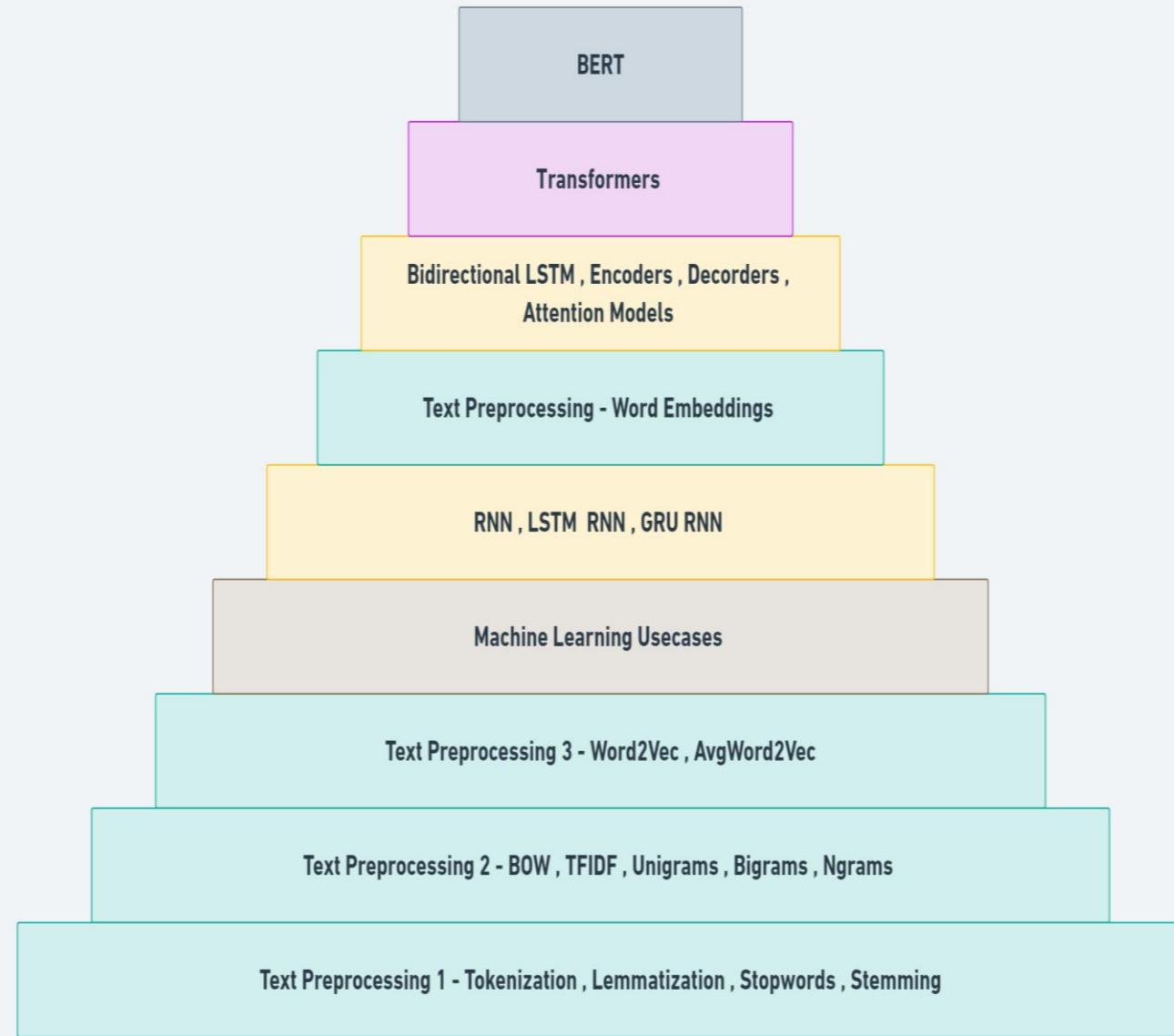


# NATURAL LANGUAGE PROCESSING

Advanced



*Libraries: NLTK , SPACY , TextBlob , Tensorflow , Pytorch , Huggingface*

# NLP

## Day 1

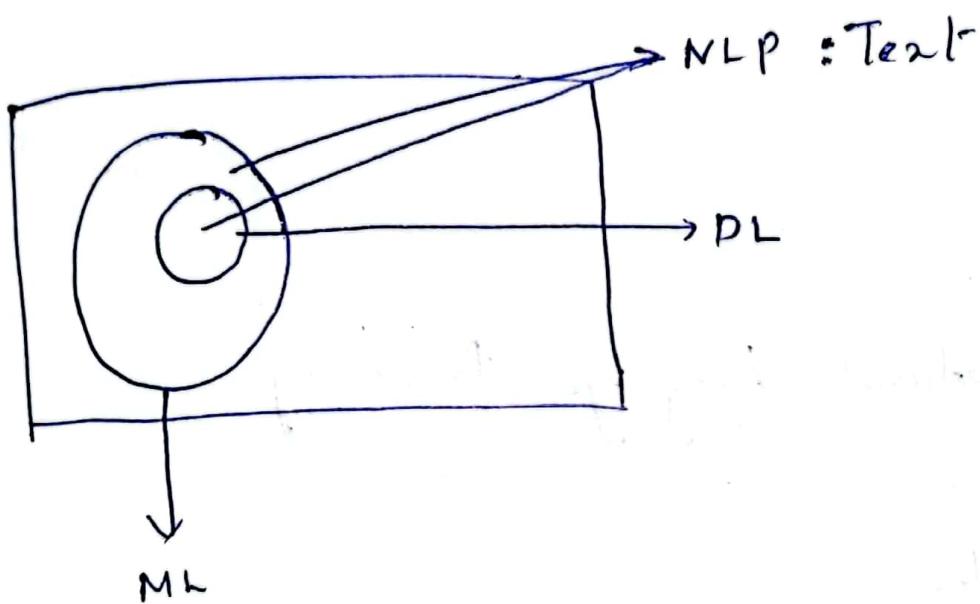
### Agenda :

1. Roadmap of Natural Language Processing
2. Why NLP
3. Lots of Examples
4. Tokenization, Stemming, Lemmatization
5. Bag of Words.

### Prerequisites

- 1) Python.
- 2) Stats
- 3) ML
- 4) Arrs, optimisers, loss functions

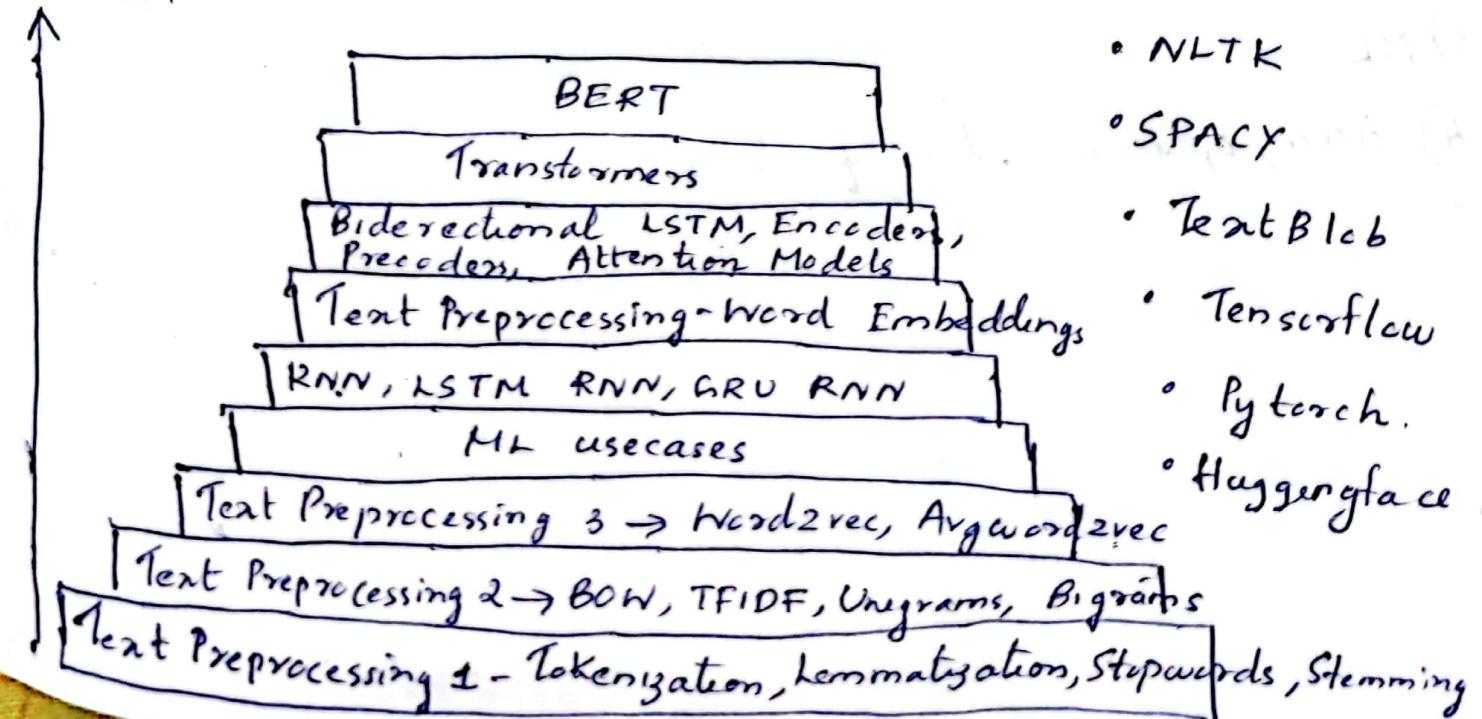
Why NLP?



How machine gonna understand our language?  
So we need NLP.

## RoadMap of NLP

Advanced



## Libraries

- NLTK
- SPACY
- TextBlob
- Tensorflow
- Pytorch
- Huggingface

## Text Preprocessing

Text may be in the form of Paragraphs, Sentences, we should convert this into specific format specifically called as vectors.

### 1. Tokenization

ML Usecase; Spam classifier.

Dataset.

Part I/p f <sub>1</sub> Email-body	I/p f <sub>2</sub> E-mail Subject	Spam/ham O/P
1) You won 100000 \$	Billionaire	Spam
2) Hey Anna, How are you?	Hello	Ham
3) Credit Cards Worth	Winner	Spam

① Tokenization → ② Stemming → ③ Lemmatization  
Stopwords

Tokenization: Converting sentence into words.

Eg: Hey Buddy I want ~~to~~ <sup>Not</sup> go ~~to~~ <sup>Important</sup> your house

"Not" Keyword can be important.

In order to remove the words we apply stopwords.  
We can use our own stopwords.

Stemming: we try to find out the base stem of the word.

histo recall

history → Historical Word Stem/Root word/Base form.

It is a process of reducing acids to their base word stem. It has a disadvantage that it has no meaning.

finally  
final  
finalized. } final (meaningless)

going      }  
goes      } go  
gone      } (meaningful)

## advantages

1. It is really fast

## Disadvantages

1. It is removing the meaning of the word.

## Lemmatization

Text pre-processing technique in Natural NLP  
that breaks down a word into its root  
meaning, also known as Lemma.

history

historical

history

finally

final

finalized

final..

## Advantages

1. Meaningfull words

## Disadvantages

1. It's slow.

## Use cases of Stemming

1. Spams Classification
2. Review classification

## Use cases of Lemmatization

1. Text Summarization
2. Language Translation
3. Chatbot.

# Text Preprocessing

Step 1

- 1) Tokenization
- 2) Stopwords
- 3) Stemming
- 4) Lemmatization

Step 2

Now we should convert the words into vectors

1. Bag of Words
2. TF-IDF
3. Word2vec.

Day 2

Agenda

- 1) Text Preprocessing → words → vectors
- a) One Hot Encoding (OHE)
- b) Bag of words (Bow)
- c) TF-IDF (Term Frequency - Inverse Document Frequency)
- d) Word2Vec.

## Basic Terminologies in NLP

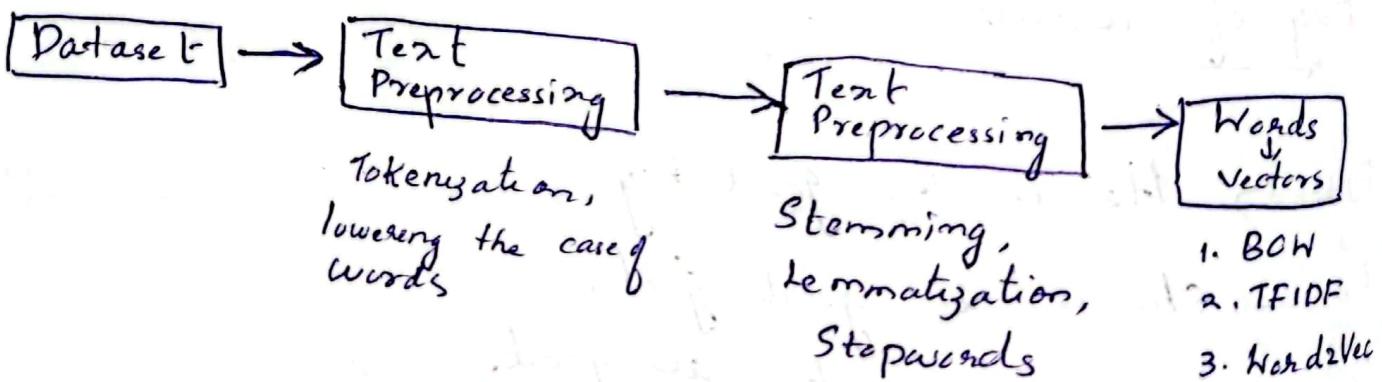
1. CORPUS → Paragraph →  $[D_1, D_2, D_3, D_4]$
2. Documents → Sentence →  $D_1, D_2, D_3, D_4$
3. Vocabulary → Total number of unique words
4. Words → word

## Sentiment Analysis

Text

O/P

D <sub>1</sub>	The food is good	1
D <sub>2</sub>	The food is bad	0
D <sub>3</sub>	Pizza is amazing	1
D <sub>4</sub>	Burger is bad.	0



## One Hot Encoding

[A man eat food  
Cat eat food  
People watch Krish Youtube]

CORPUS

vocabulary = 9

$$D_1 \left[ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \underbrace{\text{0000}}_{\text{0000}} \right] \rightarrow \text{CWE} \leftarrow D_2 = \left[ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \right]$$

### Advantages

1. Simple to Implement
2. Intuitive

Extra words that may come in the test data cannot be handled.

### Disadvantages

1. It creates sparse matrix.
2. Out of Vocabulary (OOV)
3. Not fixed size.
4. Semantic meaning b/w words not captured.

### Bag of Words

$D_1 \rightarrow$  He is a good boy

$D_2 \rightarrow$  She is a good girl.

$D_3 \rightarrow$  Boys and girls are good.

After applying Stop words

$D_1 \rightarrow$  Good boy

$D_2 \rightarrow$  Good girl

$D_3 \rightarrow$  Boys girls Good.

## Vocabulary (3)

## Frequency

Good

3

Boy

2

girl

2

Document 1: Good boy girl

$f_1 \ f_2 \ f_3$

(Order it based on the frequency in descending order)

Good boy girl

→ Whenever we use stop words make sure that lower all the words case (lower case)

$f_1 \ f_2 \ f_3$  (Assumption)

Good boy girl o/p

Doc 1 1 1 0 -

Doc 2 1 0 1 -

Doc 3 1 1 1 -

In BOW, we have an option to make it binary BOW.

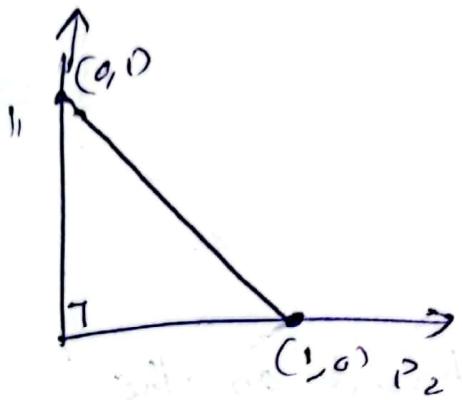
### Advantages

1. Simple and Intuitive

### Disadvantages

1. Sparsity
2. OOV
3. Ordering of the words has completely changed
4. Semantic meaning is lost

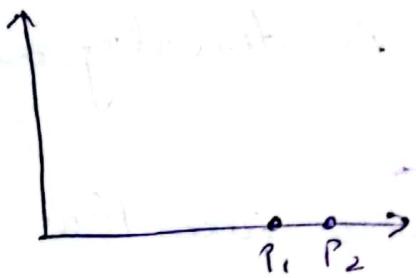
## Cosine Similarity



$$\cos 90^\circ = 0$$

$$1 - 0 = 1$$

Points are not similar.



$$\cos 0^\circ = 1$$

$$1 - 1 = 0$$

Points are almost similar.

## To Capture the Semantic Information

We use Ngrams : Bigrams, Trigrams, Ngrams

	$f_1$	$f_2$	$f_3$	$f_4$	$f_5$
good	$f_1$	$b\text{oy}$	$g\text{irl}$	$g\text{o}\text{o}\text{d}$	$g\text{o}\text{o}\text{d}$
Sent 1	1	1	0	1	0
Sent 2	0	0	1	0	1
Sent 3	1	1	1	0	0

Anna Eats Food

No. of Bigrams ? 2 .

- Anna Eats
- Eats Food

I am not feeling well.

Trigrams? 3.

- I am not
- am not feeling
- not feeling well.

- Bag of Words in text Processing is a Feature Extraction technique.

~~Day<sup>3</sup>~~

Bag of Words is a technique to convert text into vectors.

① Sent 1 → He is a good boy

Sent 2 → She is a good girl

Sent 3 → Boy and girl are good.

Applying Stopwords

Sent 1 → good boy

Sent 2 → good girl

Sent 3 → boy girl good

### ③ Frequency (Vocabulary)

	<u>frequency</u>
good	3
boy	2
girl	2

(4)

	$f_1$	$f_2$	$f_3$
good	1	0	1
boy	0	1	0
girl	1	1	1

Sent 1      1      1      0

Sent 2      0      1      1

Sent 3      1      1      1

The food is good

food good is not the

The food is not good

1      1      1      0      1

1      1      1      1      1

even though the sentences  
are opposite

→ Here it shows they  
are always similar

→ So this issue can be solved by TF-IDF

TF-IDF : Term Frequency - Inverse Document  
Frequency

After cleaning

Sent 1 : good boy

Sent 2 : good girl

Sent 3 : boy girl good

⇒ Whichever words are present rarely in the sentence we should give more weightage.

⇒ The rare words are captured by <sup>Term</sup> ~~Term~~ frequency and

Common words are captured by Inverse Document frequency.

⇒ How to calculate Term frequency?

$$\text{Term Frequency} = \frac{\text{No. of repetition of words in Sentence}}{\text{No. of words in Sentence}}$$

⇒ How to calculate IDF?

$$IDF = \log_e \left( \frac{\text{No. of Sentences}}{\text{No. of Sentences Containing the word}} \right)$$

$$TF-IDF = TF \times IDF$$

Term Frequency

	Sent 1	Sent 2	Sent 3
girl	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{3}$
boy	$\frac{1}{2}$	0	$\frac{1}{3}$
girl	0	$\frac{1}{2}$	$\frac{1}{3}$

## Inverse Document Frequency

Words      IDF

$$\text{good} \quad \log_e(3/3) = 0$$

$$\text{boy} \quad \log_e(3/2) = 0.65$$

$$\text{girl} \quad \log_e(3/2) = 0.65$$

## TF-IDF

$$f_1 \quad f_2 \quad f_3$$

Good    boy    girl

Sent<sub>1</sub>

Sent<sub>2</sub>

Sent<sub>3</sub>

Sent<sub>1</sub>      Sent<sub>2</sub>      Sent<sub>3</sub>

good      0      ~~0.325~~      ~~0.217~~

boy      0.325      0      0.217

girl      0      0.325      0.217

## Advantages

1) Inductive

2) Word Imp are getting captured.

## Disadvantages

1. Sparsity ( $\ll$  bag of words)
2. Out of Vocabulary.

~~Day~~

## Imp. Word Embeddings

2. Word2vec → CBOW (Continuous Bag of Words)  
→ Skipgrams

3. Practical implementation using Python.

## Word Embeddings

Technique to converts words into vectors

### Word Embeddings

Count or Frequency

1. Bag of Words

2. TF-IDF

3. OHE

Deep learning Trained Models

1. Word2vec

CBOW

Skipgrams

## Word2vec: Feature Representation

$f_1$	$f_2$	$f_3$	$f_4$	$f_5$	$f_6$
boy	girl	King	Queen	Apple	Mango

→ for every word set is going to create vectors but within the limited dimensions.

1. Limited Dimension
2. Sparsity is reduced.
3. Semantic meaning is maintained. (related info)

Wherever they are related the vectors will be almost same.

	$f_1$	$f_2$	$f_3$	$f_4$	$f_5$	$f_6$
boy	-0.1	1	-0.92	0.93	0	0.1
girl	0.01	0.02	0.95	0.96	-0.02	0.1
King	0.03	0.02	0.7	0.6	0.95	0.96
Queen	0.03	0.02	0.7	0.6	0.95	0.96
Apple	0.03	0.02	0.7	0.6	0.95	0.96
Mango	0.03	0.02	0.7	0.6	0.95	0.96
Gender	0.03	0.02	0.7	0.6	0.95	0.96
Royal	0.03	0.02	0.7	0.6	0.95	0.96
Age	0.03	0.02	0.7	0.6	0.95	0.96
Food	0.03	0.02	0.7	0.6	0.95	0.96
:	:	:	:	:	:	:
300 dimensions						

This is created by the model.

King  $[0.96 \ 0.95]$  Man  $[0.95 \ 0.98]$

Queen  $[-0.96 \ 0.95]$  Women  $[-0.94 \ -0.98]$

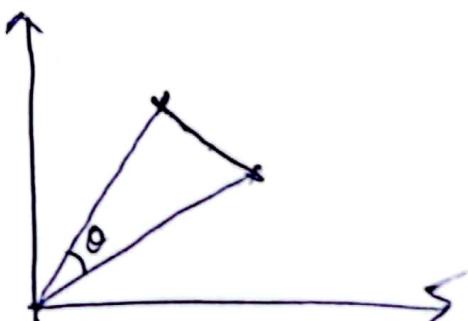
$$\text{King} - \text{Man} + \text{Women} = \text{Queen}$$

We can find the distance using Euclidean distance.

Manhattan Distance or Cosine Similarity

Distance = 1 - Cosine Similarity.

Cosine Similarity



$$\text{Cosine Similarity} = \cos \theta$$

$$\Rightarrow \text{distance} = 1 - \cos \theta$$

If Cosine-Similarity = 1  $\Rightarrow$  More Similar,  
" " " " " = 0  $\Rightarrow$  No Similarity.

# Word2vec

## 1. CBOW

CORPUS : TRAINING DATASET

[It is. a Software that does to think and learn]

o/p/target  
Content Context Context

Window Size = 5

Training Data.

Independent Feature o/p.

It, is, Software, that a

[It is. a. Software that does to think and learn]

i/p

is, a, that, true o/p  
Software.

a, software, does, to that

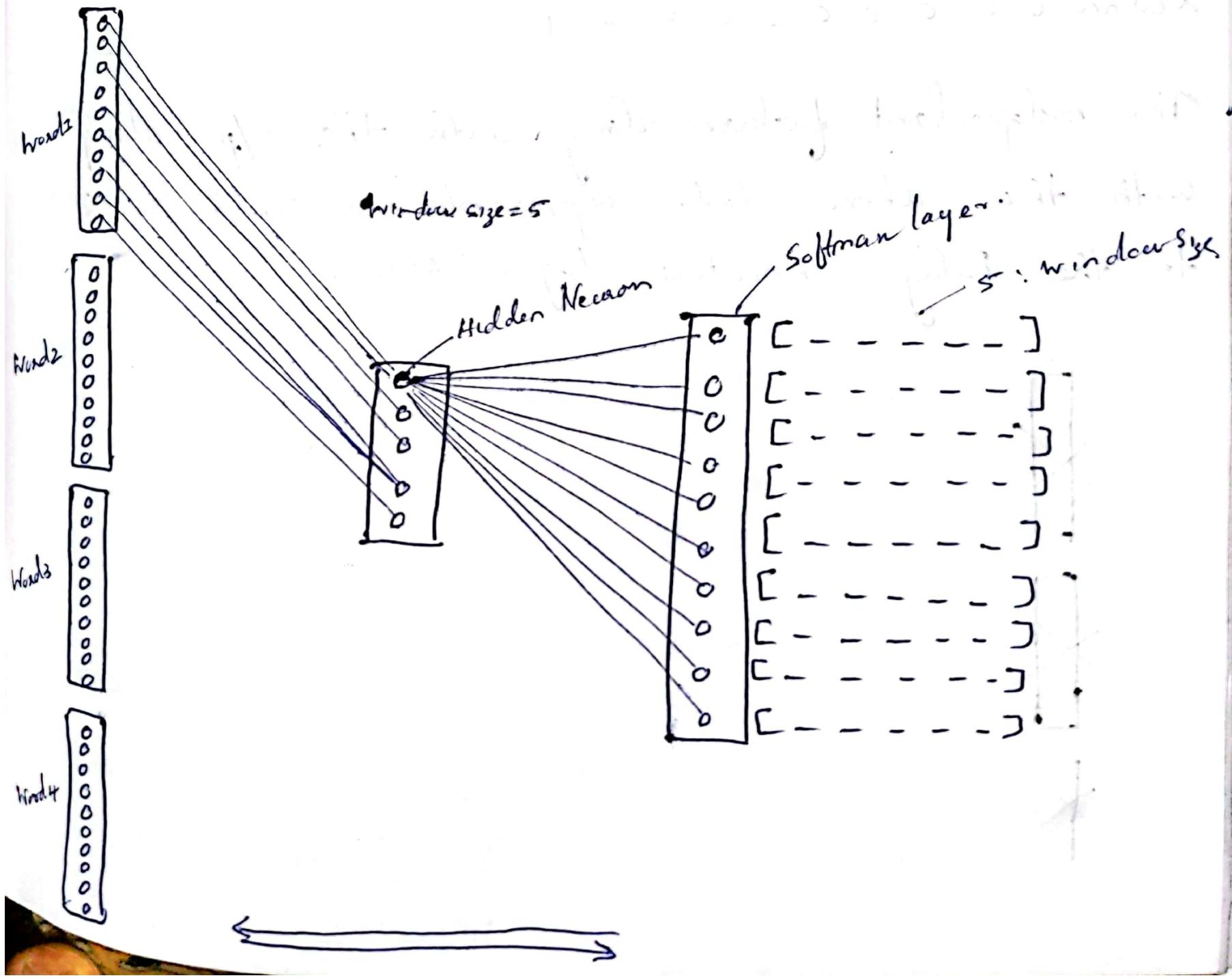
Software, that, do, think true

that, true, think, and to

does, to, and, learn think.

It 1 0 0 0 0 0 0 0 0 0  
is 0 1 0 0 0 0 0 0 0 0  
a 0 0 1 0 0 0 0 0 0 0  
Software 0 0 0 1 0 0 0 0 0 0  
that 0 0 0 0 1 0 0 0 0 0  
tries 0 0 0 0 0 1 0 0 0 0  
to 0 0 0 0 0 0 1 0 0 0  
think 0 0 0 0 0 0 0 1 0 0  
and 0 0 0 0 0 0 0 0 1 0  
learn 0 0 0 0 0 0 0 0 0 1

- The independent features along with the o/p along with the above data representation will be given to the Fully Connected layer (ANN).



Q. Skip Gram  
→ p and o/p are interchanged

Yp      O/p

Software      is, a, that, clone  
that      a, software, tries, to  
clone      Software, that, to, think  
to      that, clone, think, and  
think      tries, do, and, learn

~~Day 5~~

## 1. Practical Implementation

### Spam Classification

① Text Preprocessing → Tokenization, Stopwords,  
Stemming, Lemmatization - NLTK

② Text → Vectors : BOW, TF-IDF, word2vec,  
Avg word2vec.

Word2vec  
↓  
Skip grams      CBOW

Word2vec  
↑  
Pretrained model      Train the model  
from scratch.

- a) Should we always draw the model or use the pretrained model?
- If in the new dataset it's already captured the 75% of the words in pretrained model then opt the pretrained Model.
  - Otherwise, train your own model.

### Argword2vec

It is an extension of the Word2vec model that generates vector representations for sentences or documents instead of individual words.

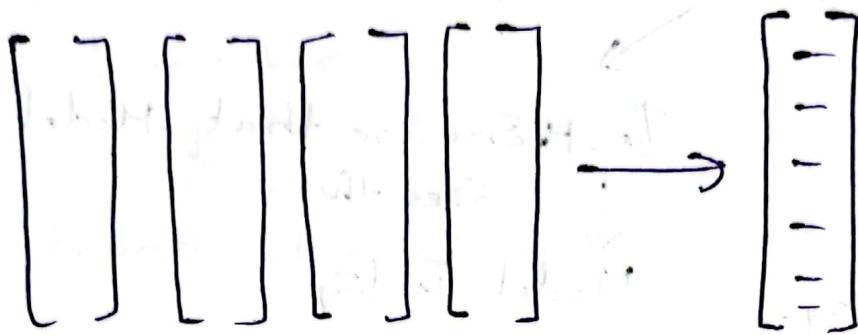
Sent. This is a ball

Word2vec converts each word into specified dimension say 300

This is a ball

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

But AvgWord2vec find out the average.



~~Day 6~~

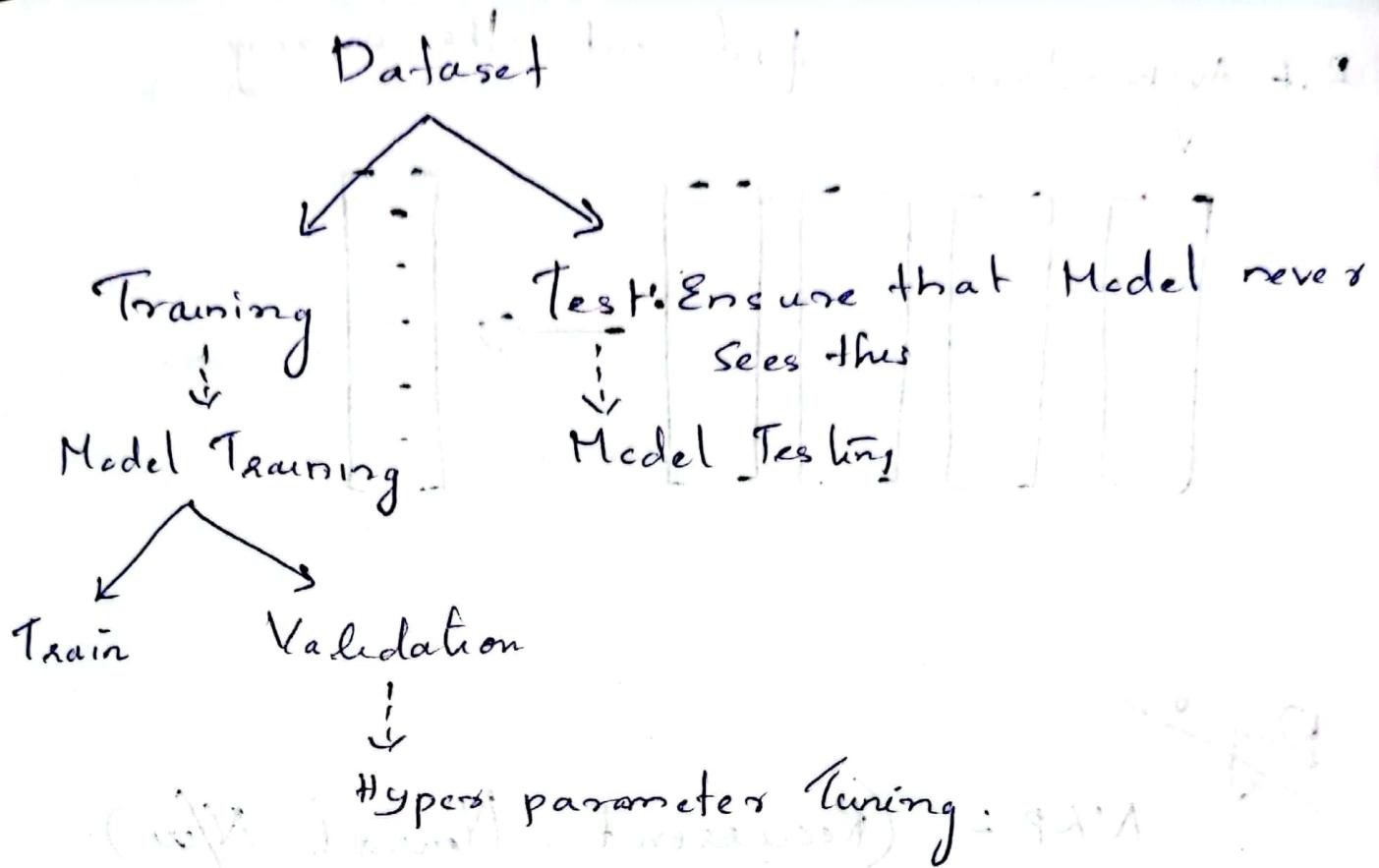
## NLP - (Recurrent Neural N/w)

### Agenda

- 1) RNN    2) LSTM RNN    3) GRU RNN
- 4) Bidirectional LSTM RNN
- 5) Encoders - Decoders
- 6) TRANSFORMERS    7) BERT.

### Interview Questions

1. Train v/s Test v/s Validation ?

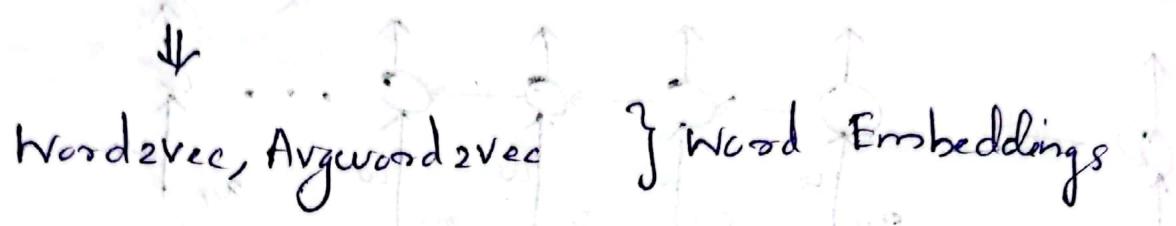


2. Why people use Random Forest instead of Decision Tree.

DT  $\rightarrow$  Low Bias and high Variance.  
 In order to reduce the Variance keeping the bias low we use Random Forest instead of Decision Tree.

# 1. Recurrent Neural N/w.

Text  $\rightarrow$  Vectors.



Applications :

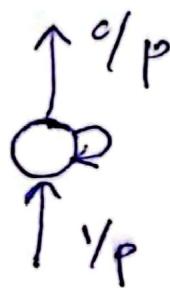
- 1) Chatbot.
- 2) Language Translation
- 3) Text Generation

ML techniques will not be able to give us good accuracy.

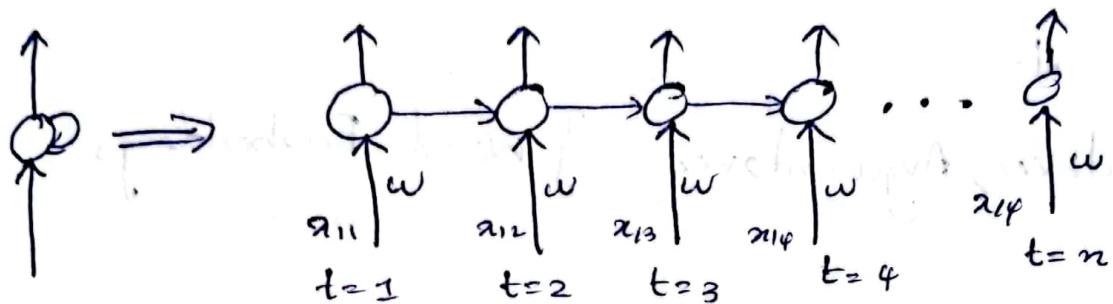
Deep Learning Techniques.

- 1) RNN
- 2) LSTM RNN
- 3) Transformer
- 4) BERT

## Recurrent Neural N/w



## Sentiment



Every o/p is given to the same neural n/u.  
with

Sentiment Analysis  $\rightarrow$  The food is so good.

O/p

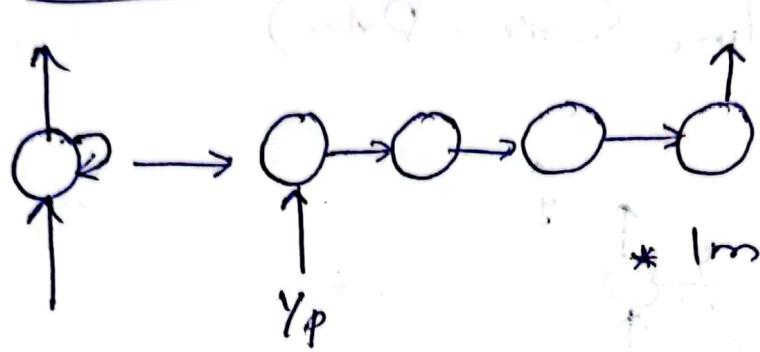
Positive

$x_{11} \rightarrow \text{word2vec} \rightarrow \text{vectors} \rightarrow d = 300$  (no. of features)

## Types of RNN

1. One to One RNN
2. One to Many RNN
3. Many to one RNN
4. Many to Many RNN

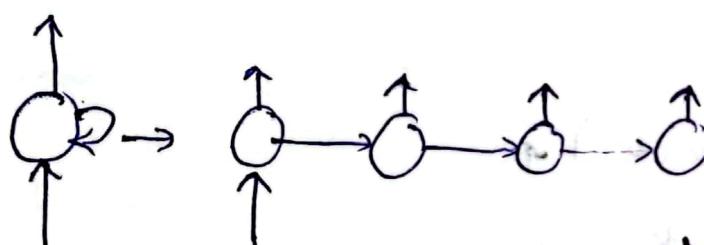
## 1. One to One RNN



1/o/p. 1/o/p and 1/o/p

\* Image classification.

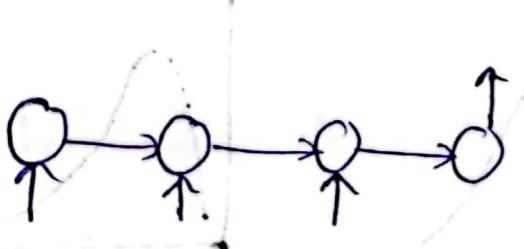
## 2. One to Many RNN



1/o/p multiple o/p

- \* Music generation
- \* Text generation
- \* Google search suggestion
- \* Movie Recommendation

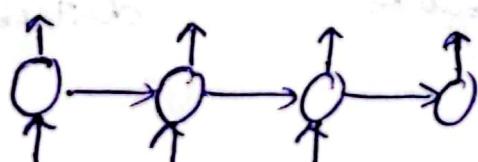
## 3. Many to One RNN



multiple i/p 1/o/p

- \* Sentiment Analysis
- \* Predict Next Day Sales

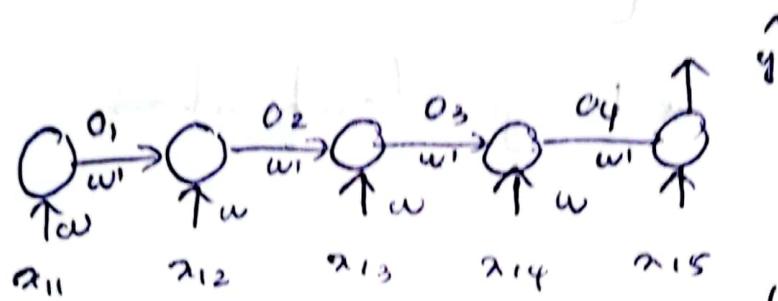
## 4. Many to Many RNN



multiple i/p multiple o/p

- \* Language Translation
- \* Question Answers
- \* Chatbots.

1. Forward Propagation in RNN  
(RNN is used for Time Series Data)



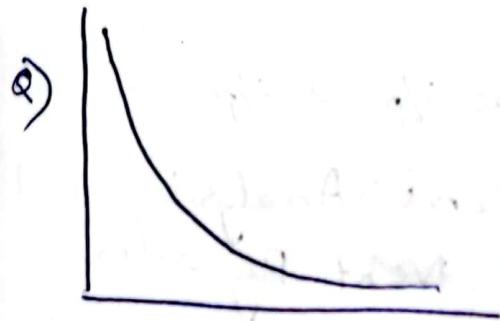
The food is very good. <sup>c/p</sup> Positive

$$o_1 = f(x_{11} * w)$$

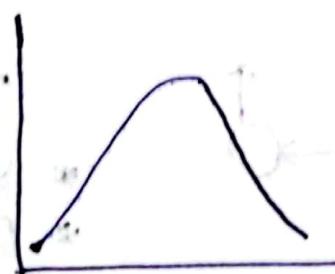
$$o_2 = f(x_{12} * w + o_1 * w')$$

$$o_3 = f(x_{13} * w + o_2 * w')$$

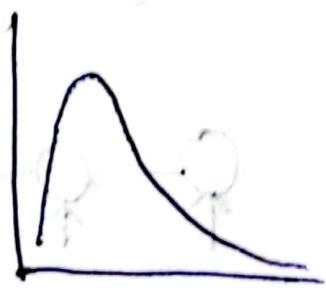
Day 4



Power law distribution  
or  
Pareto distribution

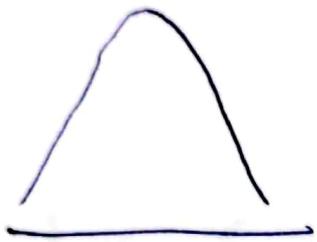


Normal  
distribution



log normal  
distribution

- Bonccon transformation is used to transform Pareto distribution  $\rightarrow$  Normal.

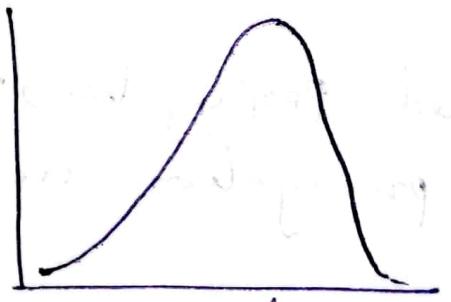
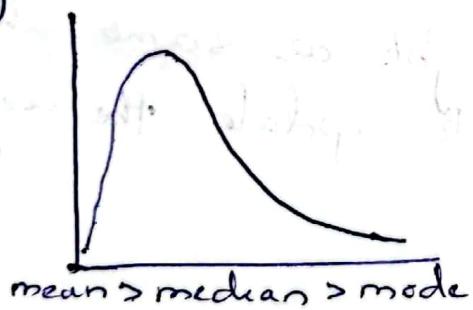


a) What techniques should we apply to check whether this distribution follows normal

ans: Q-Q plots

g) What is SND?

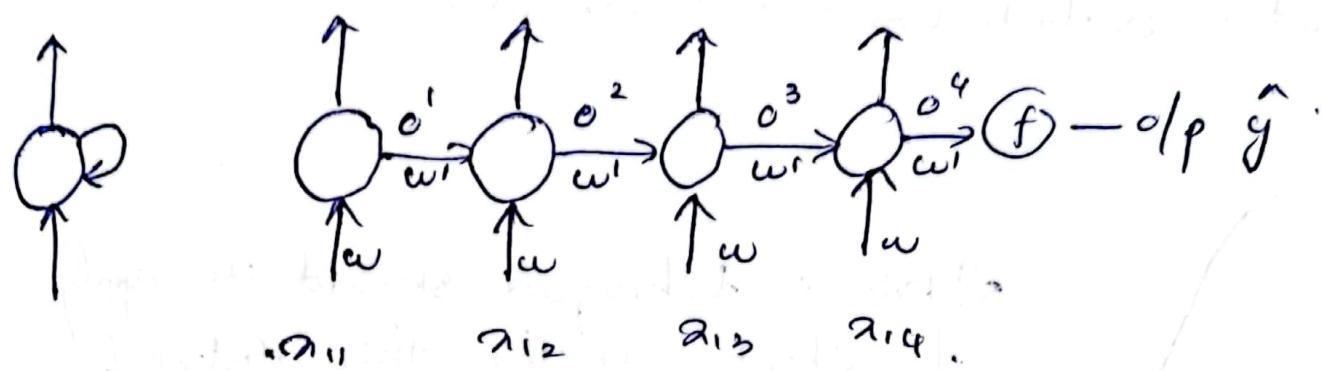
Q)



What is the relation b/w mean, median, mode in the above diagrams?

- Difference b/w transformation and transforms?
- Difference b/w Standardization and Normalization

## RNN



$$o_1 = f(x_{11} \times w)$$

$$o_2 = f(x_{12} \times w + o^1 \times w^1)$$

$$o_3 = f(x_{13} \times w + o^2 \times w^1)$$

In Forward Propagation, the weights are same while in back propagation we need to update the weights.

## Weight Updation formula

$$w'_{new} = w'_{old} - \eta \frac{\partial L}{\partial w}$$

$$\rightarrow \frac{\partial L}{\partial w^1} = \frac{\partial L}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial w^1}$$

$$w_{new} = w_{old} - \eta \frac{\partial L}{\partial w_{old}}$$

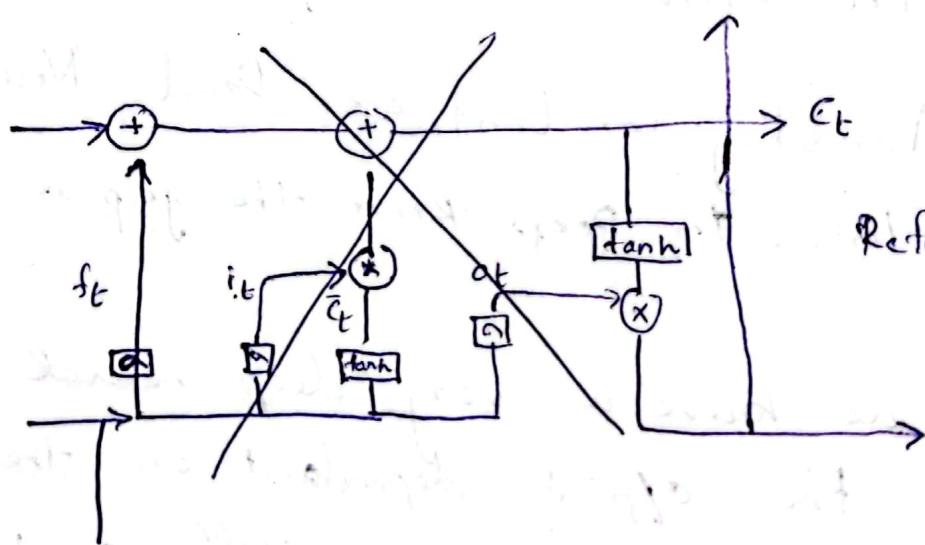
$$\rightarrow \frac{\partial L}{\partial w} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial o^4} \cdot \frac{\partial o^4}{\partial w^1}$$

If suppose the RNN is deep RNN as we move forward, vanishing gradient Problem occurs.

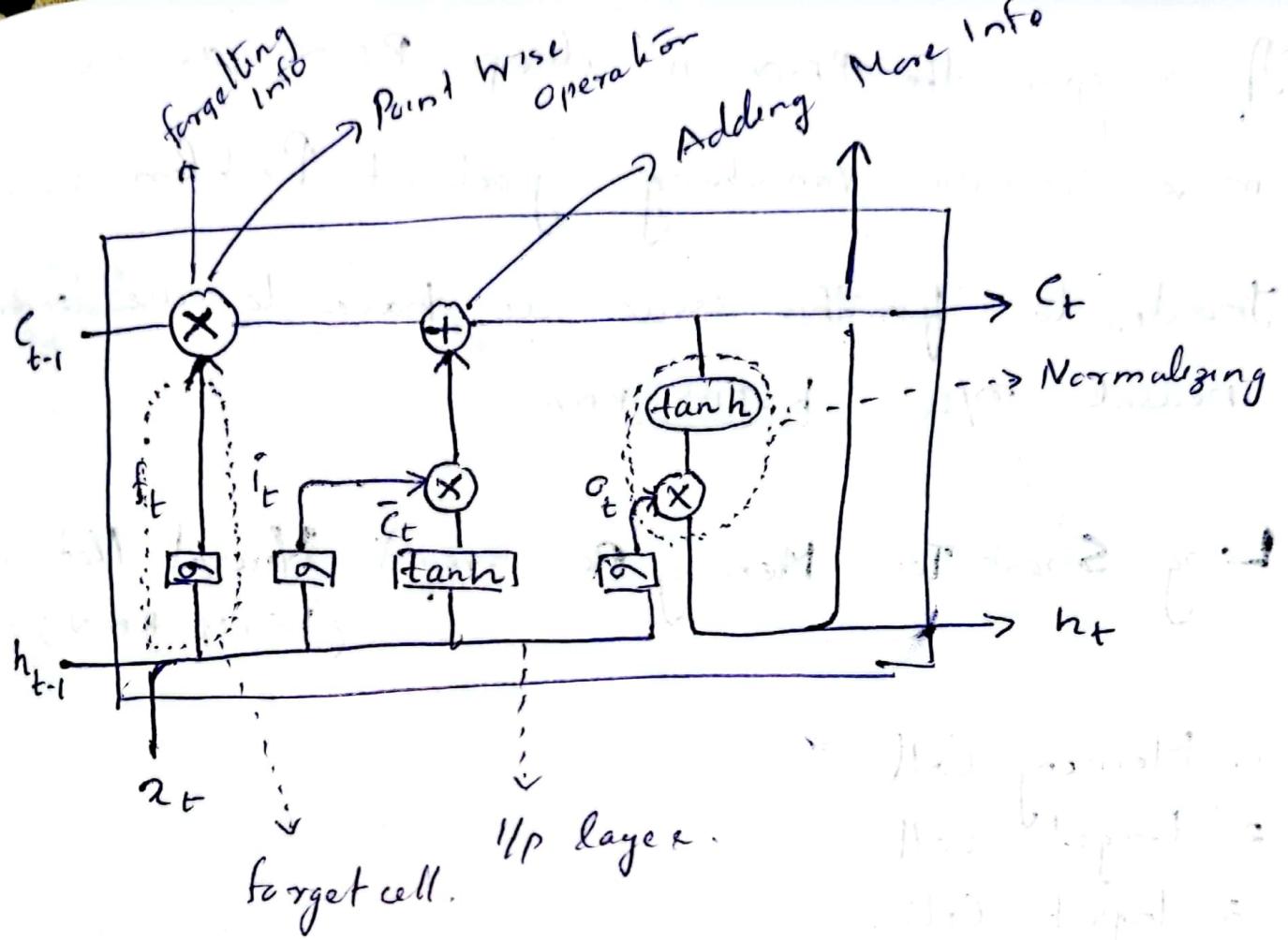
Inorder to fix this issue we have to use another neural m/w LSTM RNN.

## Long Short Term Memory Recurrent Neural Network (LSTM RNN)

1. Memory Cell
2. Forget Cell
3. Input Cell.



Refer Google



~~Imp~~ Q) Why LSTM RNN ~~and~~ not RNN?

1. Solves Vanishing Gradient or Dead Neuron.
2. Context Info. In Deep RNN the gap is huge.

Let's say we have a very deep neural n/w and some of the o/p is dependant on the first 'Up', but this context is not really captured with the help of RNN.

Day 8

Agenda

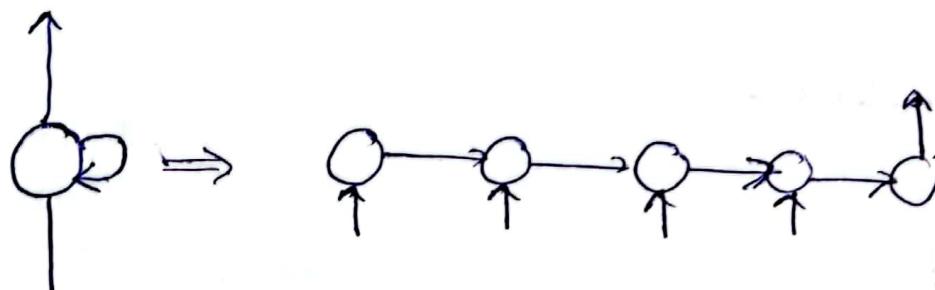
1. LSTM RNN (in-depth Architecture).

Problems with RNN

Suppose I want to predict the next word,

On Sunday I want to eat Pizza,

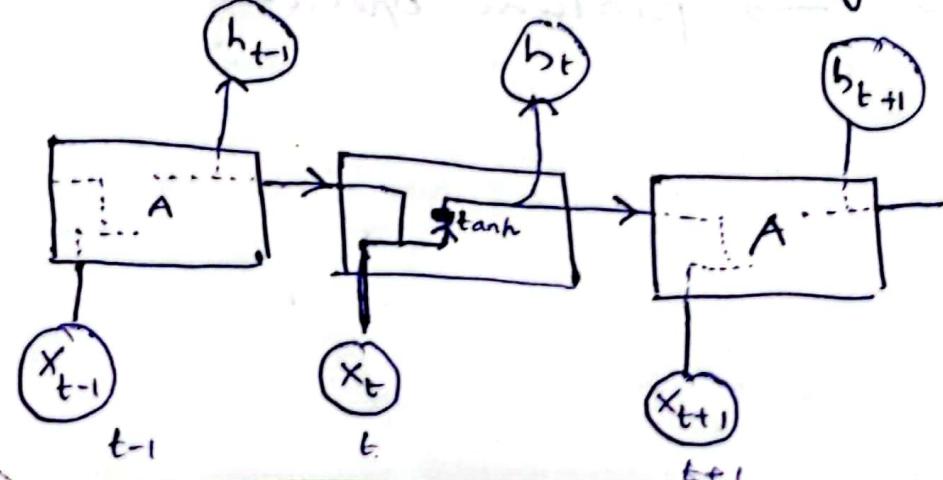
On Monday I want to eat \_\_\_\_\_



RNN can have only short-term memory.

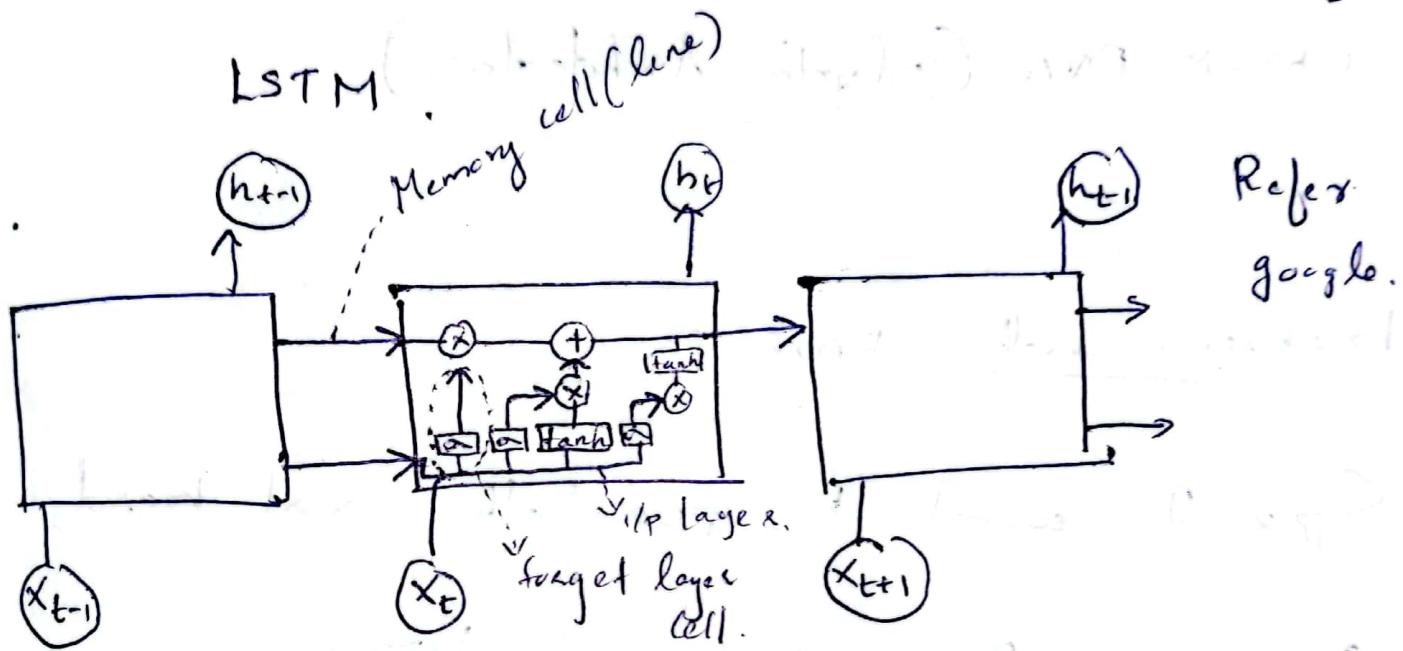
Represent  
RNN

Traditional  
RNN



Why we specifically use LSTM RNN?

i) long Sentences  $\rightarrow$  Context will be switching.



$\square \rightarrow$  Neural N/w with Sigmoid Activation fun.

$\boxed{\text{tanh}} \rightarrow$  Neural N/w with tanh Activation fun

$\top \rightarrow$  Concatenate

$\downarrow \rightarrow$  Copy

$\rightarrow \rightarrow$  Vector transform

$\begin{matrix} 0 \\ +, \cdot \end{matrix} \rightarrow$  pointwise operation

## Memory cell

- It acts like a conveyor belt in airport
- Similarly, you can add and remove information.

## Forget layer cell

- It is specifically used for one critical information.
- The Sigmoid layer cell is trained in such a way that, for most of the information we get zeros.
- We are making the memory cell forgetting the information because, forward we have a point wise operation of cross (x)

Say 2 sentences

① Anna like pizza but She doesn't like Burger

↑  
Content Switch is not happening

② Anna like pizza but her friend like Burger

Content Switch is happening.

In ①, the information is not lost.

In ②, the <sup>previous</sup> information is lost.

## O/p Gate layer

This info should be forgotten  
Consider, Anna like pizza but her friend  
like Burger.  
This information should be stored, or add it in the memory cell.

And it will be passed through an activation function and tanh activation first. And when you combine both the new content will be added to the memory cell.

## Output Gated layer

From the memory cell we get the recent information on top of that tanh is applied. And it is combined with ~~activation~~ the previous forgotten information. Then it is passed to the o/p to the next cell.

Similar things will go on at different time stamps.

"all useless layer is removed, only imp data is passed to the o/p.

Day 9

## Agenda

1. Word Embedding Layer: Tensorflow and Python.
  2. LSTM Sentiment Analysis.

## Word Embedding Layer

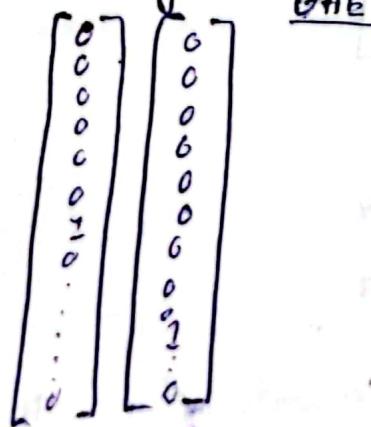
1. Sentence.
  2. OHE : One Hot Encoding (Vocabulary Size)
  3. Padding : Post Padding and Pre Padding
  4. OHE → vectors.

Sent

[Anna Likes Pizza]

[ Ram Likes Burgers ]

Say Vocabulary size = 500  
- - - ONE



By considering this OHE we have take that entire words and probably create an embedding layer and convert this into vector. → which is done by Embedding Layer in DL.

## Practical :- Word Embeddings.

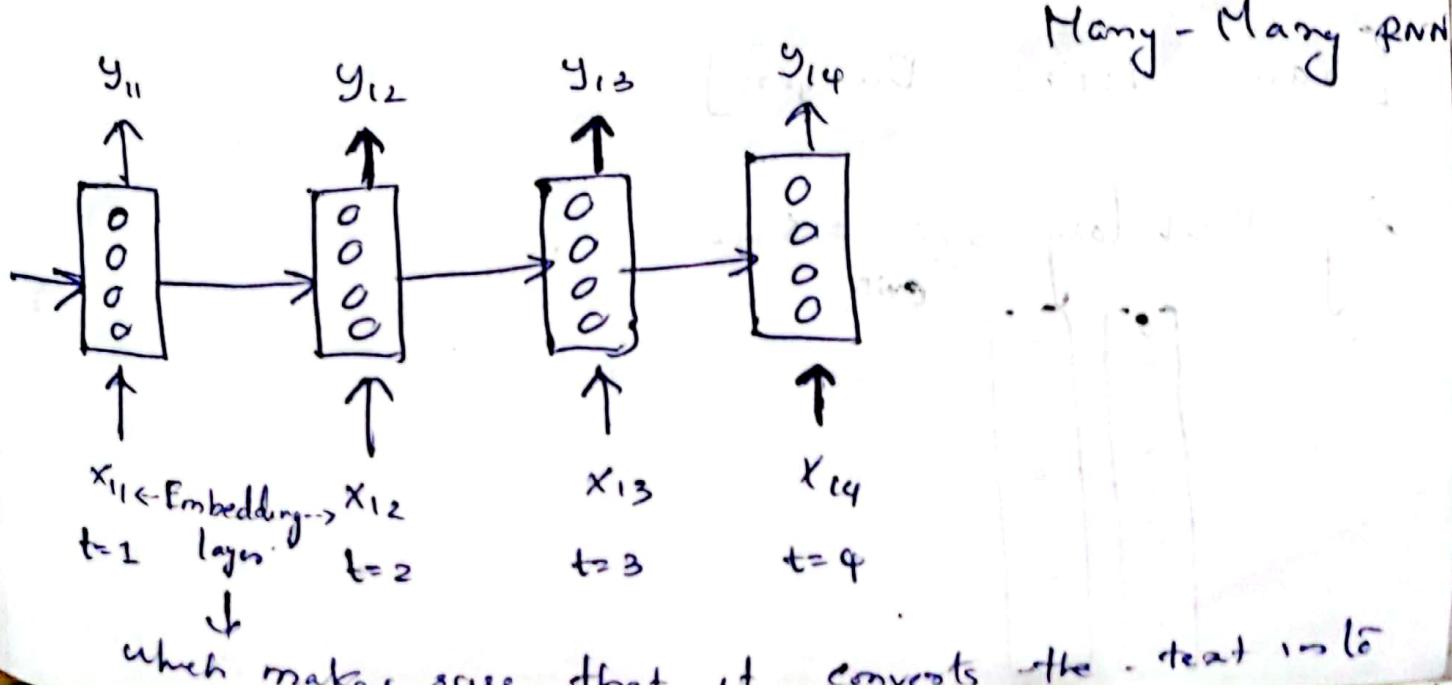
Day 10

Fake-news-classification:- LSTM.

Day 11

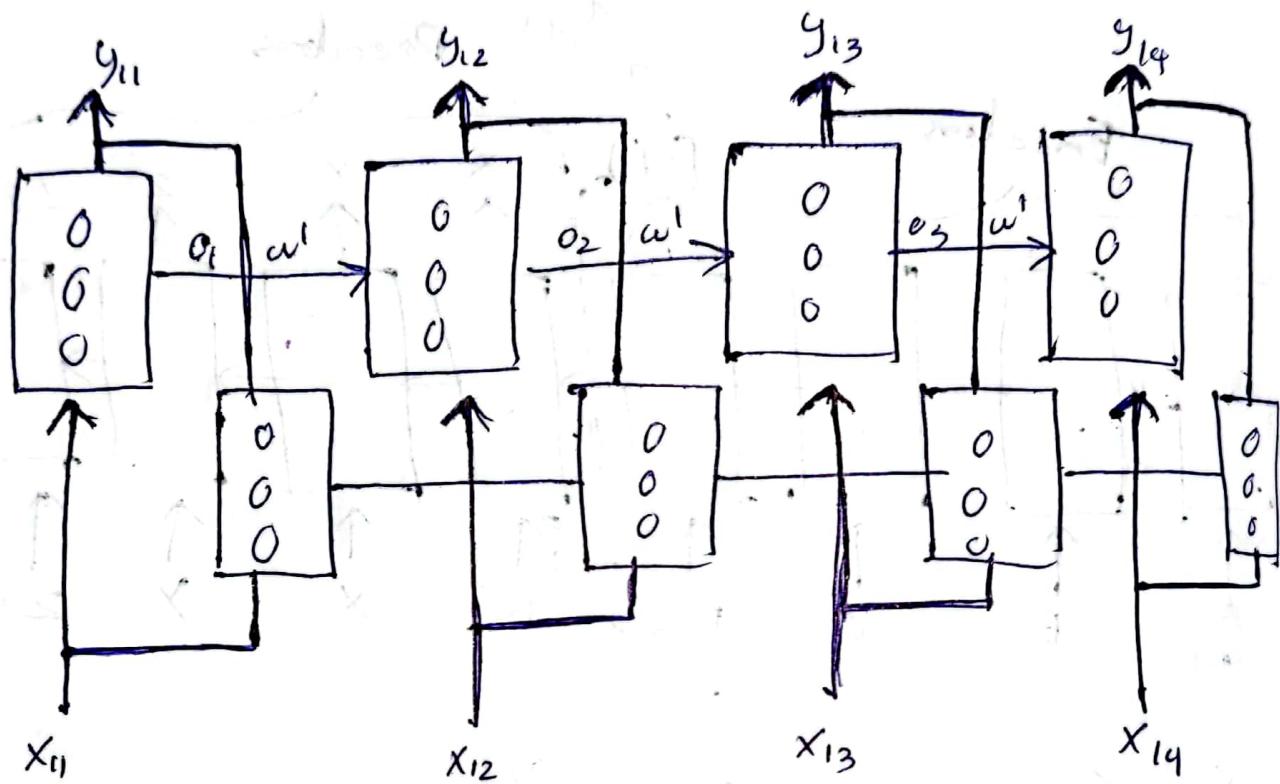
Bidirectional LSTM RNN

Anna likes to eat  $\xrightarrow{\text{O/P}} \text{in Kochi}$



vectors

$y_{13}$  have the information of  $x_{13}, x_{12}, x_{11}$

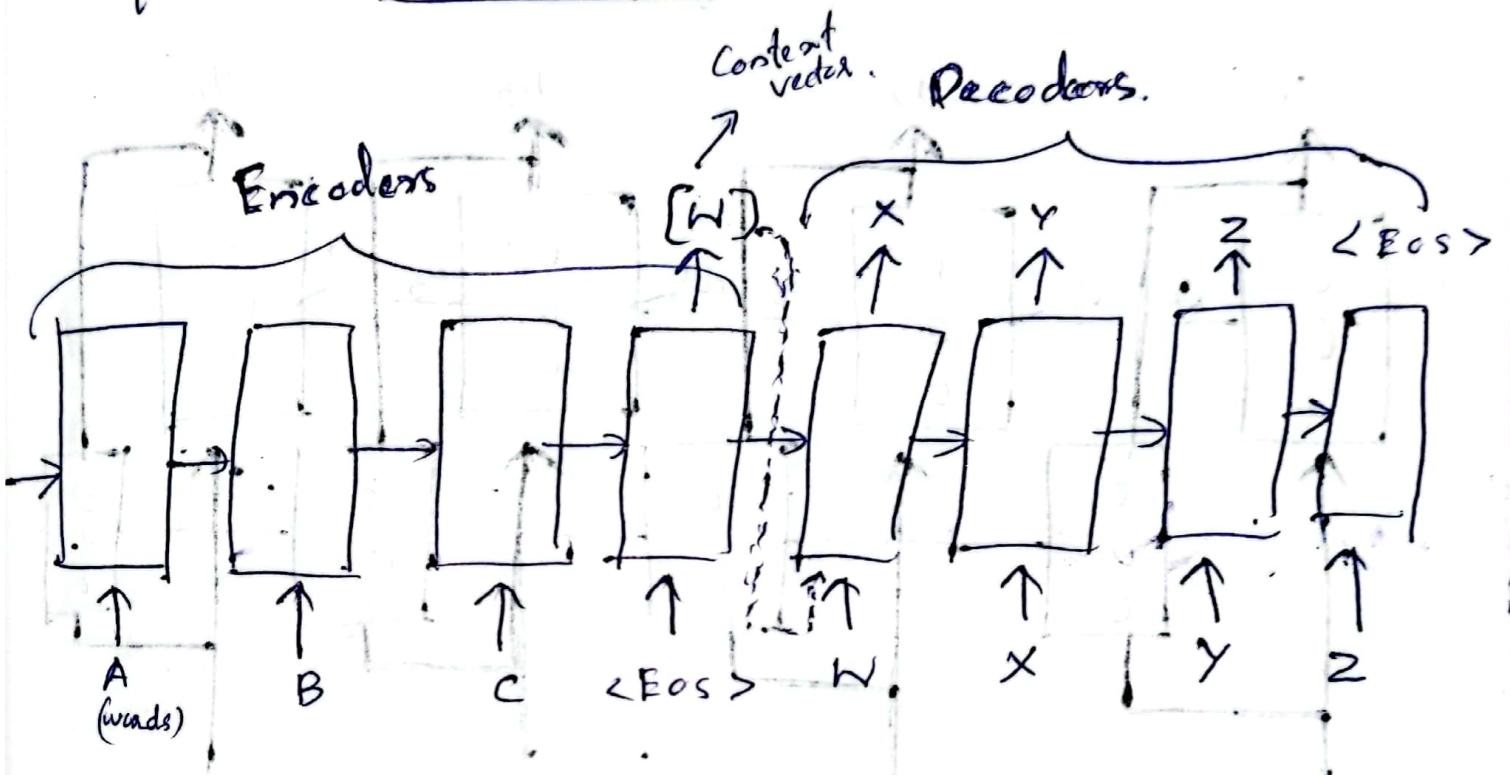


In the forward direction you are passing all the weights and in the backward direction you are passing the words.

Practical

Fake-new-classification using Bidirectional LSTM.

# Sequence to Sequence with Neural Net



$$X = \langle x_1, x_2, x_3 \rangle \quad Y = \langle y_1, y_2, y_3 \rangle$$

Vector conversion  $\rightarrow$  Embedding layer.

One hot Representation

Wordvec.

- Once we are able to get the context vector, then passed to the Decoder.

The i/p and o/p sequence may be completely different.

- When we are doing the prediction for new test data, we will be creating  $\hat{y}_1, \hat{y}_2, \hat{y}_3$  and so on.

•  $P(\hat{y} / \langle x_1, x_2, x_3, \dots, x_n \rangle)$

•  $\hat{y}$  is a combination of  $\langle \hat{y}_1, \hat{y}_2, \hat{y}_3, \dots, \hat{y}_t \rangle$ .

• Once we find out  $\hat{y}$ , we try to find out the loss,  $L(y, \hat{y})$ . And we can use an optimizer to reduce the loss through back Propagation.

• In case of Image captioning, we use advanced CNN like VGG16, RESNET instead of LSTM as an Encoder.

### Disadvantages

When you have long sentence the accuracy is not good.

So we are going to use another model called Attention Model (Bidirectional LSTM as Encoders)

### Problems with Encoders and Decoders

BLEU (Bilingual Evaluation Understudy) is a metric used to measure the quality of predicted text in NLP models by comparing

it do a sort of references.  
Ranges from 0-1, higher values indicating  
better predictions.

- 0 → Machine translated o/p has no overlap  
with the reference translation.
- 1 → perfect overlap.

## Problems with Encoders and Decoders

1. It usually do not perform with longer sentence
2.  $\downarrow$ , lower BLEU Score.

### \* Window-Size

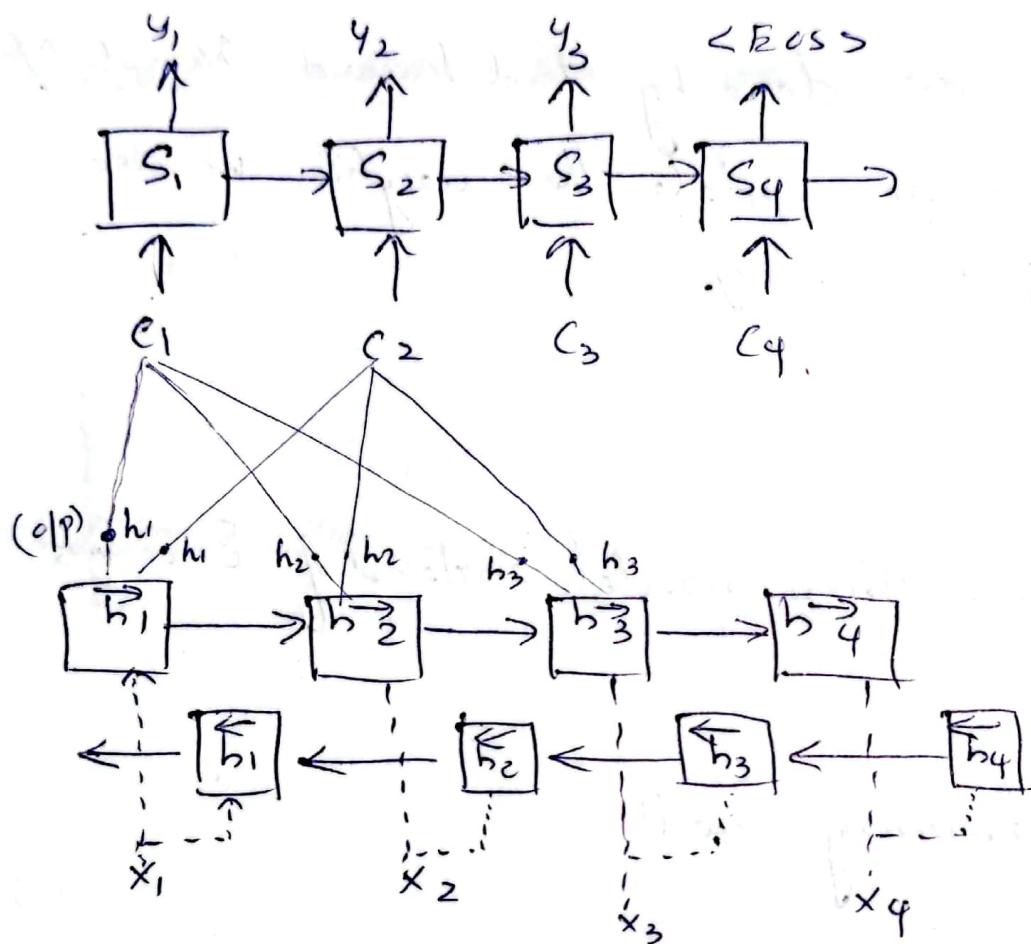
lets take an example of language translator.

FRENCH → ENGLISH.

Translator will hearing some sentence then he/  
she will be translating.

Similarly we choose 'window size': hyperparameter.

# In Attention Model : Encoders $\rightarrow$ Bidirectional LSTM



- if window size = 3.
- i.e., you are giving attention to some words (window size)
- LSTM RNN works on probability.
- $h$  based on  $T_x$  will generate some values like  $\alpha_1, \alpha_2, \alpha_3$  (weights)  
 There are some conditions;

For eg:  $T=3$

$$\textcircled{1} \quad \sum_{i=1}^T \alpha_i = 1 \quad \alpha_1, \alpha_2, \alpha_3$$

② Content vector  $c_i^o = \sum_{l=1}^{T_x} \alpha_l h_i^l$

these operations are done by feed forward Neural Net.  
Until and unless we update the weights, we are  
not gonna get good accuracy.

Interview

Why do you use these model instead of Encoders  
and decoders?

It was not performing well.

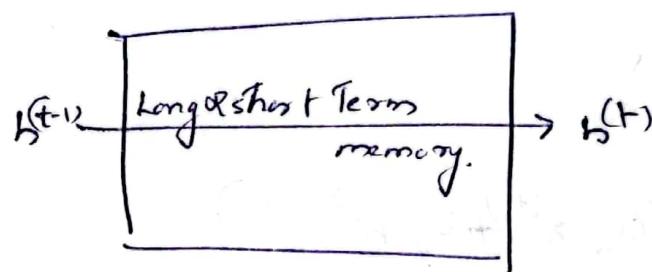
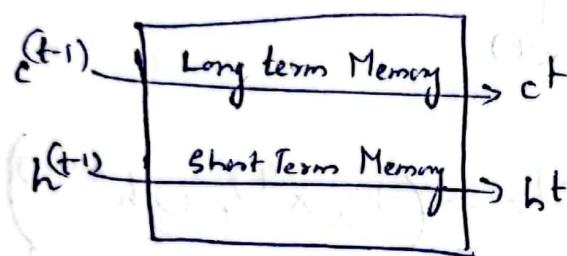
(Problems)

## CRU (Circled Recurrent Units)

Traditional RNN ~~do~~ have short term. Memory.

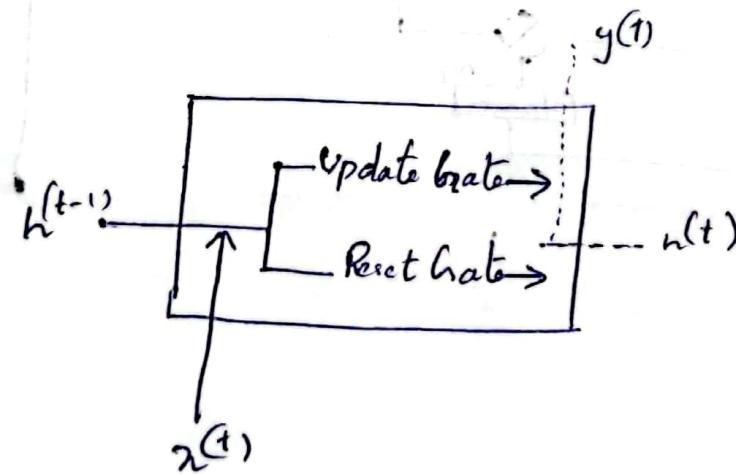
LSTM

CRU



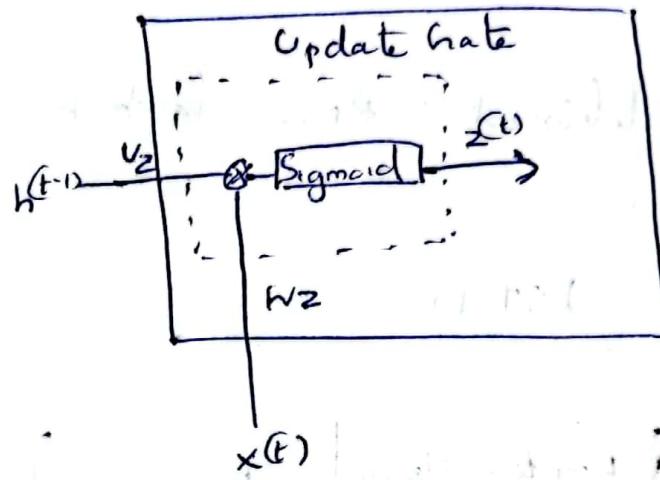
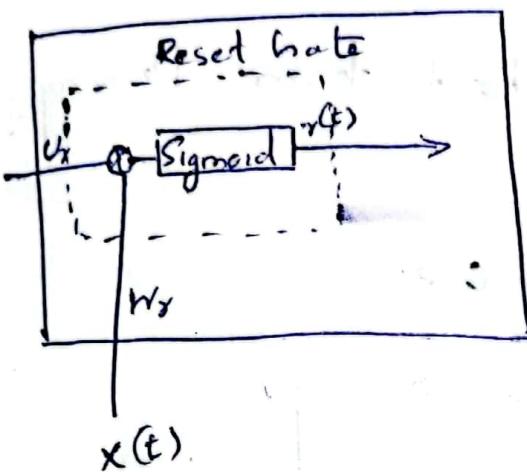
LSTM has 3 gates  $\rightarrow$  i/p, o/p, forget

CRU has 2 gates  $\rightarrow$  update and Reset.



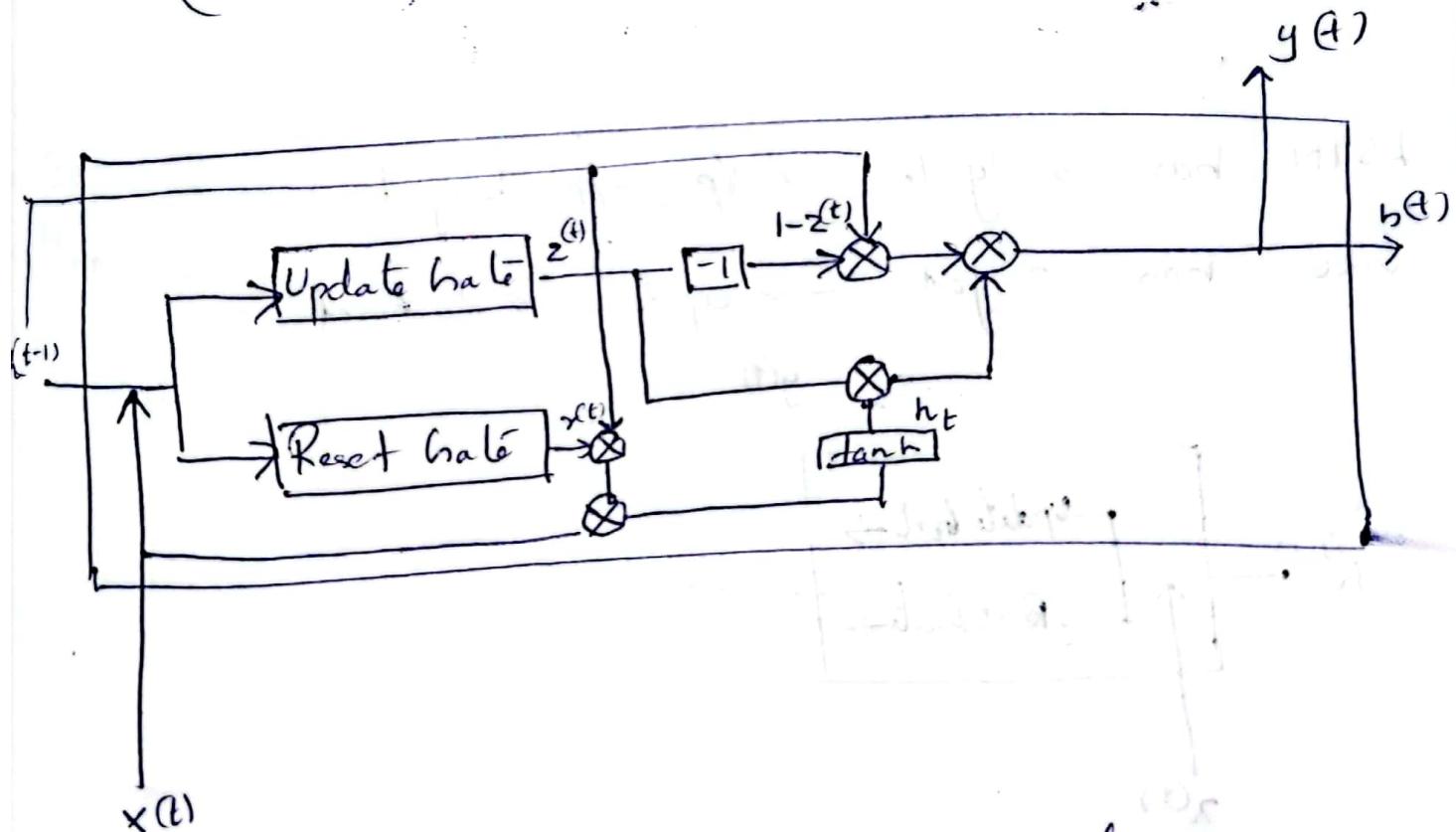
Update Gate helps : How much of Past memory to retain

Reset Gate : How much of Past memory is forgotten.



$$r^{(t)} = \sigma(w_r x^{(t)} + w_{r,h} h^{(t-1)})$$

$$z^{(t)} = \sigma(w_u x^{(t)} + w_{u,h} h^{(t-1)})$$



Difference b/w LSTM and GRU

### GRU

- 1. 3 gates:  $Y_p$ ,  $O_p$ , forget
- 2. More accurate on longer sequence, less efficient
- 1. 2 gates: reset, update
- 2. More efficient computation wise, getting more Regular.

## Transformers

- Transformers are a type of neural N/W archite.
- able that transforms or changes an i/p int. an o/p sequence.
- It enable large scale models.
- Enable faster customization
- Facilitate multi-modal AI System
- AI research and industry innovation.

### Use Cases of Transformers

1) NLP 2) Machine Translation

3) DNA Sequence Analysis

4) Protein Structure Analysis

- Self attention Mechanism is Transformer.

# Components of Transformer architecture.

for image refer AWS : Transformers  
and blog : The Illustrated Transformer by  
Jay Alammar.

## Components

- 1) Input embeddings
- 2) Positional embeddings
- 3) Transformer block
- 4) Linear and Softmax blocks

## Different Types of Transformer models?

- 1) Bidirectional Bidirectional Transformers - BERT
- 2) Generative Pretrained Transformers
- 3) Bidirectional and autoregressive transformers
- 4) Transformers for multimodal Tasks
- 5) Vision Transformers

Difference b/w transformers and CNN ?

" " " RNN ?

BERT (Bidirectional Encoder Representations from Transformers)

- Refer Blog by Jay Alammar.
- BERT can generate contextualized embeddings
- BERT is a model with absolute position embeddings so it's usually advised to pad the i/p on the right rather than the left.
- BERT was trained with the masked language Modelling (MLM) and next sequence prediction (NSP) objectives. It is efficient at predicting masked tokens and at NLU in general, but it is not optimal for text generation.