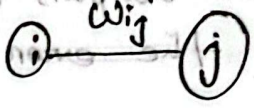
Final

Backpropagation:

$$\Delta w_{ji} = \eta \times \delta_j \times y_i$$



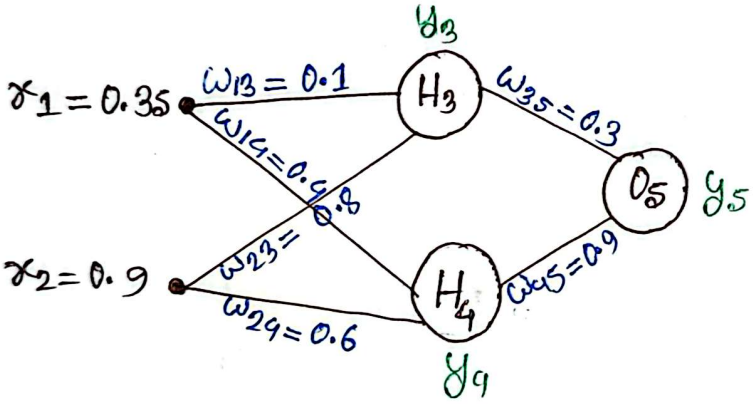
Error unit calculation:

output layer, $\delta_j = y_j(1-y_j)(y_{target} - y_j)$

hidden unit, $\delta_j = y_j(1-y_j)\sum \delta_j w_{ij}$

AI by hand

Math for Backward Propagation:



Forward pass:

$$y_3 = \sigma[(0.35 \times 0.1) + (0.9 \times 0.8)] = 0.68$$

$$y_4 = \sigma[(0.35 \times 0.4) + (0.9 \times 0.6)] = 0.6637$$

$$y_5 = \sigma[(y_3 \times w_{35}) + (y_4 \times w_{45})]$$

$$= \sigma[(0.68 \times 0.3) + (0.6637 \times 0.9)] = 0.69$$

Error unit calculation:

$y_5(1-y_5)(y_{target} - y_5)$

$$\delta_5 = 0.69(1-0.69)(0.5-0.69)$$

$$= -0.040691$$

$$\delta_3 = 0.68(1-0.68)(0.3 \times -0.0406) \longrightarrow y_3(1-y_3)(w_{35} \times \delta_5)$$

$$= -0.00265$$

$$\delta_4 = 0.6637(1-0.6637)(0.9 \times -0.0406) \longrightarrow y_4(1-y_4)(w_{45} \times \delta_5)$$

$$= -0.0082$$

Updated weights

$$\Delta w_{ij} = \eta \delta y_i \quad (y_i \rightarrow \text{input})$$

$$w_{ij}(\text{new}) = w_{ij}(\text{old}) + \Delta w_{ij}$$

i	j	w_{ij}	δ_j	x_i (input)	η	Updated w_{ij}
1	3	0.1	−0.00265	0.35	1	$w_{13} = 0.0991$
1	4	0.4	−0.0082	0.35	1	$w_{14} = 0.3971$
2	3	0.8	−0.00265	0.9	1	$w_{23} = 0.79788$
2	4	0.6	−0.0082	0.9	1	$w_{24} = 0.59262$
3	5	0.3	−0.040691	0.68	1	$w_{35} = 0.2729$
4	5	0.9	−0.040691	0.6637	1	$w_{45} = 0.8731$

Types of Neural Network: → Geeksforgeeks

⇒ Feedforward Neural Network
(training is being done using backpropagation)
When to use? ← for every type.

⇒ Convolutional Neural Network.
Convolution operation (has convolutional layer)
↳ mathematical operation.

Convolution Layer + Max-pooling

(works with grid-type data) → pixel-value.
↳ image, video.

⇒ Recurrent Neural Network. (has memory)
↳ has sequential dependency.
Problem with RNN

⇒ Long Short Term Memory ← LSTM.

⇒ Gated Recurrent Units ← GRU.
(little less parameters than LSTM)

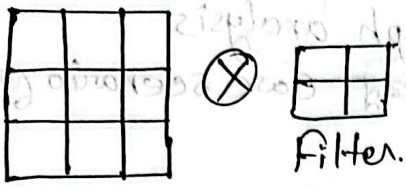
⇒ Radial Basis Function Networks

⇒ Generative Adversarial Network
Generator + Discriminator.
(generates synthetic / fake data)

Attention Mechanism → Technique of Neural Network.
↳ came from human cognition.
↳ in dL enhancing model performance
medium.com
→ transformer made with it.

CNN-

Convolution operation on Images.



Image

filter << Image

Padding

Kaggle — Deep_neural_network

Curve analysis.

Validation data → is used while training

- Training
- Testing

Verbose=0

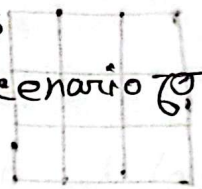
↳ প্রতি epoch এর loss দেখাবে না,

Convolutional
works well on grid-type data

Model ⇒ wider
⇒ deeper

Early stopping use করে বা না করে } Graph analysis
বিভিন্ন case scenario তে

{ Loss curve
accuracy curve.



Introduction to TensorFlow and Keras to build and train neural networks for structured data.

Report

→ Dataset details
→ variation of model, output graph, what are the findings of various model. (overfitting on not) then use early stop then result. } code snipest adding.

CNN – Convolutional Neural Network

Convolution layer → Pooling layer → Fully-connected layer.

→ down-sample (to extract feature)

→ pixel

Pooling layer → Flatten layer (matrix-কে vector-এ convert করে)

→ Binary image – (0 or 1)

→ Grey-scale image – (0-255)
→ pixel value

→ RGB image / Color image
(3 channel)

→ 3 channel. (2×2×3)
R-G-B

Feature mapping → Region-wise matching.

27-slide

⇒ will be Computational costly if convolutional layer, pooling layer are not used.

e.g. 1,20,000 pixels —(down sampling)→ 100 pixels

→ ReLU (pixel এর minus value থাকলে মুছে দেয়)

Convolutional layer → Pooling layer → Flatten layer (2D matrix কে 1D তে convert করে)

* Filter কে move করায় → known as stride (কয়টা ঘর করে move করবে)

* Diff. filter দিয়ে যে output পাই → feature maps

* 3X3X(1) → filter সংখ্যা 1

filter.

→ center by convention, hyperparameter can be changed.

(X) = sign for convolution operation

* feature map size = 1 + ((W) − (F) + 2(P))/(S) — Size. = ReLU layer.

(F) = filter size; (W) = Picture size; (P) = Padding

* Pooling layer → pools out important pixel (also works as filter) → filter doesn't have any value.

→ Max-pooling
→ Average pooling.

} Reduces Size

* CNN — uses: transfer learning approach
  - LeNet
  - AlexNet
  - ZFNet
  - VGGr
  - GoogleNet
  - ResNet

If I want to use transfer learning, image size has to be same! (disadvantage)

VGG16

CNN Explorer:

Dropout and Batch Normalization:

→ Dropout layer
→ Normalization layer

~~Feature Engineering:~~

Computer Vision | Kaggle

☆ The convolutional classifier:

☆ Convolution and ReLU

☆ Maximum pooling.

Custom Convnets (kaggle)

Random flip
Increasing contrast

Data augmentation:
(data বাড়ানোর কৌশল)

Padding → flip
→ Same

1,996

Parameter lock/freeze

How??

base.trainable=false

আগের কাজে:

Dropout, normalization

Image data chosing, multi-class classification.

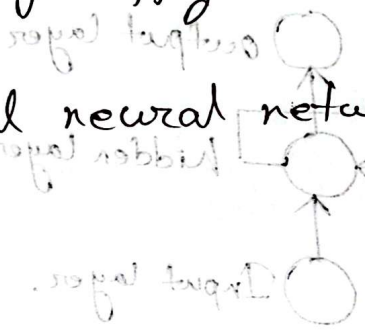
→ convolutional model → loss, accuracy

→ pre-trained → loss, accuracy.

→ data augmentation → loss, accuracy

→ early-stopping

Build convolutional neural networks with tensorflow and Keras.



RNN — Recurrent Neural Network

(Sequential type data e.g. keyboard suggestion)

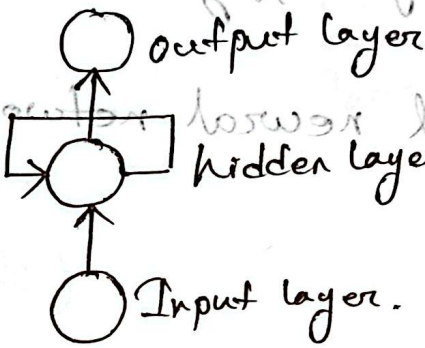
I am

Image dimension = 2 (height, width)

Video = 3 (height, width, time)

RNN application and basic structure:

⇒ Google's autosearch
⇒ Sentiment classi.
⇒ Machine translation



FNN → can't handle sequential data
→ can't memorize previous.

Architecture of RNN (medium)

Understanding RNN

String value (categorical value) has to be converted through encoding → Numerical value.

Backpropagation through time:

general workflow:

adding the loss through all layers.

RNN problem — vanishing gradient.
↳ can't handle long dependency.

Types of RNN architecture:
- Classification (one to one)
- Image caption: (one to many)
- Sentiment ana. (many to one)
- (many to many)

Advantage:
effective pixel neighborhood (uses previous pixel into to calculate the value of current pixel)

Variation of RNN → BiNN
→ LSTM

16.09.25.

Parveen Raj (Medium.)

Varishing and Exploding gradient problem of gradient descent } — RNN has it.

Introduction_to_long-term_dependency.

LSTM

↳ has dedicated memory cell.
(stores important info.)

★ has 3 gate — controls information flow

used to memory update.

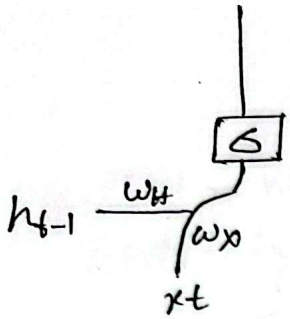
↳ input gate (decides if the coming info is important or not to store in memory) — how much.

↳ forget gate (if the stored information is further required or not)

↳ output gate (decides if any of the info of memory is required to generate prediction)

(9) → cell state and 3 gates.
(Memory cell)

Foget gate:



$$f(f) \sigma (w_x x_t + h_{t-1} w_H + b_f)$$

Input gate: (has 3 part)

→ Sigmoid layer
→ tanh layer.

} decides which to keep how much to keep.

Output gate:

Sigmoid layer — decides it will use anything from memory.

tanh layer — if needed it comes from memory.

GRU (based on LSTM)

Mastering next word prediction: (medium) — NLP domain.
AI by hand: (RNN by hand)

NLP — Embedding layer.
padsequence (to make all the words of same length)

Sequence creation — n gram sequence

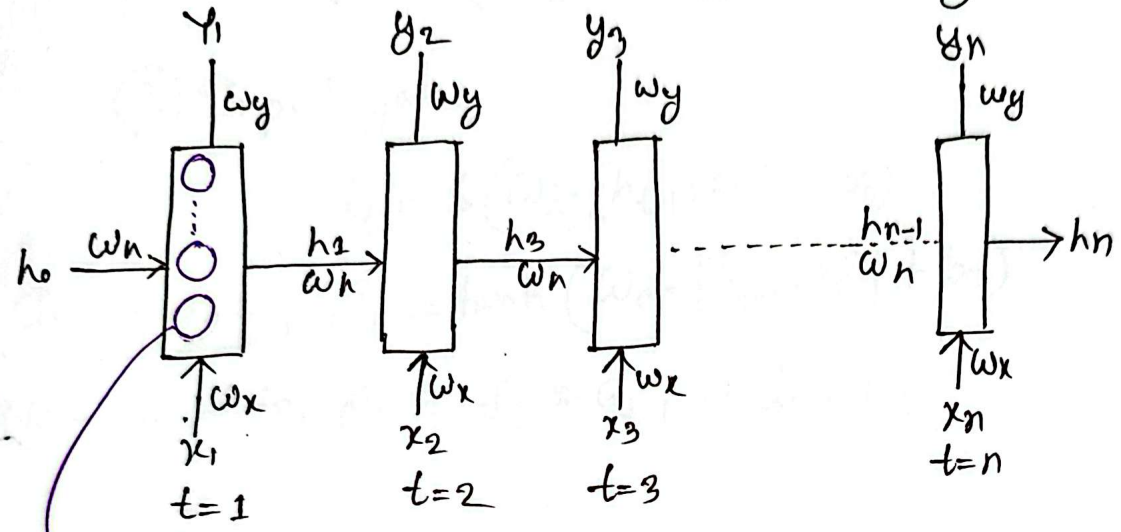
model summary
input and
output shape

Bengali word prediction
— Bangla tokenizer use করে
করতে হবে।

* Bangla next word prediction using RNN.

RNN (from 'Medium')

To remember the sequential information
RNN maintains 'hidden state'
also referred as 'memory state'



$h \rightarrow$ activation has been applied.

$h_0 \rightarrow$ vector of 0's

$W_x, W_y, W_h \rightarrow$ shared throughout the entire network.

First, one-hot encoding is applied.

hidden nodes, $a_t = W_h h_{t-1} + W_x \cdot x_t$

hidden state, $h_t = \tanh(a_t)$

prediction at time t, $y_t = \text{softmax}(W_y \cdot h_t)$

→ for multiclass classification.

W_h = square matrix.

each hidden node has a bias

LSTM (Medium)

1. Cell state (Memory cell)
2. Forget gate:

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

3. Input gate:

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

Intermediate cell state, $\bar{C}_t = \tanh\left(W_c \cdot [h_{t-1}, x_t] + b_c\right)$

New cell state, $C_t = f_t * C_{t-1} + i_t * \bar{C}_t$

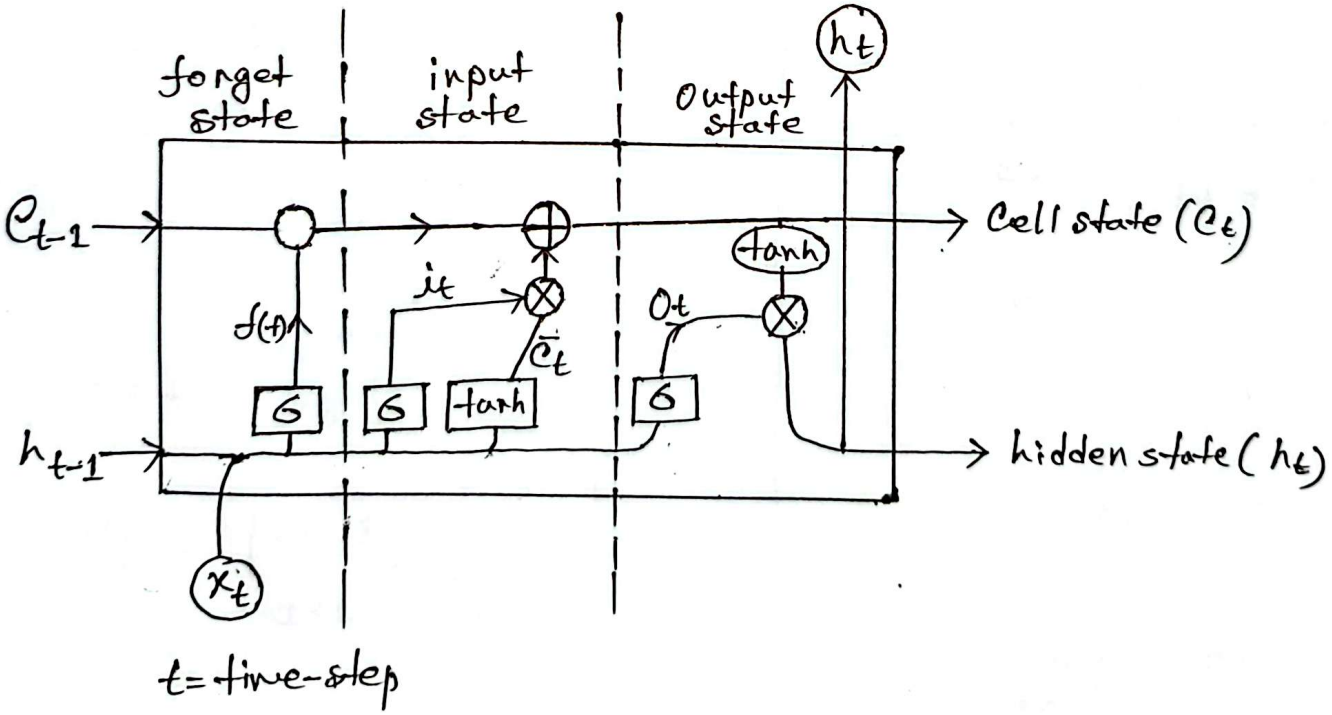
4. Output gate:

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

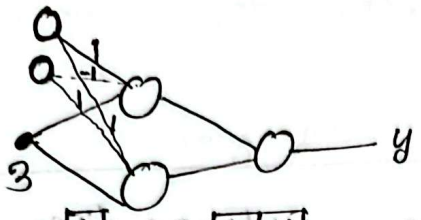
$$h_t = O_t * \tanh(C_t)$$

⊗ pointwise mul

⊕ pointwise add



input sequence, মানের input -গুলোর?



A $\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$ w_h

B $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ w_x

C $\begin{bmatrix} -1 & +1 \end{bmatrix}$ w_y

x_1 $\boxed{3}$

h_0 $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$

$$a_1 = w_x \cdot x_1 + w_h \cdot h_0$$

$$= \underset{2\times1}{\begin{bmatrix} 1 \\ 2 \end{bmatrix}} \underset{1\times1}{3} + \underset{2\times2}{\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}} \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$= 6 + 0 = 6$$

$$h_t = ReLu(a_1) = ReLU(6) = 6$$

$$y_t = w_y \cdot h_t$$

$$= \underset{1\times2}{\begin{bmatrix} -1 & 1 \end{bmatrix}} \underset{2\times1}{\begin{bmatrix} 3 \\ 6 \end{bmatrix}}$$

$$= (+3)$$

$$h_t = \begin{bmatrix} 0 \\ 28 \end{bmatrix}$$

$$y_t = \underset{1\times2}{\begin{bmatrix} -1 & 1 \end{bmatrix}} \underset{2\times1}{\begin{bmatrix} 0 \\ 28 \end{bmatrix}}$$

$$= 28$$

$$=$$

(5) input

$$h_{t-1} = \begin{bmatrix} 1 \\ 17 \end{bmatrix}$$

$$a_3 = \underset{2\times1}{\begin{bmatrix} 1 \\ 2 \end{bmatrix}} \underset{1\times1}{5} + \underset{2\times2}{\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}} \underset{2\times1}{\begin{bmatrix} 1 \\ 17 \end{bmatrix}}$$

$$= \begin{bmatrix} 5 \\ 10 \end{bmatrix} + \begin{bmatrix} 1\cdot1 + -1\cdot17 \\ 1\cdot1 + 1\cdot17 \end{bmatrix}$$

$$= \begin{bmatrix} 5 \\ 10 \end{bmatrix} + \begin{bmatrix} -16 \\ 18 \end{bmatrix}$$

$$= \begin{bmatrix} -11 \\ 28 \end{bmatrix}$$

Source — column
desti — row
natio

$W_x = [3 \times 4]$

$W_H = [3 \times 3]$

$W_y = [1 \times 3]$

Data annotation / labelling

One-hot encoding

Nominal encoding → if it has ordinal value.

Tokenization has problem

frequency বেশি যেসব ওয়ার্ডের সেগুলো নিয়ে নির্বাচন dictionary size.

max-length - এ convert padding.

flatten layer ≈ embedding layer (in NLP)

input_length = max_seq
(কম হলে padding করবে)
বেশি হলে truncate করবে

⇒ lstm for video sentiment analysis.

⇒ Human activity recognition. using LSTM-CNN (Tanmay Chauhan)

⇒ Video dim. (3)
time — temporal, dimension.

Introduction to long term memory (LSTM) — analytics vidya. medium.

RNN by hand: (math problem)

Input sequence : X [3 4 5 6]

Parameters A $\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$ (w_h) B $\begin{bmatrix} 1 \\ 2 \end{bmatrix}$ (w_x) C $\begin{bmatrix} -1 & 1 \end{bmatrix}$ (w_y)

Activation function : ReLU

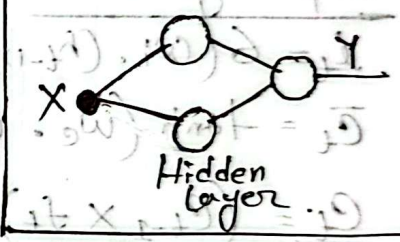
Hidden states H_0 $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$

Output sequence: Y [3 16 28 40]

$a_t = w_h h_{t-1} + w_x x_t$

$h_t = \tanh(a_t)$

$y_t = w_y \cdot h_t$



y_1 (at)

$$\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix}[3] = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 3 \\ 6 \end{bmatrix} = \begin{bmatrix} 3 \\ 6 \end{bmatrix}$$ then ReLU

2X2 2X1 2X1 1X1

= 3

y_2

$$\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} 3 \\ 6 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix}[4] = \begin{bmatrix} -3 \\ 9 \end{bmatrix} + \begin{bmatrix} 4 \\ 8 \end{bmatrix} = \begin{bmatrix} 1 \\ 17 \end{bmatrix}$$ then ReLU

= 16

y_3

$$\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ 17 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix}[5] = \begin{bmatrix} -16 \\ 18 \end{bmatrix} + \begin{bmatrix} 5 \\ 10 \end{bmatrix} = \begin{bmatrix} -11 \\ 28 \end{bmatrix} \approx$$

= 28

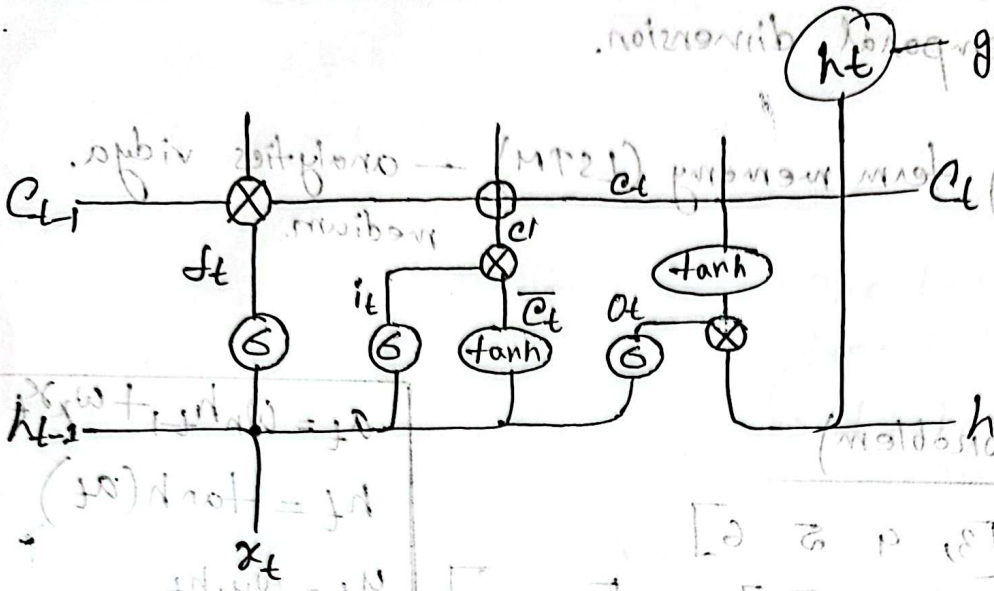
y_4

$$\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} 0 \\ 28 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix}[6] = \begin{bmatrix} -28 \\ 28 \end{bmatrix} + \begin{bmatrix} 6 \\ 12 \end{bmatrix} = \begin{bmatrix} -22 \\ 40 \end{bmatrix}$$

$\approx \begin{bmatrix} 0 \\ 40 \end{bmatrix}$

= 40

LSTM by hand



Forget gate:

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

Input gate:

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

$$\bar{C}_t = \tanh(w_c \cdot (h_{t-1}, x_t) + b_c)$$

$$C_t = (C_{t-1} \times f_t) + (\bar{C}_t \times i_t)$$

Updated memory.

$(\bar{C}_t \times i_t)$ — Selected info. to be added (c')

Output gate:

$$o_t = \sigma(w_o \cdot (h_{t-1}, x_t) + b_o)$$

$$h_t = o_t \times \tanh(C_t)$$

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

$$\text{sigmoid} = \frac{1}{1 + e^{-x}}$$

Transformer

Attention mechanism

Encoder – decoder (RNN)

Seq-to-seq.
disadvantage:
doesn't get backward context.
e.g. Google translate

github.io/visualizing neural-machine-translation mechanics
Jay Alammar.

The illustrated transformer
Jay Alammar.

Sequence-to-sequence model
Transformer.

Fuzzy Logic — partial truth.

A very brief introduction to fuzzy logic and fuzzy systems — medium.

Crisp sets and logic
↳ binary logic.

fuzzy set.
Membership function { - - - , logic
- - - - , u

Partial belonging.