Date :
Page No.

CTO Prep

Resume - Projects

- NLMTK
  - 1 DAE
  - 2 Mean
  - 3 Seq2point
  - 4 Seq2Seq
  - 5 dsc (~~Deep sparse coding~~) Discriminative sparse coding
  - 6 WindowGRU
  - 7 RNN
  - 8 Zero
  - 9 Markov
- CNN
  - Alexnet Architect
- Chatbot
  - RNN
  - GRU
  - LSTM

SVD	Eigen ~~Vector~~ Value Decomp.	PCA

$A\vec{V} = \lambda\vec{V}$ Arbitrary Real #

Eigen vectors as columns

$$AV = A[\vec{V_1}\ \vec{V_2}]$$

$$= [\lambda_1\vec{V_1}\ \ \lambda_2\vec{V_2}]$$

$$= [\vec{V_1}\ \ \vec{V_2}]\begin{bmatrix}\lambda_1 & 0\\ 0 & \lambda_2\end{bmatrix}$$

$$\begin{bmatrix}\lambda_1 & 0 & 0\\ 0 & \lambda_2 & 0\\ 0 & 0 & \lambda_3\end{bmatrix}$$

$$= V\Lambda$$

$$\Rightarrow \boxed{A = V\Lambda V^{-1}}$$

$VV^T = I$

$\boxed{V^{-1} = V^T}$ orthogonal

Scanned by CamScanner

Date :
Page No.

Orthogonal matrix

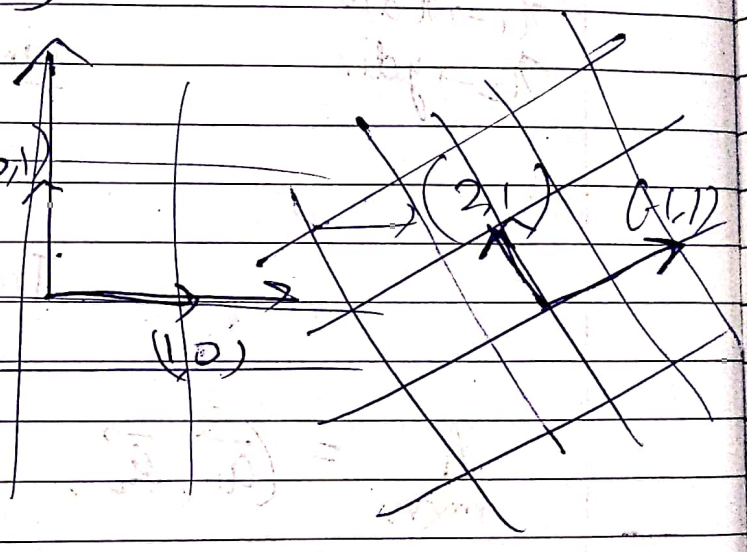
= Square matrix whose columns & rows are orthogonal unit vectors

$$Q^T Q = Q Q^T = I \qquad Q^T = Q$$

Let $Q = \begin{bmatrix} \cos x & \sin x \\ -\sin x & \cos x \end{bmatrix}$ $\qquad \cos x \sin x - \sin x \cos x$

point 1 point 2

$$\begin{bmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} & \begin{pmatrix} 0 \\ 1 \end{pmatrix} \end{bmatrix}$$



Transformation ↓ Rotation + ~~Reflection~~ ~~Linear transf.~~

$$\begin{bmatrix} 2 & -1 \\ 1 & 1 \end{bmatrix}$$

$(2,1) \cdot (-1,1)$

$-2 + 1 = -1 \quad \vec{A} \cdot \vec{B} =$

$$\begin{bmatrix} 1 & 2 \\ -2 & 1 \end{bmatrix}$$

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = x_1 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + x_2 \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \cancel{[x_1 \ x_2]} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$= \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

↓

Any 2D point

Scanned by CamScanner

Standard Basis Vectors $\begin{pmatrix}1\\0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix}$

Date :
Page No.

1 vector = linear combination of other vectors

$$T\begin{pmatrix}x_1\\x_2\end{pmatrix} = x_1\begin{pmatrix}2\\1\end{pmatrix} + x_2\begin{pmatrix}-1\\1\end{pmatrix} = \begin{pmatrix}2 & -1\\1 & 1\end{pmatrix}\begin{pmatrix}x_1\\x_2\end{pmatrix}$$

T → Linear Transformation

$\begin{pmatrix}2\\1\end{pmatrix}$ → Move point
$x_1 \to \times 2$ x dir
$\to \times 1$ y dir

$\begin{pmatrix}2 & -1\\1 & 1\end{pmatrix}$ → Linear Transf.

$x_1 \to$ x dir
$x_2 \to$ y dir

Linear Transf.

$$A = B\vec{X}$$

A → Transformed
B, $\vec{X}$ → Original basis vectors

$$\underset{m\times n}{A} = (\vec{a_1}, \vec{a_2}, \dots \vec{a_n})$$

$$\underset{n\times p}{B} = (\vec{b_1}, \vec{b_2}, \dots \vec{b_p})$$

$$AB = (A\vec{b_1}, A\vec{b_2} \dots A\vec{b_p})$$

↓

Linear combination

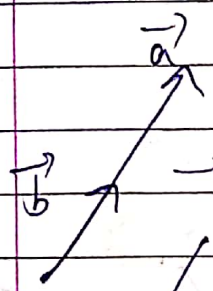
$$\vec{Y} = (AB)\vec{X}$$

Scanned by CamScanner

Linear Combination of 2 basis Vectors
= 2D space → SPAN of vectors

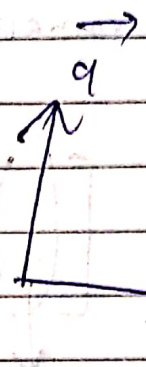
$a\vec{v} + b\vec{w}$
scalars

$a\vec{v} + b\vec{w} + c\vec{u}$ → 3D space span



→ linearly dependent
↳ Not adding new axis.

$\vec{b} = k\vec{a}$ ✓



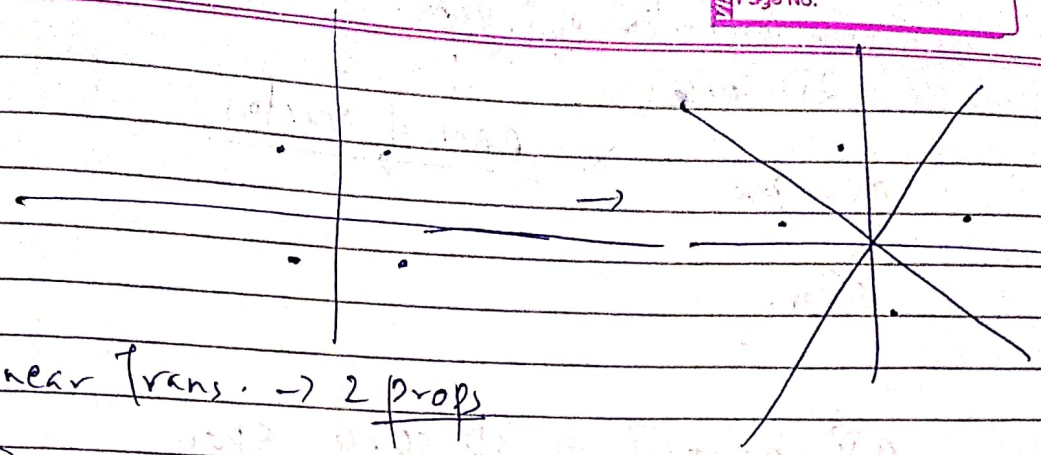
linearly independent
→ Adding new axis

$\vec{b} \neq k\vec{a}$ ✗

→ Linear Transformation

$T(\vec{A}) = \vec{B}$ → A point in space moves to other point just
↳ just a function
i/p = vector ✓

Ex



Linear Trans. → 2 props

1) All lines must remain lines (w/t getting curved)
2) Origin remains fixed

$$T\begin{pmatrix} x \\ y \end{pmatrix} = \begin{bmatrix} \begin{pmatrix} \\ \\ \end{pmatrix} & \begin{pmatrix} \\ \\ \end{pmatrix} \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

Lines lands of $\hat{i}$ & $\hat{j}$ lands $\hat{i}$ $\hat{j}$

Given pt

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \rightarrow \begin{array}{cc} \hat{i} & \hat{j} \\ \end{array} \begin{bmatrix} a & b \\ c & d \end{bmatrix} \qquad \begin{bmatrix} x \\ y \end{bmatrix} = \vec{v}$$

(Initial)

First basis vector coor after transf

Second Basis Vect after T

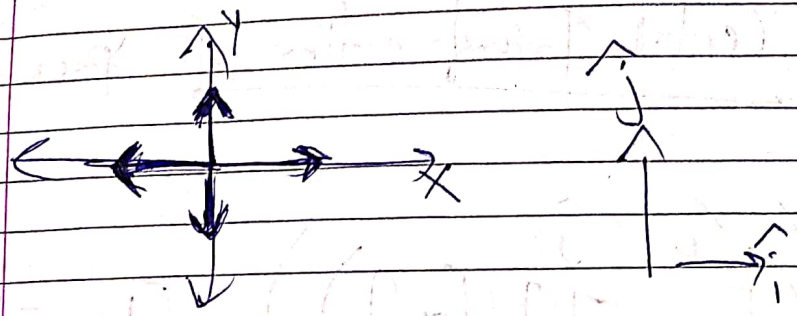
$\vec{v}(\hat{i} + \hat{j})$

$$x\begin{bmatrix} a \\ c \end{bmatrix} + y\begin{bmatrix} b \\ d \end{bmatrix}$$

$$= \begin{bmatrix} ax + by \\ cx + dy \end{bmatrix}$$

Visualize

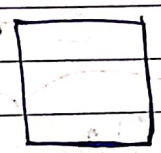
$$\begin{array}{cc} x & y \end{array}$$
$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \xrightarrow[\text{rotation}]{90^\circ} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \xrightarrow{90^\circ} \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$$



→ Shear transform

$\hat{i}$ → remains same

$\hat{j}$ → changes (~~scaled~~) transforms



$$\begin{array}{cc} \hat{i} & \hat{j} \end{array}$$
$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

→ Imagine this transformation

$$\begin{array}{cc} \hat{i} & \hat{j} \end{array}$$
$$\begin{bmatrix} 1 & 3 \\ 2 & 1 \end{bmatrix}$$

Actually transforms the axis

$$\begin{array}{cc} \hat{i} & \hat{j} \end{array}$$
$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Move $\hat{i}$ → to (1,2)

Then move $\hat{j}$ to (3,1)

Just keep grid lines parallel & evenly spaced

+ Origin remains fixed

If $\begin{bmatrix} 2 & -2 \\ 1 & -1 \end{bmatrix} \rightarrow$ linearly dependent

$\Rightarrow$ span = $\perp$ line

Matrix = Certain Transformation of Space

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}\left(\left(\begin{array}{cc} \hat{i} & \hat{j} \\ \end{array}\right.\right.$$

$$\left.\left.\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}\right)\right) = \begin{bmatrix} 1 & -1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}$$

$\leftarrow$ Rotation $\rightarrow$

Shear

Combination of transfor

Composition Matrix

Matrix Multip. = Transformation or (Compi -)

$M_1 M_2 \rightarrow$ Meaning Applying M_1 transformation & then M_2 transformation

$\underline{M_1 \cdot M_2}$

$\leftarrow$ rotation

then Shear

Right $\rightarrow$ Left Read transformation Order

Date :
Page No.

$$\begin{bmatrix} 0 & 2 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} 1 & -2 \\ 1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 2 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \begin{bmatrix} 0 & 2 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} -2 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 2 \\ 1 \end{bmatrix} \qquad \begin{bmatrix} 0 \\ -2 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 0 \\ 1 & -2 \end{bmatrix}$$

$$\begin{array}{ccc} \hat{i} & \hat{j} & \hat{k} \\ \end{array}$$
$$\begin{bmatrix} a & d & g \\ b & e & h \\ c & f & i \end{bmatrix} \rightarrow \text{3D Transformation of point} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$
or axis

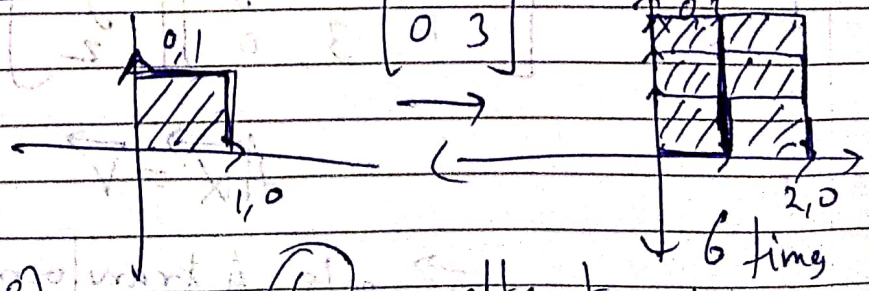
→ 3D Rotation, shear Transform → Computer Vision & Robotics.

$$\left| \begin{bmatrix} a & b \\ c & d \end{bmatrix} \right| = ad - bc$$

Determinant → New Area changed.

Ex

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$



$$\begin{array}{cc} \hat{i} & \hat{j} \\ \end{array}$$
$$\begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}$$

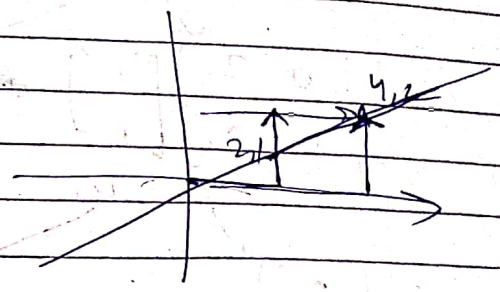
$$\begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix} = 2 \times 3 = 6$$ after trans

Scanned by CamScanner

$$\det \begin{array}{c} \begin{array}{cc} \hat{i} & \hat{j} \end{array} \\ \begin{bmatrix} 4 & 2 \\ 2 & 1 \end{bmatrix} \end{array} = 0 \rightarrow \underline{0 \cdot A} \text{ original area}$$

linearly dependent

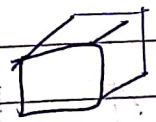
line span



$$\det(A) = \underline{-20}$$

→ Inverse orient. $(A \cdot 20)$ times

3Dimension → Volume Scales

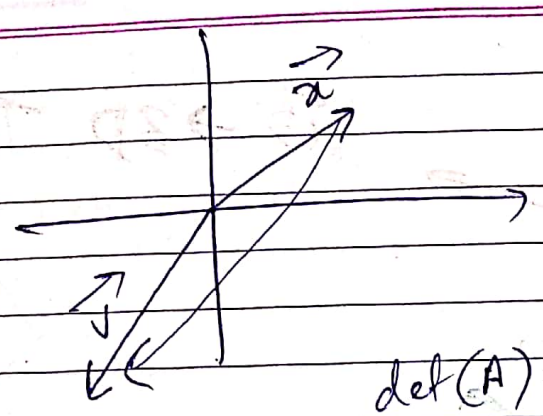


→ Linear System of Eq^{ns}

$$\begin{array}{c} \begin{array}{ccc} \hat{i} & \hat{j} & \hat{k} \end{array} \\ \begin{bmatrix} 2 & 5 & 3 \\ 4 & 0 & 8 \\ 1 & 3 & 0 \end{bmatrix} \end{array} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -3 \\ 0 \\ 2 \end{bmatrix}$$

$$A\vec{x} = \vec{v}$$

$\vec{x}$ gets A transformation to become $\vec{v}$



$\underline{\det(A) \neq 0}$

Only one & only 1 vector which lands on $\vec{v}$

<u>Ex of Inverse</u>

$AA^{-1} = I$

$\downarrow$

$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \rightarrow$ <u>Reverse Transformation</u>

$\boxed{\vec{x} = A^{-1}\vec{v}}$

Inverse Transformation exists if $\det(A) \neq 0$

If $\det(A) = 0 \longrightarrow$ Transformation squishes to lower dimension

$\rightarrow$ no inverse

Scanned by CamScanner

Non square Matrix

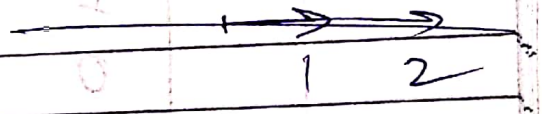
$$\begin{array}{cc} \hat{i} & \hat{j} \\ \end{array}$$
$$\begin{bmatrix} 2 & 0 \\ -1 & 1 \\ -2 & 1 \end{bmatrix} \rightarrow \quad 2D \rightarrow 3D \text{ Transformation}$$

3×2

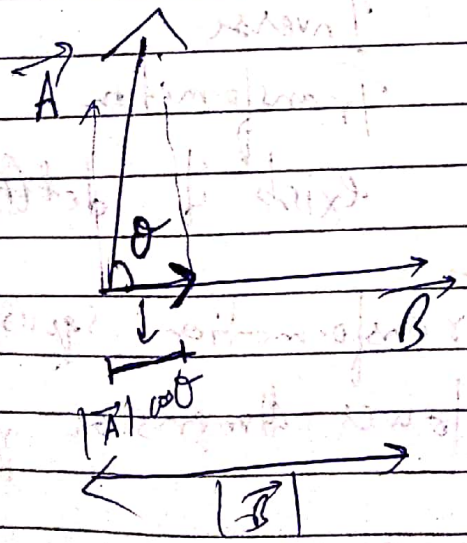
$$\begin{array}{ccc} \hat{i} & \hat{j} & \hat{k} \\ \end{array}$$
$$\begin{bmatrix} 3 & 1 & 4 \\ 1 & 5 & 9 \end{bmatrix} \quad 3D \rightarrow 2D$$

2×3

$$\begin{array}{cc} \hat{i} & \hat{j} \\ \end{array}$$
$$\begin{bmatrix} 1 & 2 \end{bmatrix} \rightarrow 2D \rightarrow 1D$$



$$\vec{A} \cdot \vec{B} = |\vec{A}| \cdot |\vec{B}| \cos\theta$$

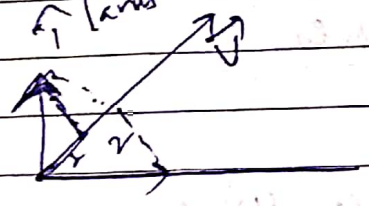


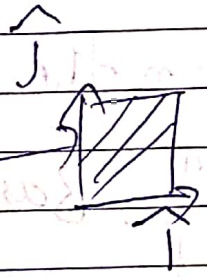
Scanned by CamScanner

$$\begin{array}{cc} \hat{i} & \hat{j} \\ \end{array}$$
$$\begin{bmatrix} 1 & 2 \end{bmatrix}$$

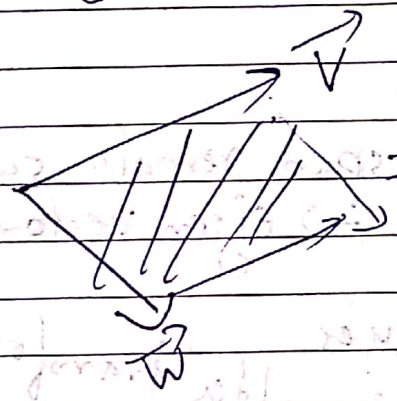
where $\hat{i}$ lands

where $\hat{j}$ lands.





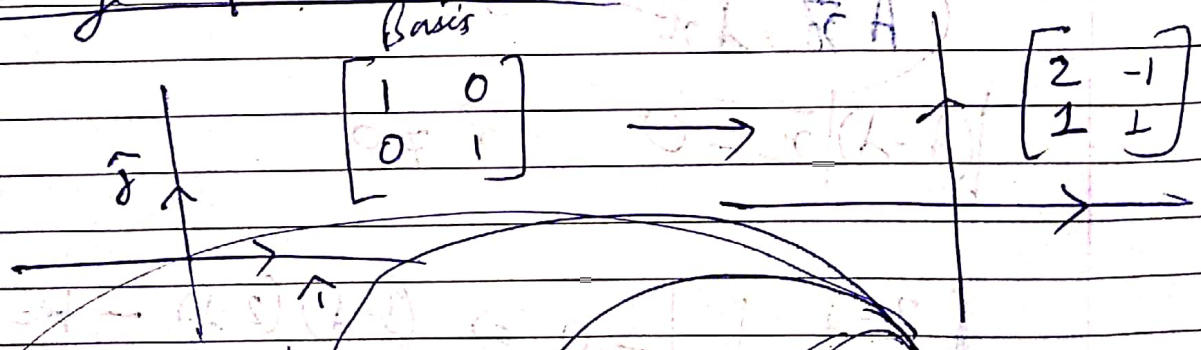
$\hat{i} \times \hat{j} = 1$



$= \vec{v} \times \vec{w}$

→ <u>Change of basis vectors</u>

Basis $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ $\longrightarrow$ Basis $\begin{bmatrix} 2 & -1 \\ 1 & 1 \end{bmatrix}$



$$\begin{bmatrix} 2 & -1 \\ 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 2 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 2 \end{bmatrix}$$

her language

vector in 2nd system

vector in 1st system

vector in 2nd system

Vector in 1st system

Date :
Page No.

$\boxed{A^{-1}KA}$ just

→ Eigen vectors & Eigen values.

Let transform the space using

$$\begin{bmatrix} 3 & 1 \\ 0 & 2 \end{bmatrix}$$ Basis vectors.

Those vectors whose span remain same even after transformation ⇒ eigen vectors

Scaled → eigen values.

Eg.

$A\vec{x} = \lambda\vec{x}$ → vector after transformation just gets scaled by λ

$|(A-\lambda)|\vec{x} = 0 \qquad \vec{x} \neq 0$

$$\begin{vmatrix} 3-\lambda & 1 \\ 0 & 2-\lambda \end{vmatrix} = 0 \Rightarrow (3-\lambda)(2-\lambda) - 1 \cdot 0 = 0$$

$$\Rightarrow (3-\lambda)(2-\lambda) = 0$$

$$\Rightarrow \underline{\lambda = 2, 3} \text{ (scaling)}$$

Eigen vectors → $\begin{bmatrix} 3 & 1 \\ 0 & 2 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = 2\begin{bmatrix} x \\ y \end{bmatrix}$

Scanned by CamScanner

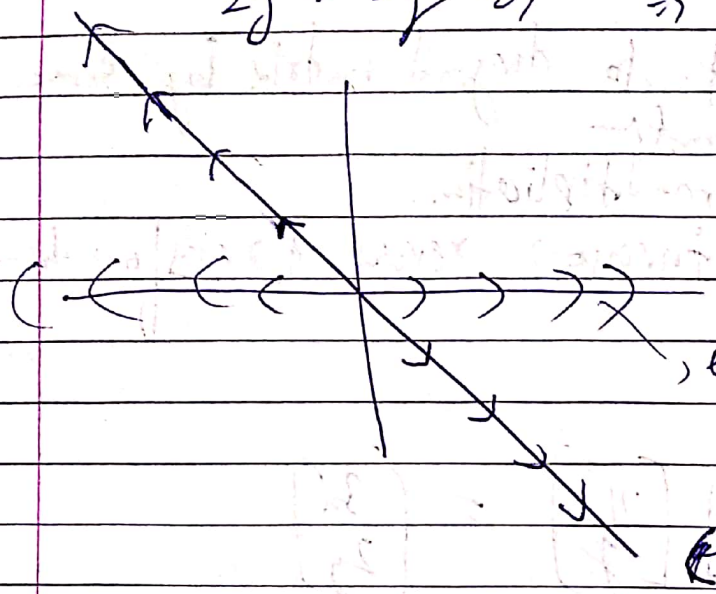
Date :
Page No.

$3x + y = 2x \Rightarrow x = -y \Rightarrow x + y = 0$

$2y = 2y \Rightarrow \quad \Rightarrow y = y$

$\begin{bmatrix} -1 \\ 1 \end{bmatrix}$

$y = -x$



, eigen vector

$$\begin{bmatrix} 3 & 1 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 3 \begin{bmatrix} x \\ y \end{bmatrix}$$

$\begin{bmatrix} \cancel{1} \\ 0 \end{bmatrix}$

$\Rightarrow \quad 3x + y = 3x \Rightarrow y = 0$

$2y = 3y \Rightarrow x =$ any real values

$\Rightarrow$ X-axis

Why Eigenvalues / vectors?

Eigen Basis

Important Property

$$\begin{bmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 4 \end{bmatrix}$$

$\rightarrow$ Eigen value = diagonal elements

Scanned by CamScanner

Date :
Page No.

make calculations easy

So idea is multiply

A → convert it to diagonal matrix by some transformation
→ Computed multiplication.
→ Compute inverse → reverse transformation

Ex.

$$\begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 3x \\ 2y \end{bmatrix}$$

$$\underbrace{\begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix}}_{A}^{3} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 3^3 x \\ 2^3 y \end{bmatrix}$$

Ex $\begin{bmatrix} 3 & 1 \\ 0 & 2 \end{bmatrix}$ → eigen vectors → Chose eigen vectors as basis vectors.

↓

C $\begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}$ ← Change of Basis Matrix

2 eigen vectors

This in C as basis

$$C^{-1} A C = \begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix}$$

$$\begin{bmatrix} 3 & 1 \\ 0 & 2 \end{bmatrix}^{100} \longrightarrow \begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix}^{100} \xrightarrow{\text{Convert back to standard system}}$$

Scanned by CamScanner

Formal definition of linearity

Additivity $L(\vec{v}+\vec{w}) = L(\vec{v}) + L(\vec{w})$

Scaling: $L(\vec{v}\cdot k) = k\, L(\vec{v})$

Derivative is linear

$$\frac{d}{dx}(f+g) = \frac{df}{dx} + \frac{dg}{dx}$$

~~rotation Stretch~~ Rotation

SVD

$$A = U \Sigma V^T$$

A → Any Matrix (May be a rectangular matrix)

U → orthogonal Matrix

Σ → Diagonal Matrix

$$\underset{m\times n}{A} = \underset{m\times m}{\begin{bmatrix} u_1 & u_2 \end{bmatrix}} \underset{m\times n}{\begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix}} \underset{n\times n}{\begin{bmatrix} v_1^T \\ v_2^T \end{bmatrix}}$$

σ_1, σ_2 → Singular values.

Scanned by CamScanner

Date :
Page No.

$$A^TA = (V\Sigma^T U^T)(U\Sigma V^T)$$

$U^TU = I$ since orthogonal

$$A^TA = V(\Sigma^T\Sigma)V^T$$

→ Positive Semidefinite Matrix

→ Symmetric

→ Eigen vectors will be orthogonal

Square matrix

λ for A^TA

Eigen vectors are σ^2 for A

Eigen vectors for A^TA

$$AA^T = U\Sigma V^T V\Sigma^T U^T$$

$$= U(\Sigma\Sigma^T)U^T$$

Eigen vector for AA^T

$$\boxed{A = U\Sigma V^T}$$

Scanned by CamScanner

Date :
Page No.

Eg $A = \begin{bmatrix} 2 & 2 \\ +1 & 1 \end{bmatrix}$ → But singular ⇒ Eigen val = 0

Not symmetric

$$A = \underbrace{\begin{bmatrix} 2 & +1 \\ -1 & 2 \end{bmatrix}}_{\text{rotate}} \underbrace{\begin{bmatrix} \sqrt{10} & 0 \\ 0 & 0 \end{bmatrix}}_{\text{stretch}} \underbrace{\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}}_{\text{rotate}}$$

$\sqrt{5}$ $\quad \Sigma \quad$ $\sqrt{2}$

Application: $A \rightarrow$ Combination of transformation

Eigen values → most information

$(U, \sigma, V^T) \rightarrow$ most info

Eigen person

Suppose:

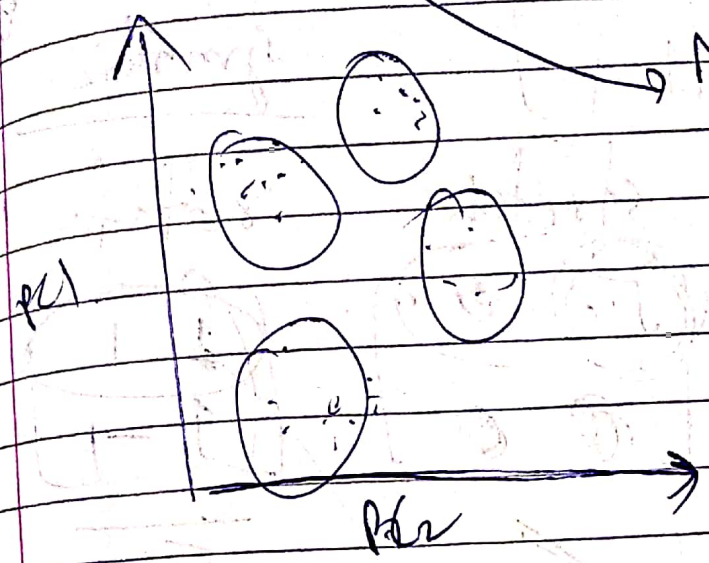
Person	P1	P2	P3
Genes.			
G_1			
G_2			
G_3			
G_4			

Most Information → PCA

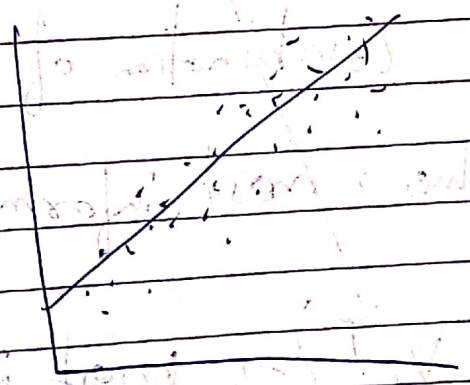
Scanned by CamScanner

PCA → Dimensionality Reduction

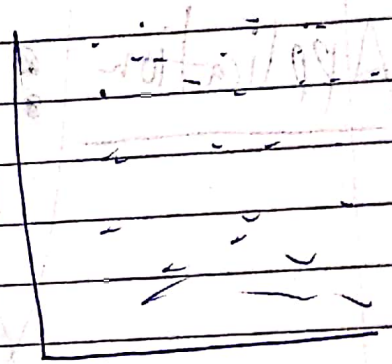
Method of ~~capturing~~ compressing a lot of data to something that captures the essence of original data



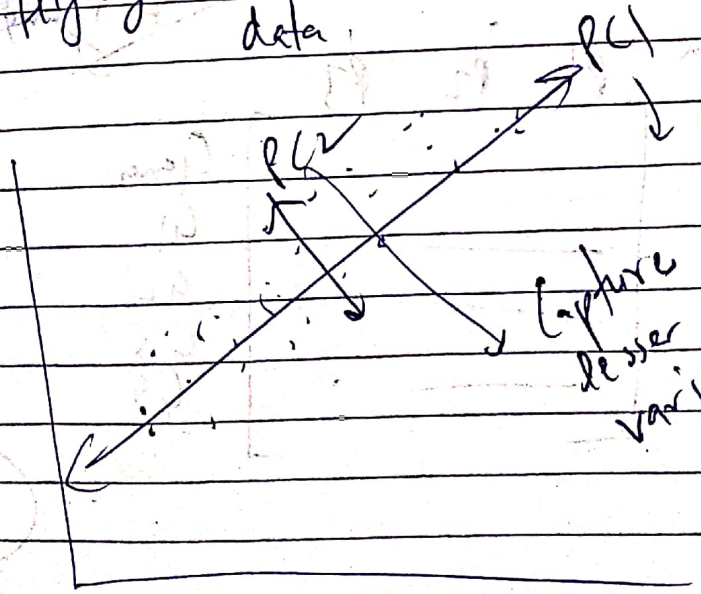
100 Genes in 1 cell → 2 PC's work



Highly co-related data



Not Corelated



Scanned by CamScanner

= Varian
auto - covar. = dispersion matrix

Date :
Page No.

Covariance Matrix

= Var- Cov.

$= E[(X-\bar{X})(Y-\bar{Y})]$

$$\boxed{Cov(X,Y) = \frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{X})(y_i-\bar{Y})}$$

Variance

$= E[(X-\mu)^2]$

$\mu = E[X]$

$X \rightarrow$ Random Var.
$Y \rightarrow$

Ex.

	X	Y	Z
1	1	1	1
2	1	2	1
3	1	3	2
4	1	4	3
	1	2.5	1.75

Var Cov.

	X	Y	Z
X			
Y			
Z			

Every var-covariance matrix = symmetric & positive semi difinite

$Cov(x_1, x_1) = Var(x_1)$

	x_1	x_2	x_3	x_4
x_1	$Var(x_1)$	$Cov(x_1,x_2)$	$Cov(x_1,x_3)$	-
x_2		$Var(x_2)$	- - -	
x_3			- - -	
x_4				

$r_{xy} \rightarrow$ Correlation $= \dfrac{Cov(x,y)}{S_x S_y}$

S_x^2 = variance

$S_x \rightarrow$ standard Dev. of x

Scanned by CamScanner

Ex. 42×13 Feature Vector $\rightarrow$ 13×1

$A = U \Sigma V^T$

Σ = Var Cov. Matrix (Diagonal Matrix)

U, V = Orthogonal Matrix

$UU^T = U^TU = VV^T = V^TV = I$

$V^{-1} = V^T$ orthogon

$$\Sigma = \frac{1}{13} \underbrace{(A^TA)}_{13\times42 \;\; 42\times13} \rightarrow 13\times13$$

Now, using SVD

$$\left\{ \begin{array}{l} A^TA = V \Sigma^T \Sigma V^T \\ AA^T = U \underset{13\times13}{\Sigma \Sigma^T} U^T \end{array} \right\}$$

Find eigen values & eigen vectors of A^TA

We get $U(42 \times 42)$ Matrix

So final feature vector after PCA

$A \cdot U[:, 1] \rightarrow (13\times42)(42\times1) \rightarrow 13\times1$

Date :
Page No.

Eigen vectors → direction for axis

" values → scaling factor.

Ex $\begin{pmatrix} 9 & 4 \\ 4 & 3 \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix} = \lambda\begin{pmatrix} u \\ v \end{pmatrix}$

$\begin{vmatrix} 9-\lambda & 4 \\ 4 & 3-\lambda \end{vmatrix} = 0 \Rightarrow (9-\lambda)(3-\lambda) - 16 = 0$

$\Rightarrow 27 - 3\lambda + \lambda^2 - 9\lambda - 16 =$

$\Rightarrow \lambda^2 - 12\lambda + 11 = 0$

$(\lambda - 11)(\lambda - 1) = 0$

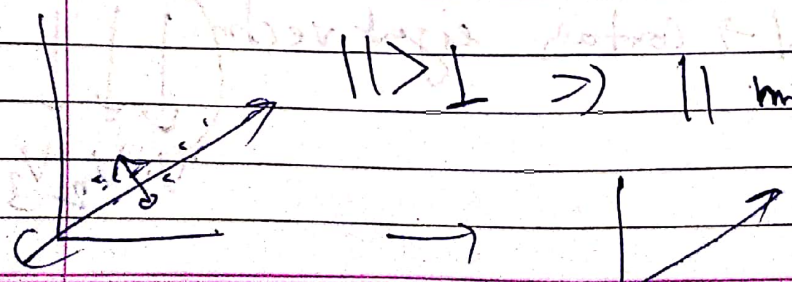
So Eigen Values → $\lambda = 1, 11$

$\Sigma \rightarrow \begin{pmatrix} 9 & 4 \\ 4 & 3 \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix} = 11\begin{pmatrix} u \\ v \end{pmatrix}$

$\Rightarrow \begin{matrix} 9u + 4v = 11u \\ 4u + 3v = 11v \end{matrix} \Rightarrow \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$ → dir^n

Eigen Vector

$11 > 1 \Rightarrow 11$ matters more



Scanned by CamScanner

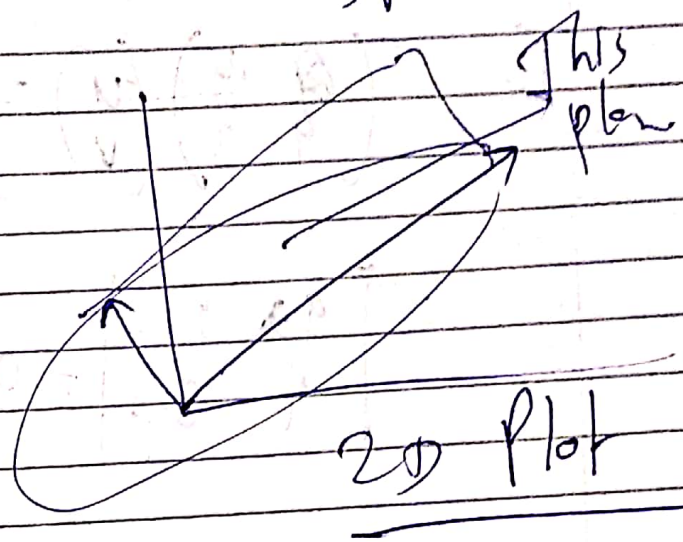
Date :
Page No.

5D → 2D

	v_i	λ_i
	v_1	λ_1
	v_2	λ_2
	v_3	λ_3
	v_4	λ_4
	v_5	λ_5

$$\begin{pmatrix} v_1 & \lambda_1 \\ v_2 & \lambda_2 \end{pmatrix}$$

↳ plot them



2D Plot

Let :

$$C = \begin{pmatrix} 5 & 5 \\ -1 & 7 \end{pmatrix}$$

Want $C = U \Sigma V^T$

$$C^T C = \begin{pmatrix} 26 & 18 \\ 18 & 74 \end{pmatrix}$$

- $$\boxed{\begin{aligned} C^T C &= V \Sigma^T \Sigma V^T \\ CV &= U \Sigma \end{aligned}}$$

$\Sigma \rightarrow$ contains eigen values. $\begin{pmatrix} \lambda_1 & & \\ & \lambda_2 & \\ & & \lambda_3 \end{pmatrix}$

$V \rightarrow$ contains eigen vectors $\begin{pmatrix} \downarrow & \downarrow & \downarrow \\ v_1 & v_2 & v_3 \end{pmatrix}$

Scanned by CamScanner

$\cancel{A - \lambda I = 0}$

Date :
Page No.

$Ax = \lambda x$

$$|C^TC - \lambda I| = 0$$

$C^TCx = \lambda x$

$\Rightarrow \lambda = 20, 80$

$$\begin{pmatrix} 26 & 18 \\ 18 & 74 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = 20\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$\Rightarrow V_1 = \begin{pmatrix} -3/\sqrt{10} \\ 1/\sqrt{10} \end{pmatrix}$ So $V = \begin{pmatrix} -3/\sqrt{10} & 1/\sqrt{10} \\ 1/\sqrt{10} & 3/\sqrt{10} \end{pmatrix}$

& $V_2 = \begin{pmatrix} 1/\sqrt{10} \\ 3/\sqrt{10} \end{pmatrix}$ $\Sigma = \begin{pmatrix} 255 & 0 \\ 0 & 455 \end{pmatrix}$

$\sigma_1^2 = 20$ $= \begin{pmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{pmatrix}$

$\sigma_2^2 = 80$

$\Rightarrow \sigma_1 = 2\sqrt{5}$

$\sigma_2 = 4\sqrt{5}$

So $CV = U\Sigma$

$\Rightarrow V = \boxed{C^{-1} U\Sigma}$

Scanned by CamScanner

SVD vs Eigen Vector Dec

Date :
Page No.

$\boxed{A = U \Sigma V^T}$

has more information

→ A → rectangular matrix

$\boxed{A = X \Lambda X^{-1}}$

A must be square matrix

$\sigma_1 \geq \sigma_2 \geq \ldots \sigma_r \geq 0$

$\sigma_1^2, \sigma_2^2 \ldots \sigma_r^2$ → non zero eigen values of AA^T & A^TA

The Orthonormal Colums of U & V are eigen Vectors of AA^T, A^TA

Scanned by CamScanner

Date :
Page No.

Customer Support Chatbot

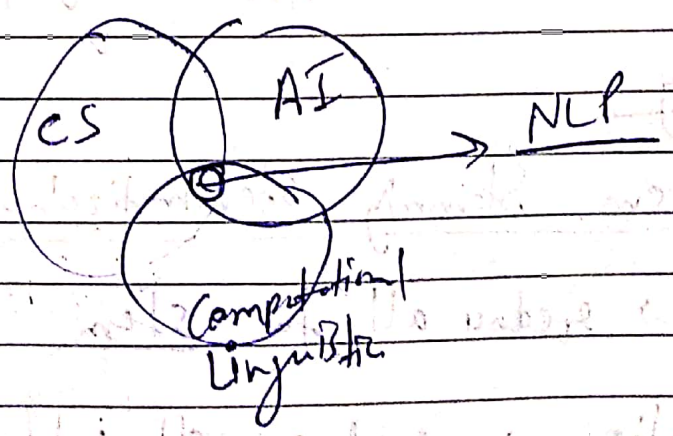
Problem: Social sites → customers need to be replied.
→ Large # of human were required to address the queries.

Chatbot handles & answers user queries on social Media.

Dataset: Kaggle → 0.1 Million Q A pairs on Apple Support.tsv

- Cornell Movie dialogue Corpus
- 1 lacs tweets (Q A) pairs

Pre-processing → Scikit & NLTK (Natural Lang Toolkit library)



NLTK provides 50 corpora/dataset + lexical resources

Scanned by CamScanner

Date :
Page No.

Text - Pre processing :

for this

① Uppercase → lowercase [Apple = apple, Crazy = crazy]

② Tokenization Ex → Chopping into words ← removing , ; etc

Friends, are, good, people ;

[Friends] [are] [good] [people]

returns tokens of words for dictionary

→ Remove noise → anything which is not a standard number or letter.

→ Removing Stop words → Ex. an, the, in, what, how, too much frequent, just remove them

→ Stemming

Stem, Stemmy, Stemmatizer, etc.

↳ reduce all to stem (Base word)

see context also

→ Lemmatization → reduce all inflectional forms to common base form. Using Lemma

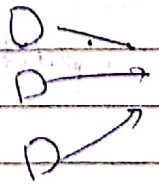
Ex Saw → S (Stemmy)

Saw → see, saw (seas whether verb or not)

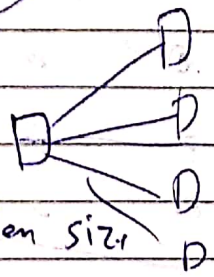
Ex. better & good → treat as same
lemma

→ Can see most frequent words using Wordcloud

CBOW →



Skipgram →



③ Embeddings

word2vec : size 640 = hidden size

I/P →

hello	how	are	you	today	
can	you	say	that	0	
you	are	good	0	0	
well	ok	0	0	0	
Sure	0	0	0	0	

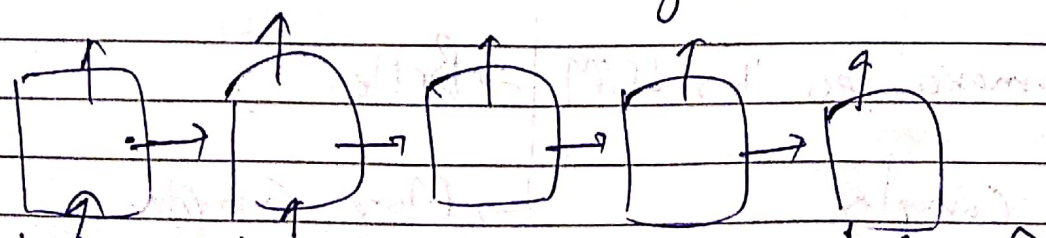
batch_size X max_length

5X5

→ Zero-padding

↓ Transpose

max length x batch size



hello	how	are	you	today
can	you	say	that	
you	are	good		
well	ok			
sure				

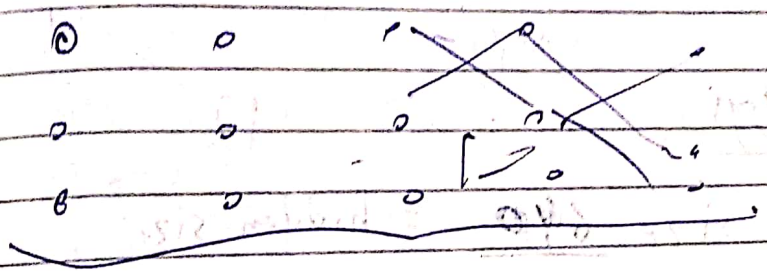
better → good
lemma

RNN → $h_t = \tanh(W_{ih}x_t + b_{in} + W_{hh}h_{(t-1)} + b_{hh})$ or relu

Disadv : Vanishing Gradient Problem

Gradients → near to 0

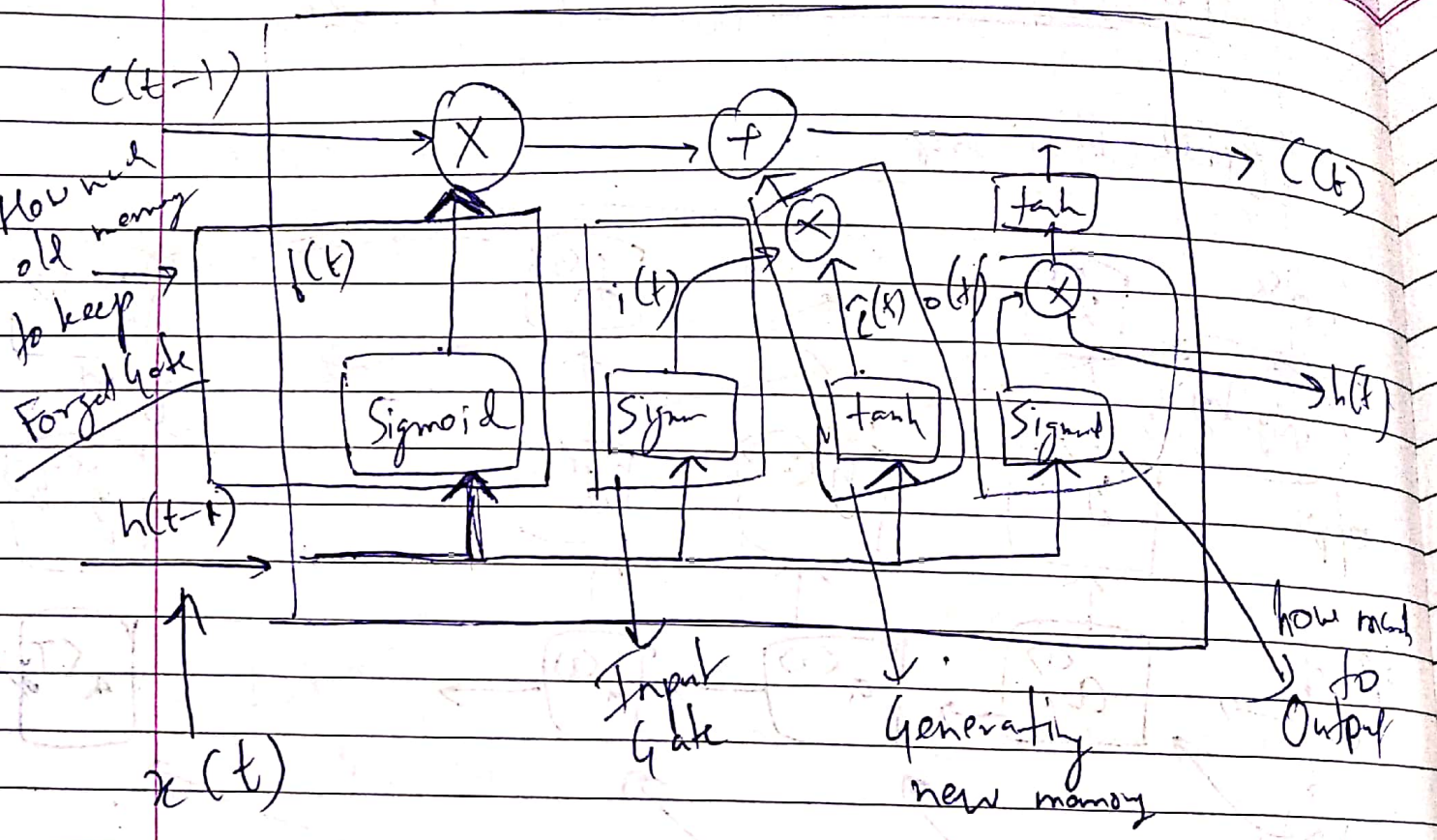
No update in previous part of N/w



GRU (New)	LSTM (old)
→ reset, update gates	i/p, output & forget gates
→ Has No memory unit	→ Has
→ Exposes full hidden content w/o any control	→ Controls info flow
→ Performance less than LSTM	→ Better
→ Less Complex	→ More Complex

Scanned by CamScanner

LSTM Cell



0, 0.5, 1 [0, 1]

None, Partially, All

$$\sigma = \frac{e^x}{1+e^x} \qquad \tanh = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad [-1, 1]$$

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

Forward Prop. → Easy

Can see directly

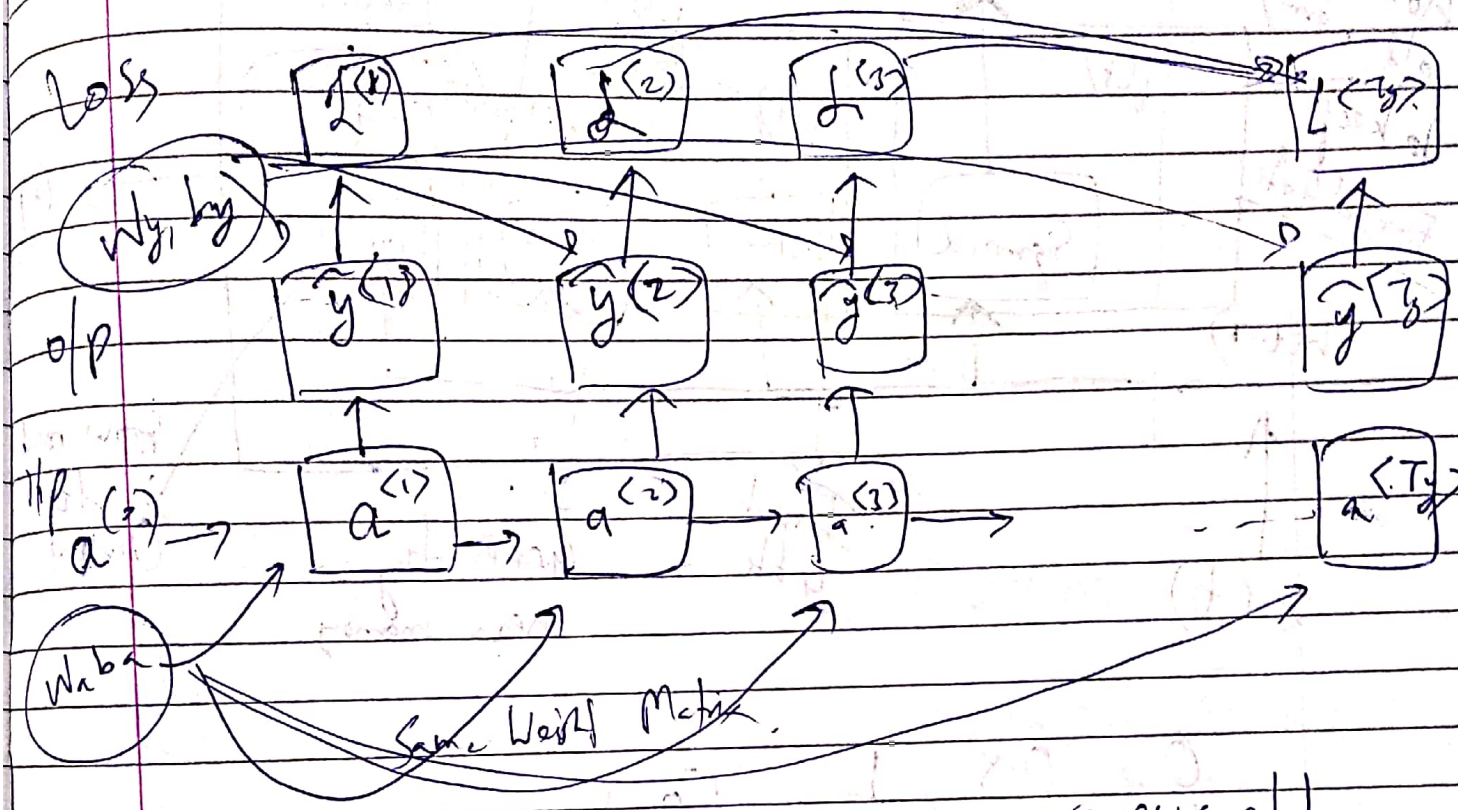
Learn Weights.

Scanned by CamScanner

Date :
Page No.

BPTT (Back Propagation thru time)

RNN Backprop



Computational Graph → reverse all edges in forward prop.

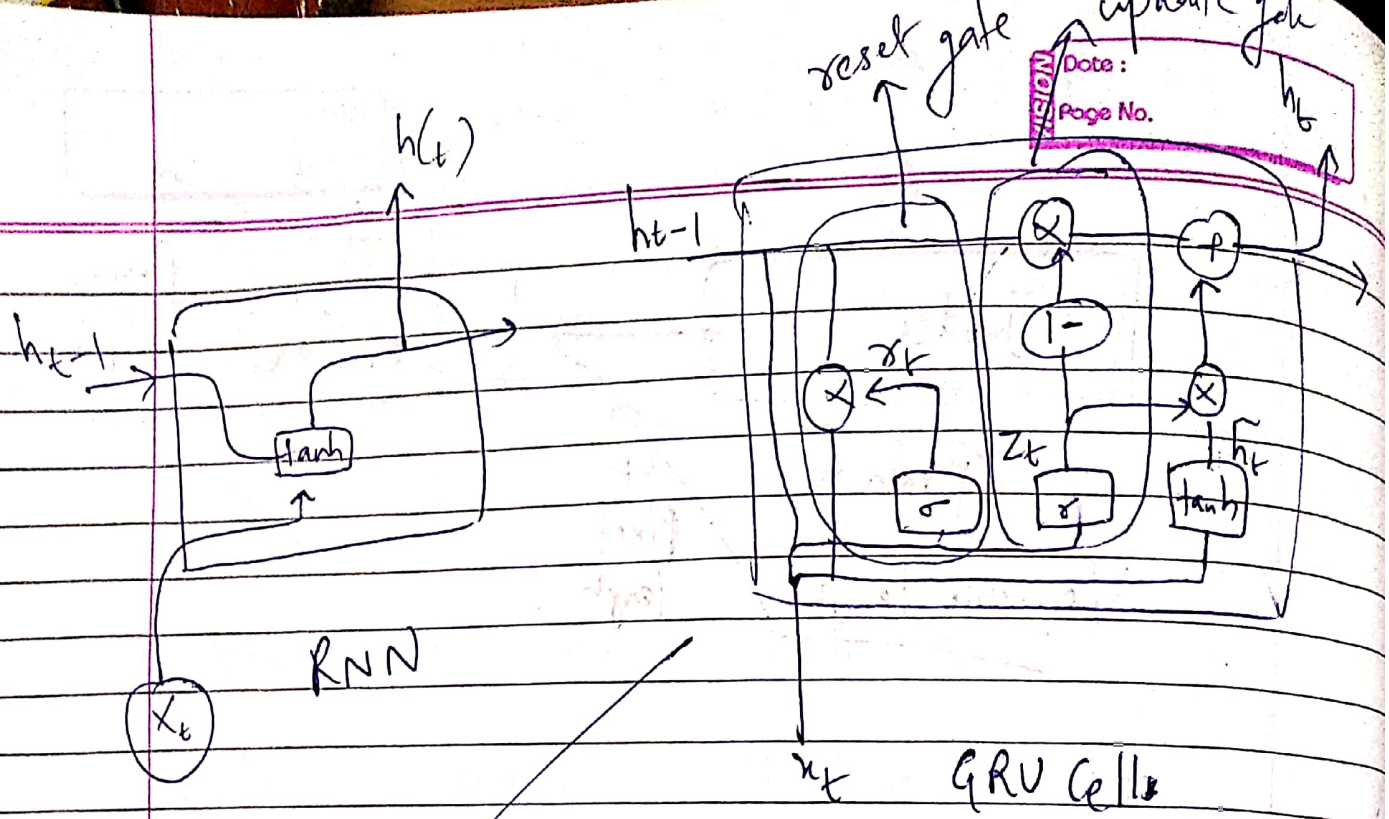
RNN: $h_t = \tanh(W_{ih} x_i + b_{ih} + W_{hh} h_{t-1} + b_{hh})$

LSTM → Traditional approach is to update weights for weight update at $t-1$

$$\frac{\partial L}{\partial W_{t-1}} = \left(\frac{\partial L_{t-1}}{\partial W_{t-1}} + \frac{\partial L_T}{\partial W_{t-1}}\right) \Big/ 2$$

Loss^n = sum of all losses

Scanned by CamScanner



Got rid of cell state

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$$

$$\tilde{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t])$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

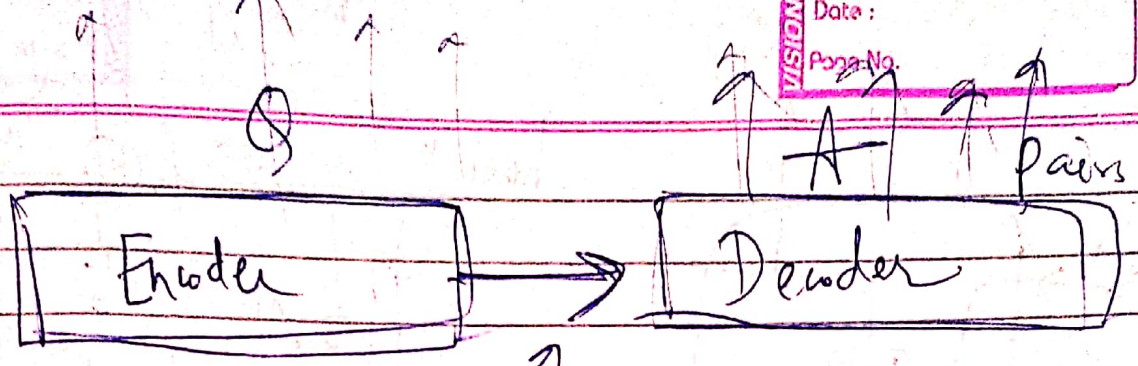
New weight θ_i = Weight θ_i − α * gradient $\left(\frac{\partial J}{\partial \theta_i}\right)$

Update gate = forget gate + ~~input gate~~ new info

Reset gate = what to forget or how much passed info to forget

Less tensor operations

Researchers run both to analyse which one is best

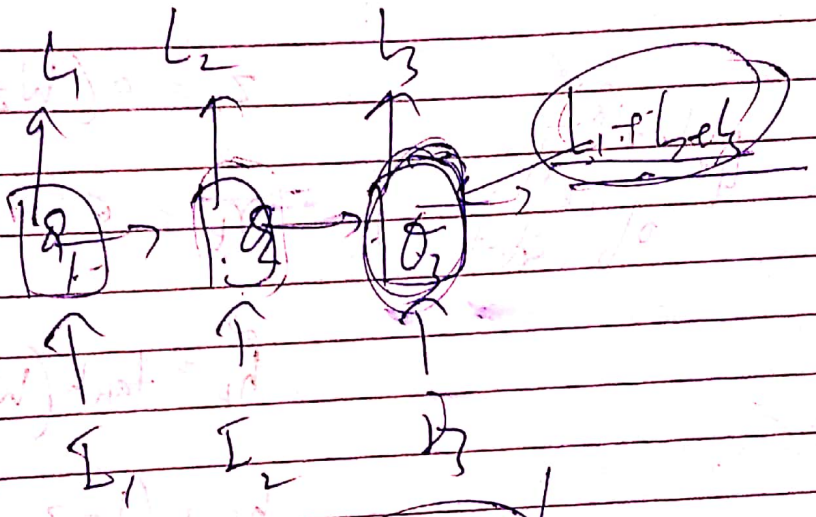


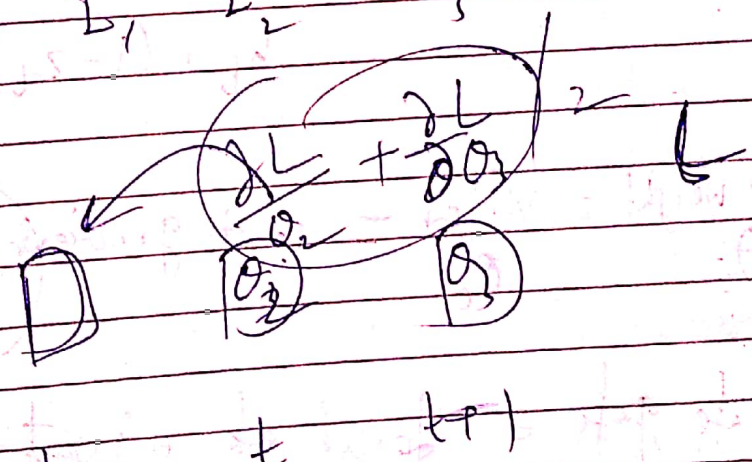
Train &

Variable length

C_t

Fixed length





word in the sequence

Date :
Page No.

Bidirectional GRU

{ Hi, How are You F→Future
You are How Hi F→Past }

Output = Sum of both i/ps

- Advantage of encoding both past & present

Encoder → fixed output vector should encode semantic similarity b/w similar meaning words.

→ Computation Graph

1) Convert word indexes to embeddings.
2) Pack padded batch of sequences for RNN module.
3) Forward pass through GRU
4) Unpack padding
5) Sum bidirectional GRU outputs.
6) Return o/p & final hidden state

Decoder Generates response sentence in a token by token fashion. It uses the encoder's context vectors, & internal hidden states. → to generate next

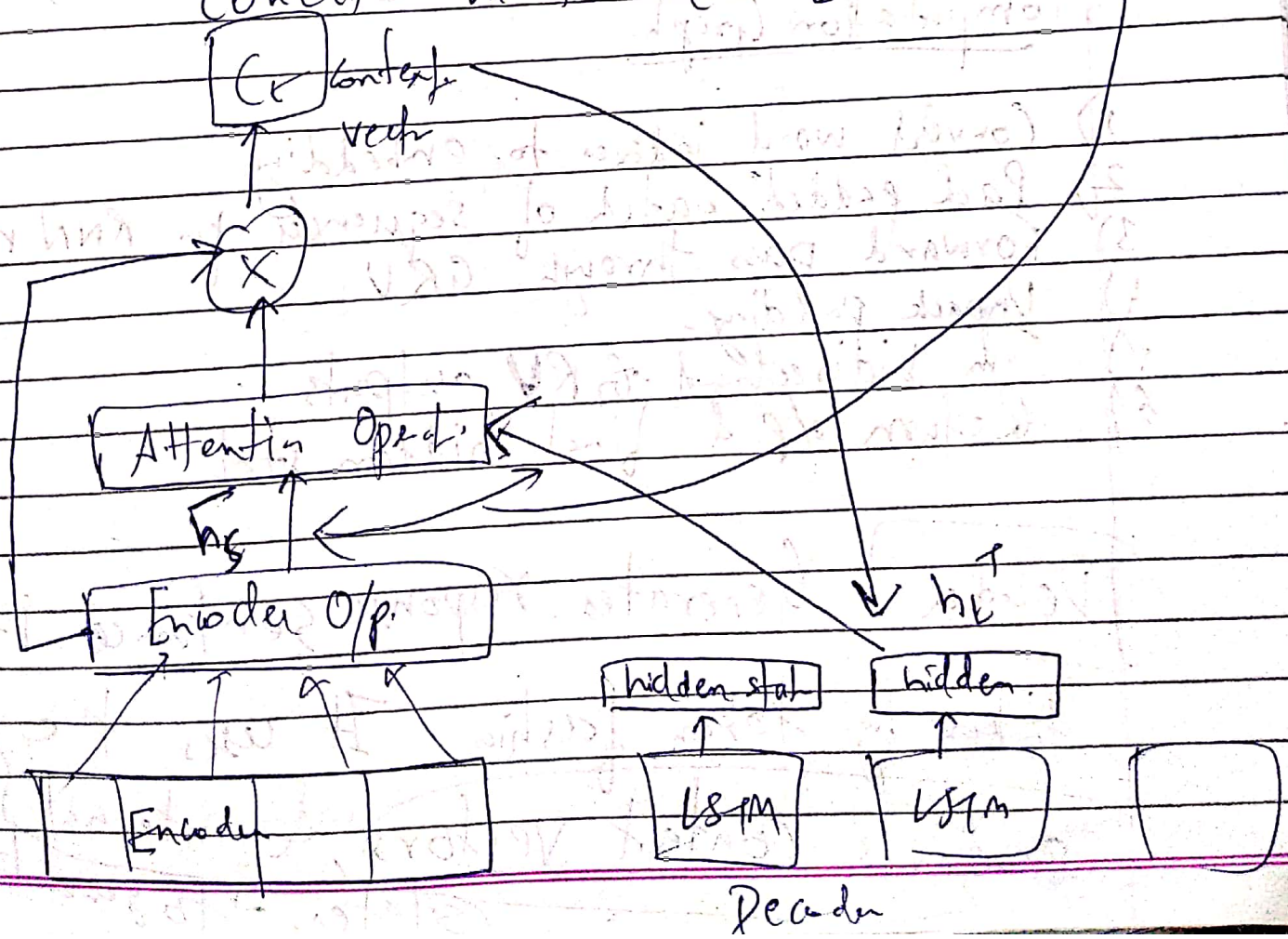
But if we use Attention mechanism, that allows decoder to pay attention to certain parts of the i/p sequence rather than using the entire fixed context at every step.

<u>Attention Types</u>

dot $h_t^T h_s$

general $h_t^T W_a h_s$

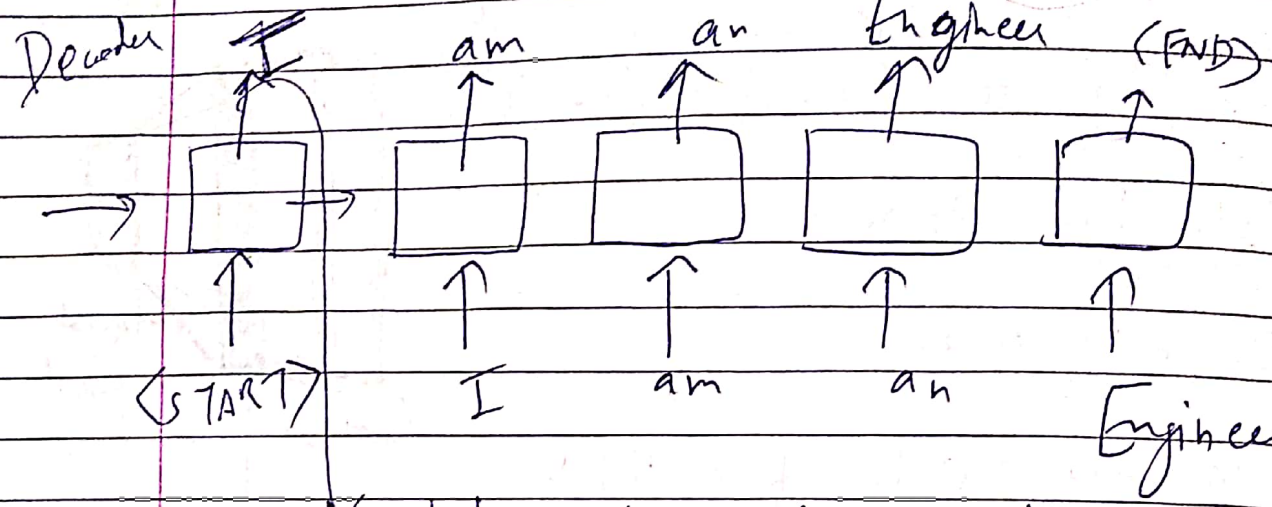
Concat $V_a^T \tanh(W_a [h_t ; h_s])$



Scanned by CamScanner

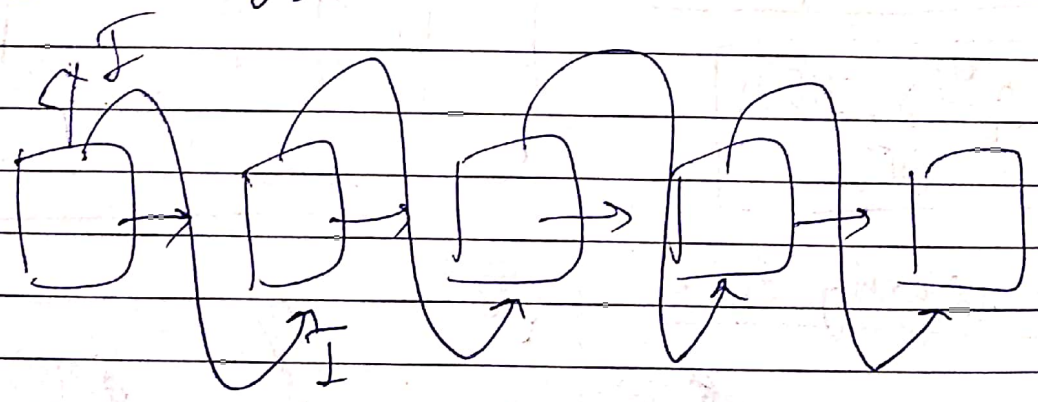
Disadv. Doesn't mimic real Time Situations.

Teacher Forcing (forcing i/ps of decoder)



whatever it outputs we force the exact feature, we want to form

No. Teacher Forcing



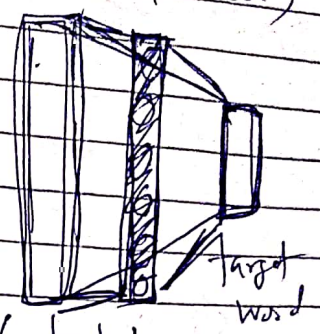
Whatever 1st LSTM o/p → is sent to i/p of other LSTM.

Generally → 0.5 (TF) + 0.5 (NTF)

Scanned by CamScanner

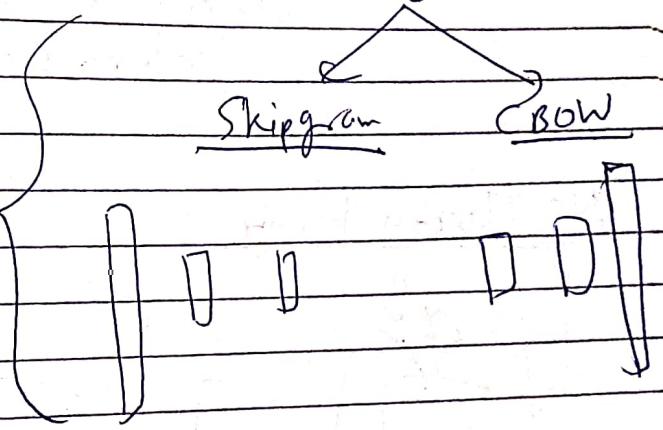
(4) Model Specification

- # of LSTM Layers = 4 (Both for Encode | Decode)
- Hidden size = 640 (Word2vec)
- Iteration = 70000
- Dropout - 0.2
- Avg Loss = 0.15 from 9.144



- Gradient Descent

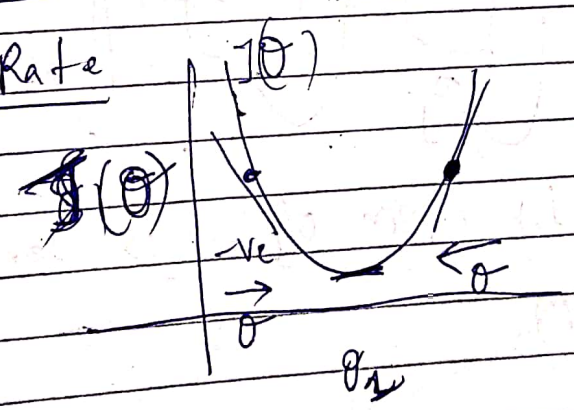
Skipgram | BOW



$J(\theta) \rightarrow$ cost function

$$\boxed{\theta_i = \theta_i - \alpha\left(\frac{\partial}{\partial \theta_i} J(\theta)\right)}$$

α = Learning Rate



$\frac{\partial J(\theta)}{\partial \theta_1} = +ve$

We want to reach minima so go in dir^n where derivative is 0

BLEU Score — Bilingual Evaluation Understudy

Date :
Page No.

Types of loss functions

Regression Loss

Regression losses

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

→ Easier to calculate gradients.

→ Due to square high penalty on predictions which are far away

② MAE $= \sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{n}$

→ Avg. mag. of error irrespective of dir^n.

- Measures Magnitude of error w/o considering dir^n
- Needs more complicated tools to compute gradients
- More robust to outliers.

Since doesn't square.

Classification Losses

dar wale ko hold karne wala

① Hinge loss / Multi Class SVM Loss $= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$

- Not differentiable
- But convex function
- Convex optimizers

	Dog	Cat	Horse
L	I1	I2	I3
Dog	-0.39	-4.61	1.03
Cat	1.49	3.28	-2.37
Horse	4.21	1.46	-2.27

$L = \max(0, (1.49) - (-0.39)) + \max(0, (4.21) - (0.39)) = 5.98$

High loss

Scanned by CamScanner

(2)

Date :
Page No.

Cross Entropy loss

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

y_i → Actual Label.

Entropy $= -\Sigma p \log p = -p\log p - (1-p)\log(1-p)$

Categorical Cross Entropy loss – Negative likelihood loss

$$-\sum_{c=1}^{M} y_{o,c} \log(p_{o,c})$$

$y_{o,c}$ → Binary (0/1) if it is correct classific

$P(A)$ = happening of event

$I(A) = -\log(P(A))$ → Info content of A

Softmax Activation at last layer (Formula)

then

$$\sigma(x_j) = \frac{e^{x_j}}{\sum_i e^{x_i}}$$

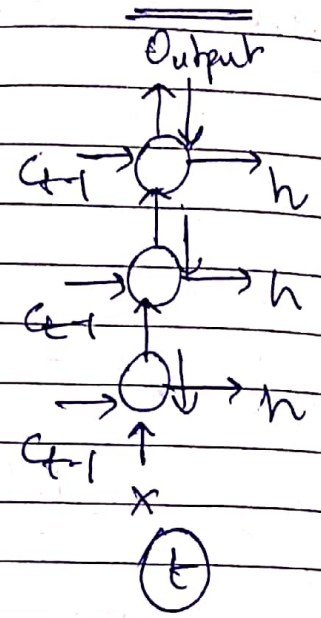
Given a vector of real #, it converts each value to a probability.

Scanned by CamScanner

Date :
Page No.

(3) KL Divergence loss f^n

Distance b/w 2 distributions.



Min $KL(Q \| P)$

$$= \sum_{H} Q(H) \ln \frac{Q(H)}{P(H|V)}$$

Entropy & Inform-

$\boxed{\text{KL Divergence} = \text{Cross Entropy} - \text{Entropy}}$

$$KL(p\|q) = H(p,q) - H(p)$$

$$KL(q\|p) = H(q,p) - H(q)$$

$$H(p,q) = -\sum_{x} p(x) \log(q(x)) \quad \text{— (A)}$$

$$H(p) = -\sum_{x} p(x) \log(p(x)) \quad \text{— (B)}$$

$$(A) - (B) = \sum_{x} p(x) \left[\log p(x) - \log q(x) \right]$$

$$= \sum_{x} p(x) \left[\log \left(\frac{p(x)}{q(x)} \right) \right]$$

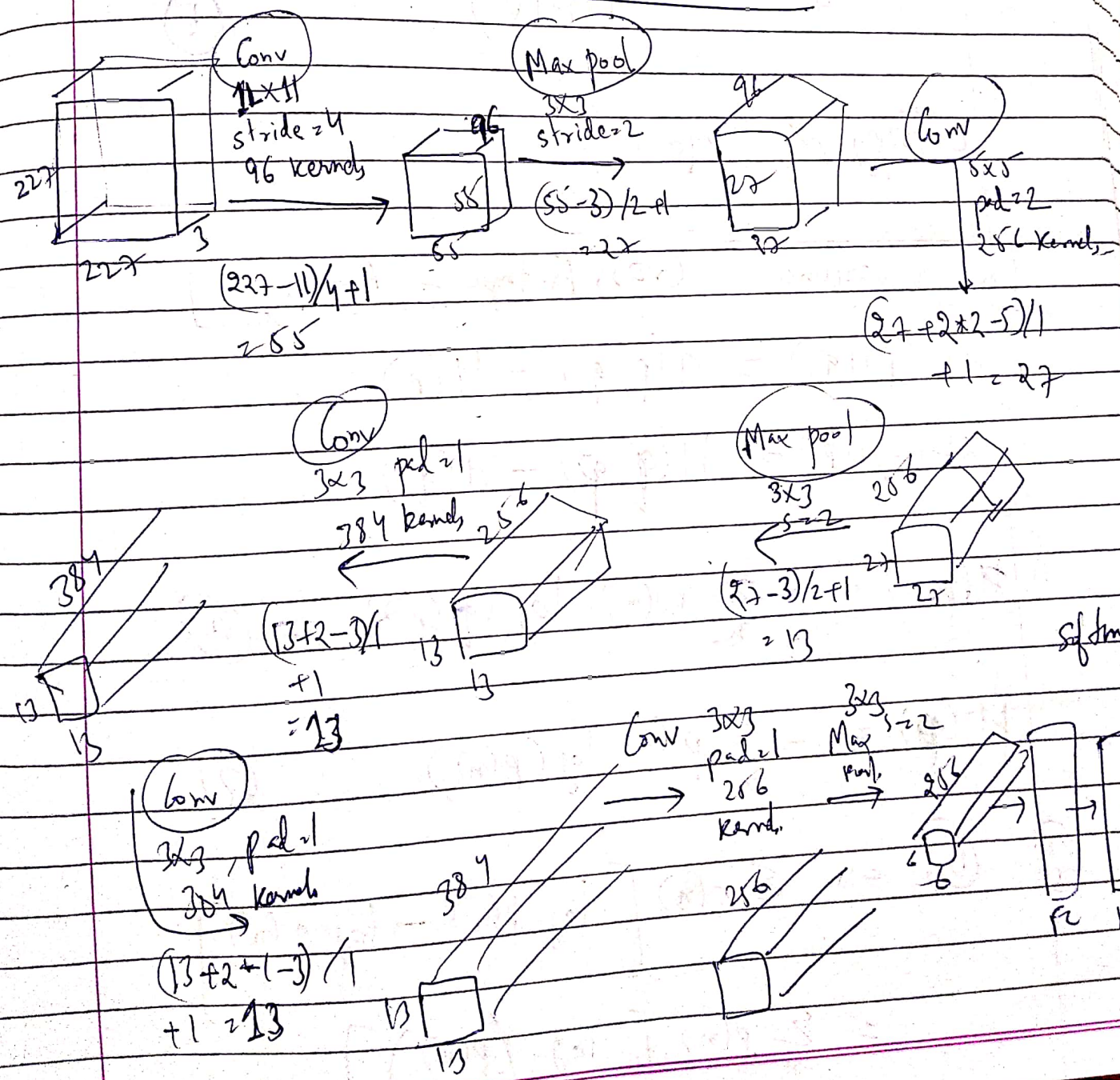
Scanned by CamScanner

$H(p,q)$

Actually, cross entropy $= KL(p \| q) + H(p)$

Dist b/w Distri — Entropy of p

CNN → Alex Net Architecture



overlap

Relu Relu Relu Relu Relu

Conv → Pool → Conv → Pool → Conv → Conv → Conv

→ Pool → FC1 → FC2 → Softmax

5 Conv, 3 Pool

Monument Recog

Fine tuning → Pre-trained used

227×227 → Alexnet →

learning rate = 0.001
Dropout = 0.2
Iteration = 10,000

62.3 million parameters

Ours →

- Dataset → 50 × 100 = 5000 Images

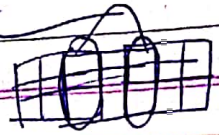
100 Monuments.

Alexnet 2012 paper

→ Used Relu Instead of tanh to add non linearity. It gets 6X accuracy

→ Used dropout instead of regularization to deal with overfitting

→ Overlap pooling to reduce size of N/W → overfitting



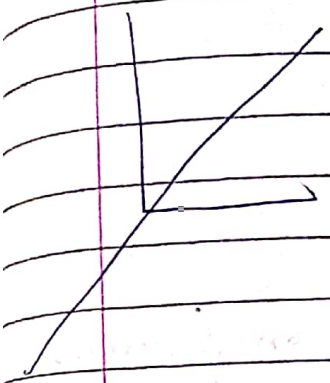
if no overlap CNN won't learn spatial features & creates bias towards dominant

Date :
Page No.

Activation functions

Linear

$f = y = f(x)$



tanh $[-1, 1]$
sigmoid $(0, 1)$

not suitable for hidden layers because if $\frac{\partial J}{\partial \theta}$ is small it slows down gradient descent.

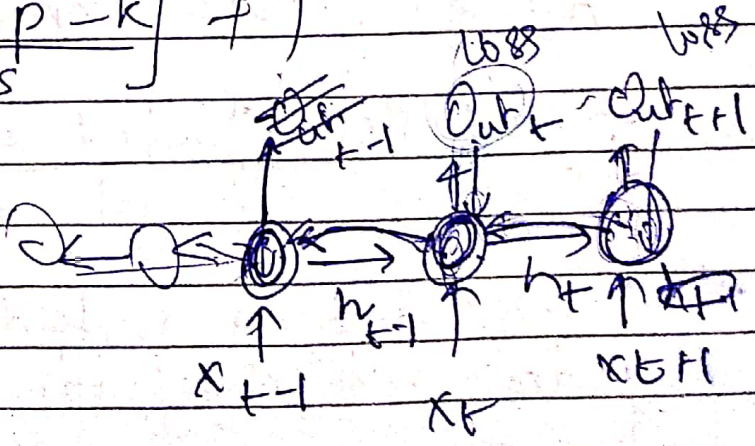
Relu $[0, \infty)$
Leaky Relu $[-\infty, \infty]$

suitable for hidden layer, paces up gradient desent

→ Formula for Convolution

$$n_{out} = \left\lfloor \frac{n_{in} + 2p - k}{s} \right\rfloor + 1$$

p = paddy size
k = kernel size
s = stride



dbptt

Scanned by CamScanner

Date :
Page No.

(3) NILMTK

- DAE
- RNN
- Mean
- Zero
- Windows GRU
- Seq2point
- Seq2Seq
- DSC
- Mark
- FHMM
- CO
- [GAN, VAE]

— Benchmark

My contribution -

- Solved Issues
- SMART* Dataset Converter
- Sankey Plot
- Hart 85
- Documentation - Demo Notebook
- Script to run all demo notebook.

Chunk → normalize → Window/Overlapping

- $\frac{x-\mu}{\sigma}$
- min-max ~~norm~~ scaling
- dropout

Datasets

- REDD
- UKDALE
- DATAPORT
- DRED

Chunk-size = 5000
Seq length = 50

→ Train & Test on same buildings
→ Train & Test across buildings from same data set
→ Train across multiple buildings from multiple data sets & test on multiple buildings across datasets
→ Train on Artificial Aggregate → sum disaggregated data over

Sum of appliance power

Scanned by CamScanner

Date :
Page No.

KW = LJ/sec

(1) Mean

Aggregate Power → [Mean] → Mean (Aggregate Power) Always ON

Predict for fridge

(2) Zero

[Zero] → 0 Always (off Always)

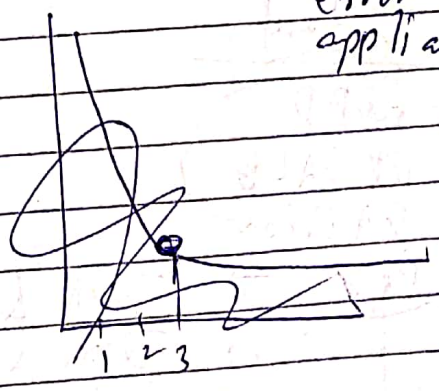
- Fridge , Furnace
- AC , AC
- Lights
- Microwave
- Washing M/c

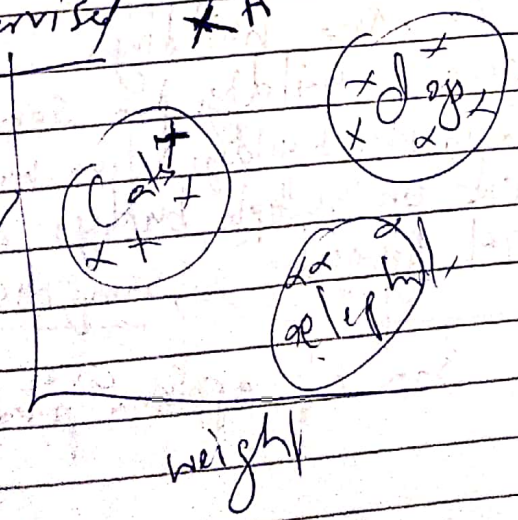
so o/p needs to be mapped to least error appliance

(3) Hart 85 (Unsupervised used KMeans)

KNN → K=2, 3, 4, ...

Supervised





1) Find A's distance from all points & store in dict
2) Sort dist (Ascending)
3) Pick up ~~to~~ min K elements
4) Majority vote

Scanned by CamScanner

Date :
Page No.

KNN

Classifier — Return mode of k min distance exmp

Regression — Return Mean of K min distance exmp

→ K-Means (Unsupervised)

for k=2 to 10

{ while (no cluster change)

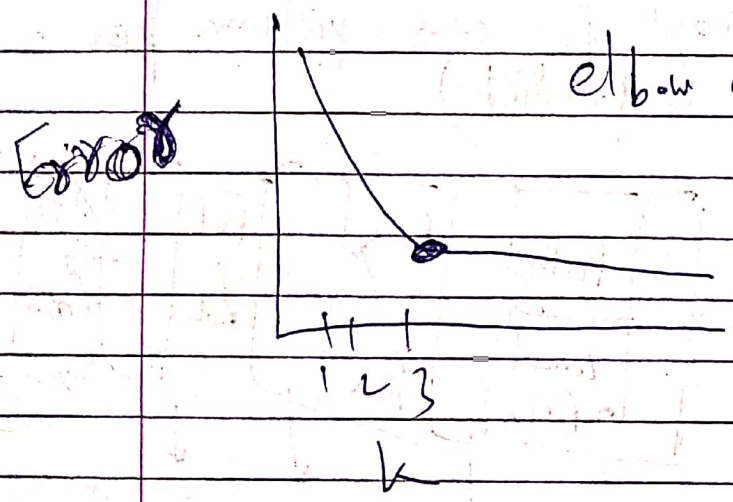
{ 1) Choose k=2 random clusters.

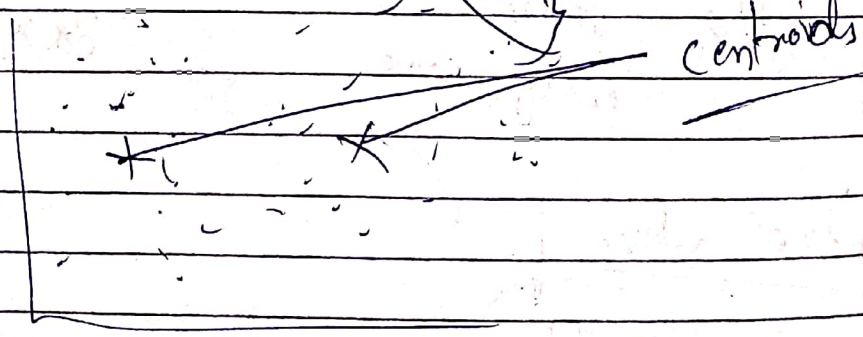
2) Assign each point to nearest cluster

3) Update centroids.

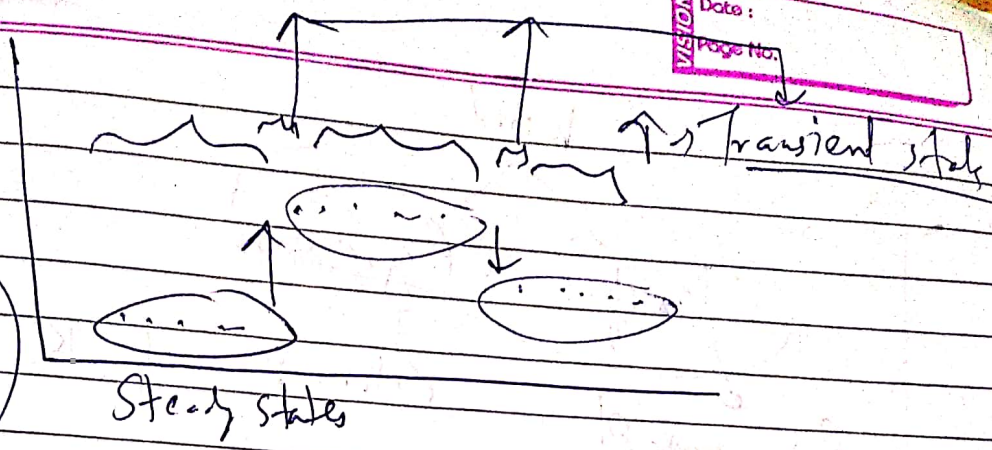
}

}





Distance = Squared Euclidean Dist



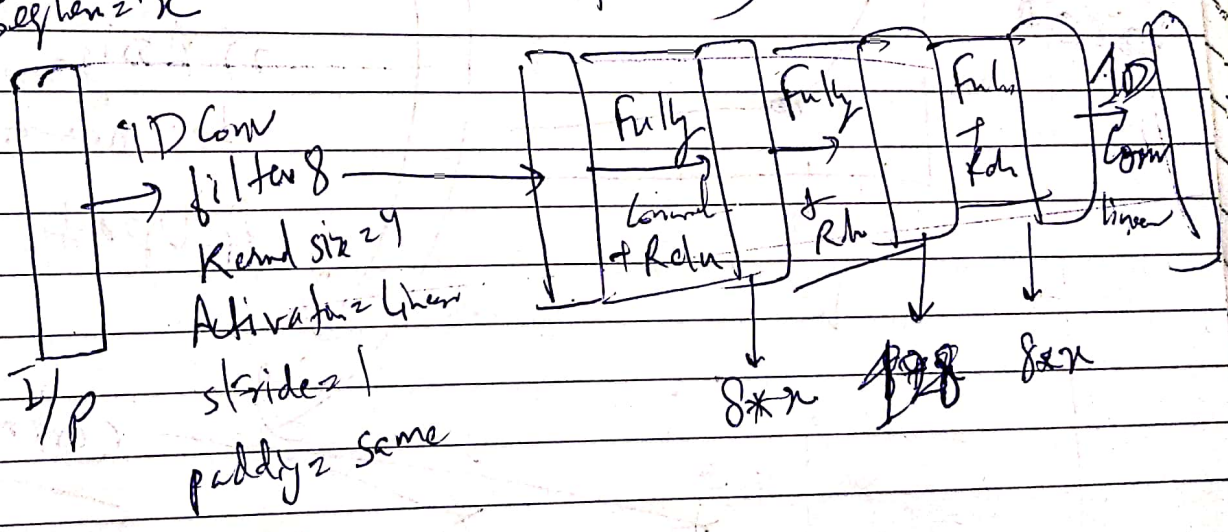
Donald Hart 88

edge = (Steady — Transient state)

↳ Clustering → o/p (not in depth)

4) DAE (Train Model for each appliance per chunk & predict)

seqlen = x

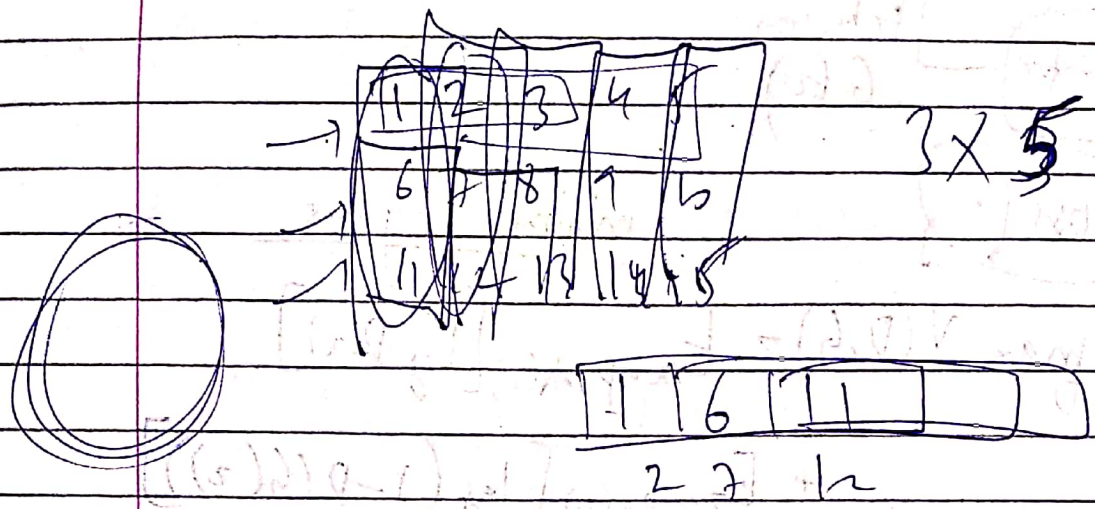


Linear activation

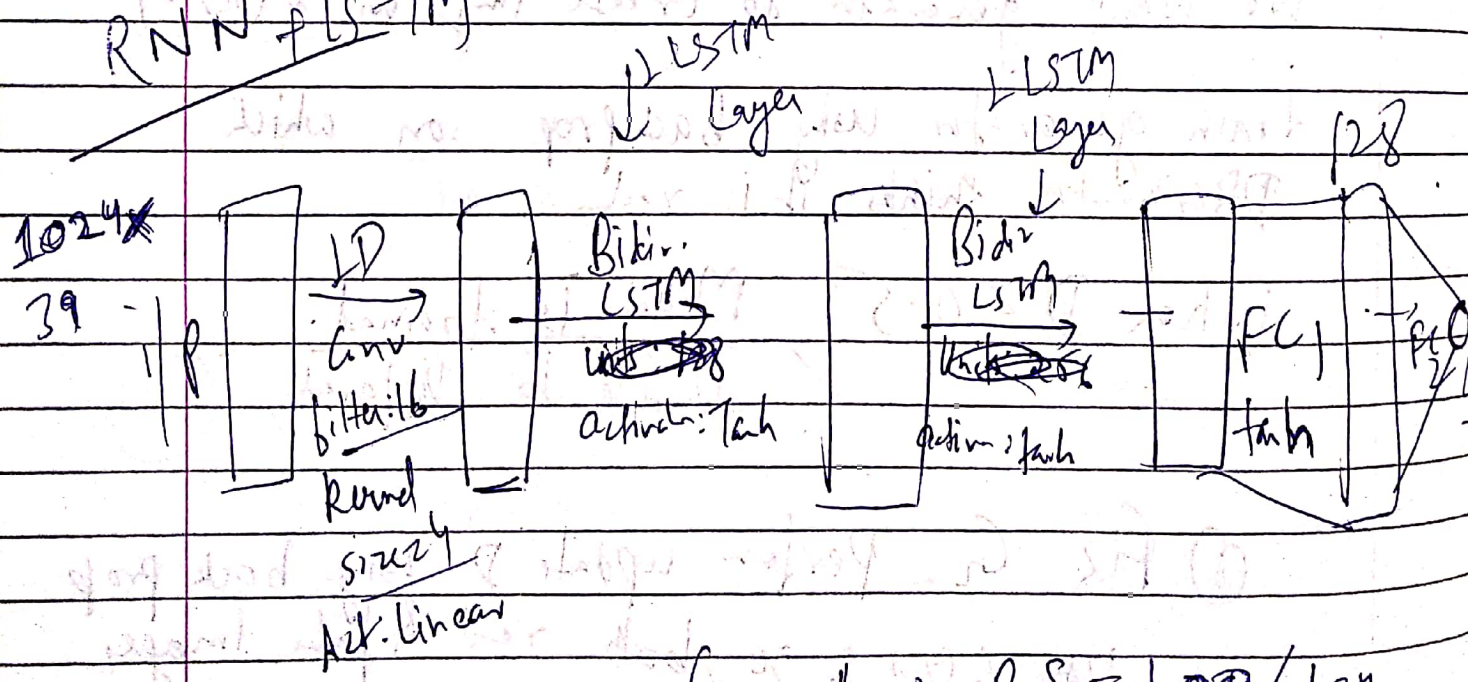
$y = ax$

→ used at o/p layer

→ not much help.



RNN + LSTM

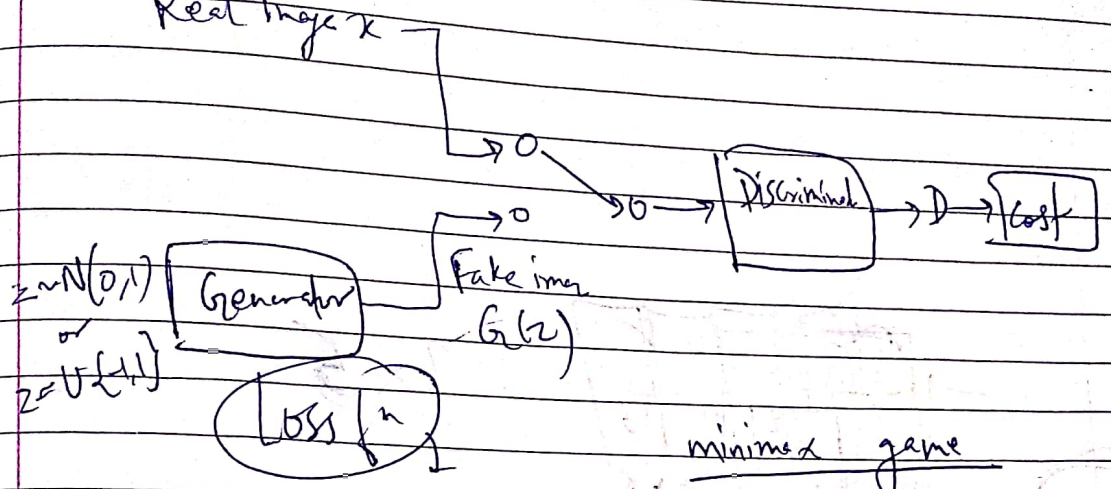


Generally → BS = 1024 / 1024
Epochs = 100

Sample rate = 60 seq length = 39
chunk size = 100000

Scanned by CamScanner

GAN



minimax game

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

Cross entropy loss

We want generator to create img with $D(x)=1$

train generator using back prop on which Discriminator thinks it is real

When $D = 0.5$, Model is trained
Unable to Differentiate

① Fix G, Perform update D using back prop
, use $G(z)$ & x, both real & fake images

② Fix D, Train G for both img & backpropagate

Date :
Page No.

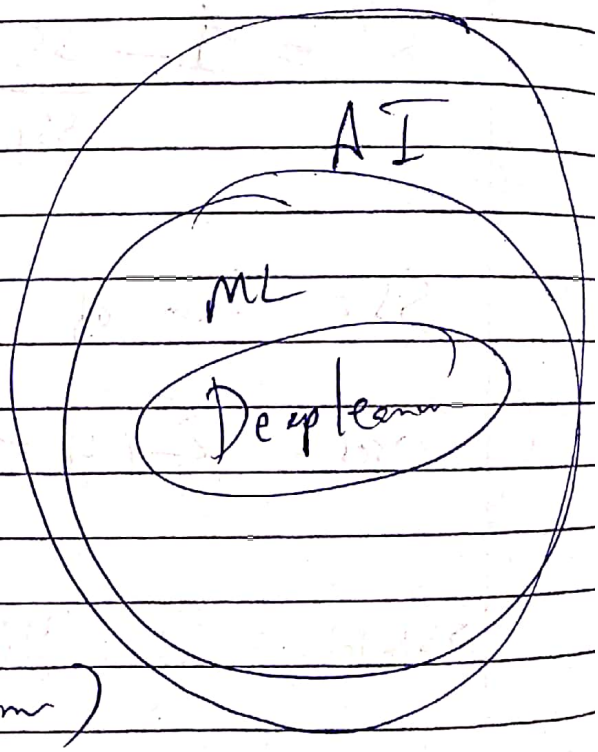
Normalization Types

① Rescaling (min-max normalization)

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

for ~~rescal~~ rescaling to range [a,b]

$$x' = a + \frac{(x - \min(x))(b-a)}{\max(x) - \min(x)}$$



② Mean Normal

$$x' = \frac{x - \text{avg}(x)}{\max(x) - \min(x)}$$

③ Standard (Z-score Norm)

$$x' = \frac{x - \bar{x}}{\sigma}$$

Scanned by CamScanner

- Bias Variance Tradeoff
- Reinf. of Transform
-

R^2 = Coefficient of Determination

$$R^2 = 1 - \frac{SS_{res}}{SS_{total}}$$

SS = Sum of squared error

$$SS_{res} = \sum_i (y_i - f_i)^2 = \sum_i e_i^2 \rightarrow \text{Error}$$

$$SS_{total} = \sum_i (y_i - \bar{y})^2 \propto \text{variance}$$

18

Scanned by CamScanner

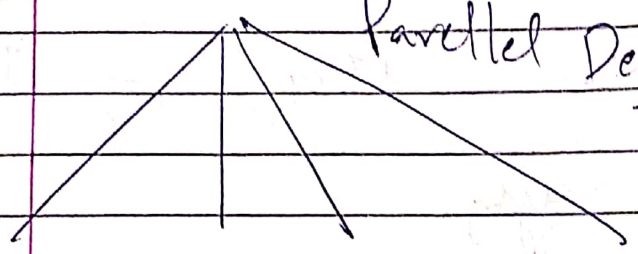
Date :
Page No.

Ensemble Technique

XGBoost

- → Bagging → Ex. Random Forest
- → Boosting → Ex. Xgboost

Random Forest

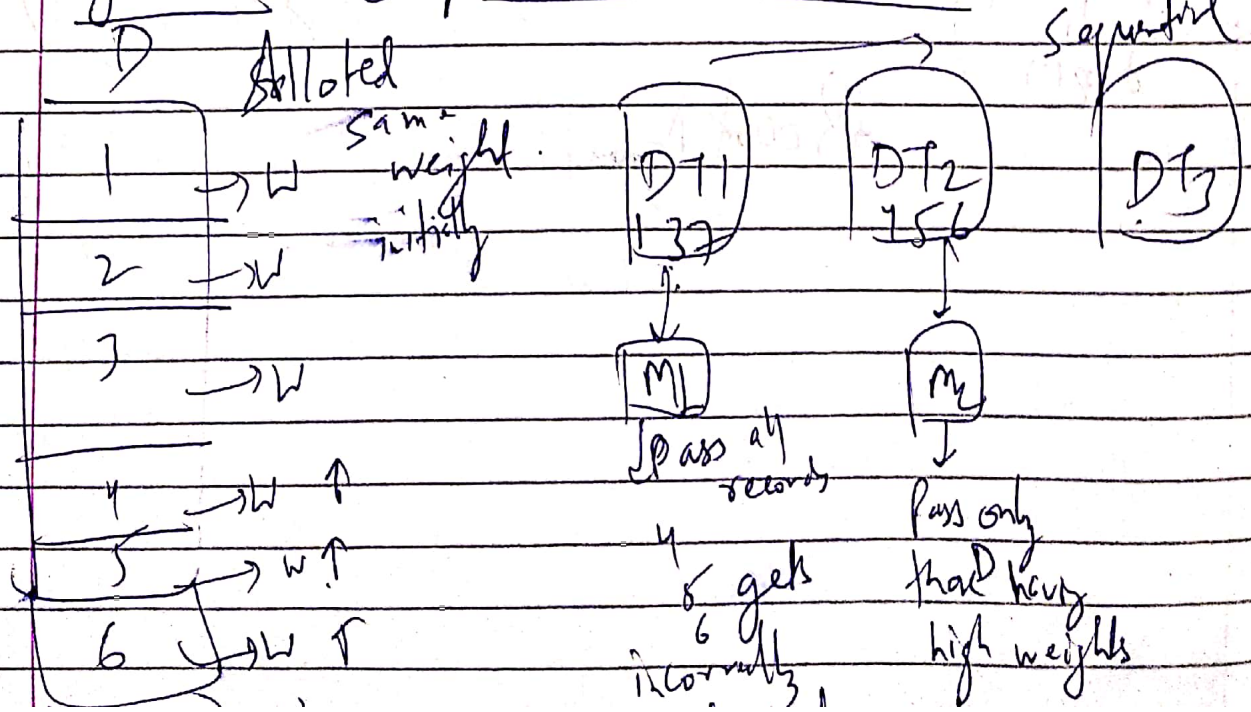


Parallel Decision Trees.

Sample Dataset 1 | 2 | 3 | 4] Sample of dataset

Sequential Ensemble technique

Xgboost → Sequential Decision Tree



Scanned by CamScanner

WS = Weighted Sample

Test data → WS1 → WS2 → WS3 — WS4 WS5

1 1 1 0 0

Classifier → output of classifier

Regressor → Continuous Values → CT1 CT2 CT3 ... CT15

Decision tree → Low Bias, High Variance → Bagging → Random forest solves.

problem

Decision Tree (Some fixed depth) ~~XGBoost~~ → High Bias, Low Variance → Low Bias by Xgboost.

Boosting

Scanned by CamScanner