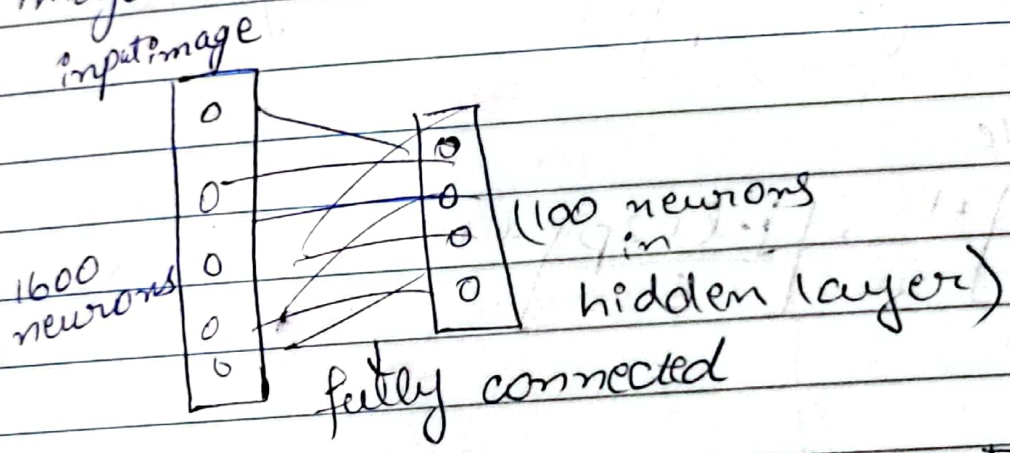
Foundation for Advancement of Science & Technology

Lecture No. 9

Convolutional Neural Network

Problems of ANN
↳ images, time series data, computational power

image : 40×40 (1600 neurons (1 pixel = 1 neuron))



Total weight = 1600 × 100 = 160,000

CNN gives better results than ANN.
↳ use filters/kernals
↓
used for feature extraction.

layers of CNN:
→ convolution
~~→ potential~~
→ pooling
→ fully connected layers.

Ex: image (5×6)

8	6	5	2	19	255
0	8	6	2	255	255
0	6	2	2	255	255
0	9	2	8	255	255
0	8	0	4	255	255

-1	-1	-1
0	0	0
1	1	1

• jb convolution krenge, agr value -ive hui to ans 0 ho jaye ga n agr 255 se value bri aa gayi to uss ko hum 255 kr dein ge.

• Convolution

image 5×6 (i, j) filter/Kernal (3×3) (k, l)

$[i-k]+1, [j-l]+1$

$[5-3]+1, [6-3]+1$

$3, 4$

Resultant image (3×4) (rows, col)

in case of padding, above formula changes to

$[i-k+2p]+1, [j-l+2p]+1$

after adding stride

$\left\lfloor \frac{i-k+2p}{s} \right\rfloor + 1, \left\lfloor \frac{j-l+2p}{s} \right\rfloor + 1$

• Pooling:

To minimize size. Average & Max Pooling are most commonly used.

Ex:

	# of parameters	dimension
input → (10, 10, 1) (height, width, channel)	0	10, 10, 1
conv → (5, 3, 3) (no. of filters; height, width)	$(3 \times 3 \times 1 + 1) * 5 = 50$	$[i-k+1], [10-3]+1 = 5 + (8 \times 8)$
activation (sigmoid)	0	$5 \times 8 \times 8$
Conv2D (2, 3, 3)	$(3 \times 3 \times 5 + 1) \times 2 = 92$	$2 \times (6 \times 6)$
activation (sigmoid)	0	$2 \times 6 \times 6$
output	$50 + 92 = 142$	$2 \times (6 \times 6)$

$(64 \times 18) + 64 =$

→ Formula to find # of parameters.
(filter width * filter height * # of filters in previous layer + 1) * # of filters in this layer

in case, agr input ke baad direct convolution aa jata ha to "# of filters in previous layer" ki jga "# of channels aaye ga.

↳ (filter width * filter height * # of channels + 1) * # of filters in this layer

	# of parameters	dimention
* pooling	0	2x(3x3)
Fully connected Dense (64)	1216	

Formula for fully connected layer: (# of parameters)
(current layer neuron × previous layer neuron) + current layer neuron

$= 64 \times (2*3*3) + 64$

$= (64 \times 18) + 64 = 1216$

Hyper parameters & parameters in ~~Con~~ CNN

→ stride, pooling types } pooling layers ki → no parameters.

filter size, height, width, stride, padding, # of filters, activation function.

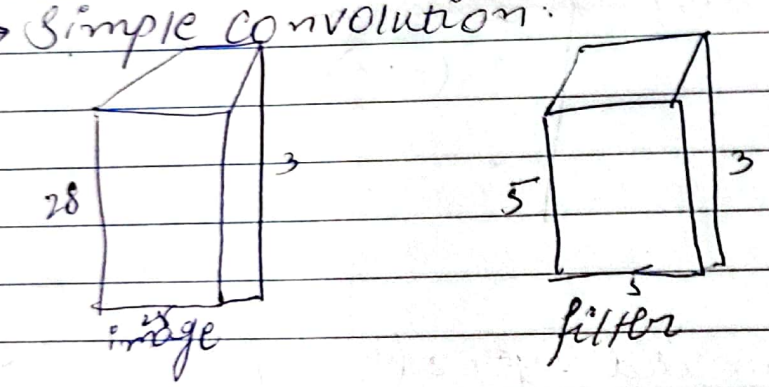
filter length, filter height, filter width in para formula

Foundation for Advancement of Science & Technology

Lecture NO. 10

Different types of convolution layers
- ↳ simple convolution
- ↳ point-wise convolution (1×1)
- ↳ Depth-wise seperable convolution
  - ↳ Depthwise convolution
  - ↳ pointwise convolution.
- ↳ Grouped convolution
- ↳ Spatial & cross channel convolution
- ↳ Shuffle grouped convolution
- ↳ Transpose convolution

→ Simple convolution:



28×28×3

5×5×3

# of filters = 16

$$\frac{28-5+2(0)}{1}+1 = 24$$

$\Rightarrow$ = (24, 24, 16)

Depth of image = Depth of filter.
OR OR
channel channel

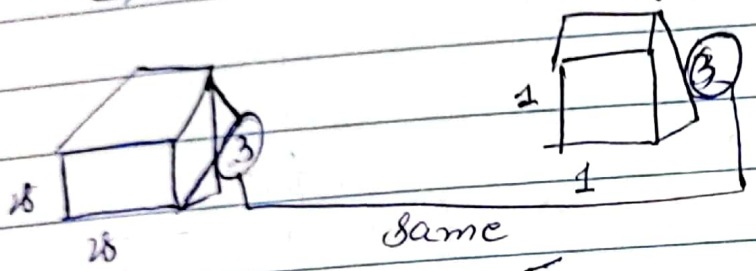
# of operations to apply 1 filter = 5×5×3 = 75

# of operations to apply 16 filters in 1st pass. → ×16 = 1200

Total no. of operations or multiplications = Output dimension × filter dimension

$= (24 \times 24 \times 16) \times (5 \times 5 \times 3)$

→ Pointwise convolution [(1×1) convolution]:
↳ reduce complexity / number of operation



$= \underbrace{28 \times 28}_{\text{same as input image}} \times 15$ ↳ no. of filter

# of filters = 15

# of operations/multiplication of one filter apply $= 1 \times 1 \times 3 = 3$

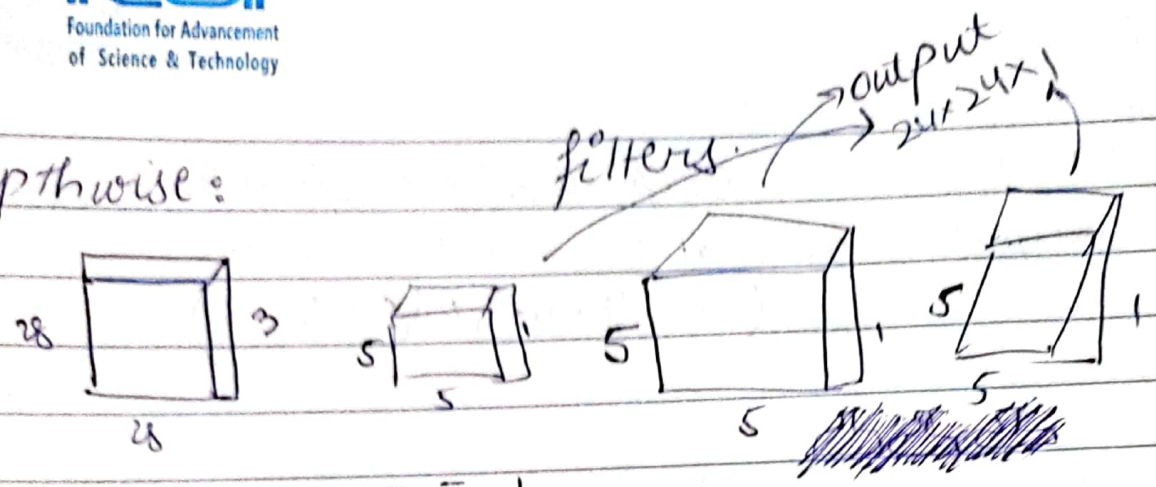
All filters $= (1 \times 1 \times 3) \times 15 = 45$

Total no. of operations = Output dimension × filter dimension

$= (28 \times 28 \times 15) \times (1 \times 1 \times 3)$

→ Depthwise Seperable Convolution:
↳ Depthwise convolution
↳ Pointwise convolution

• Depthwise:



$28 \times 28 \times 3$ $\quad$ $5 \times 5 \times 1$

# of operations in 1 pass $= 5 \times 5 \times 1 = 25$

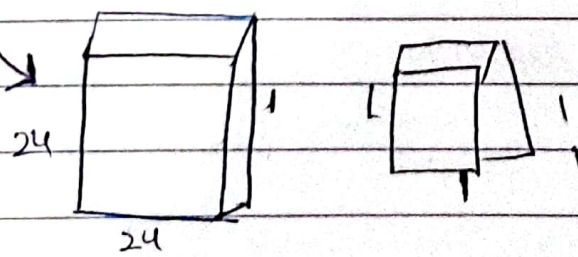
# of operations $= \underbrace{(24 \times 24 \times 1 \times 5 \times 5 \times 1)}_{\text{1 filter}} + \underbrace{(24 \times 24 \times 1 \times 5 \times 5 \times 1)}_{\text{2 filter}} + \underbrace{(24 \times 24 \times 1 \times 5 \times 5 \times 1)}_{\text{3 filter}}$

$= 43200$

Now, we will apply pointwise on the depthwise's output

when # of filters $= 256$

# of operations $(24 \times 24 \times 1 \times 5 \times 5 \times 1)(256)$



# of filters $= 256$

output $= (24 \times 24 \times 256)$

~~# of operations (one pass) $= 1 \times 1 \times 256 = 256$~~

Total # of operations $= 24 \times 24 \times 256 \times 1 \times 1 \times 1$
$= 147456$

depthwise

$\Rightarrow 28 \times 28 \times 3 \rightarrow 5 \times 5 \times 1 \rightarrow (24 \times 24 \times 1) \times 3$

org image — depth convolution

point wise convolution

$(24 \times 24 \times 256) \Leftarrow (1 \times 1 \times 256) \times 3$

output dimension — filter size — # of filters apply

# of operations in depthwise $= 3 \times (24 \times 24 \times 1) \times 5 \times 5 \times 1$
$= 43200$

# of operations in pointwise $= 24 \times 24 \times 256 \times 1 \times 1 \times 3$
$= 442368$

Total $= 43200 + 442368 = 485568$

→ with simple convolution

$28 \times 28 \times 3$

$\llcorner (5 \times 5 \times 3)$ # of filter $= 256$

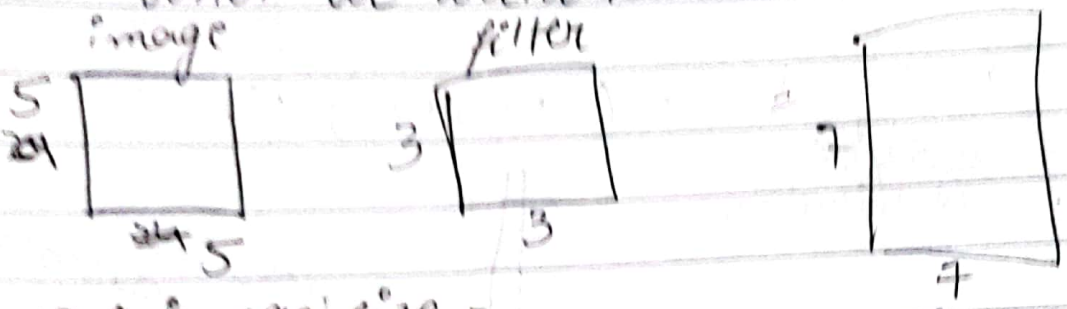
output $24 \times 24 \times 256$

# of operations $= 24 \times 24 \times 256 \times 5 \times 5 \times 3$
$= 18432000 > 485568$

better

→ Transpose Convolution:

When we want to increase the size of image



output image size =

(image width − 1) × stride + kernal width − 2p

$= (5-1) \times 1 + 3 - 2(0)$

$= 4 \times 1 + 3 = 7 \times 7$

Foundation for Advancement of Science & Technology

Lecture NO. 12

Dilated convolution → to increase the size of kernel

- dilated rate > 1 (l) → hyper-parameter
- addition of layers $(l-1)$ Ex: $l = 2$, $2-1 = 1$ layer will be added

1	2	1
9	6	7
8	3	4

→ kernel 3×3

1	0	2	0	1
0	0	0	0	0
9	0	6	0	7
0	0	0	0	0
8	0	3	0	4

→ kernel 5×5

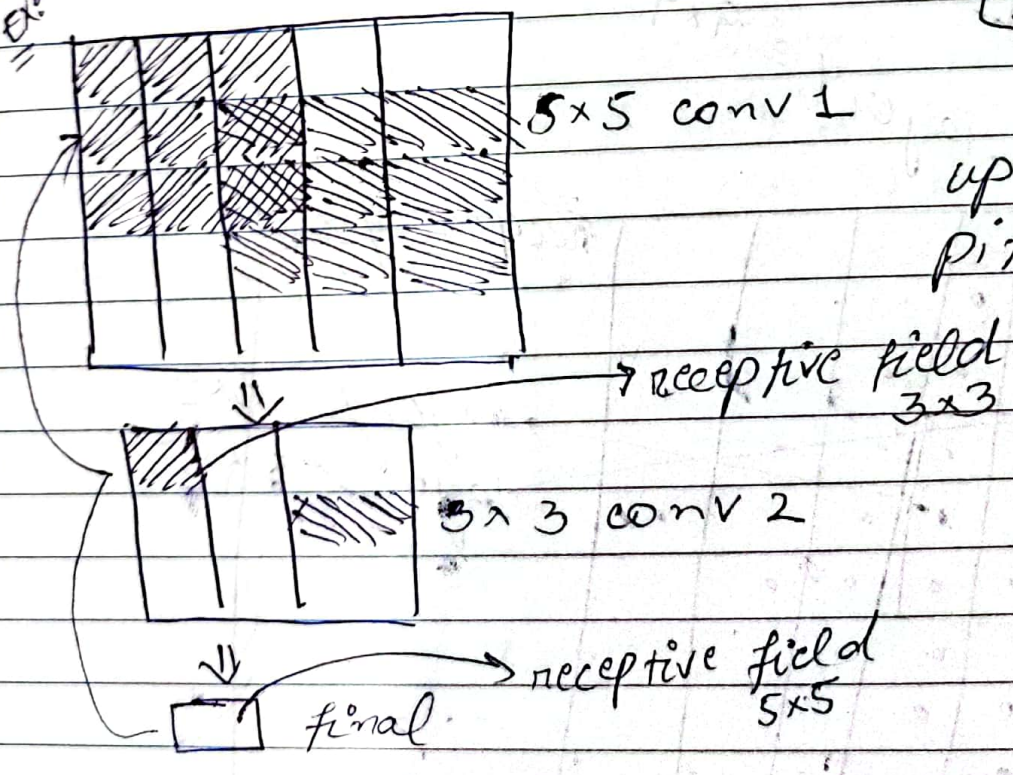
- dilated rate = 1, no layers will be added.

→ ignoring same features.

→ ~~Respective Field~~: Receptive Field

↳ 1 pixel of output is made up of how many pixels of the input (go backward)

Ex:



Ex:

image 7x7 — conv 1

⇓

image 5x5 — conv 2 → receptive field 3x3

⇓

image 3x3 — conv 3 → receptive field ~~3x3~~ 5x5

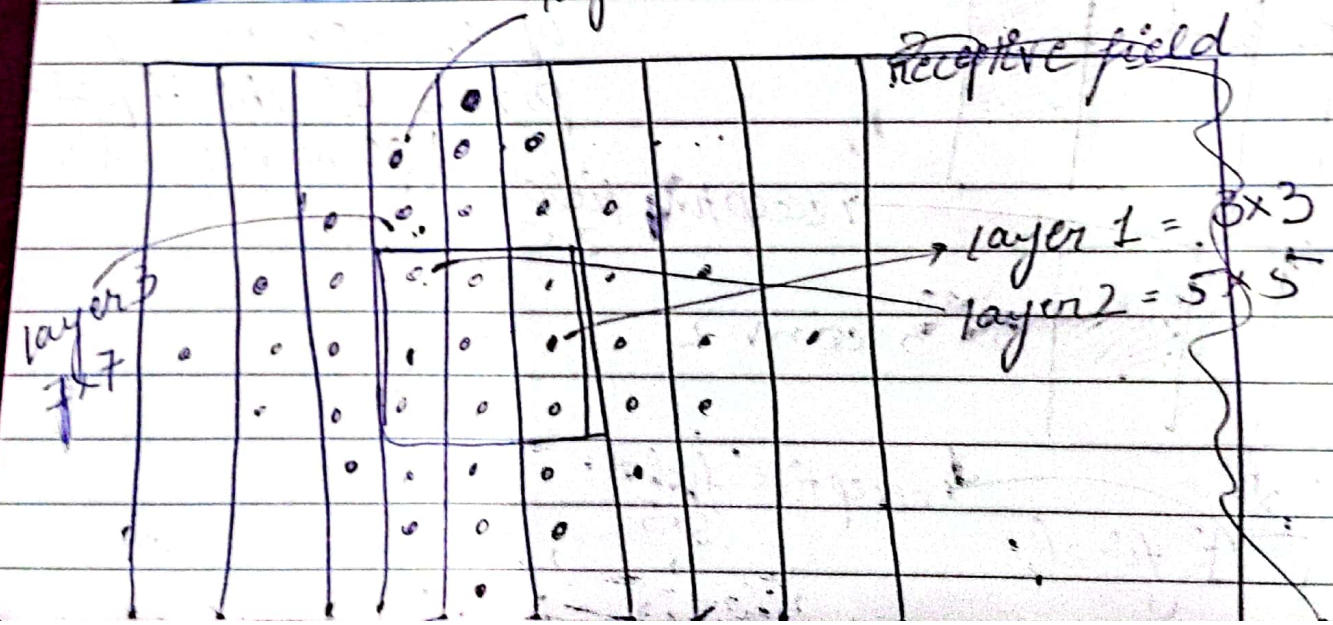
⇓

[ ] → final → receptive field 7x7

Ex:

0	1	0
1	1	1
0	1	0

layer 4 = 9x9



Transfer Learning

- using a pre-trained model
  ↳ e.g vgg-16 (16 layers (conv + FC)), imageNet dataset (1.4 Billion, 20,000 classes). Has Bounding Boxes
  uses subset (10 lakhs, 1000 categories). After model trainning, we get weights.
- we do not train model ~~for~~ from scratch to solve a new classification. Instead, we use the learned weights of vgg-16 on the dataset.

- Transfer Learning (ways of doing)

Feature Extraction
(when the classes of trained data is also ~~possible~~ present in the original ~~y~~ our own input data).

Fine-Tuning.
↳ when the thing to classify is not present in the ~~data~~ of ~~pre~~trained model based on the weights & input.

Foundation for Advancement of Science & Technology

Lecture No. 13

Image Segmentation → based on pixels.

→ give similar colors to similar pixels. It is not classification.

→ Types of Image Segmentation:

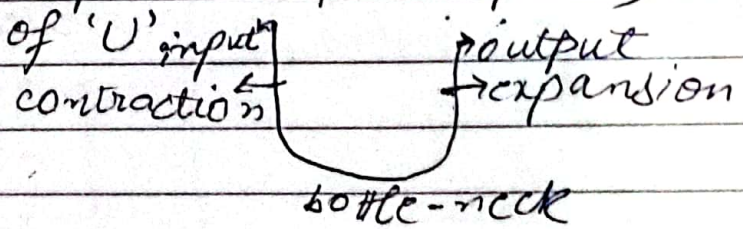
- Semantic Segmentation. → pixel wise
  classify each pixels into category.
- Instance Segmentation:
  Identifies each instance of an object in an image
- Panoptic Segmentation:
  A combination of both semantic and instance segmentation

→ What is the need?

- Automation, scalability, accuracy.
- Ability to learn complex patterns from data

→ Networks used for Image Segmentation:

- UNET (Contraction & Expansion of the path).
  ↳ It is in the shape of 'U'



↳ after bottle-neck, upsampling is used for expansion. (upsampling done in transpose

# of parameters = (filter height × no. of filter in prev layer + 1) × no. of filter in curr layer

Foundation for Advancement of Science & Technology

• image size smaller → increase depth.

convolution) • It is done to increase the size i.e expansion.

UNet

(~~UEF~~ example)

TO find:

# of dimensions

# of parameters

→ image = 572 × 572, 3 → no. of channel

⇓ → 3×3 convolution (applied) → depth = 3

~~570×5070~~ no. of filters = 1

570 × 570 × 1

parameters = (3 × 3 × 3 + 1) × 1 = 28

new size = $\frac{n - f + 2p}{s} + 1 = \frac{572 - 3 + 2(0)}{1} + 1$

= ~~569~~ 570

new size = $\frac{570 - 3 + 2(0)}{1} + 1 = 568 \times 568$

→ After Max pooling

Dimensions/size = $\frac{568}{2}$ = ~~34~~ 284

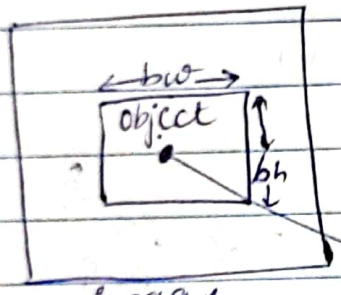
(continue from slides)

Foundation for Advancement of Science & Technology

Lecture No. 14
Image Segmentation

- Semantic
- instance
  - ⇓
  - instance (RCNN)
  - (YOLO) detection
- Panoptic

→ classification with localization (1-Object)



Image

center of the object (b_x, b_y)

$$y = \begin{bmatrix} P_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

P_c → 0 − 1

b_x, b_y → centre of object

b_h − height of object

b_w − width of object

c_1, c_2, c_3 → classes

0-7

⇓

(y) → class

⇓

cat, dog etc, pedestrian etc.

• If object exits

$$y = \begin{bmatrix} 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1, c_2, c_3 \end{bmatrix}$$

→ If object doesn't exist

$$y = \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix}$$

∵ ? → Dont care Terms

P_c → Probability that an projected object exists

- $\text{Loss} = \frac{1}{m} \Sigma (y - \hat{y})^2$

Loss: if $P_c = 1$

$$= (y_1 - \hat{y}_1)^2 + (y_2 - \hat{y}_2)^2 + \dots (y_8 - \hat{y}_8)^2$$

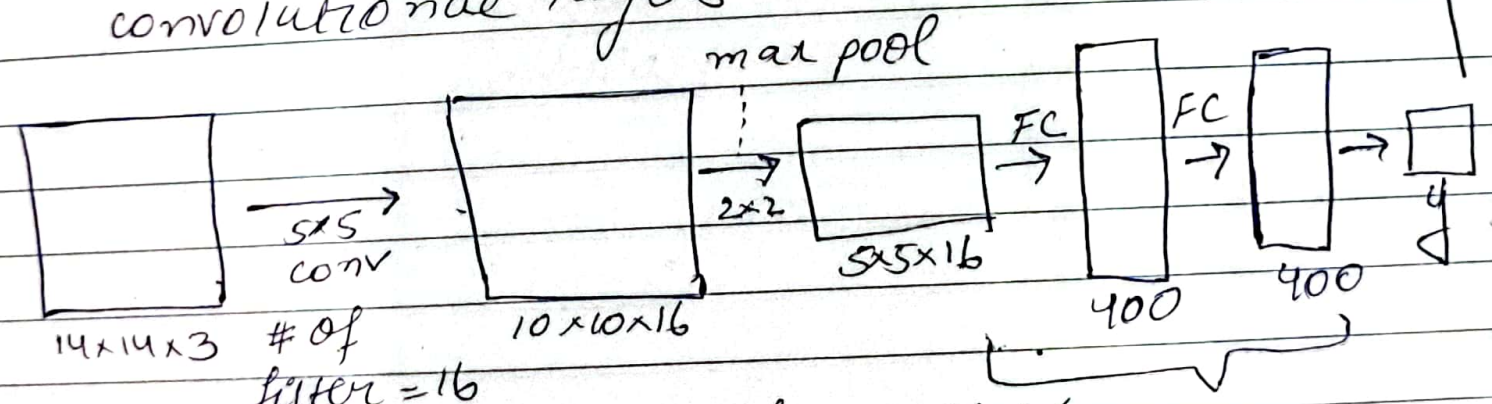
Loss: if $P_c = 0$

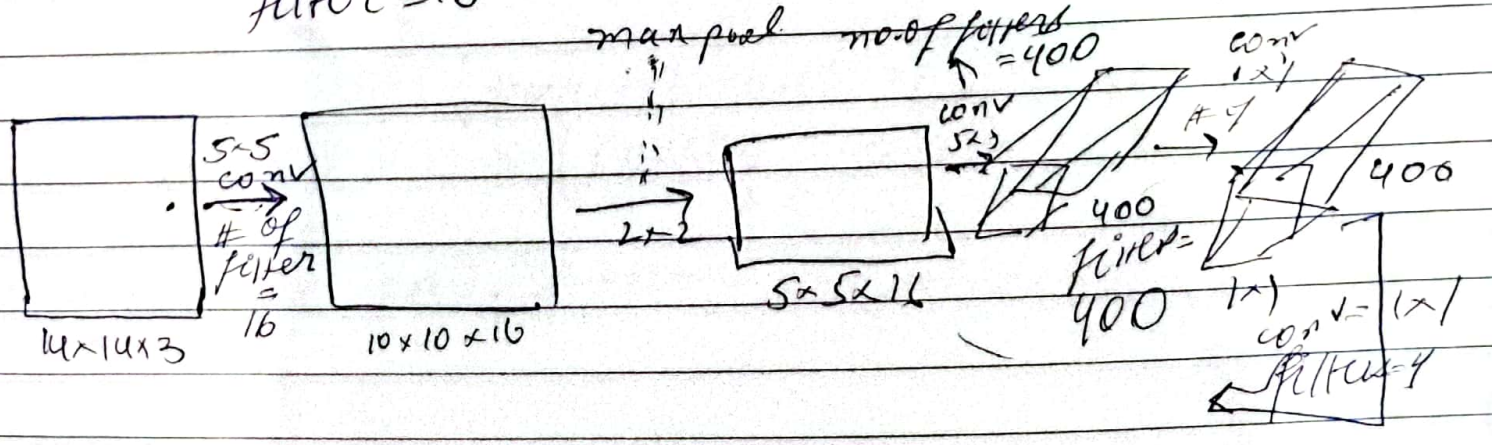
$$= (y_1 - \hat{y}_1)^2$$

Actual ↲ → Predicted

→ Object Detection (multiple objects)

- Sliding Window concept introduced & developed
- we have to convert dense layer into convolutional layers.





Foundation for Advancement of Science & Technology

- We will ~~apply convolution on~~ apply convolution on sliding ~~mata~~ window. ~~Mtrxo~~ ~~swipan~~

measure the overlap b/w 2 bounding boxes

Intersection over Union (IOU)

Lecture No. 15

$x_1^I = \max(x_1^A, x_1^B)$

$y_1^I = \max(y_1^A, y_1^B)$

$x_2^I = \min(x_2^A, x_2^B)$

$y_2^I = \min(y_2^A, y_2^B)$

$A \cap B = (x_2^I - x_1^I) \times (y_2^I - y_1^I)$

$A \cup B = \text{Area } A + \text{Area } B - A \cap B$

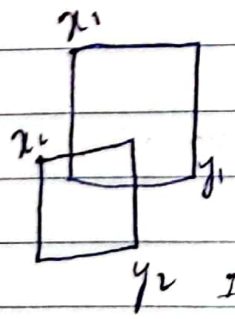
$\text{Area } A = (x_2^A - x_1^A) \times (y_2^A - y_1^A)$

$\text{Area } B = (x_2^B - x_1^B) \times (y_2^B - y_1^B)$

$\text{width} = (x_2^I - x_1^I)$ $\quad$ $\text{height} = (y_2^I - y_1^I)$

if either is negative, then there is no overlapping

$$IOU = \frac{A \cap B}{A \cup B}$$



IOU = 1 (ground truth & predicted overlap)

IOU ≥ 0.5 (pretty decent => correct)

- Higher the IOU, better the accuracy

(Quantized) ROI Pooling → Faster RCNN
(Non-Quantized) ROI Align → Mask-RCNN

Foundation for Advancement of Science & Technology

Lecture NO.16

RCNN - Regional Convolutional Neural Network

2-stage object detection method.

→ RCNN Architecture:
- Region Proposal
- Feature Extraction → feature map
- Classification (for classification: using SVM)
- Bounding Box Regression

→ Faster RCNN
↳ uses a "Region Preposal Network" (RPN)
↳ detects region using anchor boxes.

→ Feature Map:
feature of the input image obtained from a deep convolutional network.

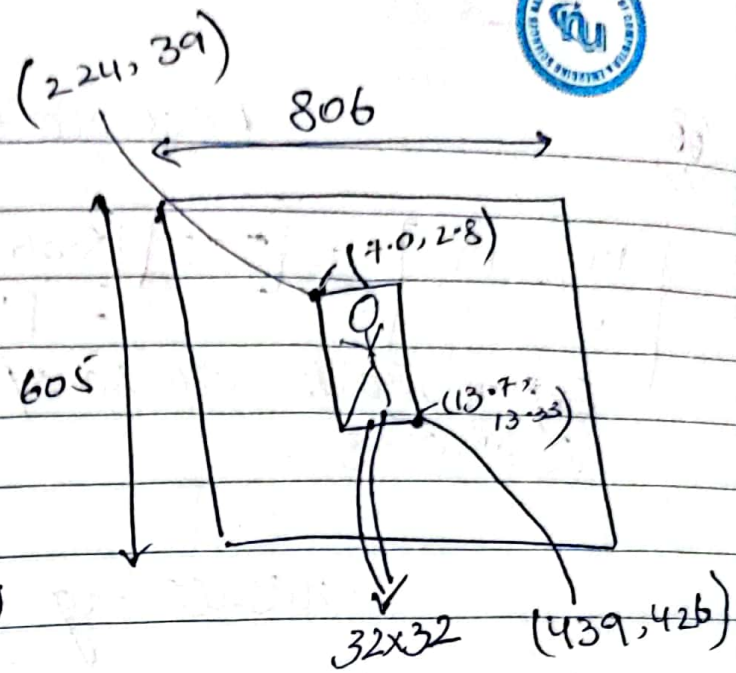
→ Masked RCNN
↳ Diff is that it uses ROI Align.

→ ROI Align:
→ Align steps
- ROI Mapping
- ROI division

CamScanner

→ Steps for RoI Align:-

- RoI Mapping
- RoI division
- Interpolation
- Max Pooling at RoI



(224, 39) (439, 426)

$$\left(\left|\frac{224 \times 1}{32}\right|, \left|\frac{89 \times 1}{32}\right|\right) = (7.0, 2.8)$$

$$\left(\left|\frac{439 \times 1}{32}\right|, \left|\frac{426 \times 1}{32}\right|\right) = (13.7, 13.3)$$

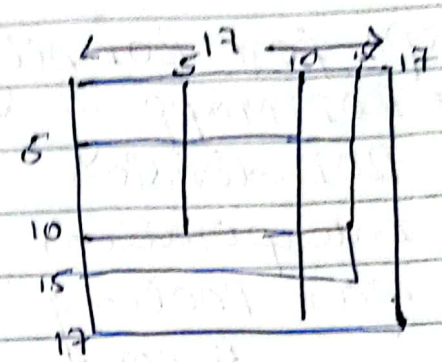
required size of image

In example, scaling factor is stride
In VGG-16, by default the stride is 32

Lecture NO. 17

$$\left[\frac{17}{3}, \frac{17}{3}\right] \therefore [5, 5] \quad [5.66, 5.66] = [5, 5]$$

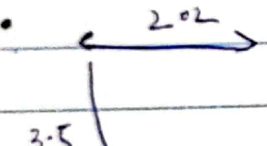
Every block will be of size 5×5



∴ resulting in info loss

For using all info, take/use actual value i.e 5.66

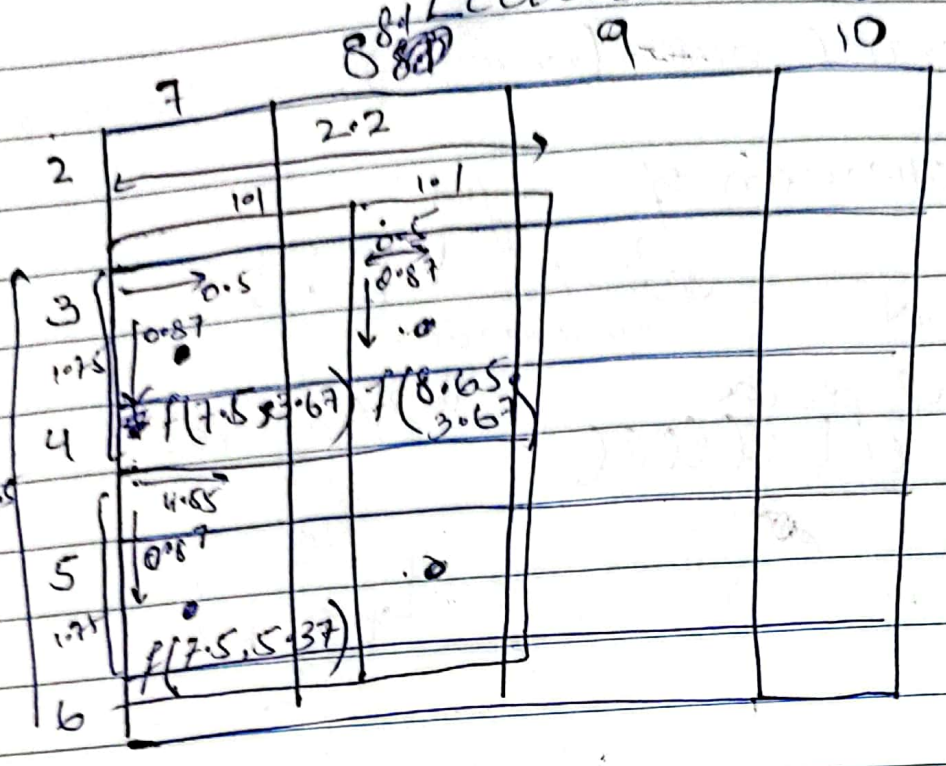
→ Linear Interpolation (Step 3 of RoI Align)



$$x = \underbrace{7.0}_{\text{staring coordinate}} + \underbrace{\left(\frac{2.2}{\beta = 2}\right) \times (1 + 0.5)}_{\text{displacement}} = 8.65$$

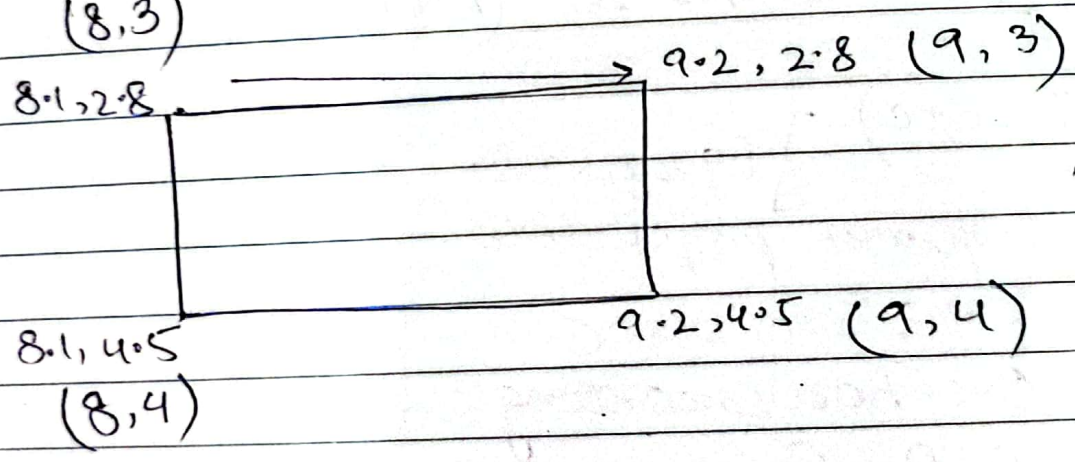
$$y = \underbrace{2.8}_{\text{staring coordinate}} + \underbrace{\left(\frac{3.5}{\beta = 2}\right) \times (0 + 0.5)}_{\text{displacement}} = 3.67$$

Lecture NO.18



•) We choose center point to balance the values intensity.

→ Horizontal & Vertical Interpolation



The intensity of the co-ordinates will be known before

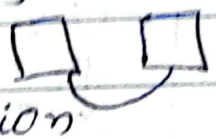
co-ordinates are discrete

Foundation for Advancement of Science & Technology

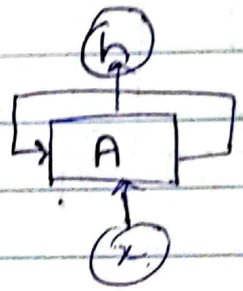
Lecture NO. 19

- ANN => not sequential data (numeric)
- CNN => images
  - ↳ collection of images
  - ↳ RCNN
  - ↳ Faster RCNN

  spatial information ⇓ ↳ features



- RNN => sequential data
  audio = lllllll



Problems of ANN:
→ variable length
→ doesn't handle sequence

Lecture NO. 20 (RNN)

$$h_t = g(Vx_t + Uh_{t-1} + c) \quad \text{(tanh)}$$
$$\hat{y}_t = Wh_t + b \quad \text{(tanh/sigmoid, softmax preferable)}$$
} Equations.

Ex:

vocab: One-hot Encoding

I →	1	0	0	0
love →	0	1	0	0
AI →	0	0	1	0
chat →	0	0	0	1

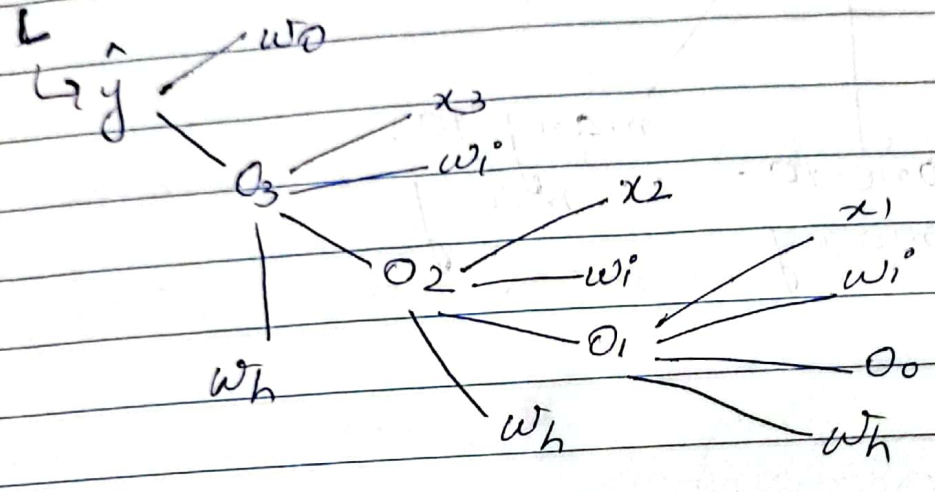
CamScanner

Foundation for Advancement of Science & Technology

Back-propagation over time:

$$\hat{y} = \text{sigmoid}(O_3 \cdot w_o)$$
$$O_3 = \tanh(x_3 w_i + O_2 w_h + b)$$
$$O_2 = \tanh(x_2 w_i + O_1 w_h + b)$$
$$O_1 = \tanh(x_1 w_i + O_0 w_h + b)$$

Loss (L)



$$\frac{\partial L}{\partial w_o} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial w_o}$$

$$\frac{\partial L}{\partial w_i} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial O_3} \cdot \frac{\partial O_3}{\partial w_i} + \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial O_3} \cdot \frac{\partial O_3}{\partial O_2} \cdot \frac{\partial O_2}{\partial w_i}$$
$$+ \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial O_3} \cdot \frac{\partial O_3}{\partial O_2} \cdot \frac{\partial O_2}{\partial O_1} \cdot \frac{\partial O_1}{\partial w_i}$$

$$= \left(\frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial O_t} \cdot \left(\prod_{i=2}^{i=t} \frac{\partial O_i}{\partial O_{i-1}} \right) \cdot \frac{\partial O_1}{\partial w_h} \right)$$

$$\Downarrow$$

$$\frac{\partial O_2}{\partial O_1} \cdot \frac{\partial O_3}{\partial O_2}$$

- Problem in RNN
  - ↳ vanishing Gradient

types

- one-to-one
- one-to-many
- many-to-many

CamScanner

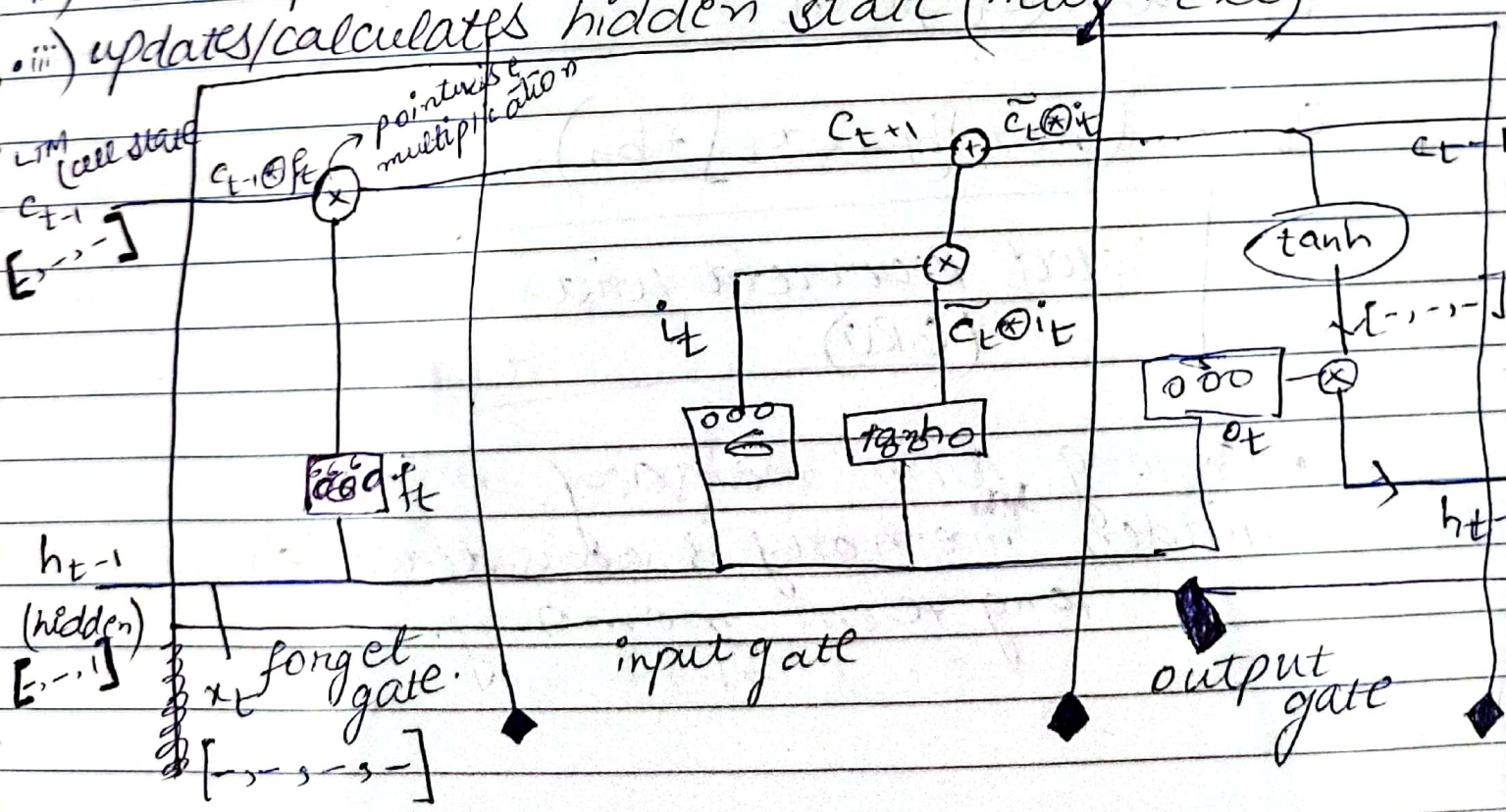
Foundation for Advancement of Science & Technology

Lecture No. 20
LSTM

→ Gates
- i) forget gate
- ii) input gate
- iii) update gate
  - ↳ hidden (update state)

→ States
- → Long Term Memory (LTM) → (cell state)
- → Short Term Memory (STM)

- i) LTM → remove based on input
- ii) insert info in LTM → based on current input
- iii) updates/calculates hidden state (new/next)



→ Equations:

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

$$\tilde{C}_t = C_{t-1} \otimes f_t$$

$$\tilde{C}_t = \tanh(w_c * [h_{t-1}, x_t] + b_c)$$

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

$$C_t = \underbrace{C_{t-1} \otimes f_t \oplus \tilde{C}_t \otimes i_t}_{\tilde{C}_t}$$

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

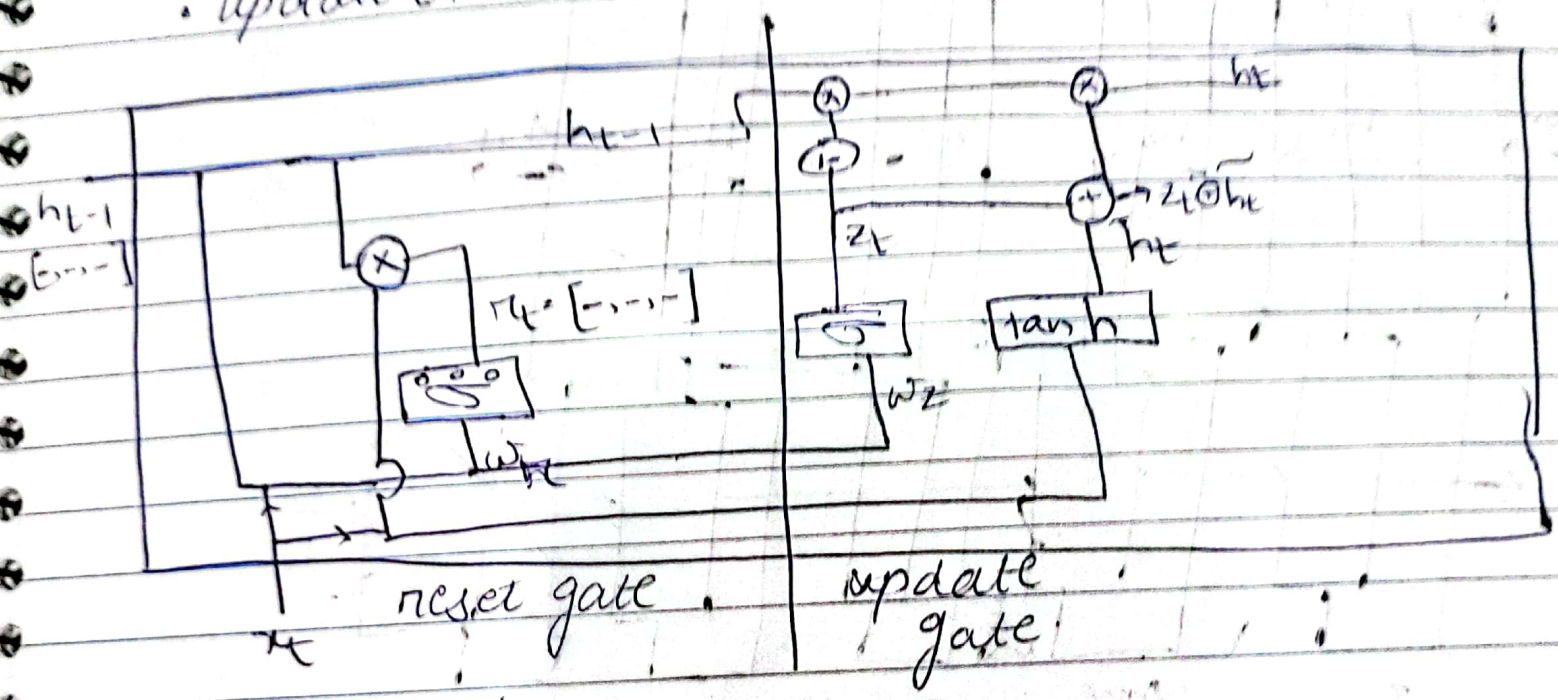
$$O_t = \sigma(w_o * [h_{t-1}, x_t] + b_o)$$

Gated Recurrent Unit (GRU)

- No long term memory
- Hidden memory is considered as long term memory.

→ Gates
- Reset Gate
- update Gate

→ State
- hidden state



→ Equations

$$r_t = \sigma(w_r [h_{t-1}, x_t] + b_r)$$

$$h_t = h_{t-1} \otimes r_t$$

$$\tilde{h}_t = \tanh(W_c [h_t, x_t] + b_h)$$

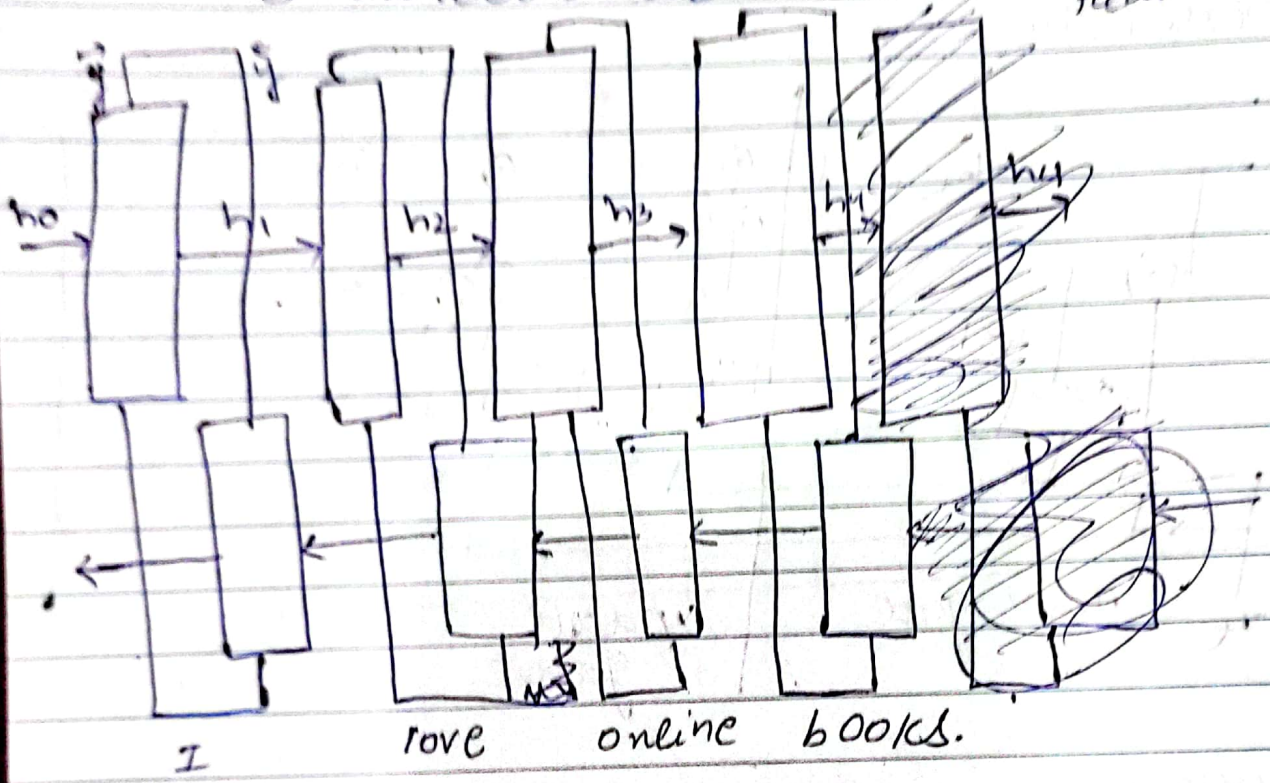
$$z_t = \sigma(w_z [h_{t-1}, x_t] + b_z)$$

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

Foundation for Advancement of Science & Technology

Lecture NO. 21

Bi-directional LSTM → doesn't work on real-time data



Ex:

I love amazon, because it is a great website
" " " " " " " " forest

→ organization

↳ area

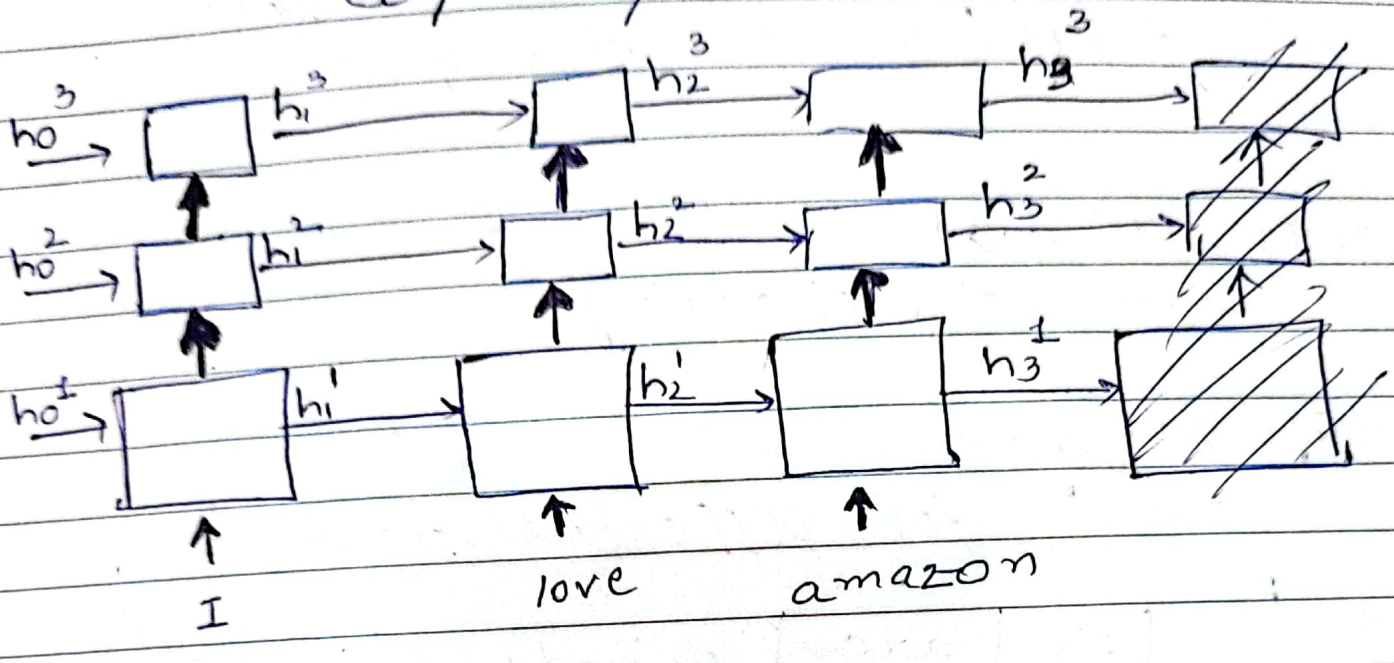
Equations

$$\vec{h}_t = \tanh\left(\vec{w}\left[\vec{h}_{t-1}, \vec{x}_t\right] + \vec{b}\right)$$

$$\overleftarrow{h}_t = \tanh\left(\overleftarrow{w}\left[\overleftarrow{h}_{t+1}, \overleftarrow{x}_t\right] + \overleftarrow{b}\right)$$

Foundation for Advancement of Science & Technology

Deep RNN/LSTM



Attention

Self-Attention

↳ Bahadanau Attention

↳ Luong Attention

Lecture No. 23

→ For similarity, we use approach

→ Multi-Head Attention and [illegible]

→ Masked Multi-Head

Eg. weather changing

→ Cross Attention

↳ used for different sequences.

Decoder				(key, value) Encoder
	we	are	friends	
	S_{11}	S_{12}	S_{13}	
	S_{21}	S_{22}	S_{23}	
	S_{31}	S_{32}	S_{33}	

→ To preserve sequence, we use positional encoding.

Lecture NO. 23

→ For similarity, we use dot product.

→ Multi-Heads Attention [illegible]

→ Masked Multi-Head.
↳ to avoid cheating.

→ Cross Attention
↳ used for different sequence.

Decoder

we	are	friends
S_{11}	S_{12}	S_{13}
S_{21}	S_{22}	S_{23}
S_{31}	S_{32}	S_{33}

(key, value)
Encoder

→ To preserve sequence, we use positional encoding.