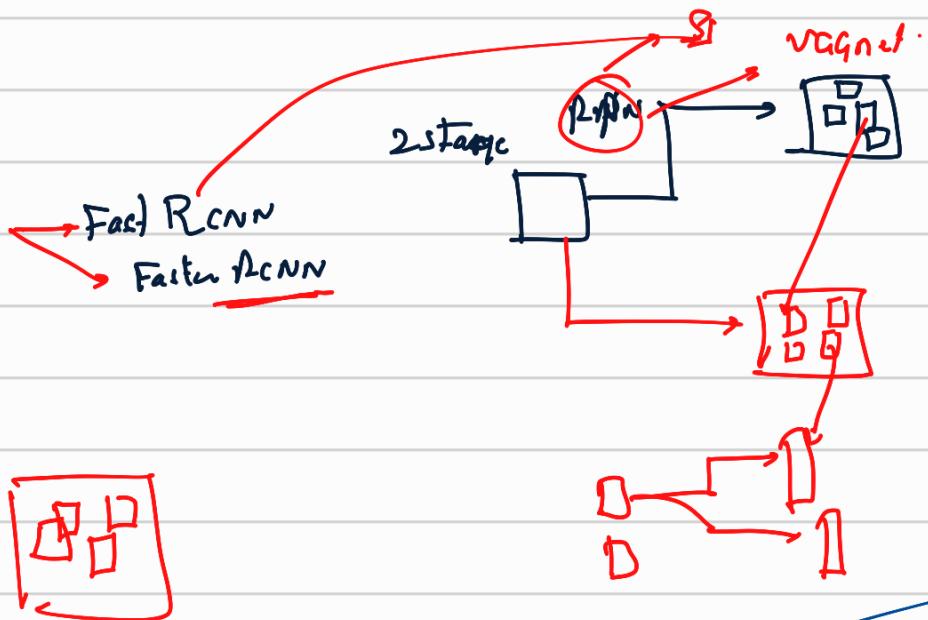


GT

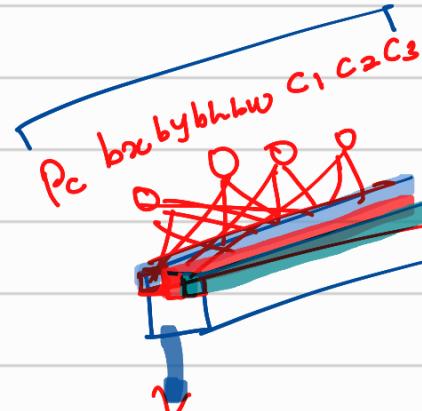
Cluster



YOLO

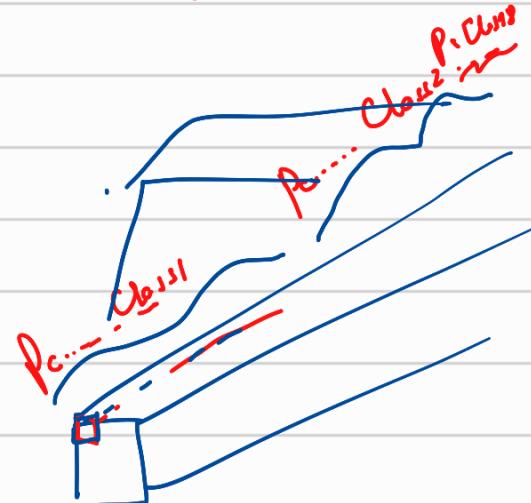
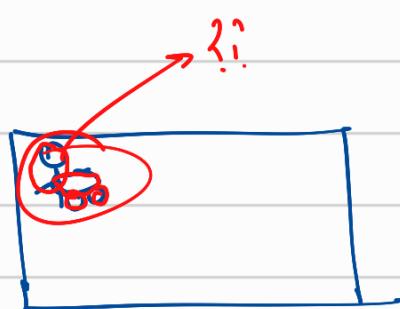


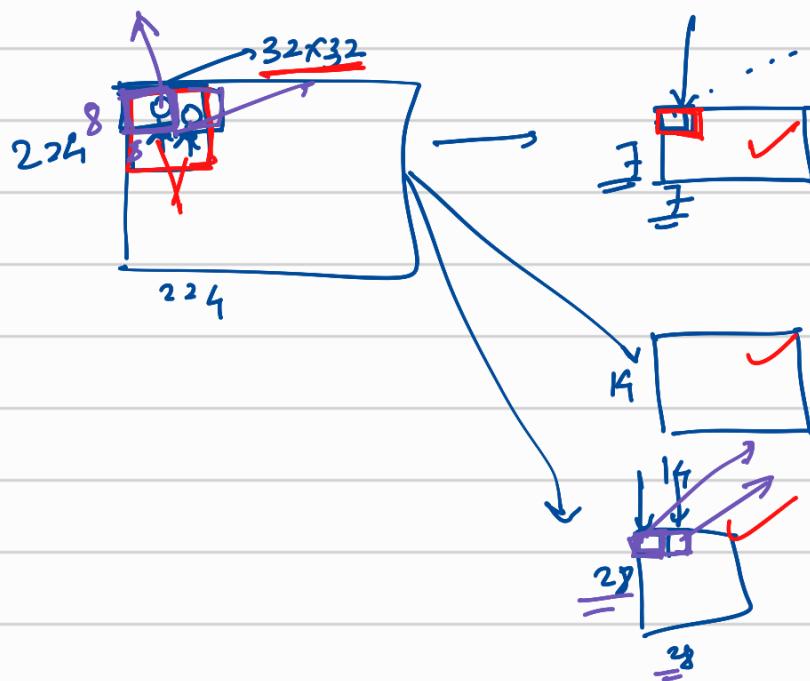
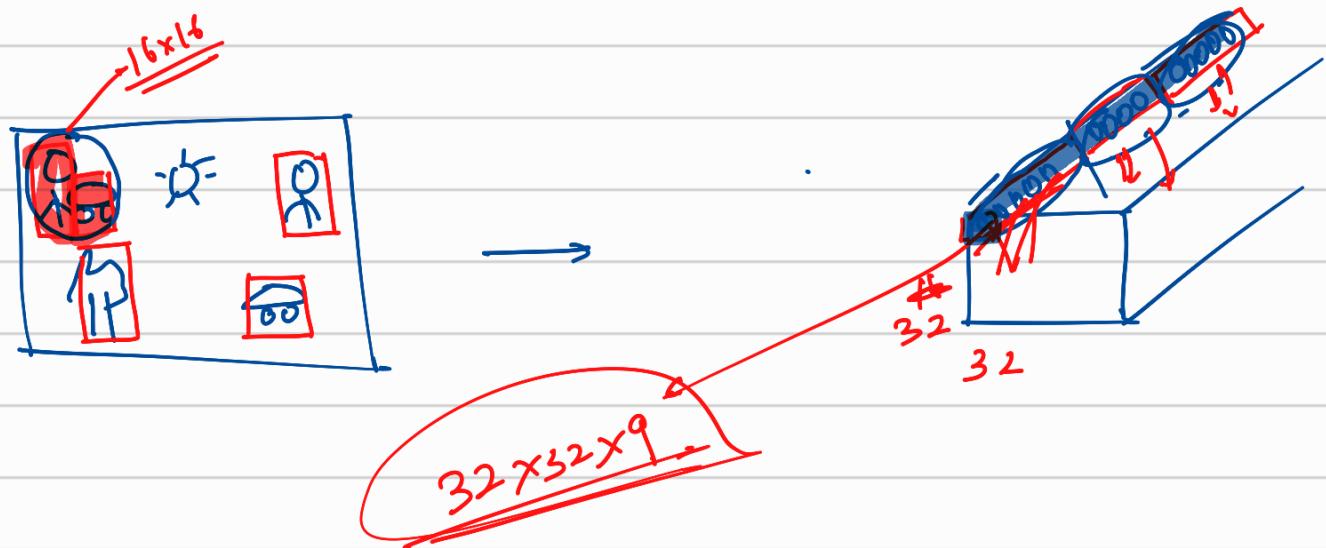
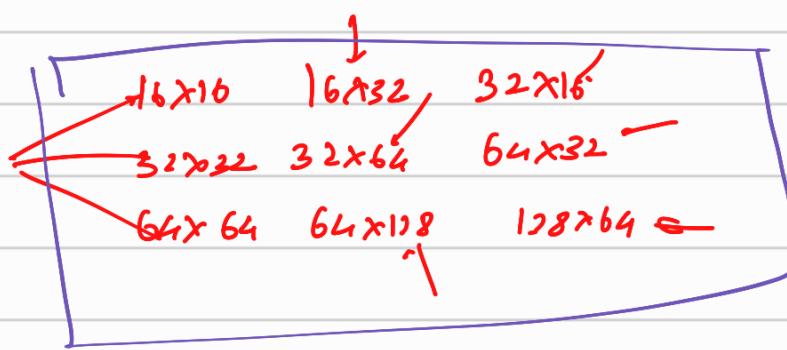
Yolo (Single Stage)

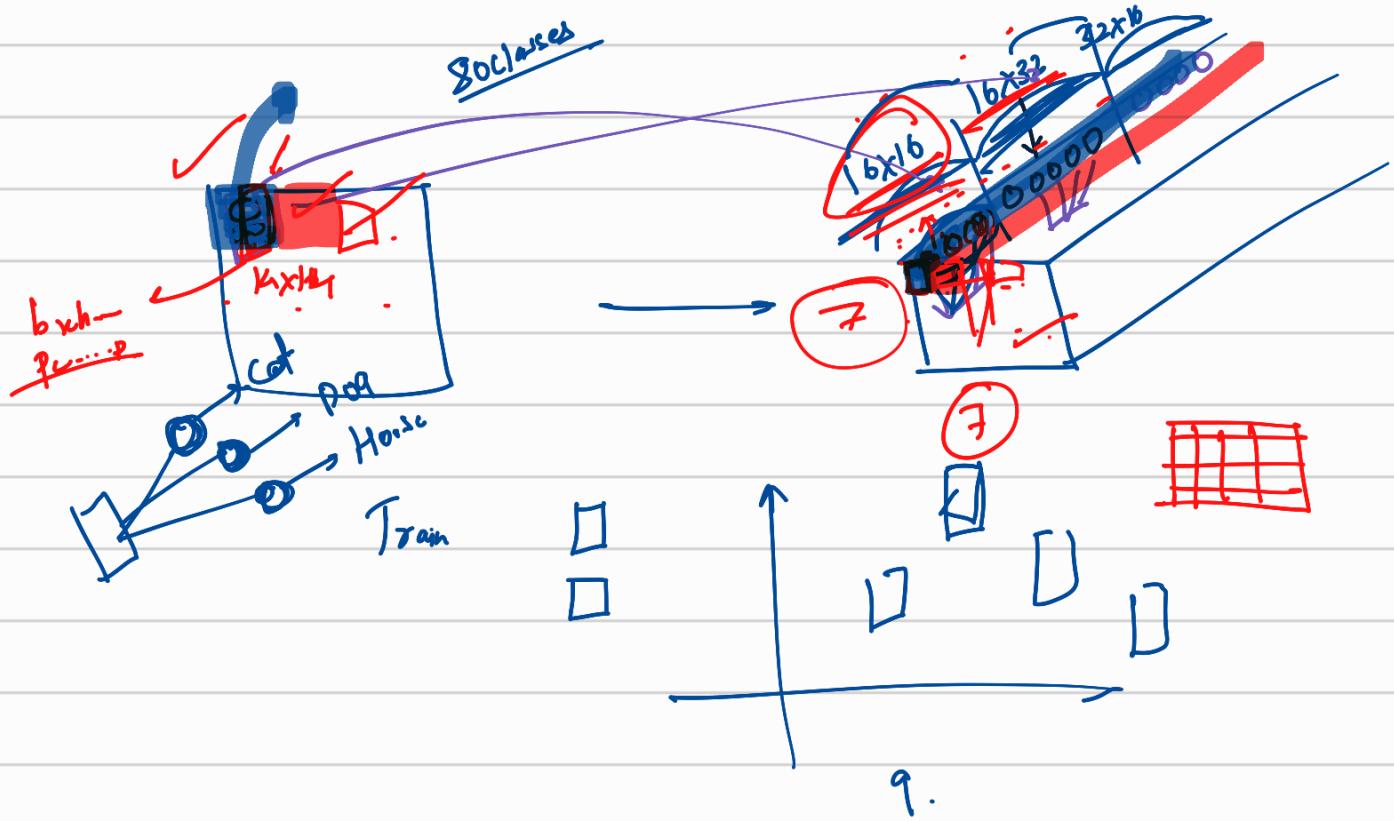


Conv2D (1x1)

Anchor Boxes



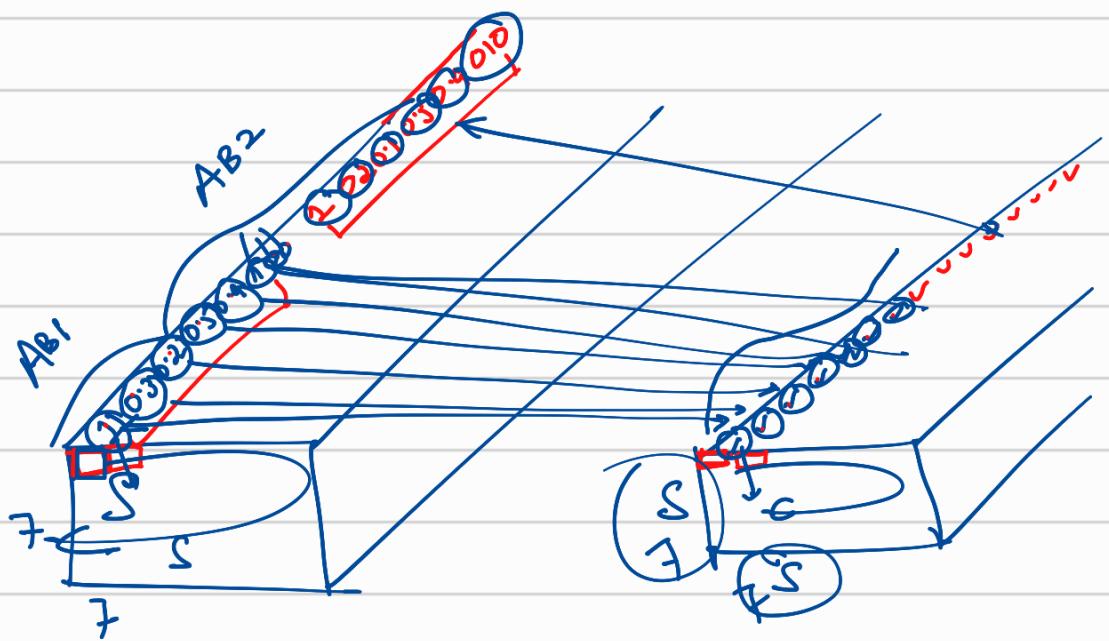


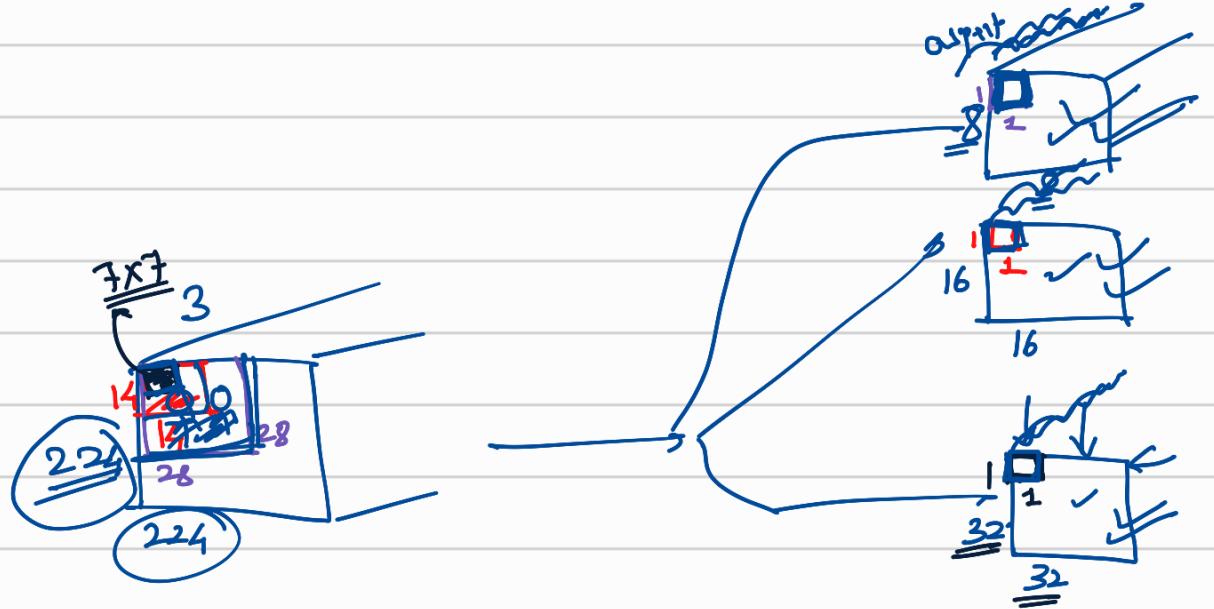


P_c b_2 b_1 b_h b_w $C_1 C_2 C_3 C_4$

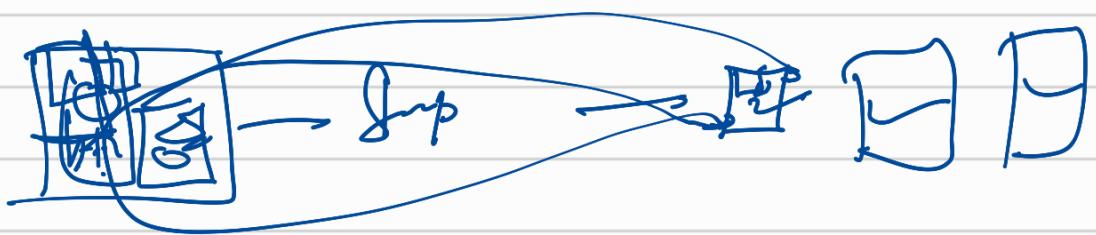
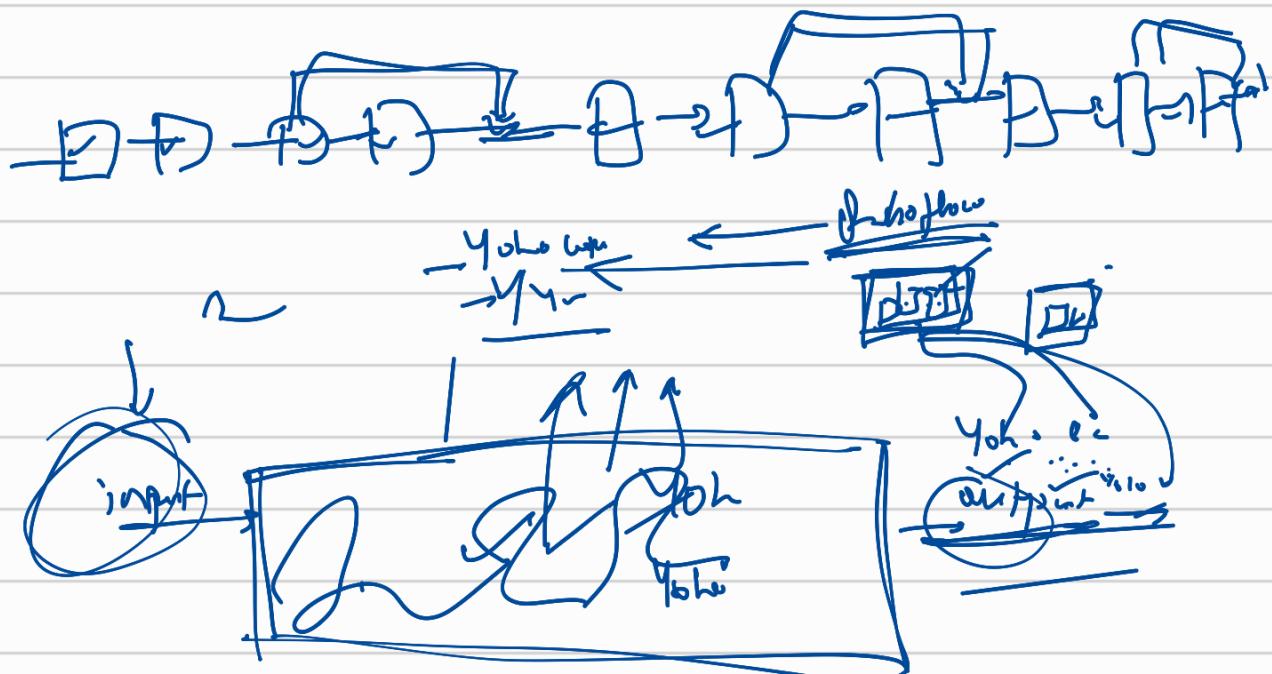
Break!

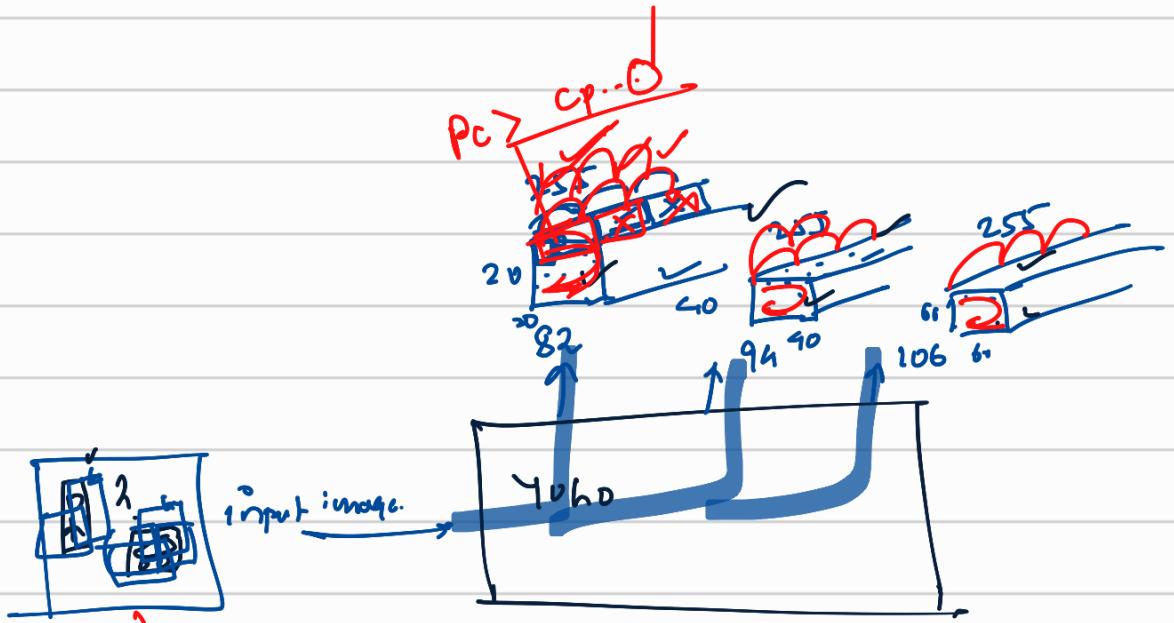
~~Yolo v3~~ Loss





Yoho





- ① $P_c > 0.6 \checkmark$
- ② np.argmax (Class) → Person
- ③ $C_p > 0.6 > 0.1 \checkmark$ AB
- ④ Bbox achieved

→ None

