

Neural Network

⁹ A system composed of many simple processing elements operating in parallel network
¹⁰ whose function is determined by neural structure connection strength and the processing performed
¹¹ at computing element.

¹²
Artificial Neuron: Info. processing system
that has certain characteristics in common
¹ with biological Neurons.

2 Steps

- 1) The processing element receives many signals.
 - 2) signals can be modified by weights at receiving synapse.
 - 3) The processing element sum the various unweighted inputs.
 - 4) Under appropriate condition , the neuron transmit the op signal.
 - 5) Output from one particular neuron may go in another neuron.

S	M	T	W	T	F	S
1	2	3	4	5		
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

JULY '21

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

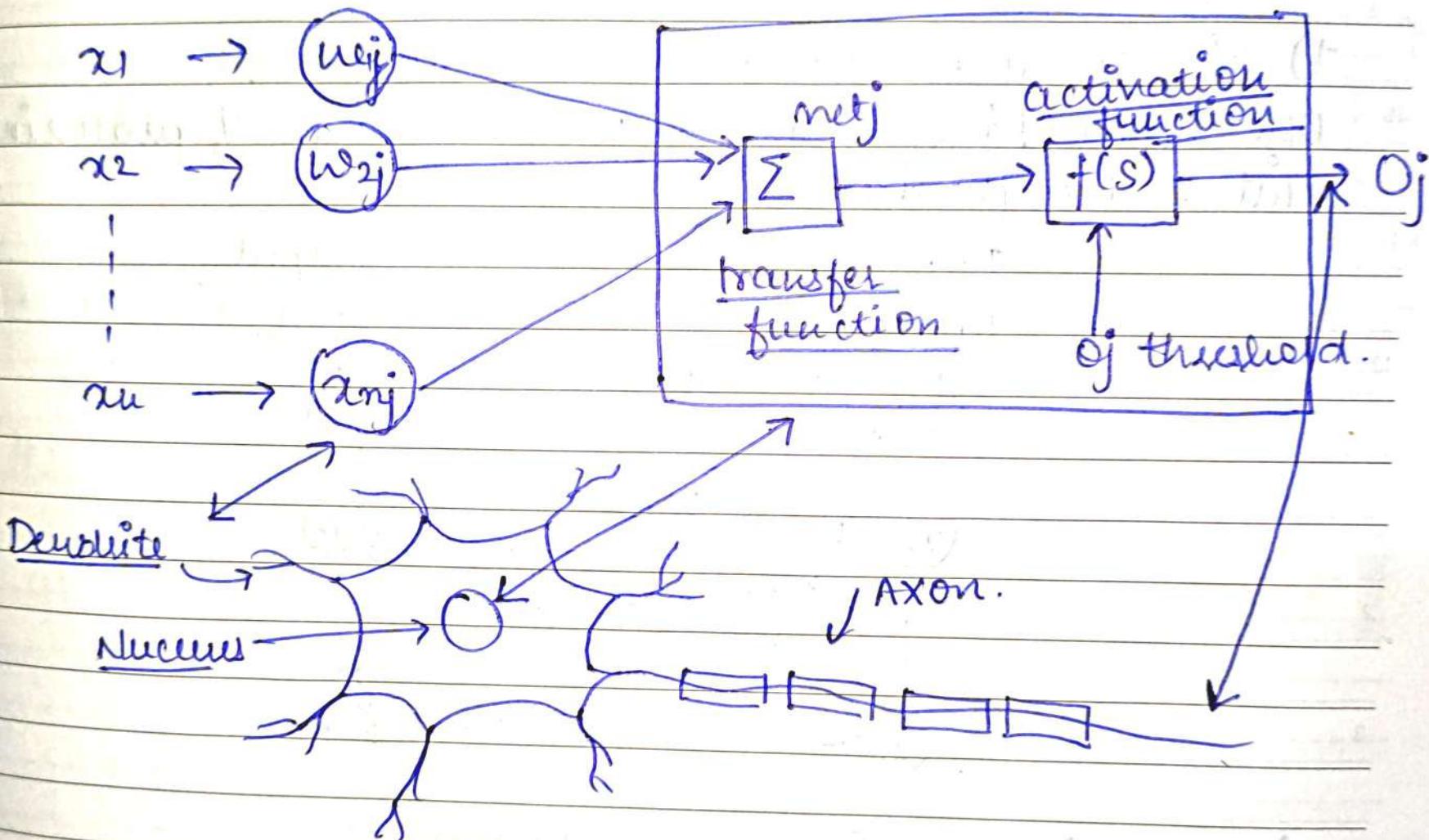
Artificial Neural Network (ANN)

Wk 20 | DAY 135-230

SATURDAY

15

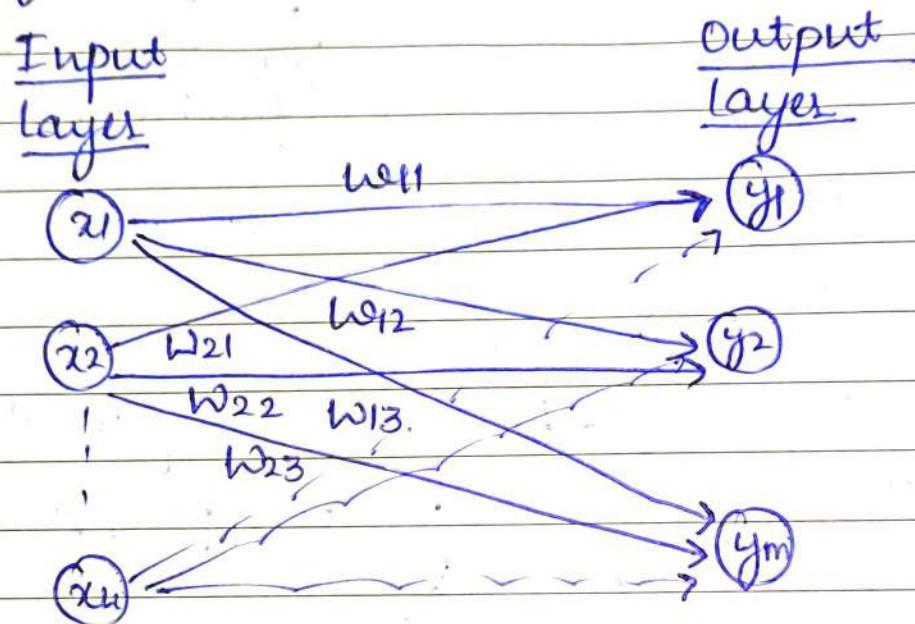
Inputs weights



Types of Neuron connections

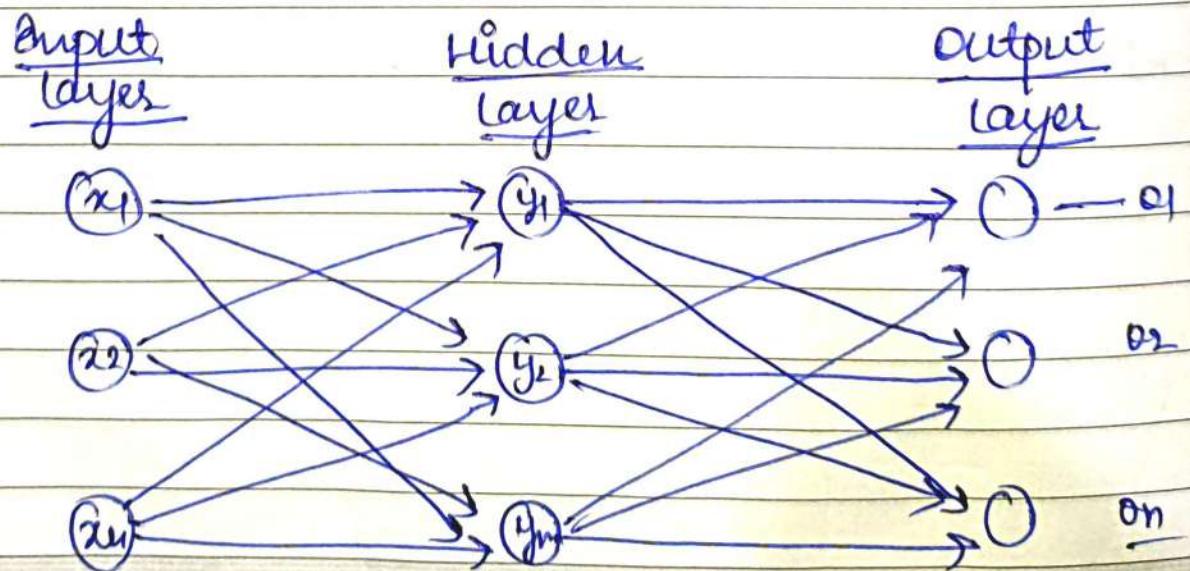
1) single layer feed-forward Network:

One input layer, one output layer of processing unit, No feedback.

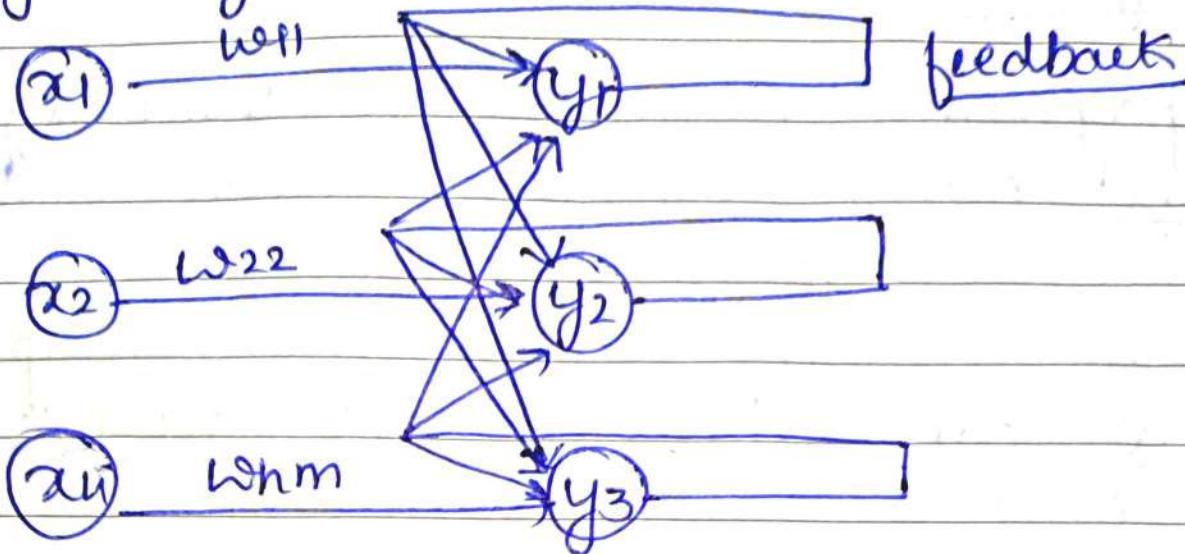


2) Multilayer Feed forward Network:

One Ip layer, One op layer and one or more hidden layer of processing unit.



3) Single layer Recurrent Network :



4) Multilayer Recurrent Network :

Backpropagation Algorithm

- ↳ Backpropagation Algo. learns the weights for multilayer neural network given with fixed set of units and interconnections.
- ↳ It employs gradient descent to attempt to minimize the square error between the network output values and the target values of those output.

In backpropagation algorithm, we consider network with multiple outputs

The sum of error of all the network o/p.

$$E(\vec{w}) = \frac{1}{2} \sum_{d \in D} \sum_{k \text{ outputs}} (t_{kd} - o_{kd})^2$$

outputs \rightarrow set of output units in network

$t_{kd} \leftarrow o_{kd} \rightarrow$ target output & network output associated with k th output unit.

$d \in D \rightarrow$ training example.

Notations :

1) each training example is a pair of (\vec{x}, \vec{t})

2) γ = learning rate

3) n_i = no. of ip units

n_o = no. of op units

Input value target value

n_h = no. of

hidden units

4) x_{ij} = ip from i to j

5) w_{ij} = net from i to j

1) Create a feed forward Network with n_i, n_o and n_h

2) Initialize all network net to some small random number.

3) until termination condition is met, Do.

for each (\vec{x}, t) in training example.

→ propagate network input fwd in network.

(i) Input \vec{x} , to network and competition
for every unit u in network.

→ propagate error backward through network

(ii) for each network output unit k , calculate error

$$\delta_k \leftarrow O_k(1-O_k)(t_k - O_k)$$

(iii) for each hidden unit h , calculate error

$$\delta_h \leftarrow O_h(1-O_h) \sum_{k \in \text{outputs}} \omega_{hk} \delta_k$$

(iv) update each network weight w_{ji}

$$\omega_{ji} \leftarrow \omega_{ji} + \Delta \omega_{ji}$$

$$\Delta \omega_{ji} = \eta \delta_j x_{ij}$$

Gradient Descent

Gradient descent is an optimized algorithm used to find the value of parameters of a function that minimizes the cost function.

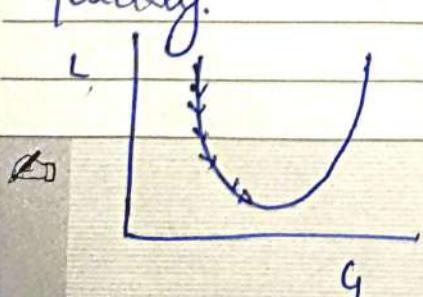
Depending upon the no. of training examples

Batch Gradient Descent

Parameters are updated after computing the gradient error w.r.t to entire training set.

$$E = \sum_{i=0}^n (t_i - o_i)^2$$

smooth & loss function converges quickly.

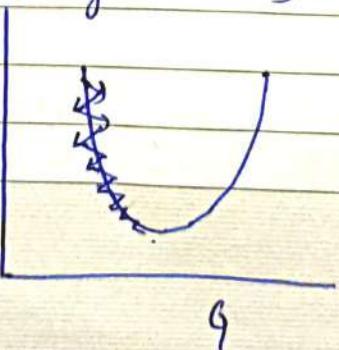


Stochastic Gradient Descent

Parameters are updated after computing the gradient of error w.r.t to single Training example

$$E = (t - o)^2$$

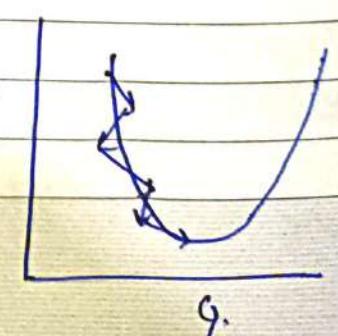
slow (large dataset)



Mini Batch Gradient Descent

Parameters are updated after computing the gradient of error w.r.t to subset of training set.

$$E = \sum_{i=0}^K (t_i - o_i)^2$$



S	M	T	W	T	F	S
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

S	M	T	W	T	F	S
						1 2 3
						4 5 6 7 8 9 10
						11 12 13 14 15 16 17
						18 19 20 21 22 23 24
						25 26 27 28 29 30 31

Wk 21 | DAY 142-223

SATURDAY

22

Stochastic Gradient DescentMini Batch GD

- 1) Take an exple.
- 2) feed it to NN
- 3) calculate its gradient
- 4) use gradient in step 3 to update the weights.
- 5) Repeat 1-4.

- 1) Pick a mini batch
- 2) Feed it to NN
- 3) calculate mean gradient
- 4) use the mean gradient to update weights
- 5) Repeat 1-4.

Difference between Batch & StochasticB.G.D.

- 1) Compute gradient of whole training set.
- 2) slow converge
- 3) Computationally exp.
- 4) for small dataset.
- 5) Deterministic in nature
- 6) Optimal sol.
- 7) Can't escape shallow local minima.
- 8) No. Random shuffling of pts are req.

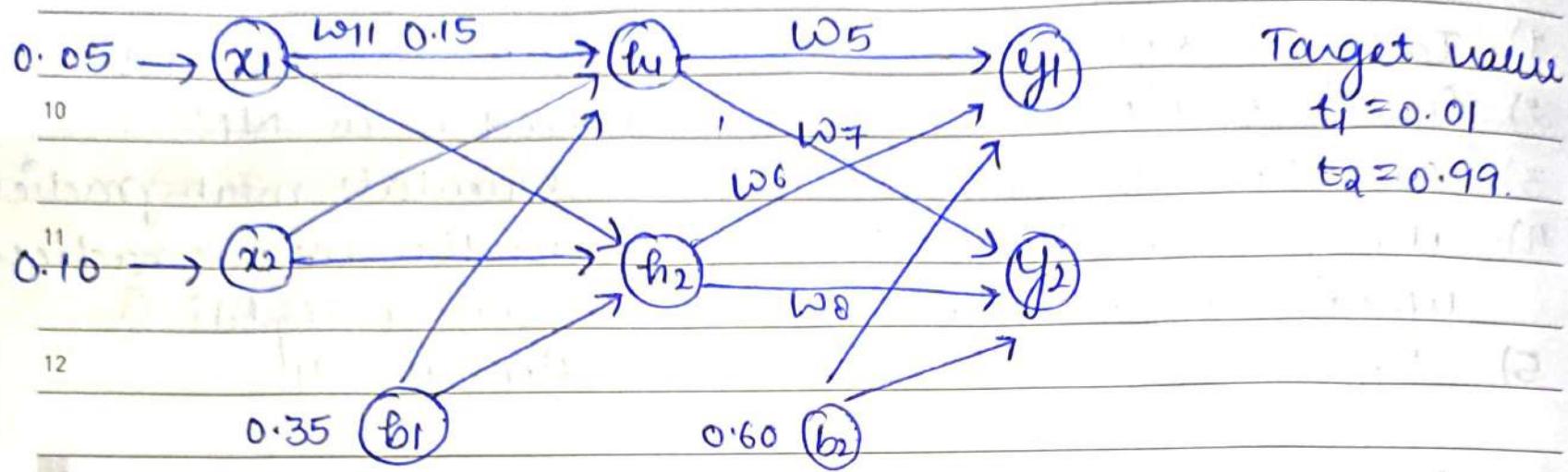
SGD

- 1) compute gradient using single training example.
- 2) faster converge.
- 3) Computationally less exp.
- 4) for large dataset
- 5) Stochastic in nature.
- 6) Not optimal.
- 7) can easily escape shallow local minima.
- 8) Shuffle dataset. (Random order preferred).

SUNDAY 23

(Noisy)

Backpropagation Numerical



$$x_1 = 0.05$$

$$w_{11} = 0.15$$

$$w_{21} = 0.20$$

$$x_2 = 0.10$$

$$w_{12} = 0.25$$

$$w_{22} = 0.30$$

Forward Pass

$$\begin{aligned} 3 \quad h_1 &= b_1 + x_1 w_{11} + x_2 w_{21} \\ &= 0.35 + 0.05 \times 0.10 + 0.20 \times 0.10 \end{aligned}$$

$$4 \quad h_1 = \underline{0.3775}$$

$$5 \quad h_{1f} = \frac{1}{1 + e^{h_1}} = \frac{1}{1 + \frac{1}{e^{0.3775}}} = \underline{0.59327}$$

$$6 \quad h_2 = b_1 + x_1 w_{12} + x_2 w_{22}$$

$$h_2 = 0.35 + 0.05 \times 0.25 + 0.10 \times 0.30$$

$$7 \quad h_2 = \underline{0.3925}$$

$$h_{2f} = \frac{1}{1 + \frac{1}{e^{0.3925}}} = \underline{0.596884}$$

$$y_1 = b_2 + h_1 \omega_5 + h_2 \times \omega_6$$

$$y_1 = 1.1059$$

$$y_2 = b_2 + h_1 \omega_7 + h_2 \omega_8$$

$$y_2 = 1.22492$$

$$y_{1f} = \frac{1}{1 + \frac{1}{e^{y_1}}} = \underline{0.751365}$$

$$y_{2f} = \frac{1}{1 + \frac{1}{e^{y_2}}} = \underline{0.77298}$$

$$E_{\text{total}} = \sum \frac{1}{2} (t - o)^2$$

$$= \frac{1}{2} (0.01 - 0.751365)^2 + \frac{1}{2} (0.99 - 0.77298)^2$$

$$\underline{E_{\text{total}} = 0.298371111}$$

Backward pass

$$\text{Error}_w = \frac{\partial E_{\text{total}}}{\partial w}$$

e.g. for ω_5

$$\frac{\partial E_{\text{total}}}{\partial \omega_5} = \frac{\partial E_{\text{total}}}{\partial y_{1f}} \times \frac{\partial y_{1f}}{\partial y_1} \times \frac{\partial y_1}{\partial \omega_5}$$

$$\frac{\partial (\frac{1}{2} (T_1 - y_{1f})^2 + \frac{1}{2} (T_2 - y_{2f})^2)}{\partial y_{1f}}$$

$$\cancel{\times} \frac{1}{2} (0.01 - 0.751365) \times -1 + 0$$

$$= \underline{0.74136507}$$

Gradient descent Algorithm (tE, η)

each TE is a pair of (\vec{x}, \vec{t})

\vec{x} = vector of ip values

\vec{t} = vector of target values.

1) Initialize each w_i to some random value.

2) until termination condition is met, do.

(i) Initialize each $\Delta w_i = 0$

(ii) for each (\vec{x}, \vec{t}) in TE

 ↳ Output \vec{x} & compute o

 ↳ for each linear unit weight w_i

$$\boxed{\Delta w_i \leftarrow \Delta w_i + \eta (t - o) x_i}$$

(iii) for each linear unit weight, do

$$\boxed{w_i \leftarrow w_i + \underline{\Delta w_i}}$$

S	M	T	W	F	S
1	2	3	4	5	
6	7	8	9	10	11
13	14	15	16	17	18
20	21	22	23	24	25
27	28	29	30		

S	M	T	W	T	F	S
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Loss function

Wk 22 | DAY 147-218

THURSDAY

27

loss function estimate how well particular algorithm models the provided data.

Regression Loss1) Mean square error

Avg. of squared diff b/w predicted & actual obs.

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

2) Mean absolute Error.

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$

3) Mean bias error.

$$MBE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)$$

4) Huber loss (smooth MAE)

$$\text{loss} = \begin{cases} \frac{1}{2} (x-y)^2 & ; |x-y| \leq \delta \\ \delta|x-y| - \frac{1}{2}\delta^2 & \text{else} \end{cases}$$

5) Mean squared logarithmic error

$$\sum_{i=1}^n \left(\log(y_i) - \log(\hat{y}_i) \right)^2$$

Classification Loss1) Cross entropy loss.

$$= -(y_i \log \hat{y}_i + (1-y_i) \log(1-\hat{y}_i))$$

2) Hinge loss

$$\text{loss} = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

$$\boxed{\sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)}$$

3) Kullback Leibler Divergence loss

$$D_{KL}(P||Q)$$

(discrete)

$$- \sum_x P(x) \log \frac{Q(x)}{P(x)} = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$

(cont)

$$- \int P(x) \log \frac{Q(x)}{P(x)} dx = \int P(x) \log \frac{P(x)}{Q(x)} dx$$

(cont)

$\frac{P(x)}{Q(x)}$ distribution

21' MAY

28

DAY 148-217 | Wk 22

FRIDAY

M	T	W	T	F	S	S
1	2	3	4			
5	7	8	9	10	11	
	14	15	16	17	18	
	21	22	23	24	25	
	26	27	28	29	30	

APRIL '21

M	T	W	T	F	S	S
31					1	2
	3	4	5	6	7	8
	10	11	12	13	14	15
	17	18	19	20	21	22
	24	25	26	27	28	29
					30	

MAY '21

Deep learning The subfield of artificial intelligence that focuses on creating large nn model that capable of making accurate data driven decisions.

Machine Learning

1) works on small amount of dataset for accuracy.

2) Dependent on low-end machine.

3) Divide the task into sub-task, solves them individually and finally combines them

4) Less time to train

5) More time to test

Deep learning

1) works on large amount of dataset.

2) Highly dependent on high-end machine

3) Solves problem end to end.

4) More time to train

5) Less time to test.

Applications

- ① speech Recognition
- ② Image Recognition
- ③ Self Driving cars & Robots.
- ④ Climate Prediction
- ⑤ Auto text gen., NLP.

JUNE '21
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30

JULY '21
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31

MAY '21

Wk 22 | DAY 149-216

SATURDAY

29

Regularization

A technique which makes slight modifications to the learning algo such that model generalizes better.

This improves model's performance on the unseen data.

Machine Learning

(Penalizes the coeff)

Deep learning

(Penalizes weight matrices of Node)

Regularization Techniques

① L1 & L2 Regularization

$$\text{cost fn} = \text{loss} + RT$$

$$L_1 = \text{loss} + \frac{1}{2m} \times \sum \|w\|_1$$

$$L_2 = \text{loss} + \frac{1}{2m} \sum \|w\|_2^2$$

λ = regularization parameter.

③ Data Augmentation

Increase the size of training Data

e.g. Aug \rightarrow flip, rotate, scale

② Dropout : At every iteration, Randomly select some nodes & remove them along with their incoming and outgoing connections.

Ensemble learning Technique

④ Early stopping

Cross-Validation strategy
Validation set
→ performance start degrading
→ stop training

NUR

Parameters in BPN

1) No. of hidden layer nodes:

Guiding criterion \rightarrow 1st & 3rd nodes are min.
11 layers

2) Momentum coefficient

To reduce training time

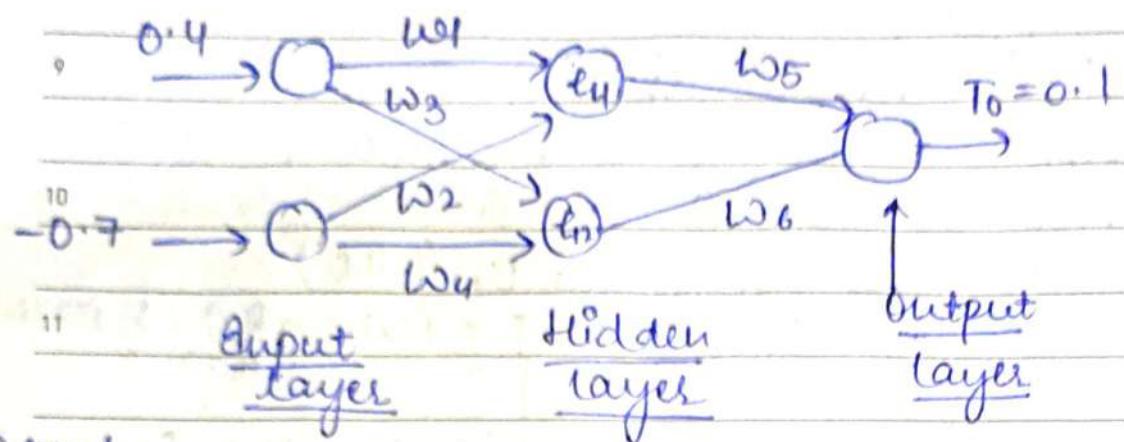
$$[\Delta w]^{n+1} = -\eta \frac{\partial E}{\partial w} + \alpha [\Delta w]^n$$

2 overcomes local minima.

3) Sigmoidal Gain λ :

When weights become large & force neurons to operate in a region where sigmoidal funⁿ falls flat, then adjust the λ . (decrease)

4) Local Minima: Change all weights randomly.



Step 1

$$w_1 = 0.1$$

$$w_3 = 0.4$$

$$w_5 = 0.2$$

$$w_2 = 0.2$$

$$w_4 = 0.2$$

$$w_6 = -0.5$$

$$\text{Step 2 } t_{i1} = b_1 * 1 + w_1 i_1 + i_2 w_2$$

$$= 0 + 0.4 \times 0.1 + (0.2) \times (-0.7)$$

$$t_{i1} = 0.18$$

$$t_{i2} = 0.4 \times 0.4 + -0.7 \times 0.2$$

$$t_{i2} = 0.02$$

5

$$\text{Step 3 } O_{H_1} = \frac{1}{1 + e^{-t_{i1}}} = 0.5448$$

$$O_{H_2} = \frac{1}{1 + e^{-t_{i2}}} = 0.5050$$

Step 4

$$\text{Net } O_1 = \sum w_i x_i$$

$$= 0.5448 \times 0.2 + 0.5050 \times (-0.5)$$

$$= -0.14354$$

Step 5

$$O_{O_1} = \frac{1}{1 + e^{-\text{net } O_1}} = 0.4642$$

Step 6 $E_{\text{total}} = \frac{1}{2} (t_{01} - o_{01})^2$

$$= \frac{1}{2} (0.1 - 0.4642)^2$$

$$= 0.0663$$

Step 7:

$$\frac{\partial E}{\partial w_5} = \frac{\partial E}{\partial o_{01}} \times \frac{\partial o_{01}}{\partial n_{01}} \times \frac{\partial n_{01}}{\partial w_5}$$

$$\frac{\partial E}{\partial o_{01}} = -(t_{01} - o_{01}) = -(0.1 - 0.4642) = 0.3642$$

$$\frac{\partial o_{01}}{\partial n_{01}} = o_{01}(1-o_{01}) = 0.4642(1-0.4642)$$
$$\frac{\partial n_{01}}{\partial w_5} = 0.2487$$

$$\frac{\partial n_{01}}{\partial w_5} = 0.5448$$

$$\frac{\partial E}{\partial w_5} = 0.3642 \times 0.2487 \times 0.5448$$
$$= 0.0493$$

Step 8: $w_5^+ = w_5 - \eta \frac{\partial E}{\partial w_5} = 0.2 - 0.6 \times 0.0493$

$$\boxed{w_5^+ = 0.17042}$$

Step 9 $\frac{\partial E}{\partial w_6} = \frac{\partial E}{\partial o_{01}} \times \frac{\partial o_{01}}{\partial n_{01}} \times \frac{\partial n_{01}}{\partial w_6}$

$$= 0.3642 \times 0.2487 \times 0.1505$$

$$= 0.0457$$

Step 10 $w_6^+ = w_6 - \eta \frac{\partial E}{\partial w_6} = -0.52742$

Step: 11 $\frac{\partial E}{\partial \omega_1} = \frac{\partial E}{\partial o_{01}} \times \frac{\partial o_{01}}{\partial \text{net}_{01}} \times \frac{\partial \text{net}_{01}}{\partial \omega_1}$

10 $\frac{\partial E}{\partial \omega_1} = \frac{\partial E}{\partial o_{01}} \times \frac{\partial o_{01}}{\partial \text{net}_{01}} \times \frac{\partial \text{net}_{01}}{\partial o_{H1}} \times \frac{\partial o_{H1}}{\partial \text{net}_{H1}} \times \frac{\partial \text{net}_{H1}}{\partial \omega_1}$

11

$$= 0.3642 \times 0.2487 \times \omega_5 \times o_{H1}(1-o_{H1}) \times i_1$$

12

$$= 0.001796$$

Step 12

$$\omega_1^+ = \omega_1 - \eta \frac{\partial E}{\partial \omega_1}$$

2

$$= 0.1 - 0.6 \times 0.001796$$

3

$$\boxed{\omega_1^+ = 0.8189}$$

Step: 13

5 $\frac{\partial E}{\partial \omega_2} = \frac{\partial E}{\partial o_{01}} \times \frac{\partial o_{01}}{\partial \text{net}_{01}} \times \frac{\partial \text{net}_{01}}{\partial o_{H1}} \times \frac{\partial o_{H1}}{\partial \text{net}_{H1}} \times \frac{\partial \text{net}_{H1}}{\partial \omega_2}$

$$= -0.0031448$$

6 $\omega_2^+ = \omega_2 - \eta \frac{\partial E}{\partial \omega_2}$

7 $\boxed{\omega_2^+ = -0.1981}$

Step: 14

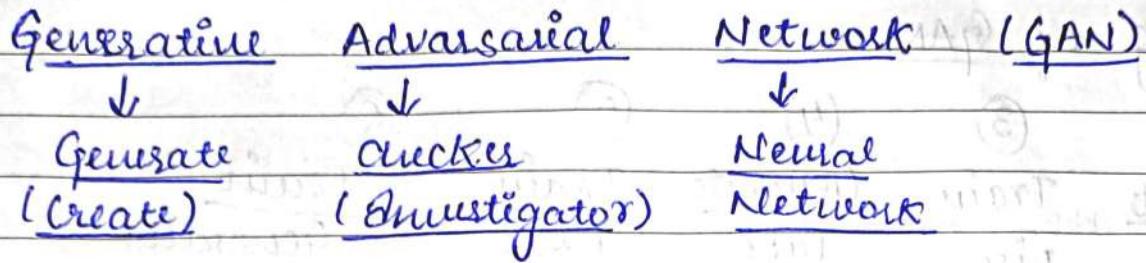
$$\frac{\partial E}{\partial \omega_3} = -0.004528$$

$$\omega_3^+ = \omega_3 - \eta \frac{\partial E}{\partial \omega_3} = 0.4027.$$

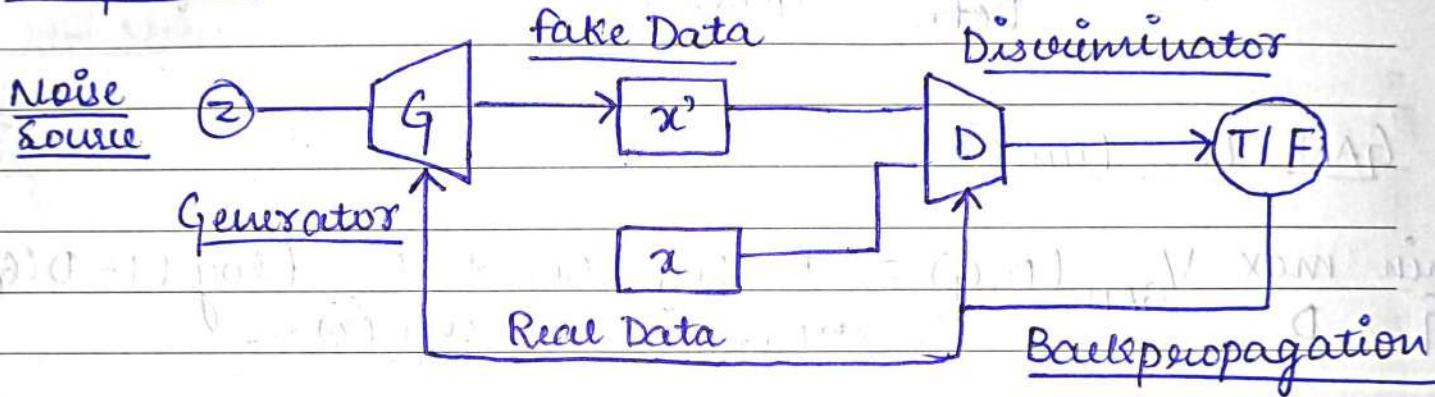
Step: 15

$$\frac{\partial E}{\partial \omega_4} = 0.00792$$

$$\boxed{\omega_4^+ = 0.1952}$$



Components

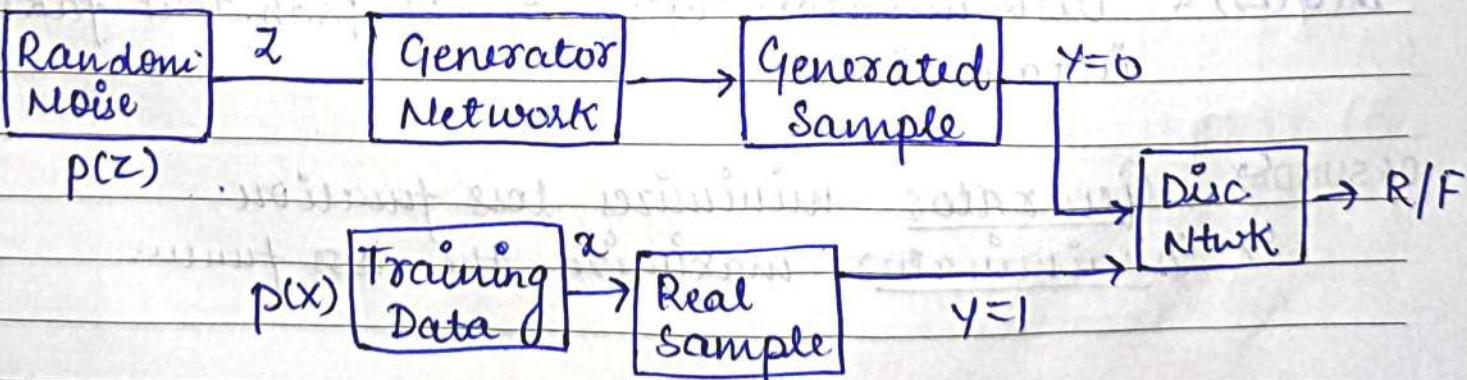


Generator:

- works on unsupervised learning approach
- It will generate fake data

Discriminator:

- works on supervised learning approach
- Discriminator does the classification



Training of GAN

- ① Define a problem
- ② Select Arch. of GAN
- ③ Train Disc.
- ④ Generate fake I/P from Generator.
- ⑤ Train Disc. on Fake Data
- ⑥ Train Generator with the O/P of Disc.

GAN Loss function

$$\min_G \max_D V_{GAN}(D, G) = E_{x \sim p_{data}(x)} \log D(x) + E_{z \sim p_z(z)} \log (1 - D(G(z)))$$

3 Formula is derived from cross-entropy b/w real and generated distributed.

4 $D(x)$ = Discriminator's estimate of prob. that real data instance is real.

5 Ex = expected value over all real data instances.

6 $G(z)$ = Generator's op when given noise z .

7 $D(G(z))$ = Discriminator's estimate of prob. that fake instance is real.

8 06 SUNDAY Generator minimizes loss function

↳ Discriminator maximizes the loss func.

Advantages

- 9 1) Better modeling of Data Distribution.
- 10 2) GANs can train any kind of Generator Network.
- 11 3) No Need to use Markov chain repeated sample.

Disadvantages

- 1) Hard to train, unstable
- 2) Good sync req. b/w Disc & Generator.
- 3) Mode collapse issues

08

DAY 159-206 | WK 24

TUESDAY

M	T	W	T	F	S	S	M	T	W	T	F	S	S
31				1			1	2	3	4	5	6	
3	4	5	6	7	8	9	7	8	9	10	11	12	13
10	11	12	13	14	15	N	14	15	16	17	18	19	20
17	18	19	20	21	22	23	21	22	23	24	25	26	27
24	25	26	27	28	29	30	28	29	30				

JUNE '21

Semi-supervised Learning

- 1) The Algo is trained upon a combination
10 of labelled and unlabelled data.

11 • Small Labelled
Data

Large amount of
Unlabelled Data

- 12 2) Semi-supervised Algo assumptions:

1 3) Continuity assumptions: The points which are
2 closer to each other are more likely to have the
same output label.

4) Cluster Assumptions: Data can be divided into
discrete clusters & points in same cluster are more
likely to share an output label.

5) Manifold Assumptions: Data lies approximately
on a manifold of much lower dimension than the
input space.

- 6) 3) Applications:

- 7) a) Speech Analysis
b) Internet Content classification
c) Protein seq. classification.

JULY '21
S M T W T F S
1 2 3
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31

AUGUST '21
S M T W T F S
1 2 3 4 5 6 7
8 9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31

JUNE '21

Wk 24 | DAY 160-205

WEDNESDAY

09

Principal Component Analysis

- 1) PCA is used to reduce dimensions of data without much loss of information.
- 2) PCA is an orthogonal linear transformation that transfers data to new coordinate system such that greatest variance by any projection of data comes to lie on first coordinate.
- 3) Second greatest variance lies on second coordinate & so on.

Algorithms:

- Step:1 Get Data
- Step:2 Subtract the means
- Step:3 Calculate covariance Matrix.
- Step:4 Calculate EV & Eigen values of covariance Matrix
- Step:5 Choosing components & forming a feature vector.
- Step:6 Diving the new Dataset.

Advantages

- 1) Lack of Redundancy
- 2) Reduced complexity in img grouping using PCA.
- 3) Reduction of Noise since max. variance.

Disadvantages

- 1) Covariance matrix is difficult to evaluate.
- 2) Simplest variance could not be captured by PCA.

$$21^{\text{ST}} \text{ JUNE} S = \sum_{k=1}^n (x_k - \bar{x})(x_k - \bar{x})^T$$

10

DAY 161-204 | Wk 24
THURSDAY

M	T	W	T	F	S	S
3	4	5	6	7	8	9
12	13	14	15	16	17	18
17	18	19	20	21	22	23
24	25	26	27	28	29	30
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

Linear Discriminant Analysis

1 LDA is used for data classification & dimensionality reduction

2 It maximizes the ratio b/w class variance to within the class variance in any particular dataset

Algorithm

1 Step:1 Compute within-class & between class scatter Matrix

Step:2 Compute eigenvalues & EV for scatter Matrix.

2 Step:3 Sort the Eigen values & selecting top k

Step:4 Create a new matrix

3 eigen vectors \rightarrow k eigenvalues.

Step:5 Obtain new features using dot product

4 of data & matrix in step 4

Advantages

1) Simple & fast,

2) Portable algo, better than some logistic reg.

Disadvantages

1) Not scalable.

2) Inefficient to update the Model.

3) Require Normal distribution assumption on features.

Application:

1) \hookrightarrow face Recognition \hookrightarrow Medical
 \hookrightarrow customer Identification.

PCA

1) Unsupervised learning
Algorithm

2) Feature classification

3) Ignores class labels &
focuses on finding the
PC that maximizes the
variance in given data.

4)

5)

PCA: component axes that
maximize the variance.

LDA

1) supervised learning
Algorithm.

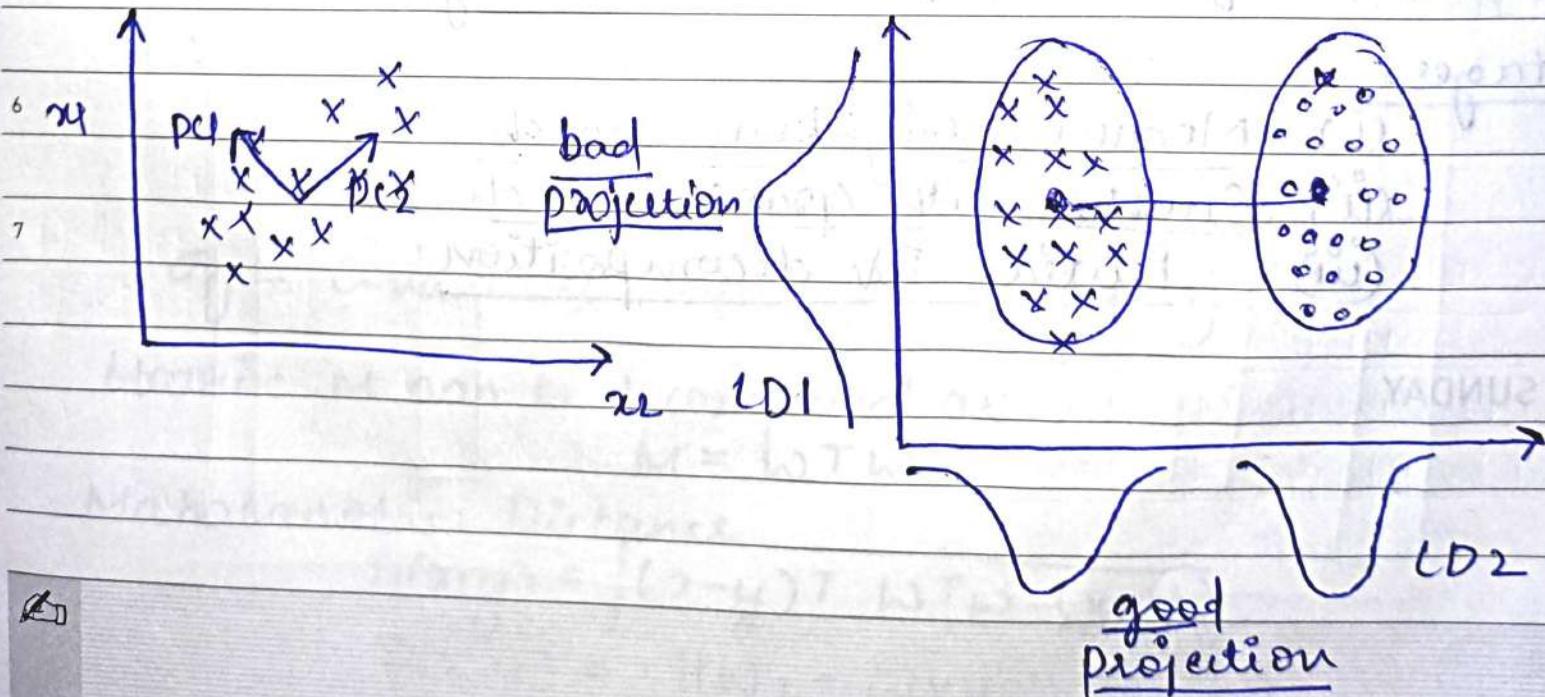
2) Data Classification.

3) It finds linear discriminant
that represents those axes
which maximizes separation
b/w diff. classes

4) Perform better multi-class
classification.

5)

LDA: maximizing the
component axes for class-sep.



Manifold learning

179

- For Non-linear dimensionality Reduction.
- It is based on the idea that idea that the dimensionality of many datasets are only artificially high.

- ① Isomap Algorithm
- ② Locally linear embedding
- ③ Hessian Eigen Mapping
- ④ Spectral embedding
- ⑤ Local Tangent Space Alignment
- ⑥ Multi-dimensional scaling
- ⑦ t-distributed stochastic Neighbour embedding

Isometric Mapping Algorithm

Extension of Multi-dimensional scaling.

Stages:

- (i) Nearest Neighbour search.
- (ii) Shortest path graph search
- (iii) Partial EV decomposition.

Metric learning

Euclidean distances are less meaningful in higher dimensions. They fail to capture the non-linearity in data because:

- they represent isotropic distance matrix
- these distances don't capture class structure.

Mahalanobis Distance Matrix:

In a 'd' dimensional space: $x, y \in \mathbb{R}^d$

$$D(x, y) = \sqrt{(x-y)^T M (x-y)}$$

M = Inverse of covariance matrix

Covariance Matrix in 'd' dimensional space:

$$M = \Sigma^{-1} = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \dots & \sigma_{1d} \\ \sigma_{21} & \sigma_{22} & \dots & \sigma_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{d1} & \sigma_{d2} & \dots & \sigma_{dd} \end{bmatrix}^{-1}$$

dxd.

σ_{ij} = covariance b/w variables i, j

Matrix M can be decomposed as

$$M = WTW$$

Mahalanobis Distance

$$D(x, y) = \sqrt{(x-y)^T W^T W (x-y)}$$

$$D(x, y) = \|Wx - Wy\|_2$$

21' JUNE

15

DAY 166-199 | Wk 25
TUESDAY

M T W T F S S
31 unsupervised
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30

M T W T F S S
1 2 3 4 5 6
7 8 9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30

Autoencoders

- Autoencoders are specific type of feed forward Neural Networks where the input is same as the output.
- They compress the input into lower dimension code and then reconstruct the output from this representation.

Autoencoder Components.

- Encoder compresses the I/P & produce the code
- Decoder then reconstruct the input using this code.

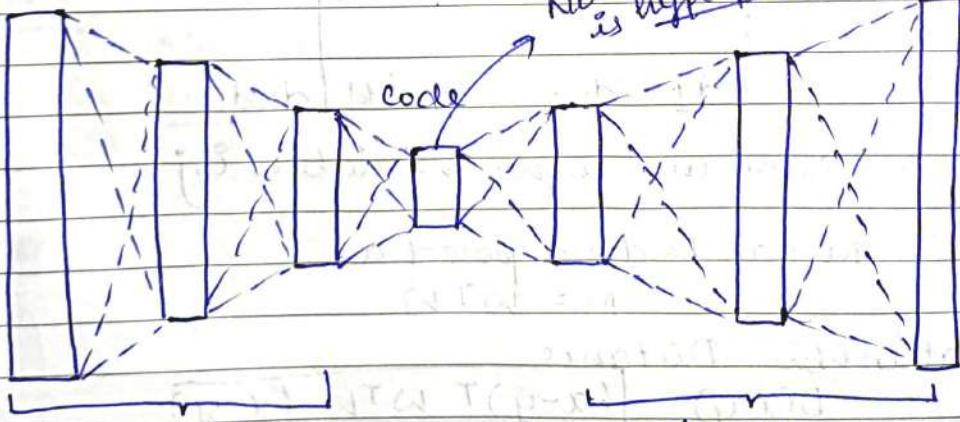
Need 3 things : Encoding Method

Decoding Method

Loss function.

Architecture:

Input



Hyperparameters:

- 1) code size : no. of nodes in middle layer.
- 2) No. of layers:
- 3) No. of Nodes per layer
- 4) Loss function

Hyperparameter Optimization: To find the hyperparameter that delivers the best performance as measured on validation set.

Model	Overview	Hyperparameters
1) C4.5	Decision tree	$C = \{0.05, 0.10, 0.20, \dots\}$
2) NNET	3 layer NN	size (4, ..., 28) decay = $\{0.10, 0.20\}$
3) KNN	KNN	$C = \{2^{*}(10 - 7) + \} \dots\}$
4) RF	Random forest	$mtry = \{10, 50, 100, 200, \dots, 1000\}$
5) SVM	SVM	$C = \{2^{-6}, \dots, 2^{10}\}$

Application: Autoencoders:

- (i) Image Generation
- (ii) Image colourization
- (iii) Dimensionality Reduction
- (iv) Data compression
- (v) Feature Extraction
- (vi) Image Denoising

Types

- (i) Denoising
- (ii) sparse
- (iii) Deep
- (iv) Contractive
- (v) Undercomplete.
- (vi) Convolutional
- (vii) Variational

SEPTEMBER
12 13 14
19 20 21 22 23 24 25
26 27 28 29 30

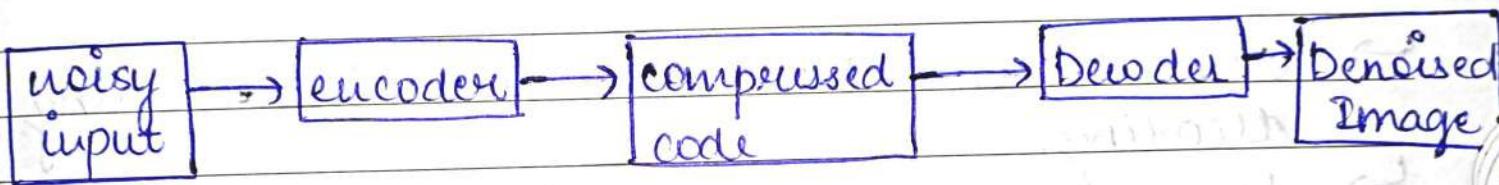
OCT
17 18 19 20 21 22 23
24 25 26 27 28 29 30

FRIDAY

Autoencoder (Types)

1) Denoising Autoencoder:

- These autoencoders take partially corrupted input while training to recover the original undistorted input
- Data corruption can be done by making some inputs zero.



2) Sparse Autoencoders:

(Hidden nodes > input nodes)

Sparcity constraint is applied on Hidden layers

to prevent output layer copy input layers

It takes frequent Activation value in HLT zero out rest of hidden nodes at a time.

3) Deep Autoencoder:

(Restricted Boltzman Machine)

Consist of 2 deep belief Net, one for encoding other

for decoding (Typically 4-5 layers).

4) Contractive Autoencoder:

To have a robust learned representation which is less sensitive to small variation in data

This can be done by applying penalty term in loss function

#(Frobenius Norm of Jacobian Matrix)

5) Undercomplete Autoencoders:

- ↪ To capture most imp features present in data.
- ↪ Have smaller dimension for hidden layer as compared to input layer.
- ↪ penalize $g(f(x))$ for being diff. from ip x .

6) Convolutional Autoencoders:

- ↪ Convolution networks are used in encoding & decoding.
- ↪ learn to encode the ip in set of simple signals & try to reconstruct ip from them.

7) Variational Autoencoders:

- ↪ Make strong assumptions concerning the dist. of latent variables.

1) Denoising (DAE)

$$L = \|x - g(f(x))\|$$

2) Sparse SAE

$$L = \|x - g(f(x))\| + \Omega(\alpha)$$

3) Deep

$$L = \|x - g(f(x))\| + \lambda \|J_f(x)\|_F^2$$

4) Contractive (CAE)

$$L = \|x - g(f(x))\|$$

5) Undercomplete (UAE)

6) Convolutional (convAE)

7) Variational (VAE)

Batch Normalization

- ↪ A technique for training DNN that standardizes the inputs to a layer for each mini-batch.
- ↪ BN makes the input to each layer to have zero mean and unit variance.
- ↪ Stabilizing the learning process and reducing the no. of training epochs required to train deep Networks.

Normalization of Input:

$$h_i = \text{layers} \quad \mu = \frac{1}{m} \sum h_i$$

$$h_i = \frac{h_i - \mu}{\sigma + \epsilon}$$

$$\sigma = \sqrt{\frac{1}{m} \sum (h_i - \mu)^2}$$

$$h_i = \frac{(h_i - \mu)}{\sigma + \epsilon}$$

ϵ = smoothing term

Rescaling of offset :

$$h_i = \gamma h_{i(\text{norm})} + \beta$$

γ = rescaling β = shifting

Advantage

- 1) Reduce covariance to internal covariance shift
- 2) Reduce dependencies of gradients on the scale of parameters and their initial values.
- 3) Allow use of saturating nonlinearities & higher learning rate.

Disadvantages

- 1) Computational overhead during training
- 2) Difficult to estimate mean & SD.
- 3) Batch size $\neq 1$

3

4

5

6

- | | |
|---|--|
| <ul style="list-style-type: none">① <u>Reduce</u> <u>Internal</u> <u>covariance</u> <u>shift</u>② <u>Reduces</u> <u>dependencies</u> <u>of gradient</u> <u>on the scale</u> <u>of parameters</u>③ <u>Allows</u> <u>use</u> of <u>saturating</u> <u>Non-linearities</u> & <u>higher</u> <u>learning</u> <u>rate.</u> | <ul style="list-style-type: none">① <u>Computational</u> <u>overhead</u>② <u>Difficult</u> to estimate mean & SD③ <u>Batch size</u> $\neq 1$. |
|---|--|

SATURDAY

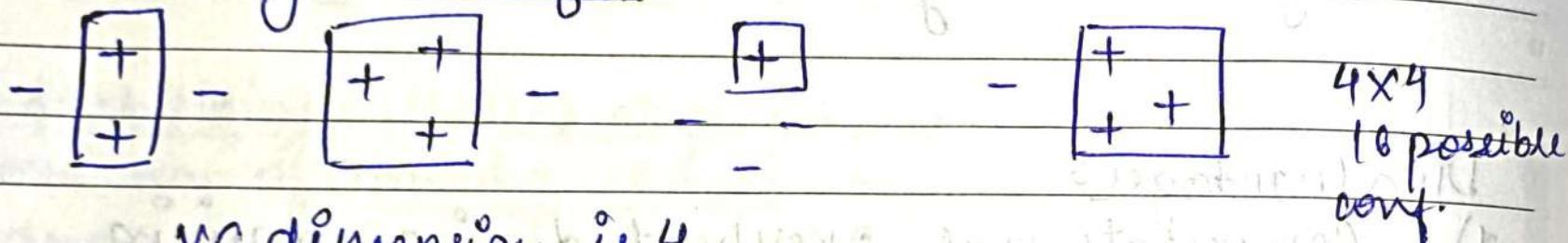
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 3014 15 16 17 18 19
21 22 23 24 25 26
28 29 30

JUNE

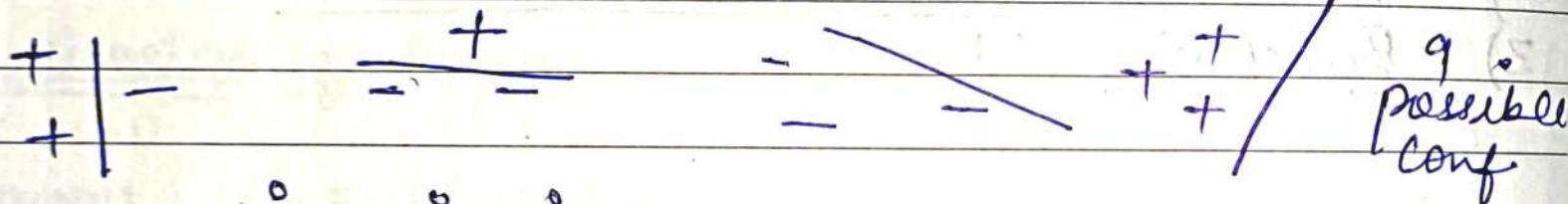
VC Dimension (Vapnik & Chervonenkis)

VC Dimension of a Hypothesis set H , denoted by vc_H
 is the largest value of N for which H can shatter all N training examples.

Rectangle classifier



VC dimension is 4



VC dimension is 3

	S	M	T	W	T	F	S
JULY '21				1	2	3	
	4	5	6	7	8	9	10
	11	12	13	14	15	16	17
	18	19	20	21	22	23	24
	25	26	27	28	29	30	31

	S	M	T	W	T	F	S
AUGUST '21				1	2	3	4
	5	6	7	8	9	10	11
	15	16	17	18	19	20	21
	22	23	24	25	26	27	28
	29	30	31				

Wk 26 | DAY 172-193

MONDAY

21

PCA

	X	Y	$x - \bar{x}$	$(x - \bar{x})^2$	$y - \bar{y}$	$(y - \bar{y})^2$	$(x - \bar{x})(y - \bar{y})$
9	2.5	2.4	0.69	0.4761	0.49	0.2401	0.3381
10	0.5	0.7	-1.31	1.7161	-1.21	1.4641	1.5851
	2.2	2.9	0.39	0.1521	0.99	0.9801	0.3861
11	1.9	2.2	0.09	0.0081	0.29	0.0841	0.0261
	3.1	3.0	1.29	1.6641	1.09	1.1881	1.4061
12	2.3	2.7	0.49	0.2401	0.79	0.6241	0.3871
	2.0	1.6	0.19	0.0361	-0.31	0.0961	-0.0589
1	1.0	1.1	-0.81	0.6561	-0.81	0.6561	0.6561
	1.5	1.6	-0.31	0.0961	-0.31	0.0961	0.0961
2	1.1	0.9	-0.71	0.5041	-1.01	1.0201	0.7171
\bar{x}	1.81	1.91		<u>5.549</u>		<u>6.449</u>	<u>5.329</u>
3				<u>9</u>		<u>9</u>	<u>9</u>
4				<u>0.6165</u>		<u>0.7165</u>	<u>0.6154</u>

$$C_{2 \times 2} \begin{bmatrix} \text{cov}(x, x) & \text{cov}(x, y) \\ \text{cov}(y, x) & \text{cov}(y, y) \end{bmatrix} = \begin{bmatrix} 0.6165 & 0.6154 \\ 0.6154 & 0.7165 \end{bmatrix}$$

$$\text{cov}(x, y) = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{n-1}$$

eigen
values

$$|C - \lambda I| = 0 \quad \begin{vmatrix} 0.6165 - \lambda & 0.6154 \\ 0.6154 & 0.7165 \end{vmatrix} = 0$$

$$\lambda^2 - 1.33\lambda + 0.0630 = 0 \quad \lambda_1 = 1.28393$$

$$\lambda_2 = 0.49068$$

JUNE '21

Wk 26 | DAY 174-191
WEDNESDAY

23

S	M	T	W	T	F	S
			1	2	3	
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

	x	y	$x - \bar{x}$	$y - \bar{y}$	$(x - \bar{x})^2$	$(y - \bar{y})^2$	$(x - \bar{x})(y - \bar{y})$
9	2	4	-2.5	-4	6.25	16	10.0
10	3	5	-1.5	0	2.25	0	0
11	4	3	-0.5	-2	0.25	4	1
12	5	6	0.5	1	0.25	1	0.5
13	6	7	1.5	2	2.25	4	3
14	7	8	2.5	3	6.25	9	7.5
15	<u>27</u>	<u>30</u>			17.5	34	22

1 $\mu_1 = \frac{27}{6} = 4.5$

2 $\mu_2 = \frac{30}{6} = 5$

3 Covariance Matrix = $\begin{bmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{bmatrix}$

4 Characteristic eqn: $|M - \lambda I| = 0$

5
$$\begin{vmatrix} 2.92 - \lambda & 3.67 \\ 3.67 & 5.67 - \lambda \end{vmatrix} = 0$$

6 $\lambda^2 - 8.59 + 3.09 = 0$

$\lambda_1 = 8.22$

$\lambda_2 = 0.38$ (very small) left out.

Eigen Vector corresponding to greatest eigen value
is PC

$$\therefore MX = \lambda X$$

$$\begin{bmatrix} 9.92 & 3.67 \\ 3.67 & 5.67 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 8.22 \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$9.92x_1 + 3.67x_2 = 8.22x_1$$

$$3.67x_1 + 5.67x_2 = 8.22x_2$$

$$5.3x_1 = 3.67x_2 \quad \text{--- (1)}$$

$$3.67x_1 = 2.55x_2 \quad \text{--- (2)}$$

$$\text{from } x_1 = 0.69x_2$$

$\textcircled{1} + \textcircled{2}$

$$\text{from (2)} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2.55 \\ 3.67 \end{bmatrix}$$

$$\text{Principal component } \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2.55 \\ 3.67 \end{bmatrix}$$

\Rightarrow Plot graph

out let feature vector $\begin{bmatrix} 2 \\ 1 \end{bmatrix}$

$$\text{feature vector} = (\mathbf{v}_1)^T (FV - MV)$$

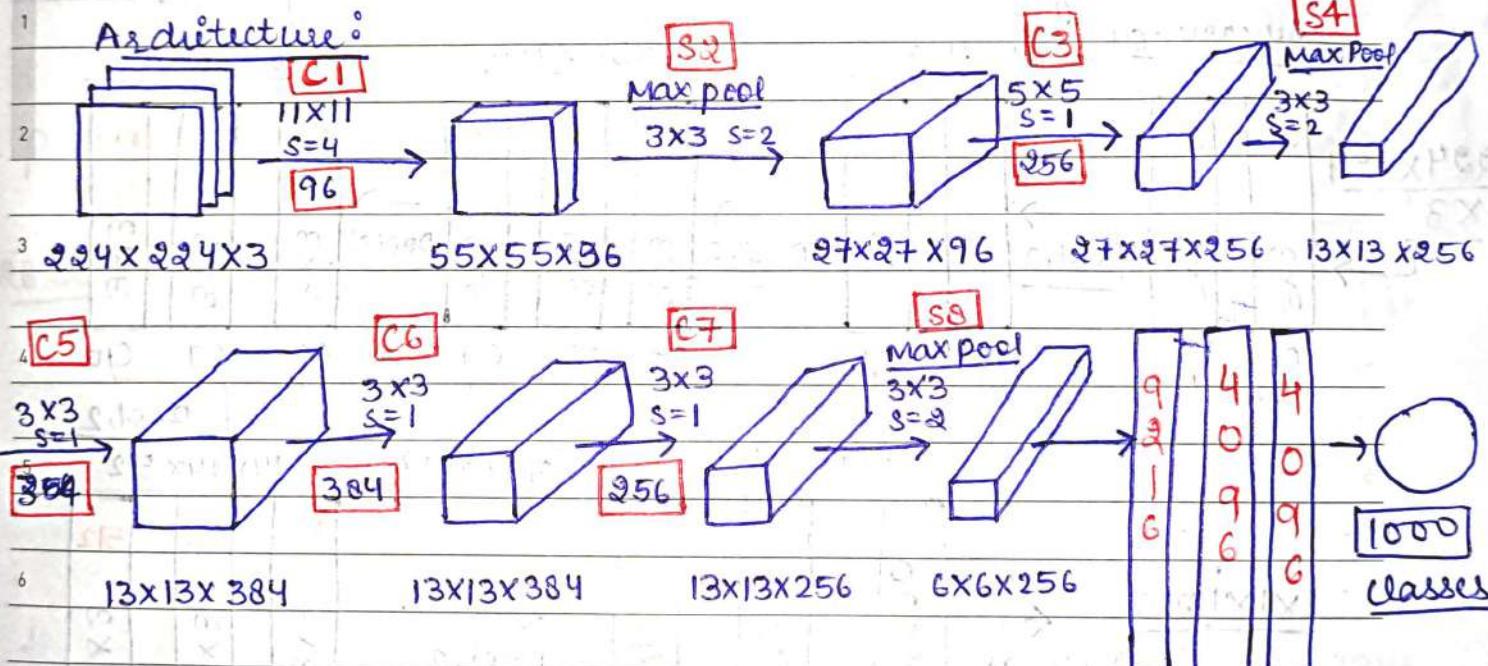
gets transformed to

$$= [2.55 \ 3.67] \left[\begin{bmatrix} 2 \\ 1 \end{bmatrix} - \begin{bmatrix} 4.5 \\ 5 \end{bmatrix} \right]$$

$$= -21.055$$

AlexNet

- ↳ first CNN which used GPU to boost performance
- ↳ 5 convolutional layers
- 3 Max Pooling layers
- 2 Normalization Layers.
- 3 fully connected layers
- 1 softmax layers.

Architecture:

Total No. of Parameters : 69.3 Million

3 hidden layers

21 JUNE

26

DAY 177-188 | Wk 26
SATURDAY

M	T	W	T	F	S	S
31				1	2	
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
						MAY '21
						1
						2
						3
						4
						5
						6
						7
						8
						9
						10
						11
						12
						13
						14
						15
						16
						17
						18
						19
						20
						21
						22
						23
						24
						25
						26
						27
						28
						29
						30
						JUNE '21

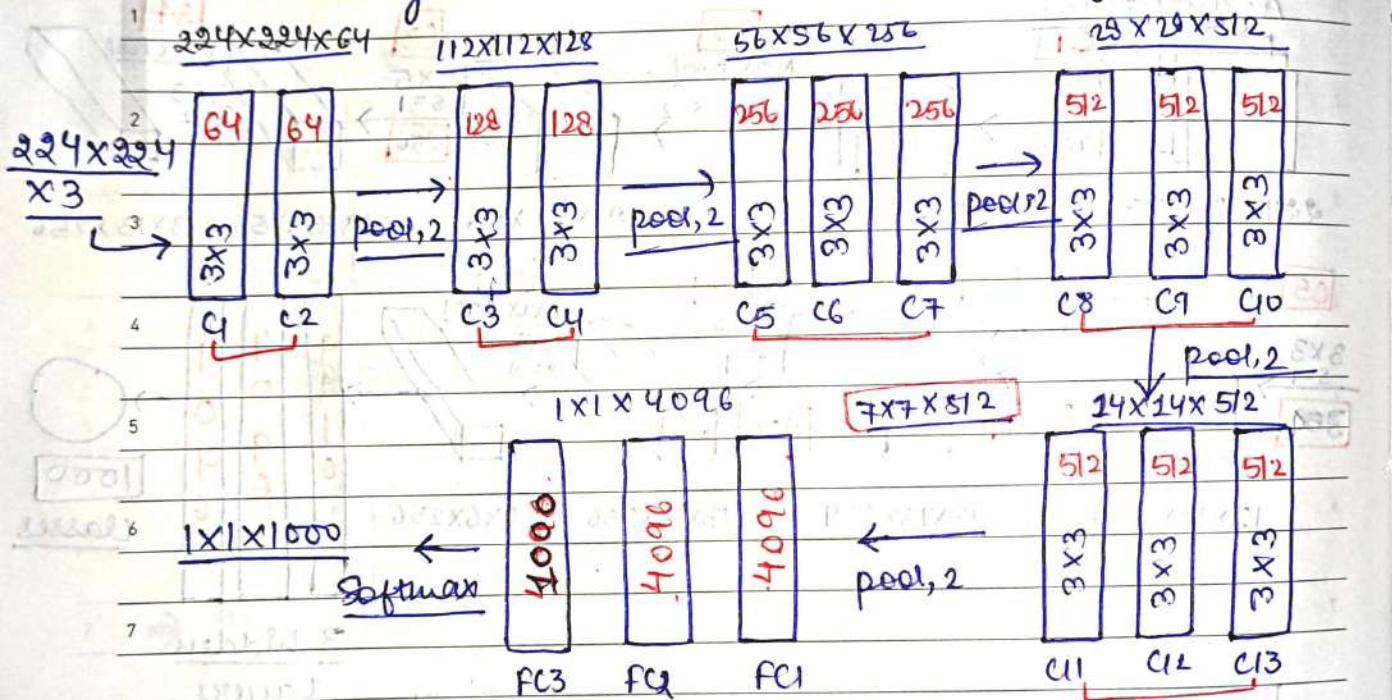
Visual Geometry Group Architecture

↳ VGG16 has total of 16 layers, only convolution and pooling layers are used.

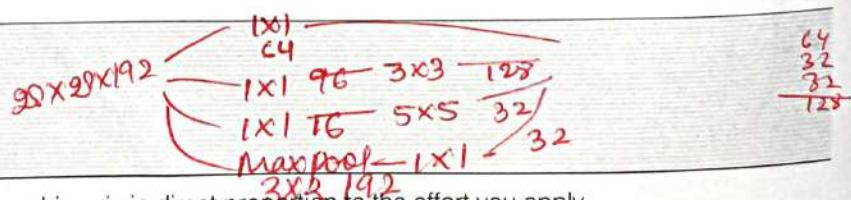
↳ Kernel: 3×3 stride = 1

Max pooling: 2×2 stride = 2

↳ Accuracy: 92.7% trained on ImageNet Dataset



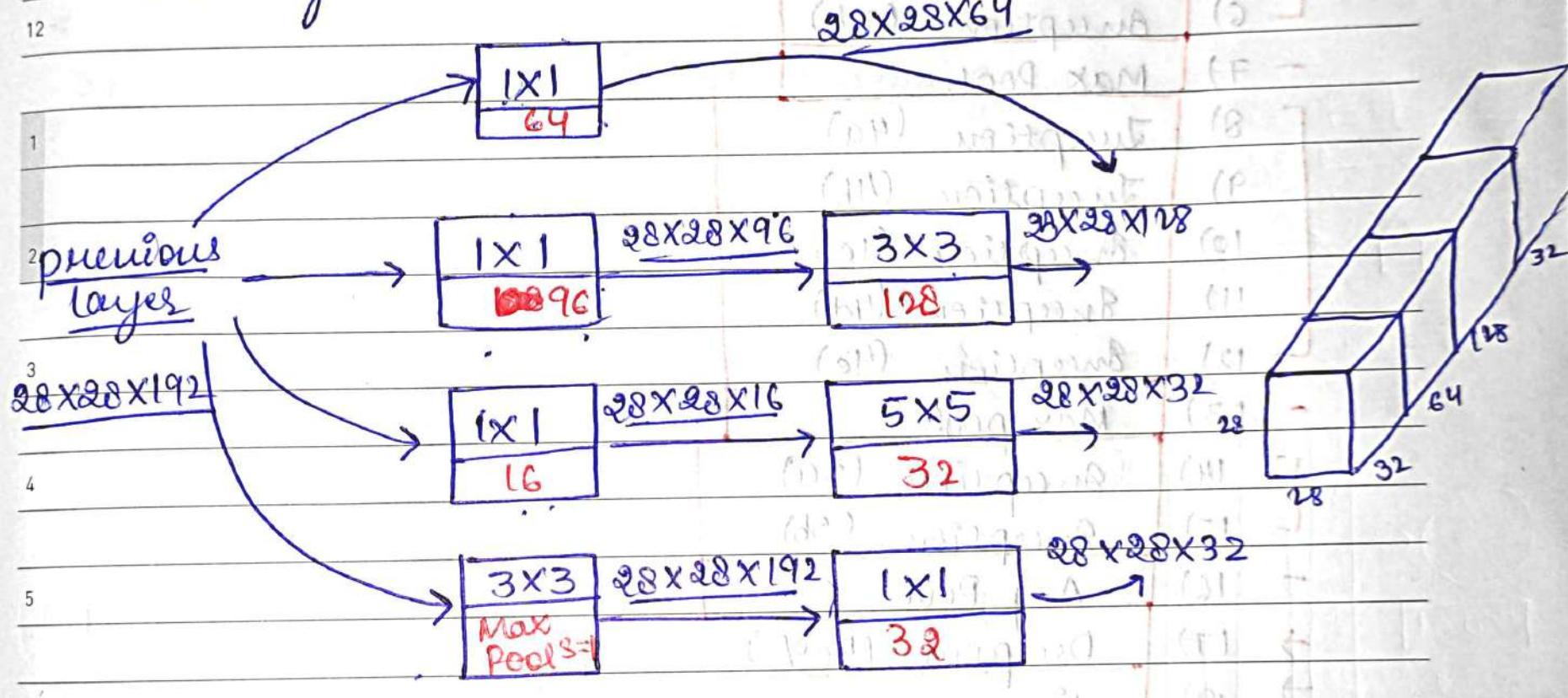
27 SUNDAY



The result you achieve is in direct proportion to the effort you apply

Inception Architecture

Inception layer is the combination of all those layers (1x1 conv. layers, 3x3 conv. layer, 5x5 conv. layer) with their output filter banks concatenated into a single output vector forming the ip of next stage.



Same padding helps in getting contribution from boundary pixels.

GoogleNet designed Inception Module that Approximate a Sparse CNN with a Normal Dense Construction.

29

DAY 180-185 | Wk 27

TUESDAY

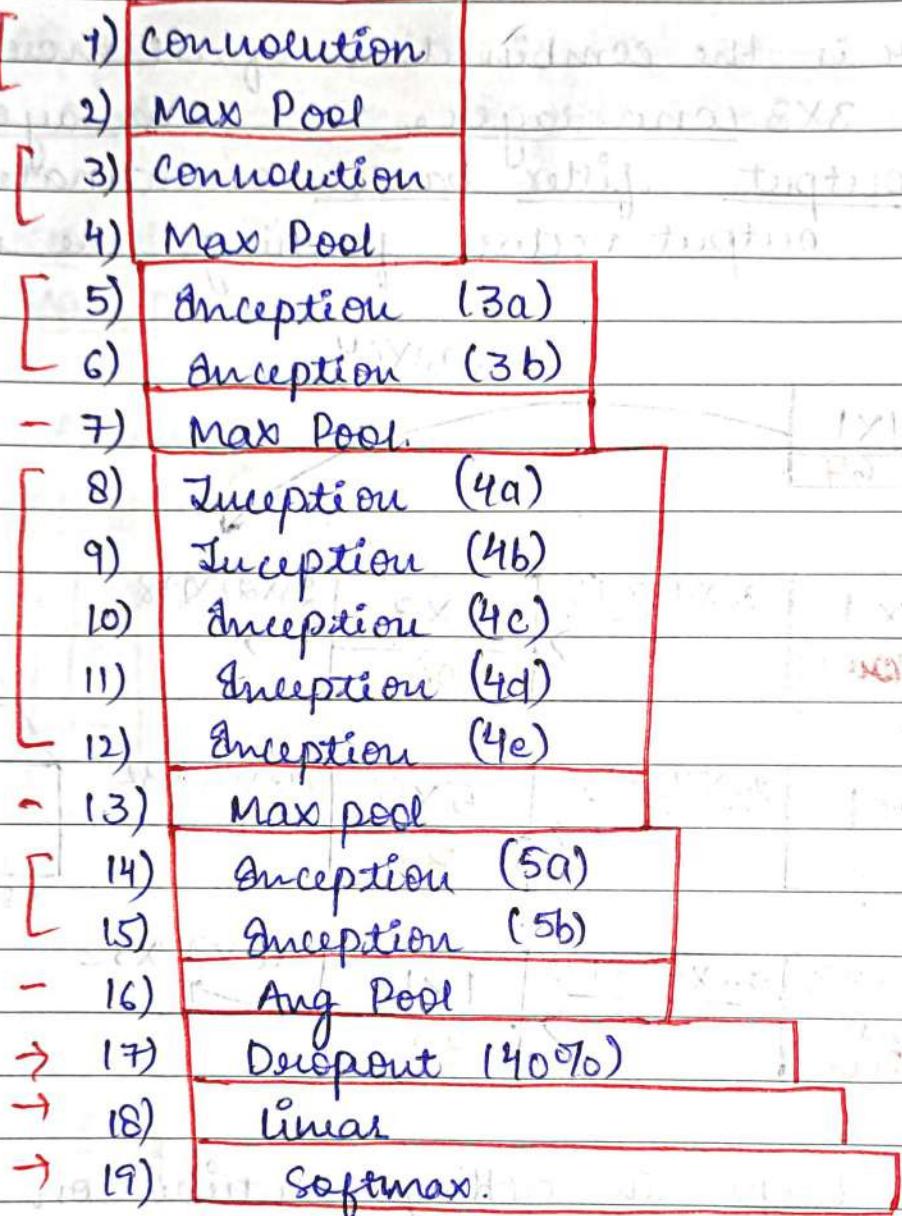
GoogleNet

M	T	W	T	F	S	S
31				1	2	
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

MAY '21

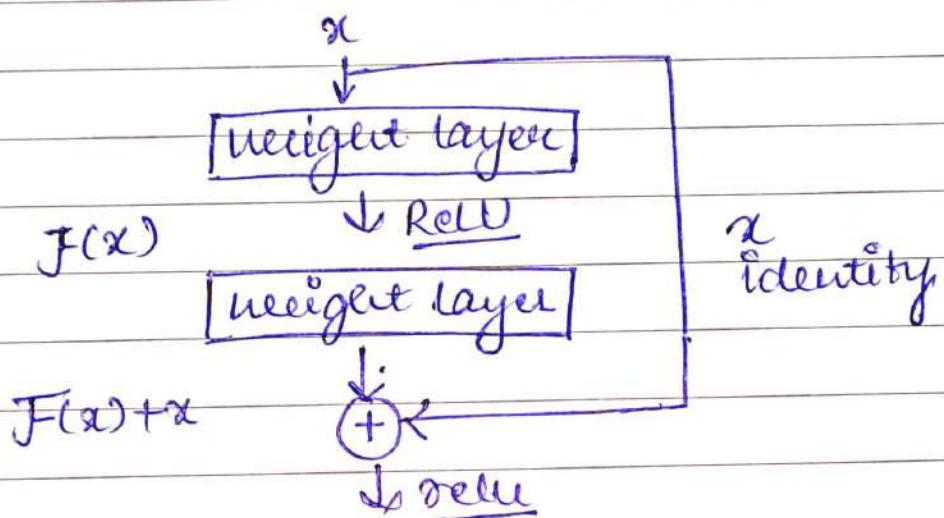
M	T	W	T	F	S	S
1	2	3	4	5	6	
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

JUNE '21

Inception Architecture : 27 Layer Deep.

ResNet (Residual)

- 9
- 10 1) In order to solve the problem of vanishing gradient this Network uses the technique of skip connections
- 11 2) Skip training from few layers & connect them directly to output.
- 12 3) Network fit Residual Mapping.



21' JULY Geometric
invariance

02

DAY 183-182 | Wk 27

FRIDAY

M	T	W	T	F	S	S
1	2	3	4	5	6	
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

M	T	W	T	F	S	S
1	2	3	4			
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	JULY 21

Spatial Transformer Networks

9

STN allows a neural network to learn how to perform spatial transformation on the input image in order to enhance the geometric invariance of Model.

11

e.g. it can crop a region of intersection, scale, correct the orientation of image.

Components:

Localization Network

Grid
Generator

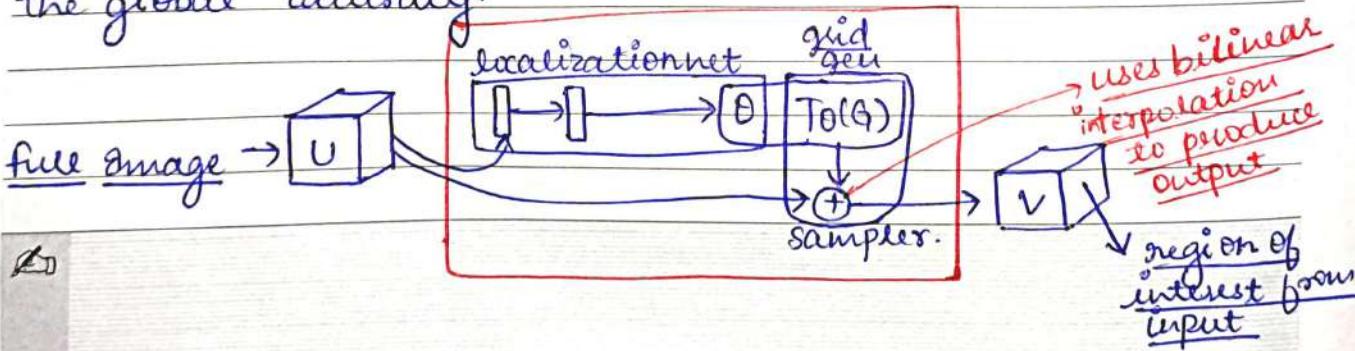
Sampler

Regular CNN which
regress transformation
parameters

Generate a grid
of coordinates in
input images
corresponding to
each pixel from
output image.

It uses the
parameters of
transformation
and applies
it to the
input img

transformation is never
learned explicitly from
dataset, instead netwk
learns automatically the
spatial trans. that enhances
the global accuracy.



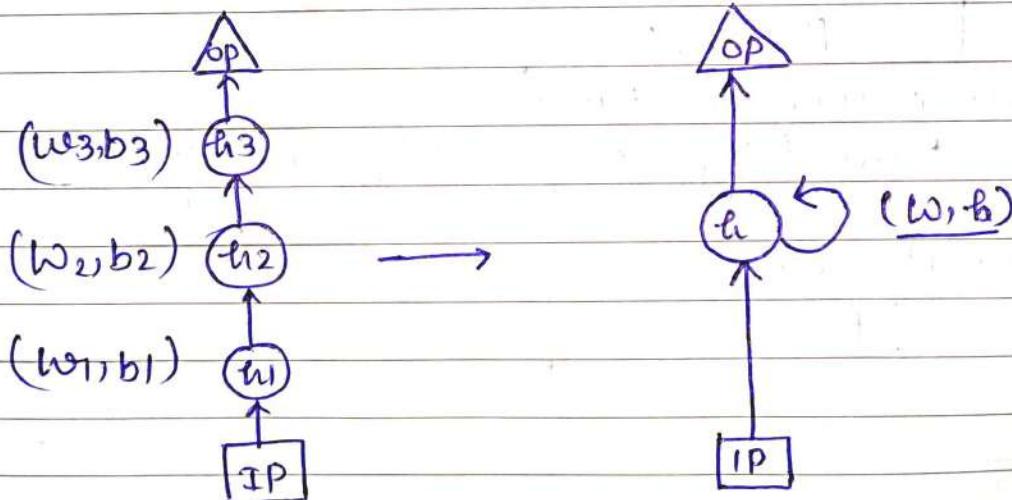
Recurrent Neural Network

→ RNN are a type of NN where the output from previous step are fed as input to current step.

→ RNN has hidden state (memory) which remembers all the info about what has been calculated.

e.g. let DN having one IP, 3 HL, One OP layer having weight & bias as (w_i, b_i) , $i \rightarrow \{1, 2, 3\}$

RNN converts independent activations into dependent activations by providing same weight & bias to all layers, thus reducing the complexity of increasing parameters, & memorizing each previous output by giving it as IP to next HL



SUNDAY 04

→ calculate current state:

$$h_t = f(h_{t-1}, x_t)$$

$$h_t = f(h_{t-1}, x_t)$$

$h_t \rightarrow$ current state
 $h_{t-1} \rightarrow$ previous state
 $x_t \rightarrow$ IP state.

21' JULY

05

DAY 186-179 | Wk 28

MONDAY

JUNE 21						JULY 21					
M	T	W	T	F	S	M	T	W	T	F	S
1	2	3	4	5	6	7	8	9	10	11	12
14	15	16	17	18	19	20	21	22	23	24	25
28	29	30				5	6	7	8	9	10
						12	13	14	15	16	17
						19	20	21	22	23	24
						26	27	28	29	30	31

B) Applying activation function.

$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t)$$

W_{hh} → net at recurrent neuron

W_{xh} → net at input neuron

c) calculate output

$$y_t = W_{hy} h_t$$

y_t → output

W_{hy} → net at op neuron.

1) calculate current state:

$$h_t = f(h_{t-1}, x_t)$$

2) Apply Activation function

$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t)$$

3) calculate output

$$y_t = W_{hy} h_t$$

AUG
22 23 24 25 26 27 28

SEPT
19 20 21 22 23 24 25
26 27 28 29 30

TUESDAY

Advantages of RNN

- 1) Remembers info \rightarrow useful for series prediction
 \rightarrow LSTM

- 2) RNN used with conv. layers to extend powerful pixel neighbourhood.

Disadvantages of RNN

- 1) Gradient vanishing and exploding problem
- 2) Training RNN is difficult.
- 3) long sequence if using tanh / ReLU as AF

Applications \rightarrow Text Generation \rightarrow Machine Translation

\rightarrow Speech Recognition \rightarrow Sentiment Analysis \rightarrow Anomaly Detection

\rightarrow Visual search \rightarrow Face Recognition \rightarrow Stock price forecasting

LSTM Network

- ↳ Special kind of RNN, capable of learning long-term dependencies.
- ↳ LSTM make small modifications to the info by Multiplication and Addition.
- ↳ LSTM flow information in the form of cell state.

LSTM deal with both LTM and STM for making calculations simple and effective.

- 1) forget Gate: LTM goes to FG and it forgets info that is not useful.
- 2) Learn Gate: Current input and STM combined so that necessary info we recently learned from STM can be applied to current IP.
- 3) Remember Gate: LTM info. that we haven't forgot and STM + Event in RG which works as updated LTM.
- 4) Use Gate: This gate also uses LTM, STM & event to predict OIP of current event which works as an update STM.

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

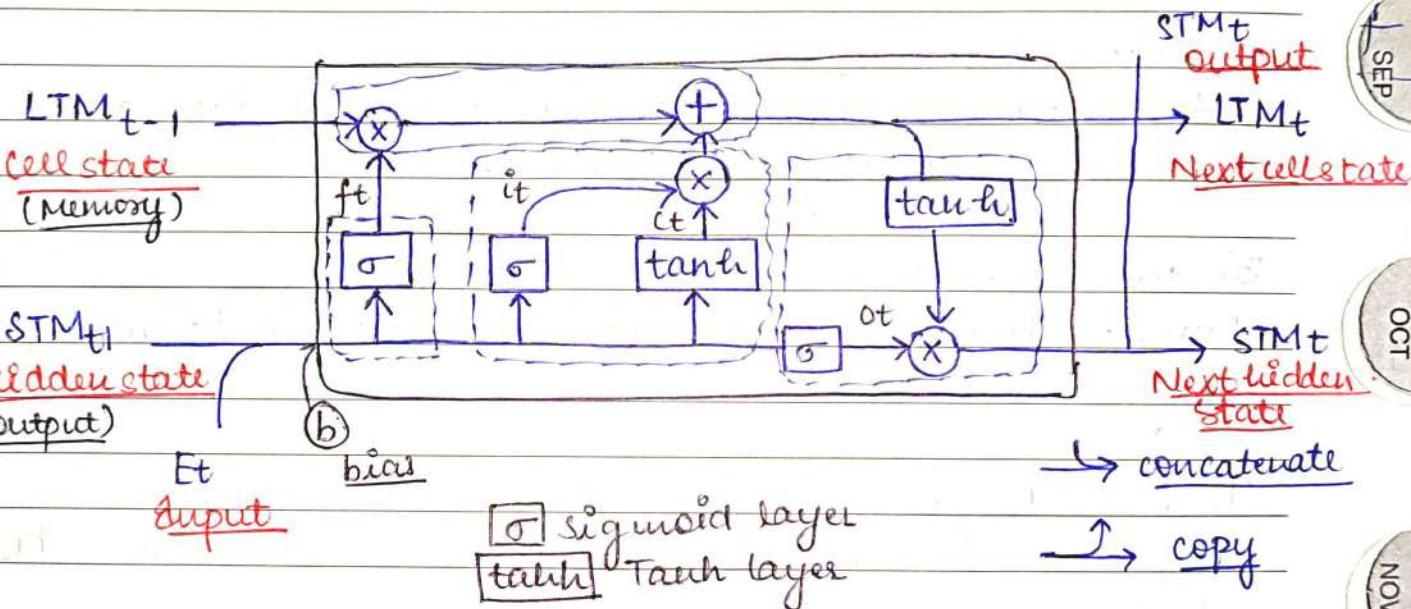
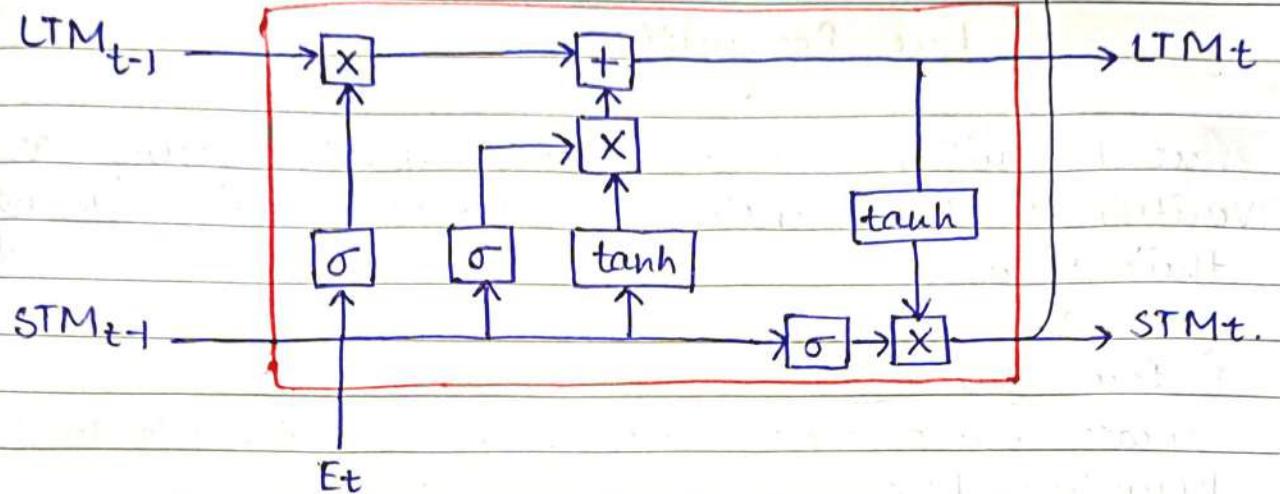
S	M	T	W	T	F	S
					1	2
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

3-σ 1-t
3-X 2-table

JULY '21

Wk 28 | DAY 189-176
THURSDAY

08



Applications:

- 1) Time Series prediction
- 2) Robot control
- 3) Speech Recognition
- 4) Handwriting Recognition
- 5) Music composition
- 6) Drug Design
- 7) Object segmentation
- 8) Market prediction
- 9) Semantic parsing
- 10) Protein homology detection

Face Recognition

9
10 Face Recognition is a method of identifying or verifying the identity of an individual using their face.

11

1) face detection:

12 Locate one or more faces in image and mark with a bounding box.

1

2) face alignment:

2 Normalize face to be consistent with the database such as geometry and photometrics.

3

3) face extraction:

4 Extract features from the face that can be used for the recognition task.

5

4) face Recognition:

6 Perform matching of face against one or more known faces in a prepared database.



Audio WaveNet

- 1) WaveNet is deep Neural Network for generating raw audio. It was created by researchers at London-based AI firm DeepMind.
- 2) The technique is able to generate relatively realistic sounding human like voices by directly modelling waveforms using a NN method trained with recordings of real speech.
- 3) Tests with US English and Mandarin reportedly showed that system outperforms Google's best existing (TTS) system.
- 4) WaveNet is an audio generative model based on the Pixel CNN.
In this generative model, each audio is conditioned on previous audio sample.
The conditional prob. is modelled by a stack of convolutional layers.
- 5) This network does not have pooling layers, and output of model has same time dimensionality as IP.

FPNN
dilations

④ Output - Input
Same dimension

① Deep Mind ② AGM
② Pixel CNN

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

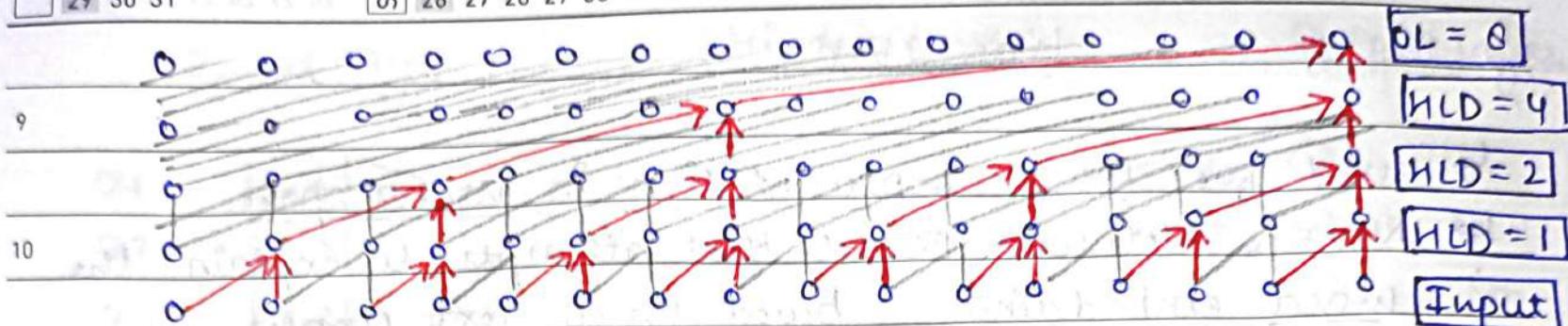
S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

OL - output dilation JULY '21
HLD - Hidden Layer Dilation

Wk 29 | DAY 194-171

TUESDAY

13



11

6) fully convolutional NN, where convolutional layers have various dilations factor that allows its receptive field to grow exponentially with depth and covers thousands of time steps.

7) Joint probability of waveform $x = \{x_1, \dots, x_T\}$ is factorized as product of conditional probabilities

$$p(x) = \prod_{t=1}^T p(x_t | x_1, \dots, x_{t-1})$$

each audio sample x_t is therefore conditioned on samples at all previous timesteps.

so based on the above diagram, we can say that x_t is conditioned on x_1, x_2, \dots, x_{t-1}

if we want to get x_t then we need to take input x_1, x_2, \dots, x_{t-1}

so if we want to get x_t then we need to take input x_1, x_2, \dots, x_{t-1}

so if we want to get x_t then we need to take input x_1, x_2, \dots, x_{t-1}

so if we want to get x_t then we need to take input x_1, x_2, \dots, x_{t-1}

so if we want to get x_t then we need to take input x_1, x_2, \dots, x_{t-1}

21 JULY

Word Embedding based on text corpus

14

DAY 195-170 | Wk 29

WEDNESDAY

	M	T	W	T	F	S	S
JUNE '21	1	2	3	4	5	6	1
JULY '21	5	6	7	8	9	10	4
	12	13	14	15	16	17	18
	19	20	21	22	23	24	25
	26	27	28	29	30	31	

Word2Vec

- 9 ↳ used for word representation in vector space
- 10 ↳ Neural Network Model that attempts to explain the word embeddings based on a text corpus.
- 11 ↳ 2 layer Neural network that processes text.
- 12 Input: text corpus
- 13 Output: set of vectors
- 14 ↳ Output of W2V NN is vocabulary in which each item has vector attached , which can be fed into DLN to detect relationship b/w words.
- 15 ↳ To implement word embedding using Word2Vec:
- 16 (i) Continuous Bag of words (CBOW)
- 17 (ii) Skip-Gram.

- 18 1) CBOW Model: Takes the context of each word as input and tries to predict the target word in its neighbourhood.

Objective function $-\log \left(P\left(w_0|w_i\right) \right)$

w_0 = output words

w_i = context words

$$P(w_0|w_i) = \frac{\exp(v_{w_0}^T v_{w_i})}{\sum_{w=1}^W \exp(v_w^T v_{w_i})}$$

Disadv

① slower to train

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

② context

S	M	T	W	T	F	S
1	2	3	4	5	6	7
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

Advantages

① faster

② Better for frequent words
representation | DAY 196-169

③ low on Memory

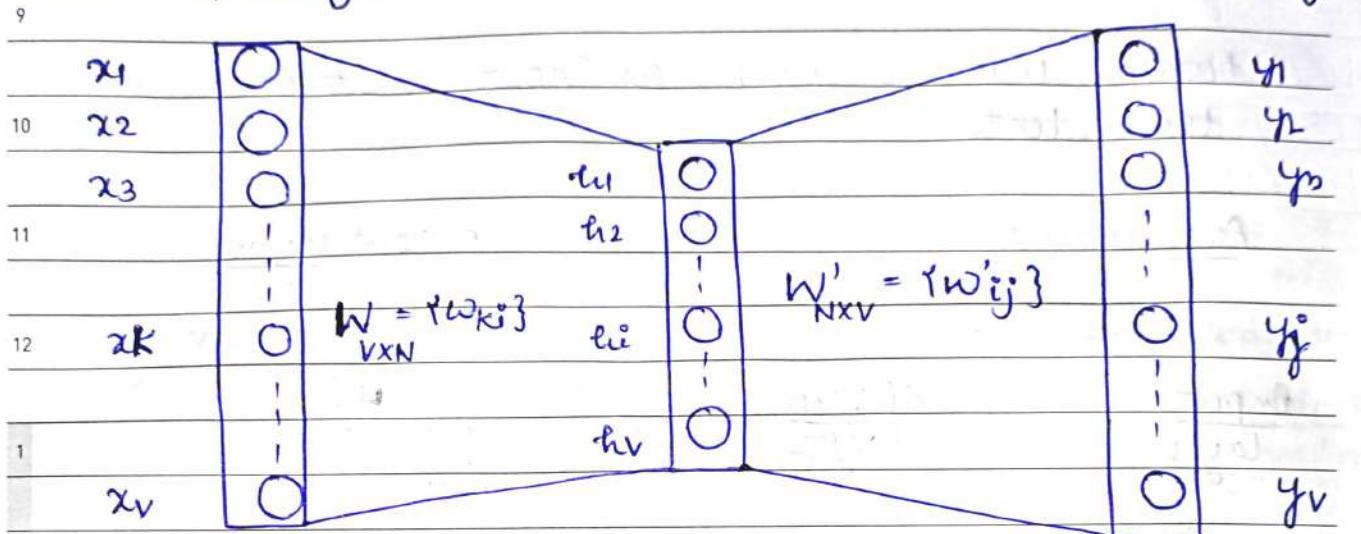
JULY '21

15

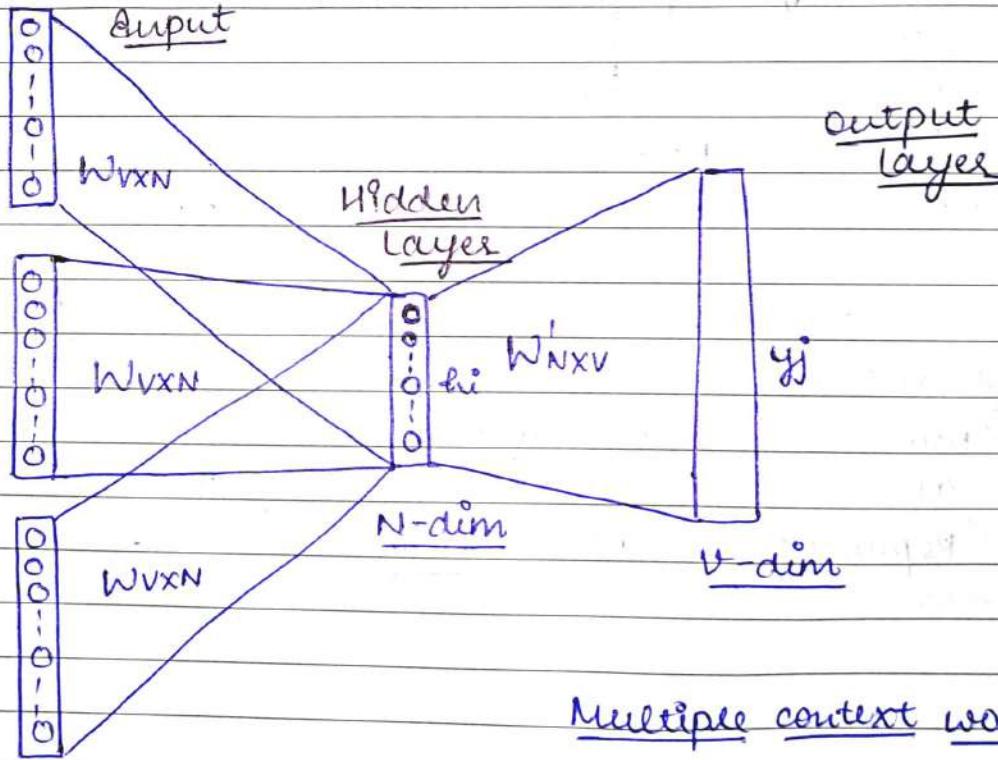
Input layer

Hidden layer

Output layer



single context-word



Multiple context words

C X V-dim

2) Skip-Gram Model

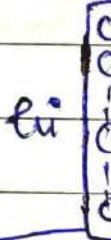
target word is given as input, it predicts the context.

Architecture:

Output layers

Input layer

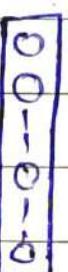
Hidden layer



w_{NXN}

e_i

w_{NXV}



y_{1j}

w_{NXV}



y_{2j}

w_{NXV}



y_{cj}

$N\text{-dim}$

$v\text{-dim}$

Advantages:

- 1) ✓ works well with small amount of data
- 2) ✓ Represent rare words well.

$C \times V\text{-dim}$

AUG

22 23 24 25 26 27 28
29 30 31

SEPTEMBER

19 20 21 22 23 24 25
26 27 28 29 30

SATURDAY

Image captioning

The process of generating textual desc. from an image based on objects and actions in image.

Divide the task into 2 Modules:

Image Based model

Extract the features and nuances out of an Image.

Encoder



Convolutional NN

Language Based Model

translate features and objects given by IBM to a natural sentence)

Decoder



Recurrent NN

AUG

OCT

NOV

ImageNet

9 ↳ Imagenet is a large dataset of images which are originally labelled with synsets of wordnet lexicon tree.

10 ↳ For computer vision Research:

11 ↳ Object detection, Image classification & object localization.

12 ↳ 14,197,122 images organized into 21,841 subcategories.
1 These sub-categories can be considered as subtree of
2 27 high-level categories.

3 ↳ 14,197,122 images → 21,841 categories → 27 high level categories

AUG
22 23 24 25 26 27 28
29 30 31

SEPTEMBER
19 20 21 22 23 24 25
26 27 28 29 30

TUESDAY

20

Bioinformatics

Bioinformatics is the field of study that deals with computational & mathematical approaches to analyse & extract knowledge from Biological data.

Application of ML in Bioinformatics:

- ↳ Diseases prediction
- ↳ Diagnosis Analysis
- ↳ Survival Analysis

Bioinformatics Tools:

1) Homology and similarity Tools:

Computational Analysis → protein & DNA Seq → similarity → common ancestry

2) Protein function Analysis:

Protein (size, structure, physiochemical properties) → Identification → mass spectroscopy
separation → electrophoresis

3) Structural Analysis:

Seismic Response → Vulnerability of Building → Topologies

4) Sequence Analysis:

RNA, DNA, peptide seq → analytics → search, align

Advantages

Disadvantages

- 1) provides systematic info about genomics, proteomics and metabolomics.
 - 2) helps in finding out evolutionary relationship between 2 species.
 - 3) Gene Mapping & Sequencing
 - 4) To reconstruct genes from expressed sequence tags.
 - 5) Storing, organizing & indexing huge database.
- 1) Require sophisticated lab - require lots of funds.
 - 2) Need uninterrupted power supply.
 - 3) Regular check of computer viruses.
 - 4) High Maintenance.

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

S	M	T	W	T	F	S
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

LSTM Network

- 1) special kind of RNN, capable of learning long-term dependencies.
- 2) makes small modifications to the info by multiplication and Addition
- 3) LSTM flow information in form of all states
- 4) Deal with both STM and LTM for making calculations simple and effective.

(A) forget gate: Decides which info needs attention and which can be ignored.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

f_t = forget gate at t

W_f = weight matrix between forget gate & input gate

h_{t-1} = previous hidden state

x_t = input

b_f = connection bias

(B) Input gate: perform 2 operations to update cell status

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

i_t = input gate at t

b_i = bias vector at I

$$C_t^u = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$

W_i = weight matrix (input gate to OP gate)

C_t^u = value generated by \tanh

W_c = weight matrix (cell state info)

A good plan violently executed now is better than a perfect plan next week Network op)

23

DAY 204-161 | Wk 30

FRIDAY

1	2	3	4	5	6
7	8	9	10	11	12
14	15	16	17	18	19
21	22	23	24	25	26
28	29	30			

JUNE '21

1	2	3	4
5	6	7	8
12	13	14	15
19	20	21	22
26	27	28	29

JULY '21

(C) cell state : To decide new cell state

[forget gate & input gate of previous info] [new cell state]

$$C_t = f * C_{t-1} + i_t * \tilde{C}_t$$

C_t = cell state information

f_t = forget gate at t

i_t = input gate at t

C_{t-1} = previous cell state info.

(D) Output gate : Determine value of next hidden state (contains info on previous inputs)

$$O_t = \sigma(W_o [h_{t-1}, x_t] + b_o)$$

$$h_t = O_t * \tanh(C_t)$$

O_t = output gate at t [current state & previous hidden state] [new cell state]

W_o = weight matrix - output gate

b_o = bias vector

h_t = LSTM output

AUGUST
8 9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31

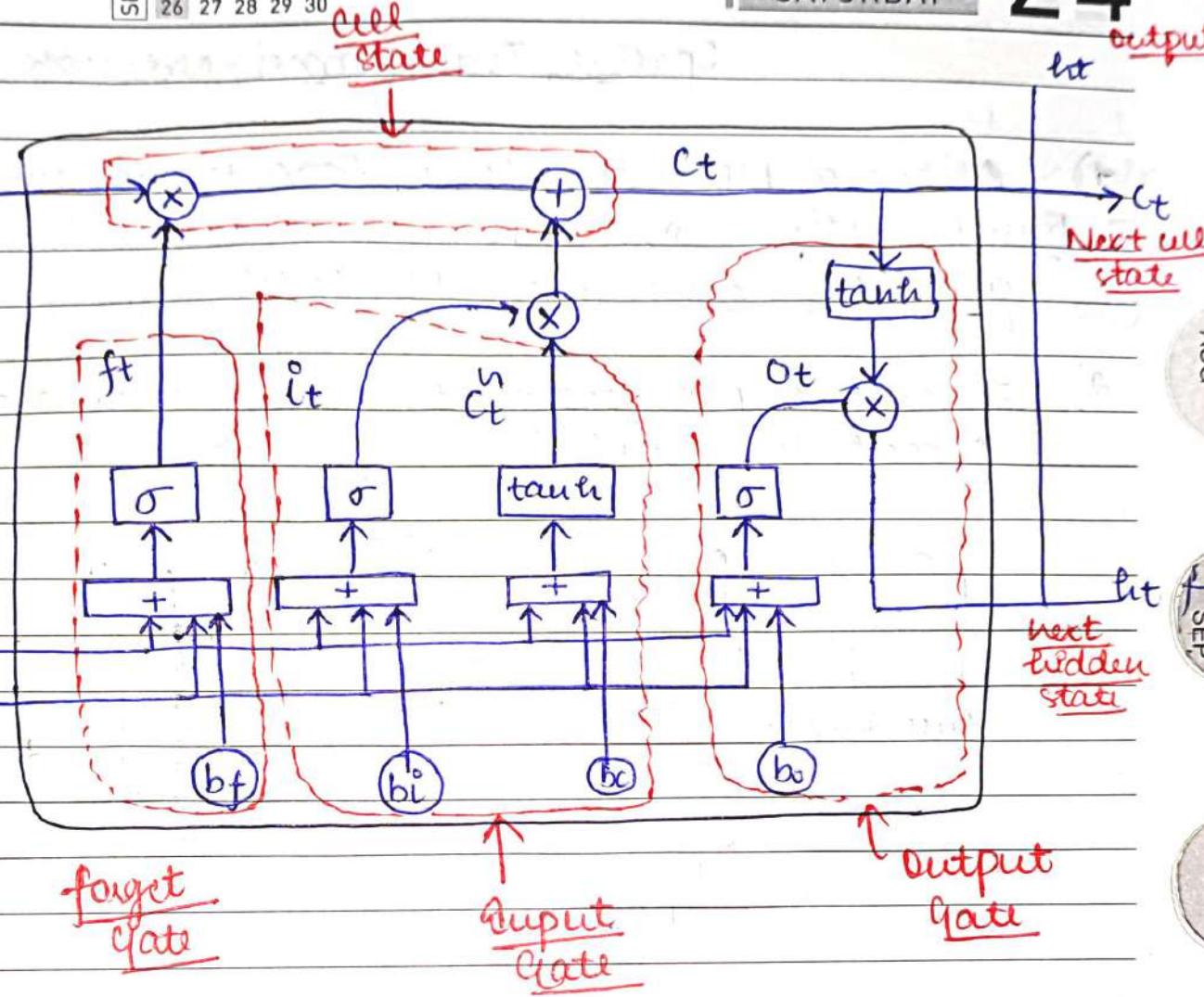
SEPTEMBER
5 6 7 8 9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30

Wk 30 | DAY 205-160

SATURDAY

24

output



Advantages:

- 1) solves the problem of vanishing gradient.

Applications:

- ↳ Language Translation
- ↳ Speech & Handwriting Recognition
- ↳ Music Generating
- ↳ Image processing
- ↳ Language Modelling

Disadvantages:

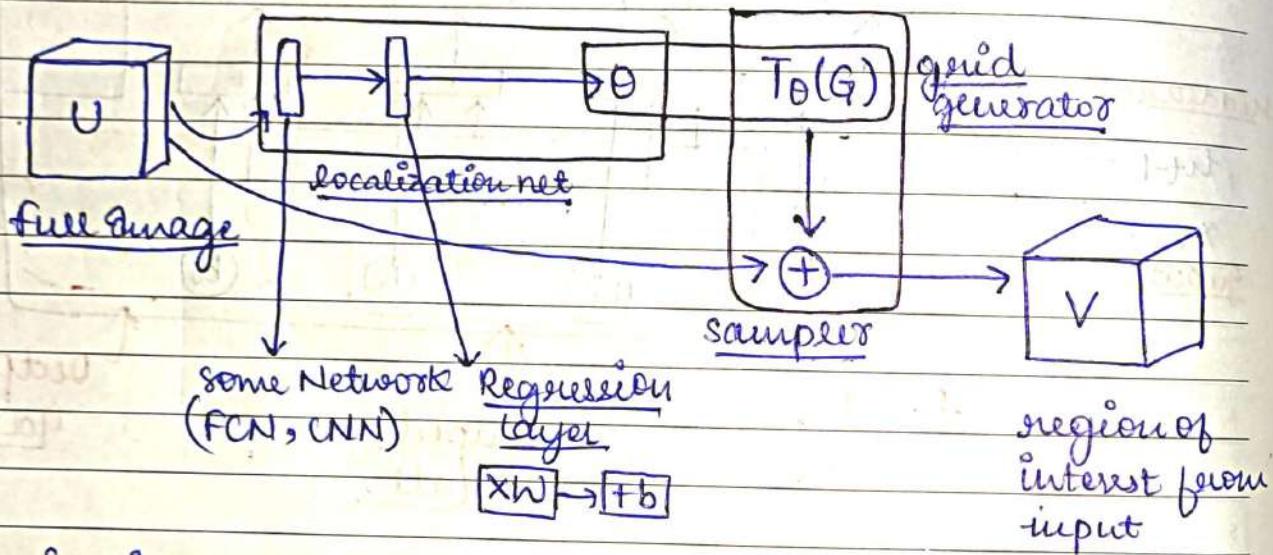
- 1) Takes longer to train
- 2) Requires more memory
- 3) Easy to overfit
- 4) Sensitive to random weight initialization
- 5) Difficult to apply dropout algorithm

SUNDAY 25

DEC

Spatial Transformer Networks

- 1) Allow a NN to learn how to perform spatial transformation on input image to enhance geometric invariance of Model.
- 2) It can crop of region of intersection, scale correct the orientation of image.



(A) Localization Net

- 1) Regular Neural Networks that regress transformation parameters.
- 2) Transformation is never learned explicitly, instead Network learns automatically the spatial transform that enhances global accuracy.

transformation parameters

	S	M	T	W	T	F	S
AUGUST '21	1	2	3	4	5	6	7
	8	9	10	11	12	13	14
	15	16	17	18	19	20	21
	22	23	24	25	26	27	28
	29	30	31				

	S	M	T	W	T	F	S
SEPTEMBER '21					1	2	3
	5	6	7	8	9	10	11
	12	13	14	15	16	17	18
	19	20	21	22	23	24	25
	26	27	28	29	30		

27

(B) Grid Generator:

Transformation parameters predicted by IN are used in form of Affine Transformation Matrix for each image in batch.

(Affine transformation preserves points, straight lines + planes)

$$\begin{pmatrix} z_i^s \\ y_i^s \end{pmatrix} = T_0(g) = A\theta \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix} = \begin{bmatrix} \theta_{01} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$

source

angle
by which
img has to
be rotated

translation
along
width &
height

target

Thus, we obtain sampling grid

(C) Sampler: Iterate over entries of sampling grid and extract corresponding pixel value from IP map using Bilinear Interpolation.

- ① find 4 neighbouring points
- ② for each neighbouring point calculate weight
- ③ take weighted avg. to produce output.

$$w_{ij} = U_{yx} \cdot dy \cdot dx + U_{y\bar{x}} \cdot dy (1-dx)$$

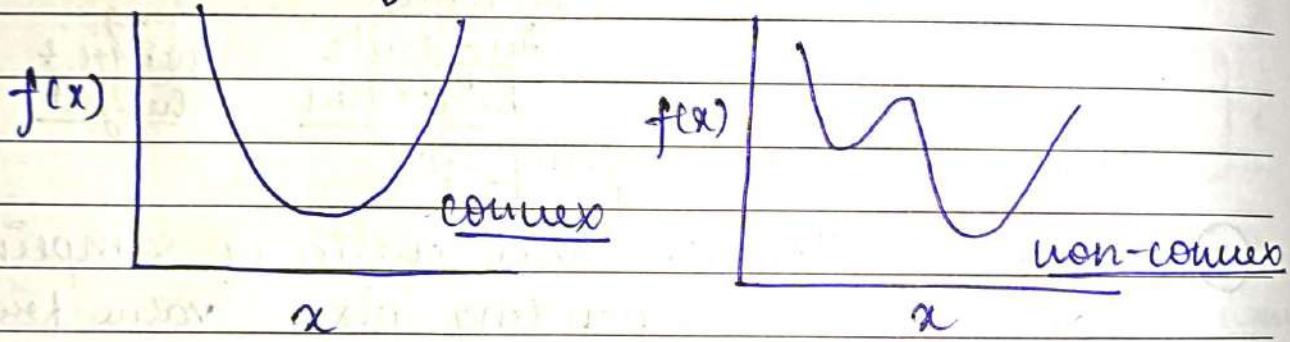
$$+ U_{\bar{y}x} \cdot d(1-dy)dx + U_{\bar{y}\bar{x}} (1-dy)(1-dx)$$

Non convex optimization

- 1) Potentially many local minima
- 2) saddle points

(A point on the surface of graph of a function where slopes in orthogonal directions are all zero, but which is not the local extremum of a function).

- 3) Very flat region (plateaus)
- 4) widely varying curvature



Gradient Descent optimization Algo.

- (i) Momentum
- (ii) Nesterov Accelerated Gradient
- (iii) Adagrad
- (iv) Adadelta
- (v) RMS Prop
- (vi) Adam
- (vii) Adam Extension

Momentum

9

SGD has trouble navigating Ravines ; momentum helps
accelerate.

10

11

Adds a fraction γ of updated vector of past step
 v_{t-1} to current vector v_t

12

$$v_t = \gamma v_{t-1} + \gamma \nabla_{\theta_t} J(\theta_t)$$

$$\theta = \theta - v_t$$

1

2 ↳ Reduce update for dimension whose gradient changes directions.

3 ↳ Increase updates for dimension whose gradients points in the same direction.

4

AUG

SEP

OCT

Adagrad

Adagrad adapts the learning rate to the parameters

SGD update

$$\theta_{t+1} = \theta_t - \eta g_t$$

$$[g_t = \nabla_{\theta_t} J(\theta_t)]$$

Adagrad divides learning rate by square root of sum of squares of historic gradients.

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_t + \epsilon}} \cdot g_t$$

$G_t \in \mathbb{R}^{d \times d}$ diagonal matrix where each diagonal matrix element i_i is the sum of sq. of gradients w.r.t θ_i

$$\theta_{t+1} = \theta_t - \eta g_t = \theta_t - \frac{\eta}{\sqrt{G_t + \epsilon}} g_t$$

Square root of sum of squares of historic gradients

AUG
22 23 24 25 26 27 28SEP
29 30 31

SATURDAY

RMS prop

↳ Divides learning rate by running average of squared gradients

$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1-\gamma) g_t^2$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$$

γ = decay parameter, set 0.9

η = learning rate, 0.001

SUNDAY 01

$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1-\gamma) g_t^2$$

Adam (Adaptive Momentum)

Stores running average of past squared gradients
¹⁰ v_t like adadelta & RMSprop.

$$\begin{aligned} \text{11} \quad m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t \\ \text{12} \quad v_t &= \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \end{aligned}$$

m_t = first moment (mean) of gradients

v_t = second moment (uncentered variance) of gradients

β_1, β_2 = decay rates.

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

Adam update rule:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t$$

SUNDAY 01

$$\begin{aligned} \text{m}_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t & \hat{m}_t &= \frac{m_t}{1 - \beta_1^t} \\ \text{v}_t &= \beta_2 v_{t-1} + (1-\beta_2) g_t^2 & \hat{v}_t &= \frac{v_t}{1 - \beta_2^t} \\ \theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t \end{aligned}$$

Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
30	31	$g_t = \nabla_{\theta_t} J(\theta_t)$ $\Delta \theta_t = -\eta g_t$ $\theta_t = \theta_{t-1} + \Delta \theta_t$				1
2	3	<u>Momentum</u>				8
4		$\Delta \theta_t = -\gamma v_{t-1} - \eta g_t$ <u>Adagrad</u> $\Delta \theta_t = \frac{-\eta}{\sqrt{G_t + \epsilon}} \cdot g_t$				
9	10	<u>Adadelta</u>	12	$\sqrt{G_t + \epsilon}$	14	15
					$G_t = \sum_{i=0}^t \text{diag}(g_i^T g_i)^2$	
16	17	<u>RMSprop</u>	19	20	21	22
			$\Delta \theta_t = \frac{-\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$			
23	24	<u>Adam</u>	25	26	27	28
			$\Delta \theta_t = \frac{-\eta}{\sqrt{\hat{v}_t + \epsilon}} \hat{m}_t$			29

Anniversaries

Birthdays
 $G_t = \sum_{i=0}^t \text{diag}(g_i^T g_i)^2$

03

DAY 215-150 | Wk 32

TUESDAY

M	T	W	T	F	S	S
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

JULY '21

M	T	W	T	F	S	S
30	31					
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29

AUGUST '21

Weight Initialization

unlike convex optimization, it matters where you start.

- 1) Initialize with zero → get stuck at big saddle pt.
- Initialize with constant value → difficult to break symmetry.
- Initialize with large values → off on the great plateaus, small gradient, slow converge.

Initialize the net of network with small Random values.

eg. zero mean Gaussian noise with constant variance

5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29

Natural Language Processing

1) Sentence segmentation:

Breaking ^{text} piece of sentence in to various sentences

2) Word tokenization:

Breaking sentence into individual words called tokens.

3) Predicting part of speech for each token:

Predicting whether the word is noun, verb, adverb, adjective, pronoun, preposition etc.

4) Lemmatization:

feeding the Model with root word.

5) Identifying stop words:

most frequently occurring words are stop words.

6) Dependency parsing:

finding out relation between words in a sentence.

7) Finding noun phrases:

grouping words that represent same idea

8) Name Entity Recognition:

NER maps words with the real world places.

	S	M	T	W	T	F	S
MAY '21	30	31		1			
	2	3	4	5	6	7	8
	9	10	11	12	13	14	15
	16	17	18	19	20	21	22
	23	24	25	26	27	28	29

	S	M	T	W	T	F	S
JUNE '21		1	2	3	4	5	
	6	7	8	9	10	11	12
	13	14	15	16	17	18	19
	20	21	22	23	24	25	26
	27	28	29	30			

Convolutional Neural Network

9

CNN are a class of Deep Neural Networks that can recognise and classify particular features from images and are widely used for analysing visual images.

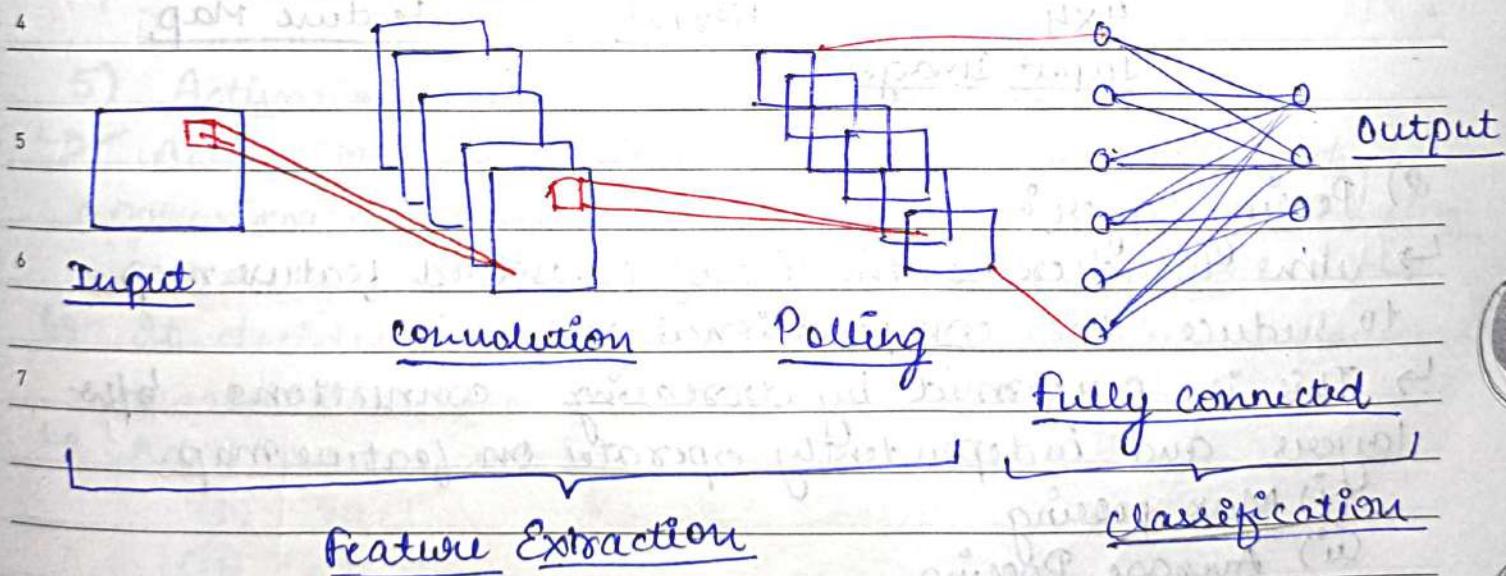
11

CNN Architecture:

12

- 1) A convolution tool that separates and identifies the various features of image for analysis in a process called Feature Extraction.
- 2) A fully connected layer that utilizes the O/P from Convolution process and predicts the class of image based on features extracted in previous stages.

4



$$21^{\text{st}} \text{ APRIL} \quad S \quad [i - k + 2p] + 1$$

$$[i - k + 2p] + 1 \quad 4 - 2$$

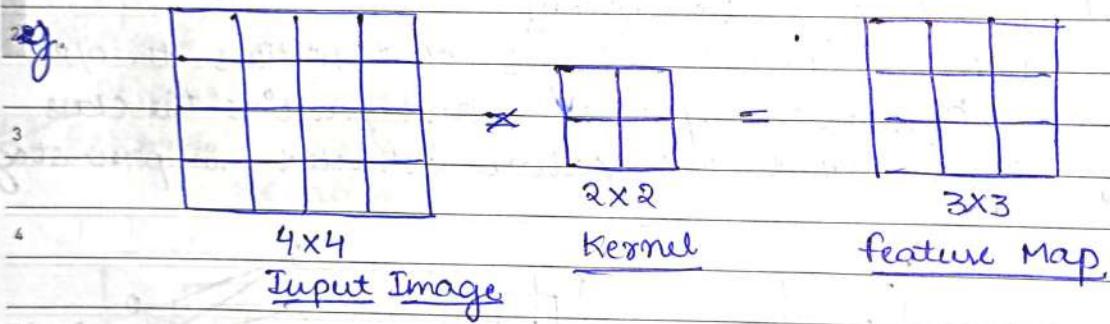
06

DAY 096-269 | Wk 15
TUESDAY

M	T	W	T	F	S	S		M	T	W	F	S	S	
1	2	3	4	5	6	7		1	2	3	4			
8	9	10	11	12	13	14		5	6	7	8	9	10	11
15	16	17	18	19	20	21		12	13	14	15	16	17	18
22	23	24	25	26	27	28		19	20	21	22	23	24	25
29	30	31						26	27	28	29	30		

1) Convolution Layer:

- ↳ used to extract various features from I/P image.
- ↳ Mathematical operation of convolution is applied b/w the input image and filter of particular size $M \times M$ called Kernel filter.
- ↳ By sliding kernel over input image, dot product is taken b/w the filter and parts of image.
- 12. Wrt to size of filter.
- ↳ The output is feature Map which gives info about img such as corners, edges etc.



2) Pooling Layer:

- ↳ aims to decrease the size of convolved feature map to reduce the computational cost.
- ↳ This is performed by decreasing connections b/w layers and it independently operates on feature map.
- (i) Max pooling
- (ii) Average Pooling

eg Max pooling:

19	2	6	47
9	6	13	2
0	7	12	4
0	21	33	3

filter (2x2)
= stride(2,2)

19	47
21	33

A strategy of rapid global growth require a different culture than one of focused innovation

3) Fully connected layer:

- ↳ consist of the nets and biases along with the neurons and is used to connect the neurons b/w different layers.
- ↳ These layers are usually placed before output layers and forms the last few layers of CNN Arch.
- ↳ It work on the flattened image fed to it from previous layers.

4) Dropout layer:

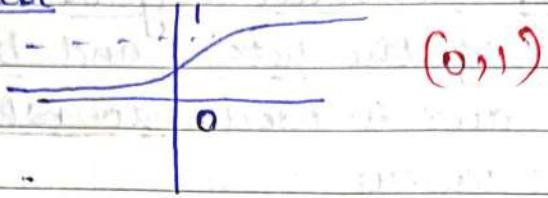
- ↳ When all the features are connected to FC layer, it can cause overfitting of training dataset.
- ↳ To overcome this, Dropout layer is used wherein few neurons are dropped from NN during training process resulting in reduced size of Model.

5) Activation Layer:

- ↳ Activation functions are used to learn and approximate any kind of continuous and complex relationship b/w the variables of Network.
- ↳ It decides which info. of model should fix in forward direction.
- ↳ Adds Non-linearity to Network.
 - (i) ReLU
 - (ii) softmax
 - (iii) tanh
 - (iv) Sigmoid function.

(i) Sigmoid function

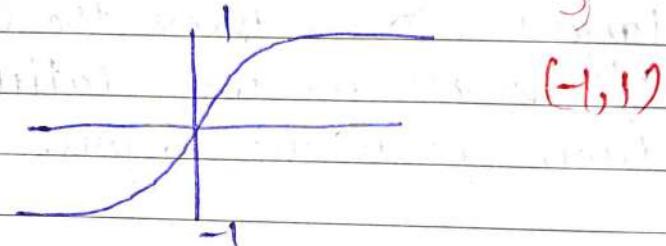
$$f(x) = \frac{1}{1 + e^{-x}}$$



(0, 1)

(ii) Tanh function

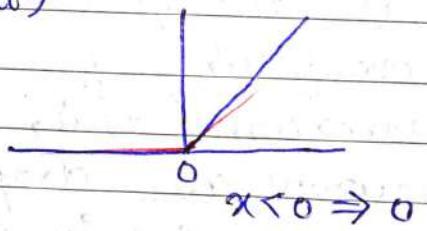
$$f(x) = \frac{x}{(1 + e^{-2x})^{-1}}$$



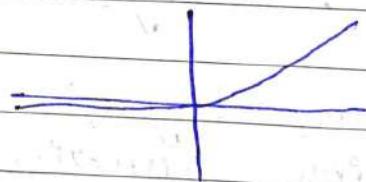
(-1, 1)

(iii) ReLU (Rectified linear unit)

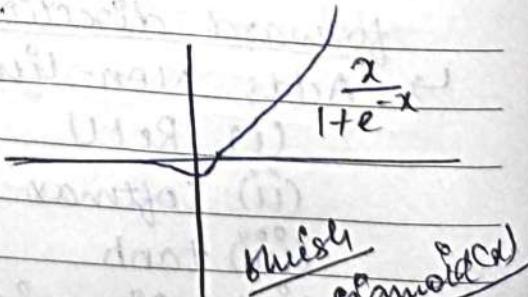
~~used in hidden layer only~~
~~max(0, x)~~

 $x < 0 \Rightarrow 0$ (iv) leaky ReLU

$$f(x) = \begin{cases} 0.01x & x < 0 \\ x & x \geq 0 \end{cases}$$

(v) softmax function : combination of multiple sigmoids.

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}} \quad \text{for } j=1 \dots K.$$



$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}} \quad j=1 \dots K$$

~~softmax~~
~~in sigmoid~~
~~form~~

Delta Rule:

↳ Delta Rule is a gradient descent learning rule for updating weights of I/P to Artificial Neurons in a single layer Neural Network.

↳ Special case for More general Backpropagation Algorithm.

$$\Delta w_i = -n \sum_{d \in D} (t_d - o_d) x_{id}$$

D = set of training examples

η = learning Rate

t_d = target output for training example d

o_d = Output of linear unit for training example d .

Delta Rule is Gradient Descent learning rule for updating weights of Input to Artificial Neural in a single layer Neural Network.

It is a special case for More generalized Backpropagation Algorithm

$$\Delta w_i = -\eta \sum_{d \in D} (t_d - o_d) x_{id}$$

D = set of training Data

η = learning Rate

$t_d \Rightarrow$ target output for training example d

$o_d \Rightarrow$ Output of linear unit of training example d

Delta Rule is derived by attempting to minimize

the error in O/P of neural Network using gradient descent

①

$$E[\vec{W}] = \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2$$

11

12 The direction of steepest can be found by computing derivative of E wrt each component of vector \vec{W}
1 The vector derivative is called gradient of E wrt \vec{W}

②

$$\nabla E[\vec{W}] = \left[\frac{\partial E}{\partial W_0}, \frac{\partial E}{\partial W_1}, \dots, \frac{\partial E}{\partial W_n} \right]$$

3

The gradient descent specifies the direction of steepest increase of E, the training rule for gradient descent is

5

$$\vec{W} \leftarrow \vec{W} + \Delta \vec{W}$$

6

$$\text{where, } \Delta \vec{W} = -\eta \nabla E(\vec{W})$$

6

$$\vec{W}_i \leftarrow \vec{W}_i + \Delta \vec{W}_i$$

$$\Rightarrow \Delta \vec{W}_i = -\eta \frac{\partial E}{\partial W_i}$$

} component form

7

④

Calculating gradient at each step. The vector of derivatives that form gradient can be obtained by differenting E

S	M	T	W	T	F	S
30	31	1	2	3	4	5
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

JUNE '21

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

Wk 17 | DAY 111-254
WEDNESDAY

21

$$\frac{\partial E}{\partial w_i} = \frac{\partial}{\partial w_i} \frac{1}{d} \sum (t_d - o_d)^2$$

$$= \frac{1}{d} \sum \frac{\partial}{\partial w_i} (t_d - o_d)^2$$

$$= \frac{1}{d} \sum \cancel{\partial} (t_d - o_d) \frac{\partial}{\partial w_i} (t_d - \vec{w} \cdot \vec{x}_d)$$

$$= \frac{1}{d} \sum (t_d - o_d) \frac{\partial}{\partial w_i} (t_d - \vec{w} \cdot \vec{x}_d)$$

$$\boxed{\frac{\partial E}{\partial w_i} = \sum_d (t_d - o_d) (-x_{i,d})} \quad \text{--- ②}$$

from ① & ②

$$\Delta w_i = -n \frac{\partial E}{\partial w_i}$$

$$\boxed{\Delta w_i = -\eta \sum_d (t_d - o_d) x_{i,d}}$$

MAY

JUN

21 APRIL

24

DAY 114-251 | Wk 17

SATURDAY

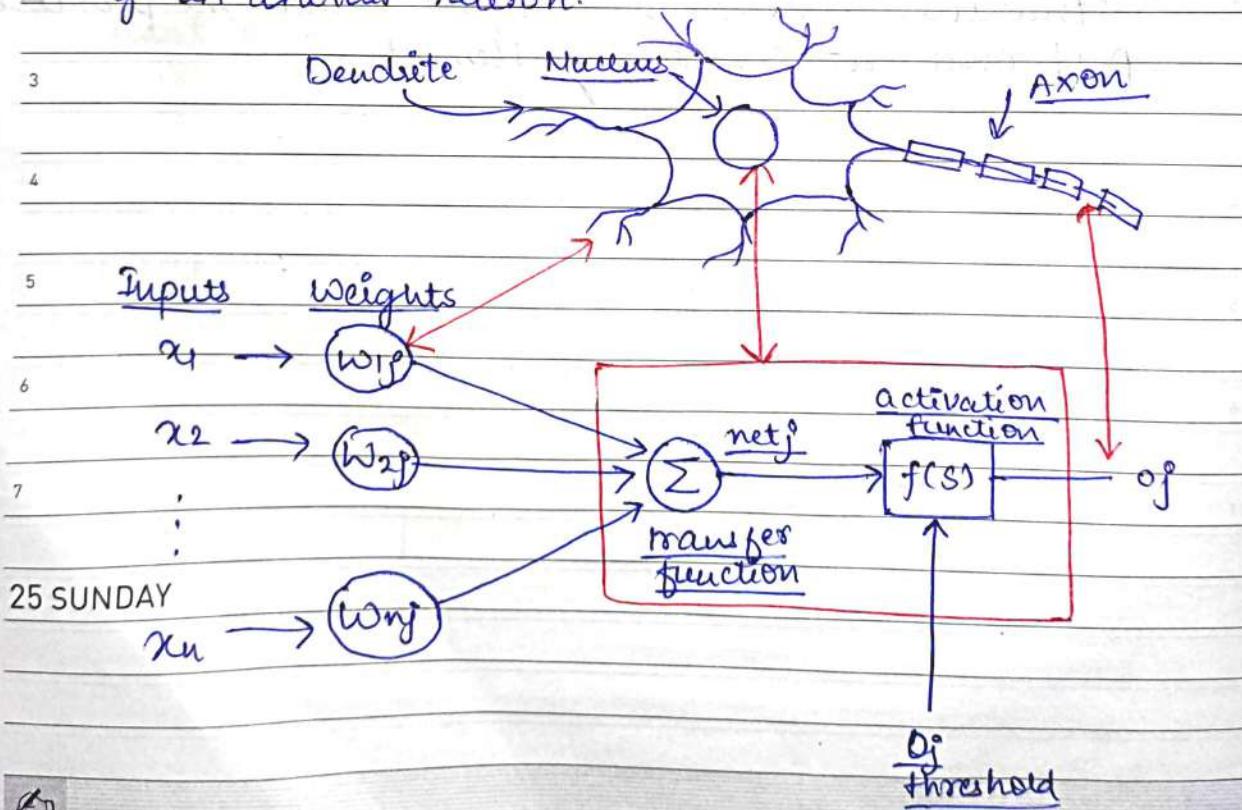
M	T	W	T	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

M	T	W	T	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

APRIL '21

Artificial Neuron is information processing system that has certain characteristics in common with biological Nets.

- 1) Processing elements receive many signals.
- 2) Signals can be modified by a weight at receiving synapse.
- 3) The processing element sum the weighted input
- 4) Under appropriate condition, the neuron transmits the output signal.
- 5) The output from one particular neuron may go in another neuron.



25 SUNDAY

There are two ways of exerting one's Strength: one is pushing down, the other is pulling up

S	M	T	W	T	F	S
30	31		1			
1	2	3	4	5	6	7
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29

JUNE '21

S	M	T	W	T	F	S
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

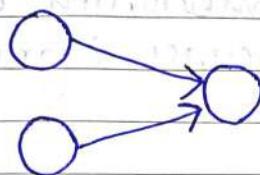
APRIL '21

Wk 18 | DAY 116-249

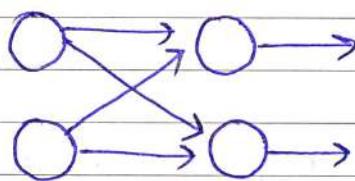
MONDAY

26

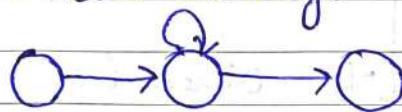
Single layer: One input layer, one output layer of processing unit. No feedback connection.
eg. perceptron.



Multi layers: One input layer, one output layer, and one or more hidden layers of processing unit.
No feedback connection.
eg. Multilayer perceptron.



Recurrent: A network with atleast one feedback connection. It may or maynot have hidden layers.
units. RNN have memory.
eg. RNN



21' APRIL

27

DAY 117-248 | Wk 18

TUESDAY

M	T	W	T	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

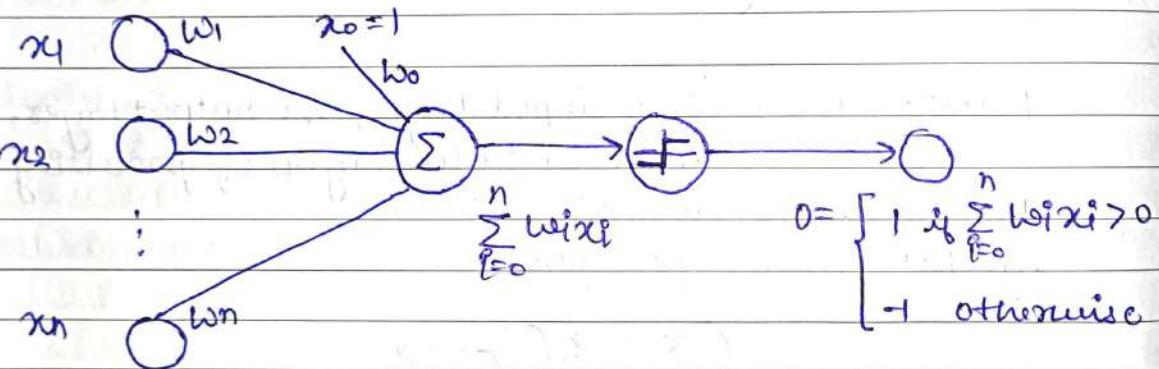
MARCH '21

M	T	W	T	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

APRIL '21

Perception : single layer Neural Network

Perception takes a vector of real valued inputs and calculates a linear combination of these I/P then output 1 if the result is greater than some threshold otherwise -1.



Given input x , Output $O(x_1, x_n)$ generated by perception.

$$O(x_1, \dots, x_n) = \begin{cases} 1 & \text{if } w_0 + w_1 x_1 + \dots + w_n x_n > 0 \\ -1 & \text{otherwise} \end{cases}$$

$$O(\vec{x}) = \text{sgn}(\vec{w} \cdot \vec{x})$$

w_i = real valued constant, that determines the contribution of input x_i to perception output.

w_0 is threshold that weighted combination of inputs must surpass in order for perception to O/P 1.

learning a perceptron involves choosing weights

w_0, w_1, \dots, w_n

∴ The space H of candidate hypotheses considered
in perceptron learning is set of all real-valued
weights vectors.

$$H = \{ \vec{w} \mid \vec{w} \in \mathbb{R}^{n+1} \}$$

21' MAY

	M	T	W	T	F	S	S
APRIL '21	31			1	2		
	3	4	5	6	7	8	9
	10	11	12	13	14	15	16
	17	18	19	20	21	22	23
	24	25	26	27	28	29	30

MAY '21

03

DAY 123-242 | Wk 19

MONDAY

Single Perceptron can represent Boolean function

AND

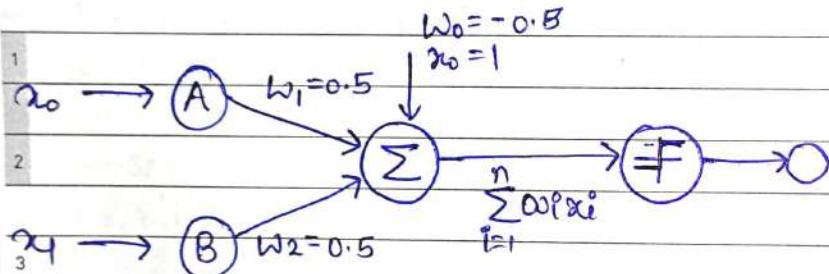
$$A \quad B \quad A \wedge B$$

$$0 \quad 0 \quad 0$$

$$0 \quad 1 \quad 0$$

$$\emptyset \quad 0 \quad 0$$

$$1 \quad 1 \quad 1$$



$$O(x_1, x_2) = \begin{cases} 1 & \text{if } w_0 + w_1 x_1 + w_2 x_2 > 0 \\ -1 & \text{otherwise} \end{cases}$$

(i) If $A(x_0)=0 \wedge B(x_1)=0$

$$-0.8 + (0.5)(0) + (0.5)(0) = 0 \quad \text{Output} = 0$$

(ii) If $x_0=0 \quad x_1=1 \Rightarrow -0.8 + (0.5)*1 + (0.5)*0 = -0.3 < 0$

$$\text{Output} = 0$$

(iii) If $x_0=1 \quad x_1=0 \Rightarrow -0.8 + (0.5)1 + (0.5)*0 = -0.3 < 0$

$$\text{Output} = 0$$

(iv) If $x_0=x_1=1 \Rightarrow -0.8 + 0.5x_1 + 0.5x_1 = 0.2 > 0$

$$\text{Output} = 1$$

S	M	T	W	T	F	S
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

S	M	T	W	T	F	S
					1	2
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

04

OR

$$x_0 \quad x_1 \quad \text{OR} \vee x_1$$

$$0 \quad 0 \quad 0$$

$$0 \quad 1 \quad 1$$

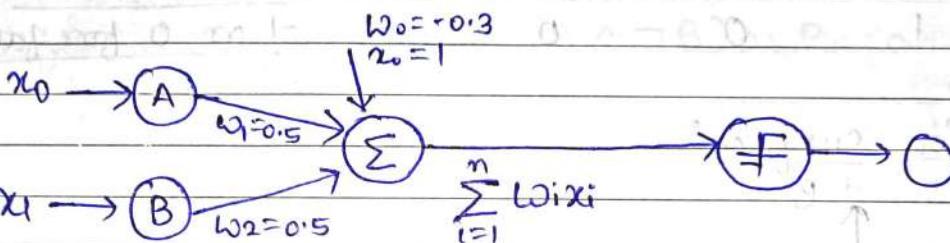
$$1 \quad 0 \quad 1$$

$$1 \quad 1 \quad 1$$

$$\text{let } w_0 = -0.3$$

$$w_1 = 0.5$$

$$w_2 = 0.5$$



(i) $A=0 \wedge B=0 \Rightarrow -0.3 + (0.5 \times 0) + (0.5 \times 0)$
 $-0.3 < 0 \quad \underline{\text{Output}} = 0$

(ii) $A=0 \wedge B=1 \Rightarrow -0.3 + (0.5 \times 0) + (0.5 \times 1)$
 $= 0.2 > 0 \quad \underline{\text{Output}} = 1$

(iii) $A=1 \wedge B=0 \Rightarrow -0.3 + (0.5 \times 1) + (0.5 \times 0)$
 $= 0.2 > 0 \quad \underline{\text{Output}} = 1$

(iv) $A=1 \wedge B=1 \Rightarrow -0.3 + (0.5 \times 1) + (0.5 \times 1)$
 $= 0.7 > 0 \quad \underline{\text{Output}} = 1$

21' MAY

05

DAY 125-240 | Wk 19

WEDNESDAY

M	T	W	T	F	S	S
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

APRIL '21

M	T	W	T	F	S	S
31					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

MAY '21

Ques. $A \wedge \neg B$ A B $\neg B$ $A \wedge \neg B$

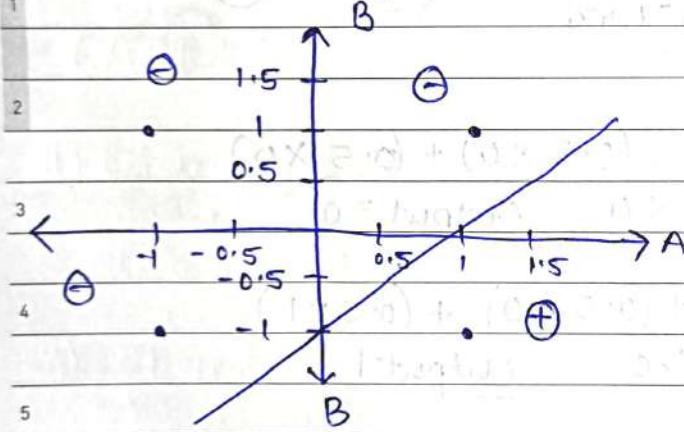
10 0 0 1 0

11 0 1 0 0

12 1 0 1 0

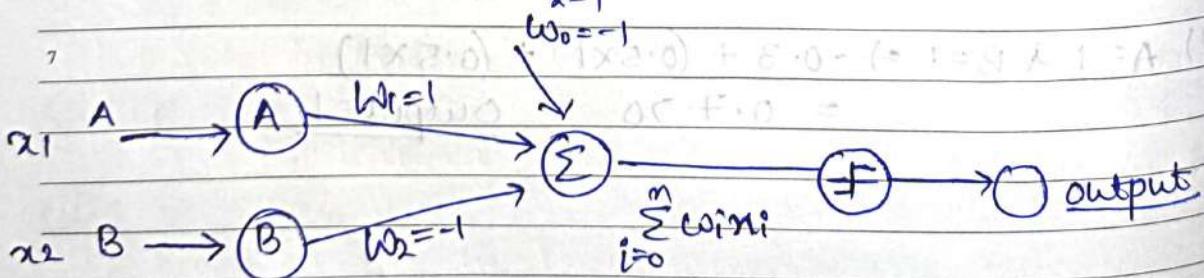
13 1 1 0 0

The values of
A and B are
1 (true) or
 $\neg 1$ or 0 for false

Decision Surface:

let Weights are
 $w_0 = -1$
 $w_1 = 1$
 $w_2 = -1$

The line crosses A axis at 1
B axis at -1



S	M	T	W	T	F	S
1	2	3	4	5		
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

S	M	T	W	T	F	S
1	2	3	4	5		
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

06

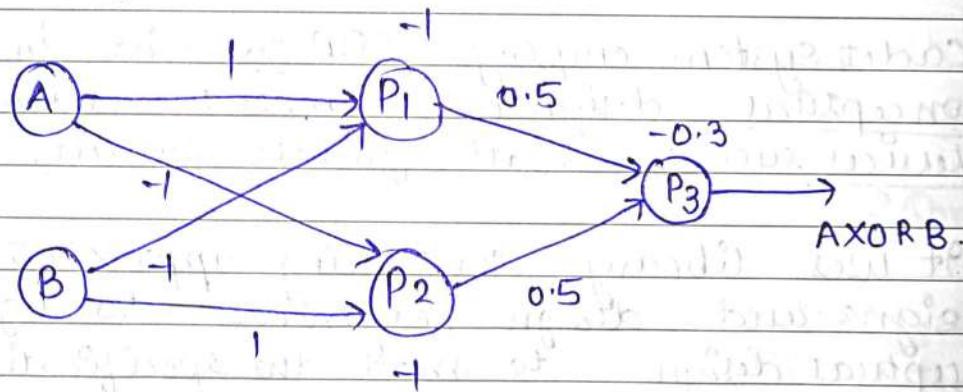
(ii) $A \oplus B$ cannot be calculated by single perceptron so we build a 2 layer network of perceptron.

Expressing $A \oplus B$ in terms of logical connections

$$A \oplus B = (A \wedge \neg B) \vee (\neg A \wedge B)$$

① Define perceptron for $(A \wedge \neg B)$ as P_1 and perceptron for $(\neg A \wedge B)$ as P_2 .

② Composing the output of P_1 and P_2 into P_3 that implements $O(P_1) \vee O(P_2)$.



APRIL '21
S M T W T F S
1 2 3
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30

MAY '21
S M T W T F S
30 31
1 2 3 4 5 6 7 8
9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29

23

Backpropagation Algorithm

- 1) The Backpropagation Algo. learns the weights for Multilayer Network, given a network with a fixed set of units and interconnections. It employs gradient descent to attempt to minimize the sq error b/w the network O/P values & the target values for those outputs.

- 2) In Backpropagation Algorithm, we consider networks with multiple O/P units rather than single units. The sum of errors of all the network O/P units

$$E(\vec{w}) = \frac{1}{2} \sum_{d \in D} \sum_{k \in O} (t_{kd} - o_{kd})^2$$

outputs = set of O/P units in the network

t_{kd} and o_{kd} = the target and O/P values associated with kth output unit.

d = training example.

3 phases of computation.

Forward Pass :

- 1) Run the NN and compute error for each neuron of the output layer.

Backward pass :

- start at the O/P layer and pass errors backward through the network, layer by layer, by recursively computing local gradient descent.

21' MARCH

24

DAY 083-282 | Wk 13

WEDNESDAY

M	T	W	T	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

FEBRUARY '21

MARCH '21

BACKPROPAGATION (training-examples; η , n_{in} , n_{out} , n_{hidden})

- 9) Each training example is pair of form (\vec{x}, \vec{t}) ,
 i) vector of network input values, \vec{t} vector of target network output values.

10) η = learning rate n_{hidden} = no. of hidden layer units.

11) n_{in} = no. of network inputs, n_{out} = no. of output units

x_{ji}^i = input from unit i into unit j

12) w_{ij}^i = weight from unit i to j .

- 1) ↳ Create feed forward Network with n_i , n_{hidden} , n_{out} .
- 2) ↳ Initialize all Network weights to small random no.
- 3) ↳ Until termination condition is met, Do
- 4) ↳ for each (\vec{x}, \vec{t}) in training examples, Do
- 5) ↳ Propagate IP forward through network
- 6) 1) Input \vec{x} , to network and compute O_u of every unit u in the network.
- 7) 2) Propagate errors backward through network.
- 8) 3) for each network output unit k calculate error
- 9) $\delta_k \leftarrow O_k (1 - O_k) (t_k - O_k)$
- 10) 3) for each hidden unit h , calculate error
- 11) $\delta_h \leftarrow O_h (1 - O_h) \sum_{k \text{ outputs}} w_{hk} \delta_k$ $w_{hk} \delta_k$
- 12) 4) Update each network weight w_{ji}^i
 $\Delta w_{ji}^i \leftarrow w_{ji}^i \leftarrow w_{ji}^i + \Delta w_{ji}^i$
 Where, $\Delta w_{ji}^i = \eta \delta_j x_{ij}$

MARCH '21
1 2 3 4 5 6
7 8 9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 31

APRIL '21
1 2 3
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30

FEBRUARY '21

Wk 07 | DAY 041-324
WEDNESDAY

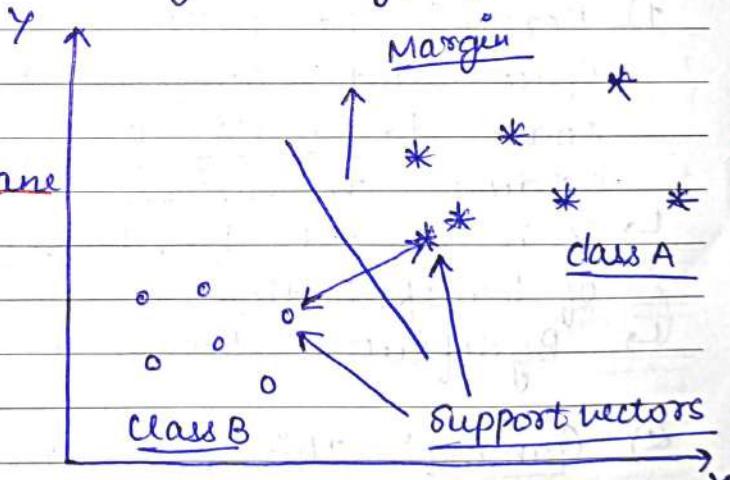
10

Support Vector Machine

- ↳ SVM is powerful and flexible supervised ML Algorithm used for classification and regression analysis.
- ↳ An SVM Model is basically representation of different classes in Hyperplane in multidimensional space.

- ↳ The hyperplane will be generated in iterative manner by SVM so that the error can be minimized.
- ↳ The goal of SVM is to divide the dataset into classes to find a maximum marginal Hyperplane (MMH)

- ↳ Support vectors:
Data points that are closest to the hyperplane Separating line will be defined using these points



- ↳ Hyperplane:
A decision plane or space which is divided by the set of objects of diff. classes.

- ↳ Margin: The gap b/w 2 lines on the closest datapts of different classes. It can be calculated as the L dist. from the line to support vectors.

21' FEBRUARY

11

DAY 042-323 | Wk 07

THURSDAY

	M	T	W	T	F	S	S		M	T	W	T	F	S	S	
JANUARY '21								JANUARY '21								FEBRUARY '21
									1	2	3			1	2	
	4	5	6	7	8	9	10							2	3	4
	11	12	13	14	15	16	17							5	6	7
	18	19	20	21	22	23	24							8	9	10
	25	26	27	28	29	30	31							11	12	13
														14		
														15	16	17
														18	19	20
														21		
														22	23	24
														25	26	27
														28		

Linear SVM:

- 9 It is used for linearly separable data, which means if a dataset can
- 10 be classified into 2 classes using single straight line then such data is linearly
- 11 separable data & classifier is linear SVM.

Non-linear SVM
It is used for non-linearly separable data, i.e. if a dataset cannot be classified into by using straight line then such data is non-linear data & classifier is non-linear SVM.

Parameters used in Support vector classifier:

- 1) Kernel: It takes lower dimensional input space and transform it into higher dimensional space to make problem separable by adding dimensions.
 - ↳ It is selected based on type of data and type of transformation.
 - ↳ By default, Kernel is RBF Kernel.

- 2) Gamma: This parameter determines how far the influence of single training example reaches during transformation, which in turn affects how tightly decision boundaries end up surrounding points in input space.
 - ↳ 'Y' increases, model gets overfit, 'Y' decreases model gets underfit.

MARCH '21
S M T W T F S
1 2 3 4 5 6
7 8 9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 31

APRIL '21
S M T W T F S
1 2 3
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30

12

3) c parameter: This parameter controls the amount of regularization applied on data.

↳ Large value of c = low regularization = cause overfitting.

↳ Lower value of c = Higher Regularization = cause underfitting (may lead to lower accuracy).

12 Types of Kernels:

1) Linear Kernel: $K(x, x_i) = \sum (x * x_i)$.

2) Polynomial Kernel:

$$K(x_i, x_j) = (x_i \cdot x_j + 1)^d$$

3) $d = \text{degree of polynomial}$
popular in Image processing.

4) Gaussian Kernel: $K(x, y) = \exp\left(\frac{-||x-y||^2}{2\sigma^2}\right)$

5) Gaussian RBF: $K(x_i, x_j) = \exp(-\gamma ||x_i - x_j||^2)$

6) $\gamma > 0$ good value 0.1

7) Laplace RBF: $K(x_i, y) = \exp\left(-\frac{||x_i - y||}{\sigma}\right)$

8) Hyperbolic Tangent Kernel: $K(x_i, x_j) = \tanh(K x_i \cdot x_j + c)$
 $K > 0$ (for some) & $c < 0$

9) Sigmoid Kernel: $K(x_i, y) = \tanh(\alpha x_i^T y + c)$

10) ANOVA radial basis kernel:

regression $K(x, y) = \sum_{k=1}^n \exp(-\sigma (x^k - y^k)^2)^d$

21' FEBRUARY

13

DAY 044-321 | WK 07

SATURDAY

M	T	W	T	F	S	S	M	T	W	T	F	S	S
JANUARY '21							FEBRUARY '21						
1	2	3	4	5	6	7	8	9	10	11	12	13	14
15	16	17	18	19	20	21	22	23	24	25	26	27	28
29	30	31	22	23	24	25	26	27	28	29	30	31	

AdvantagesDisadvantages

- 1) SVM is more effective in High dimensional space.
- 2) SVM can be used for linearly separable and non linearly separable data.
- 3) SVM is relatively memory efficient.
- 4) Guaranteed optimality
Owing to nature of convex Optimization , the soln will always be Global Min.
- 5) SVM can carryout feature mapping using simple dot product using kernel Trick.
- 6) Not suitable for large datasets.
- 7) SVM doesn't perform well when data set has noise target classes are overlapping.
- 8) SVM doesnot give best performance for handling text structures.
- 9) The choice of kernel is perhaps the biggest limitation of SVM.
- 10) In case where no. of features for each datapoint exceeds the no. of training data samples SVM will under perform.

14 SUNDAY

- ⑤ ① More efficient in HDS
 ② Memory efficient
 ③ Guaranteed optimality

- ④ feature mapping using simple dot product ..

1	2	3	4	5	6
7	8	9	10	11	12
14	15	16	17	18	19
21	22	23	24	25	26
28	29	30	31		

APRIL '21	4	5	6	7	8	9	10
	11	12	13	14	15	16	17
	18	19	20	21	22	23	24
	25	26	27	28	29	30	

Properties of SVM

- 1) Flexibility in choosing similarity function.
- 2) Sparcity of solution when dealing with large datasets.
only support vectors are used to specify the separating hyperplane.
- 3) Ability to handle large feature space.
Complexity does not depend on dimensionality of feature space.
- 4) Overfitting can be controlled by softmargin approach.
- 5) A simple convex optimization problem which is guaranteed to converge to single global soln.
- 6) Feature selection.

REGRESSION

Regression Analysis or Regression is a statistical method and a predictive Modelling Technique which investigates the relation between one dependent and a series of other independent variables.

e.g. Relationship b/w rash driving & no. of road accidents by a driver is best studied through regression

Advantages:

- 1) It indicates the significant relationship b/w dependent and independent variables
- 2) It indicates the strength of Impact of Multiple Independent variables on dependent Variables.

21' JANUARY

22

DAY 022-343 | Wk 04

FRIDAY

M	T	W	T	F	S	S
1	2	3	4	5	6	
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

M	T	W	T	F	S	S
1	2	3				
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

JANUARY '21

linear Regression A supervised ML algorithm where the predicted output is continuous and has a constant slope.

Simple Regression

$$y = mx + b$$

Multivariable Regression

$$f(x, y, z) = w_1x + w_2y + w_3z$$

Logistic Regression: A supervised learning classification algorithm used to predict the probability of target variable.

Binary (Binomial) Regression

O/P: 1/0

Multinomial Regression

O/P 'Type A'
Type 'B'
Type 'C'

Ordinal Regression

Types having Quantitative significance
O/P 'Poor' / 'Good'

$$\log \left[\frac{y}{1-y} \right] = b_0 + b_1 x_1 + b_2 x_2 + \dots + b_n x_n$$

S	M	T	W	T	F	S
1	2	3	4	5	6	
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28						

MARCH '21

S	M	T	W	T	F	S
2	3	4	5	6		
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

straight
line

IV

DV

DV

JANUARY '21

Wk 04 | DAY 023-342
SATURDAY23
IVParameterLinear
RegressionLogistic
Regression1) ModelSupervised Regression
ModelSupervised Classification
Model.2) UsedTo predict values with
in a constant rangeTo predict the
probability of a
new target variable.3) Activation
FunctionNo activation function
is usedActivation function is
used to convert linear
regression to logistic
regression4) Threshold
value

Threshold value is added

No threshold is needed

5) PurposeTo find best-fitted
lineTo fit the line values
to sigmoid curve.6) Loss Function
MethodMean square Error
MethodMaximum likelihood
Estimation.7) OutputContinuous ValuesCategorical Values8) Collinearitycollinearity b/w the
Independent variablesNo collinearity b/w
independent variables9) MUATPLOCGaussian / Normal
distribution of the
dependent var.Binomial Distribution
of dependent var.