Natural Language Processing

① NLP problem use case.

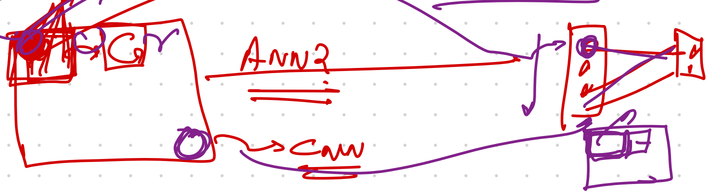
⊙ How do we structure words / sentence

↓

Feature engineering

○ Sequence models → RNN, LSTM, GRU, Transformer

Attention



vectors

words → Embeddings → it's a way to structure our

language

words / sentences / documents → Embedding Vectors → Sequence model

Large language models

RLHF → ChatGPT

Img → Text

multimodal

Question → Encoder → [ ] → Decoder → Answers

Prompt

Natural language → is a Sequence

This is a great batch

Great this is batch

Natural language Processing

Understanding information from text

- Unstructured →
- Content / Sentiments / slangs / Sarcasm

- Indigu . . . . . .
- Romanised words
- Tujhe main dekh lunga → English

मैं देख लूंगा

तुझे

Pyinclic translation

Phonetics

Sender

- Structured

Use Cases:

Translation

gmail auto type → auto fill

Search engines

Sentiment analysis

Gmail / spam → spam → HAM

Chatbots

Text classification, sum

NER, information

$y = f(x)$ → semantics

Language model.

Language model

Sentences are very unclear → Social media, Human Conversations

Preprocessing techniques:

Document classification

Document → Sports, Entertainment

Special characters @ https...

Removal of unwanted characters or patterns of characters from our source sentence.

Ticket classifier

Ticket → IT OS, IT App

Sentiment classification

+ve
-ve

Regex → a process of defining certain patterns using certain rules. (logic)

"I am very hungry. I will order pizzas and give to my friend."

Regex → Extract some defined patterns
from sentences

[w]
[s]

Name: Subhadeep
Age: 30
Education: master

~~Stop~~ words [Name]
words [Age]

Stopwords an, a, the
because and then

Translation, chatbot → Temp 25°C
→ I am ...

What's the temp here

मैं घर जा रहा हूँ → Tom I am going home

Stemming / Lemmatization

Embedding
words → vectors → model

Embedding

Frequency based
Count of the words
in the sentence

Prediction based.
vectors generated
through NN.

Content?

I am going home., home is where heart is

I : 1 , am : 1, going : 1 home 2 , is : 2
where 1., heart : 1

Run, Running, Ran, Run

Go, gone, going

Go

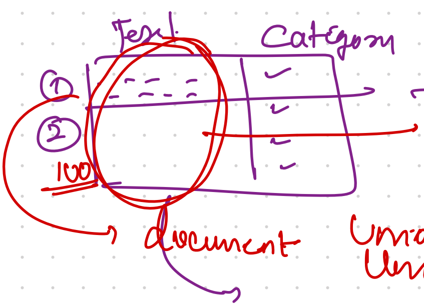
helpful

Sentences
vector
form

10:33 → 10:40 pm

1

How to represent sentences / documents in a vectorised form using frequency embeddings.



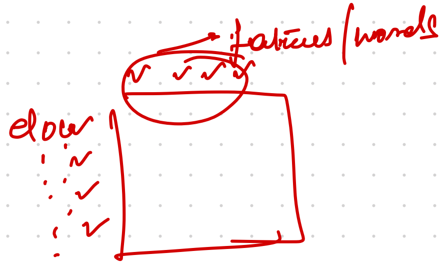
Union of all the words → Vocabulary.
Union of all the documents → Corpus
one word → Token

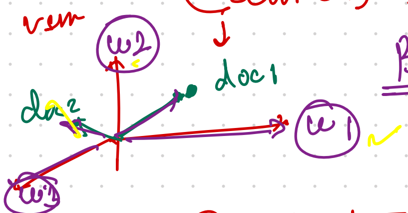
Bag of words model

Document classification

(Document) → Outcome
↓
doc 1

BoW

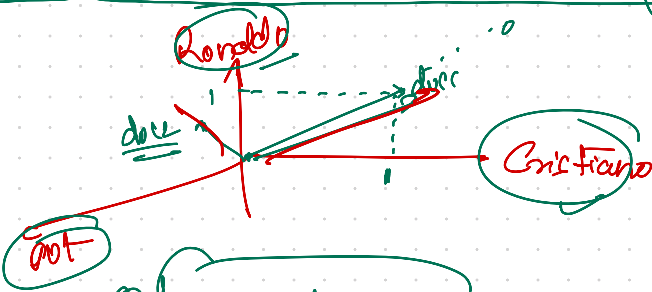




① Cristiano Ronaldo got transferred. Cristiano is great
→ Sports

② Puja hegde performs in Cannes.

	Cristiano	Ronaldo	got	transferred	Dua	Lipa	Performs	in	Concert
①	1	1	1	1	0	0	0	0	0
②	0	0	0	0	1	1	1	1	1

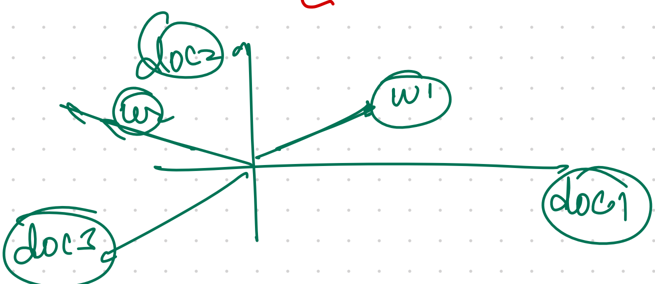


Problem: ① too many dimensions

② does not maintain content

→ order not maintained

By definition you and all words are independent



Stemming / Lemmatization

Go, going, went → Go

Run, Running, Ran → Ran

Stemming & Lemmatization Enforcement

Porter Stemmer

Lemmatization

Running → Run (Verb)

Ran

going → go

gone

Speeding

Speeds

Caresses

Cry → cri

I am go(verb) to visit my home and will

I am going to visit my home and will make → make

Sun my ac is running

sun my ac is run

Dependency Par

my run was awesome

POS

root

"I am great at public speaking"

Stemming & Lemmatization

(word)

word tokenizer

Sentence → words

sentence tokeniser

documents → sentence

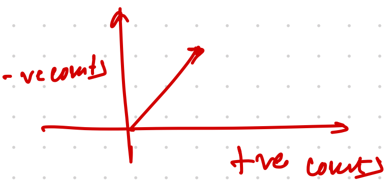
.split(' ')

"I | am | going | to | my | home" .split(' ')

['I', 'am', 'going', 'to', 'my', 'home']

~~w~~ nltk → word tokeniser (' ')

tweet → words → (count of the word in +ve tweets, count of the word in negative tweets)



tweet → (count of the +ve word, count of -ve words, 1)

+ve 50 10

60 100

+ve -ve

tweet: The game

+ve

tweet (110, 110)

freqs

{(game, 0): 20
(game, 1): 40
(awesome, 0): 50
(awesome, 1): 90}

tweet: game is awesome

game awesome

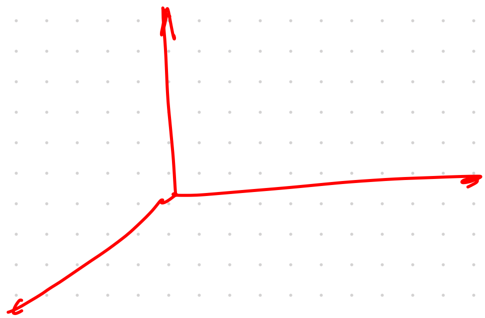
pos = freqs[(game, 1)] = 40 + freqs[(awesome, 1)] ,90

neg = freqs[(game, 0)] = 20 + freqs[(awesome, 0)] 50

pos = 130

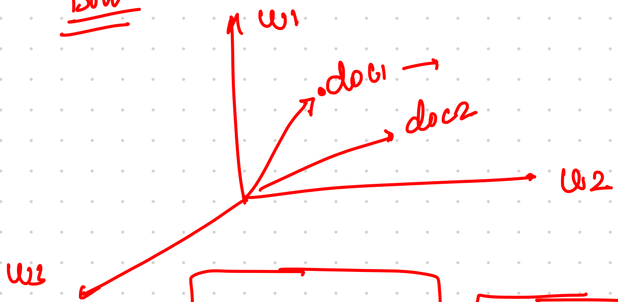
neg = 70

tweet = (1, 130, 70)



NLP → words/docs into vector space models

BoW



Sentence 1 / doc1 : I am happy → Vocabulary

Sentence 2 / doc2 : I am sad.

Union of all docs

	w1	w2	w3	w4
doc1	1	1	0	0
doc2	0	0	1	1

Count vectorizer

n	w1	w2	w3	w4
doc1	2/n	3/n	1/n	1/n
doc2 / n2	1/n2	1/n2	1/n2	1/n2

① donot maintain any order

doc1 I am happy today. today happ I am

	I	am	happy	today
doc1	1	1	1	1

frequency based embeddings donot capture the context

Term Frequency Inverse Document Frequency.

Football

Football.

Count(Football)

count(Football)

high ↑↑

Entertainment, Sports, Science.

Football

count()

x

Count()

x

Football important word.

tf(Term frequency) × idf

$$\frac{f(w_2, doc_1)}{\text{Total number of tokens in } doc_1} \times \log\left(\frac{N}{n}\right)$$

$w_1 \quad w_2 \quad w_3 \ldots w_n$

doc1
doc2
doc3
⋮

$N = 100$

total frequencies of all the tokens in doc1

number of documents in which w_2 occurs.

doc1 : Ronaldo is lovely, Ronaldo is rich.

doc2 : Ronaldo is famous.

doc3 : Messi is famous

doc1 : Ronaldo is lovely, Ronaldo is rich.

doc2 : Ronaldo is famous.

doc3 : Messi is famous

$$\boxed{\text{Normalised term frequency} \times \log\left(\frac{N+1}{n}\right)}$$

$1/3$

	Ronaldo	Messi	lovely	Rich	famous
doc1	$\frac{2}{3}\times$	0			
doc2		0			
doc3		$\frac{1}{2}\times\log(4/1)$			

$\log(4/2)$

$\log\left(\frac{N+1}{n}\right) = 4/1$, $3+1$

$$\frac{2}{4} \times \log(4/2)$$

$$\boxed{\frac{2}{4} \times \log(3/2) \rightarrow n \quad \underline{\log(1)} \rightarrow 0}$$

$N+1$

$\log(N/n) \rightarrow n$ is N, $n \geq 1$

TF-IDF $(t, doc1)$

$$= \text{Normalised term frequency} \times \log\left(\frac{N+1}{n}\right)$$

$\underline{N}$ → total number of doc in corpus.

$\underline{n}$ → total number of docs in which term occurs.

Amazon

	w1	w2	w3	...	wn	Class
doc1						Spam
doc2						Y
doc2						N
docn						N
						Y

find out similar documents

doc1 (Product)

doc3 (Product)

addresses 1 2 3

Using frequency based embedding we can find similar documents

Convosight

Brands

J&J Shampoo

Facebook Communities

4000

1.5m

1.0m mothers

Baby Products

Target audiences → mothers of Baby child 0-5 years

Admin

10 → 20 → J&J

10 25

14000 comm

→ 1.6 billion computations

seed communities

7

J&J

→ 0.8

Similarity

7

14000

500 communities

15000

7

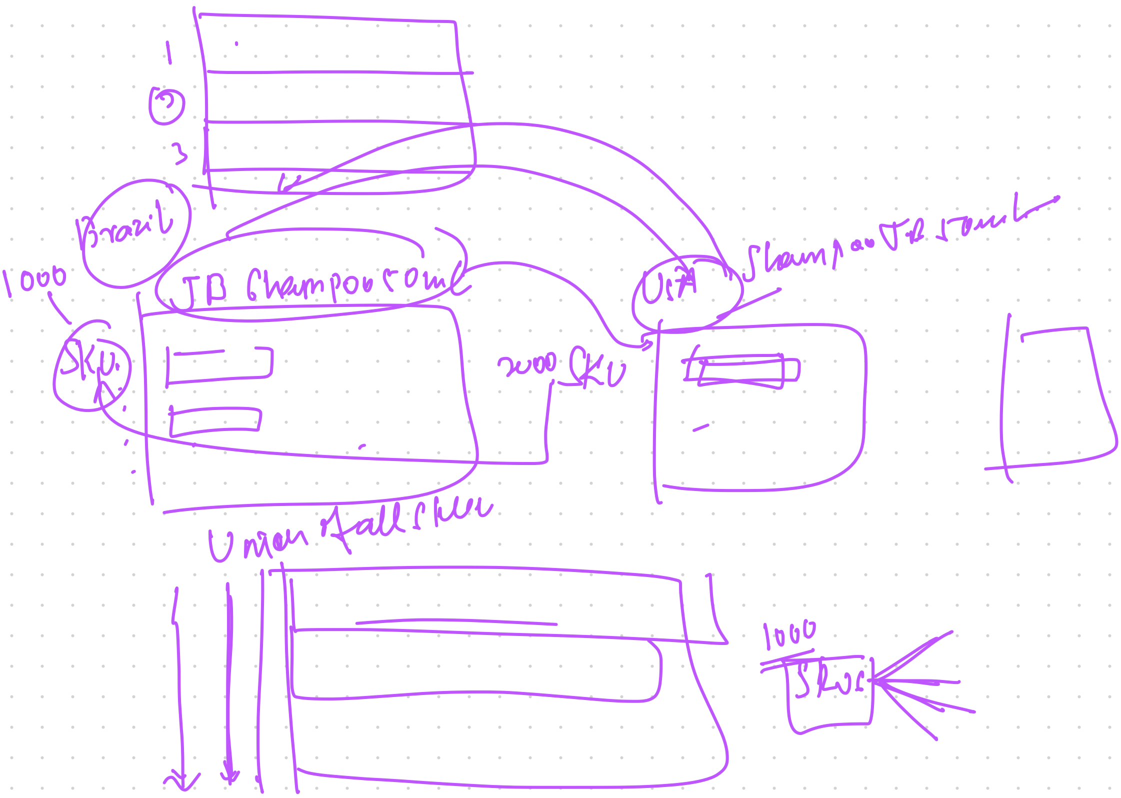
14000

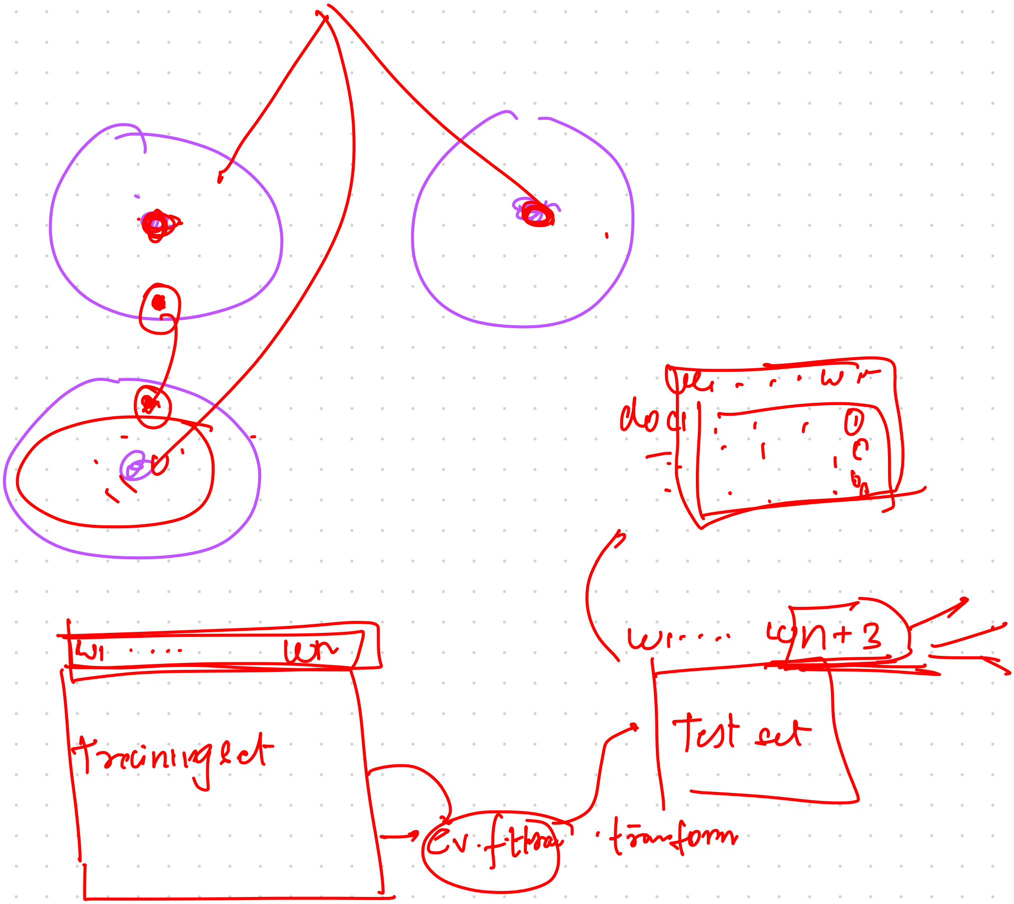
I am happy to teach you

Food and Recipes for mother, nutrition for babies

Hey mother, see this new recipe

I created




doc1
w1 . . . . wn
Training set
Test set
w1 . . . . wn+3
cv.fit
.transform

corpus = ["I am ... , " ...]

2 for sentence in corpus

log (N/n)

two happy friends

doc1 → x log N/n

doc2 → x log(N/n)

x

I am great, thank you.

Unigram:

'I' 'am' 'great' 'thank' 'you'

Bigram

'I am' 'am great' 'great thank' 'thank you'

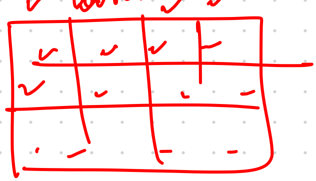
'I' 'I am' 'am' 'am great'

Embedding → word2vec

Frequency based embedding

words

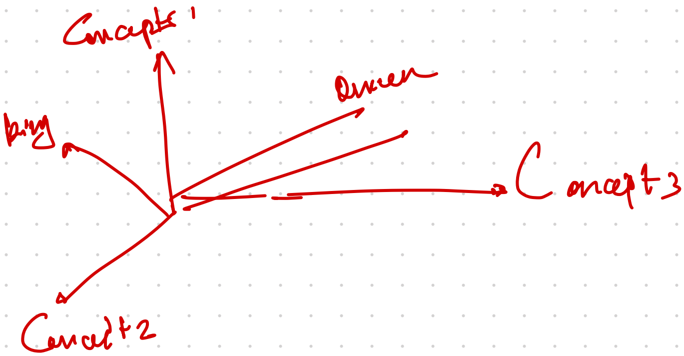
docs



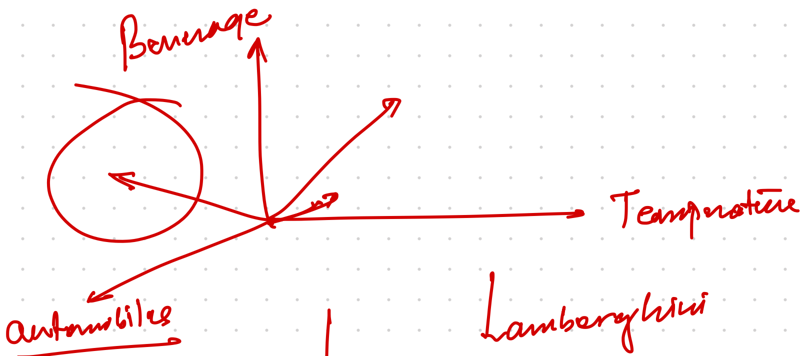
Prediction based Embedding

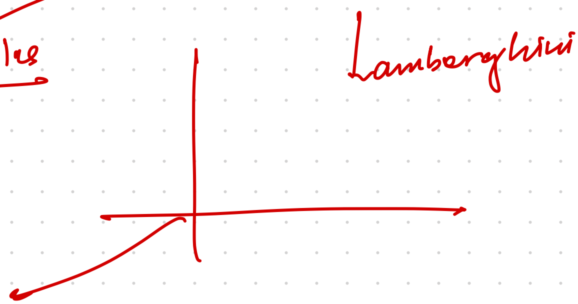
↓

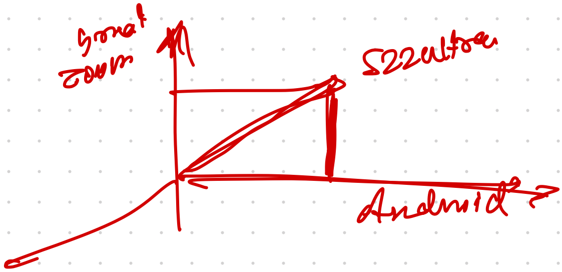
Prediction based embedding



Hot coffee







Embeddings are dense representation of concepts

Abstract non-linear combinations of multiple words.

Embeddings (unsupervised)

Word2vec an model

Embeddings SVD ~~x~~

Embedding from sequence models → (Supervised way)

LM, Seq (RNN, LSTM)

Word2vec

CBOW
Continuous Bag of words

Skip gram model.

The class was excellent we cannot achieve word2vec model. Next class Bhubaneswar make talk 2nd (drone).

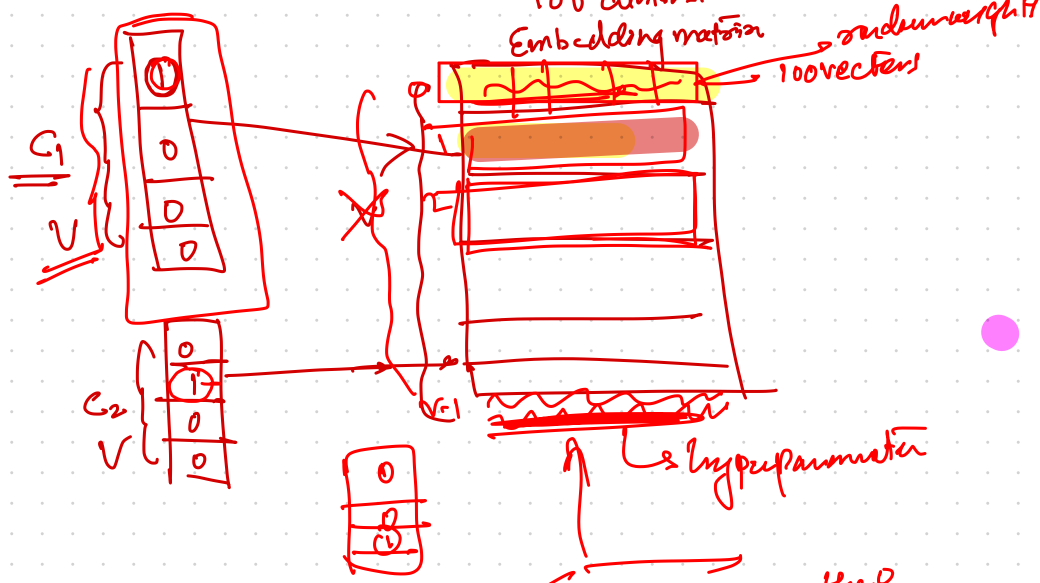
Context words	Target word
The, was	Class
Class, excellent	was
was, we	excellent

Context window =1

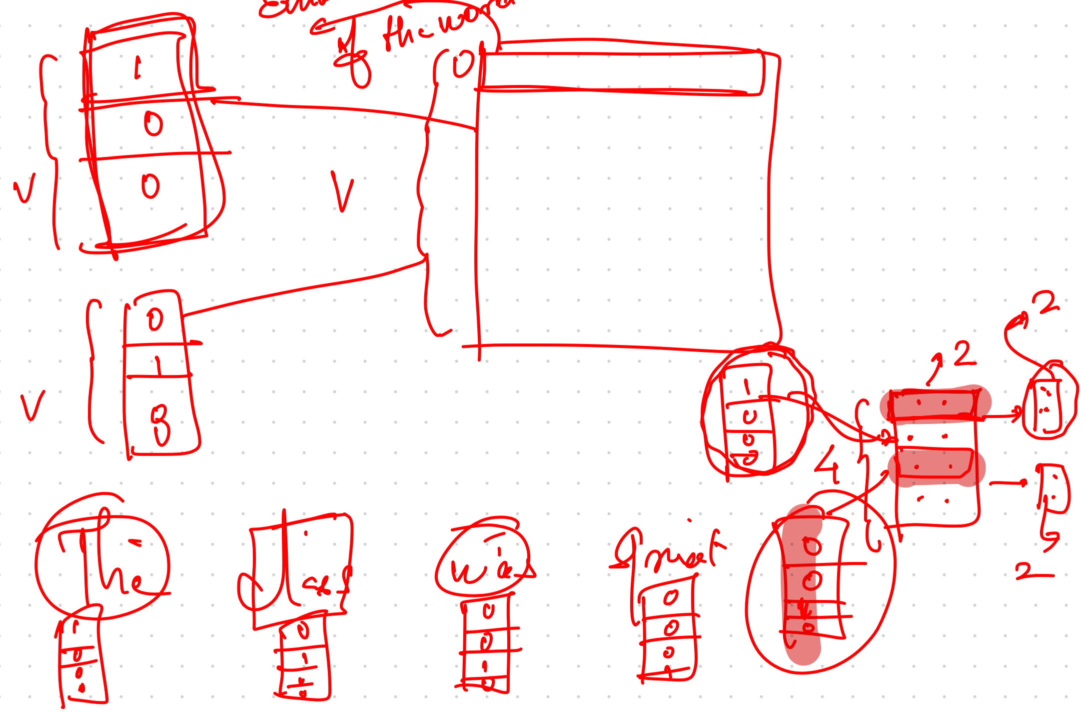
Context window =2

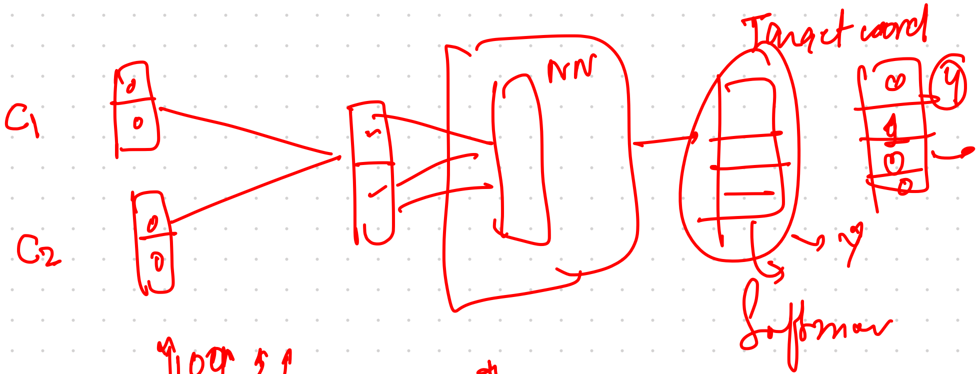
Class The

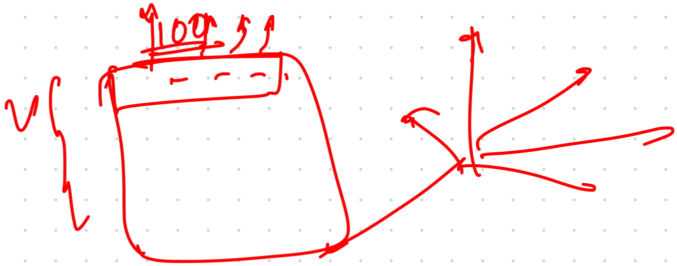
Each word is represented as a one hot encoded vector

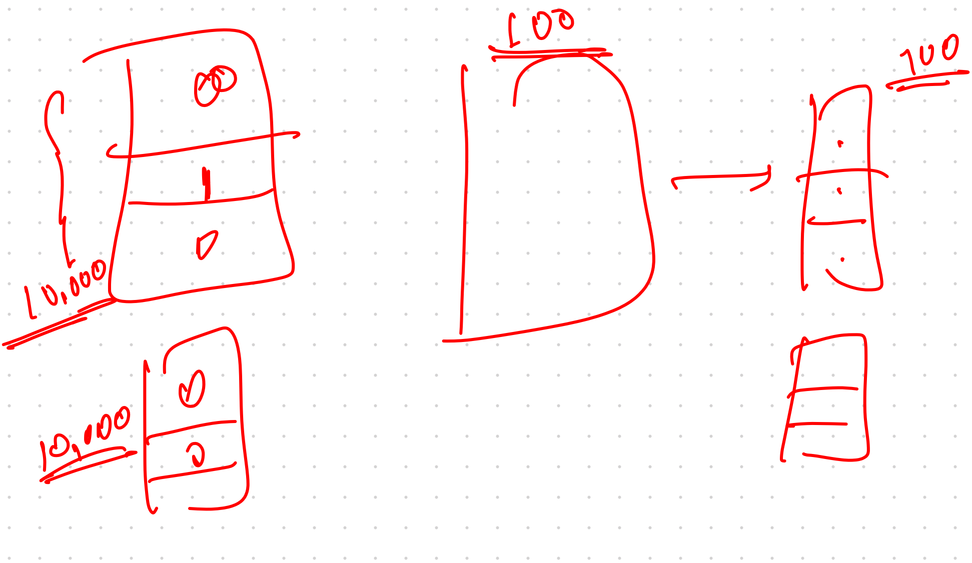


Embedding vector of the word which has 1 in the 0 index




C1
C2
NN
Target word
y
y'
Softmax


100
V


10,000
10,000
100
100

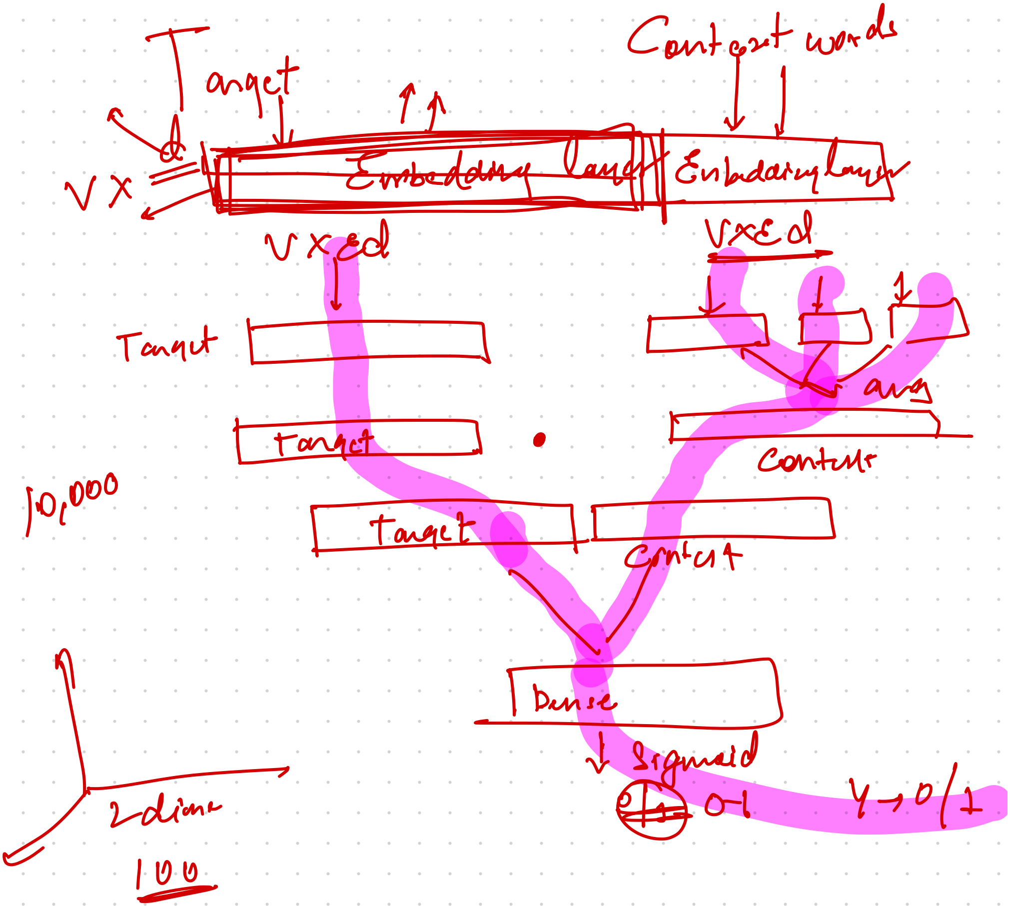
Skip-gram model

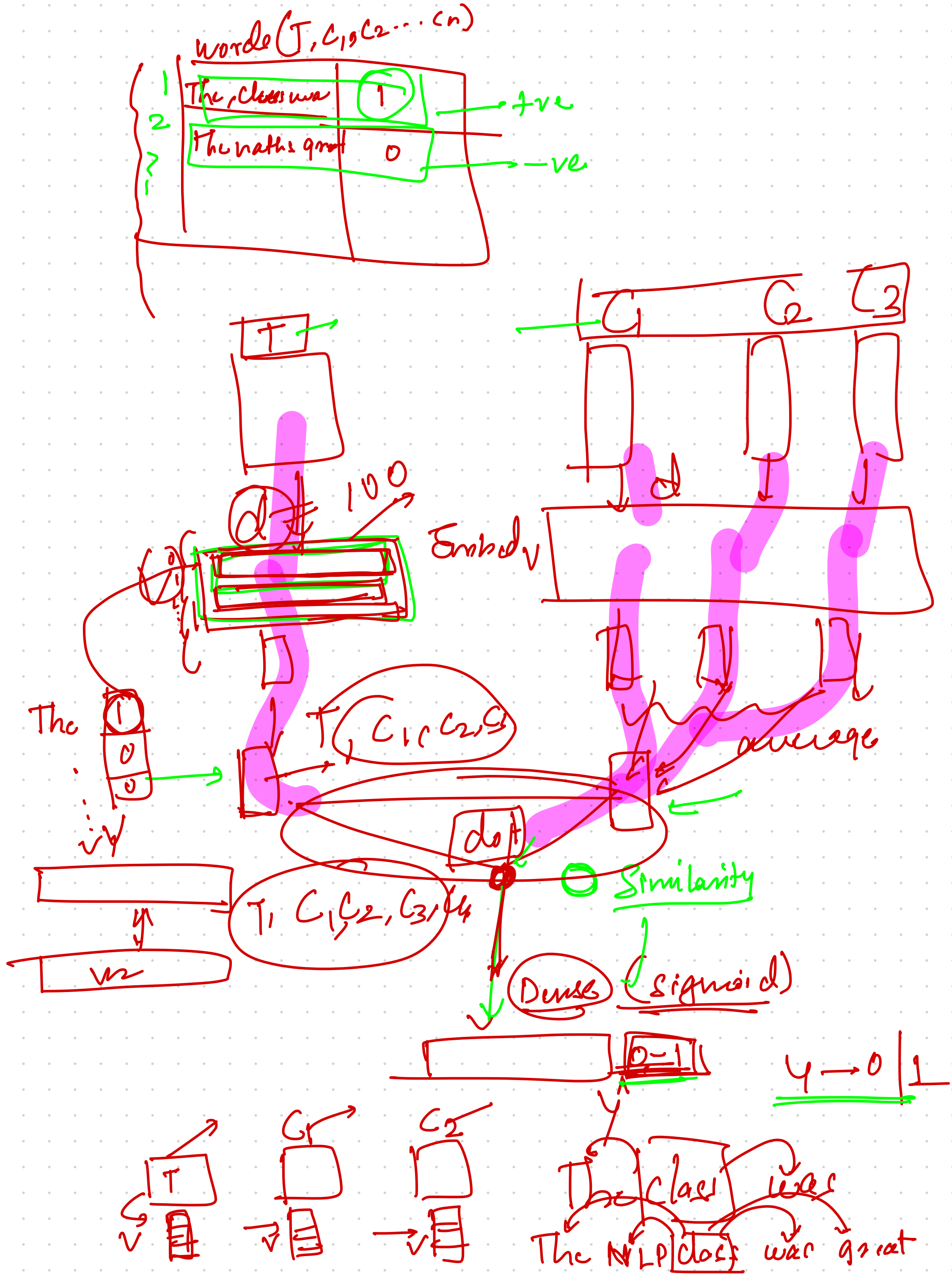
The class was great and we got to know new concepts

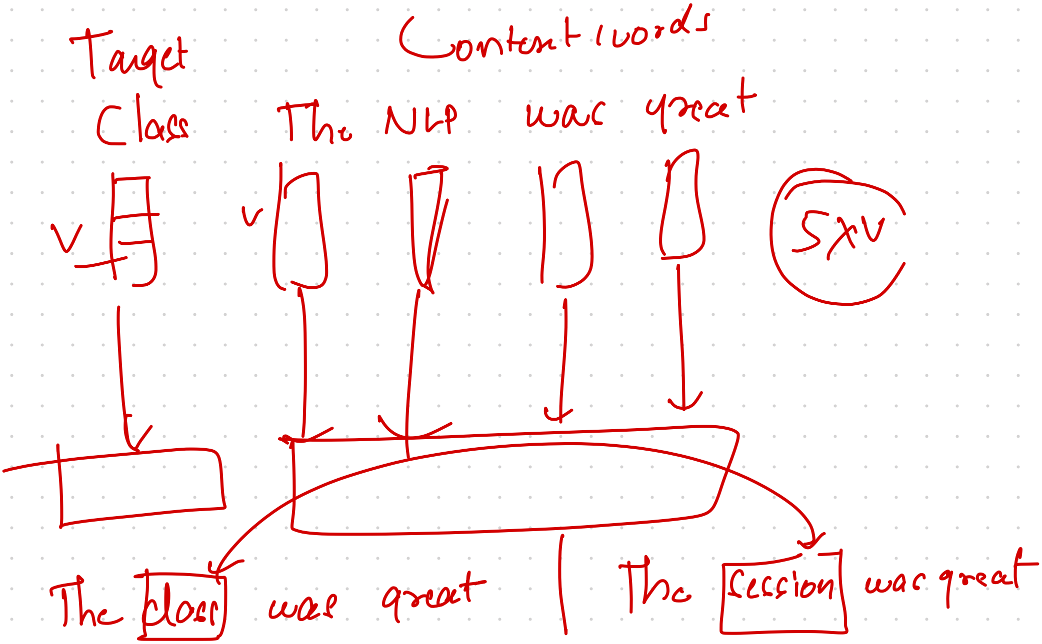
Target	Context		
class	The, was	(class, The, was)	1
was	class, great	(class, everything, ...)	0
great	was, and		

+ve sample

-ve sample
-ve sample



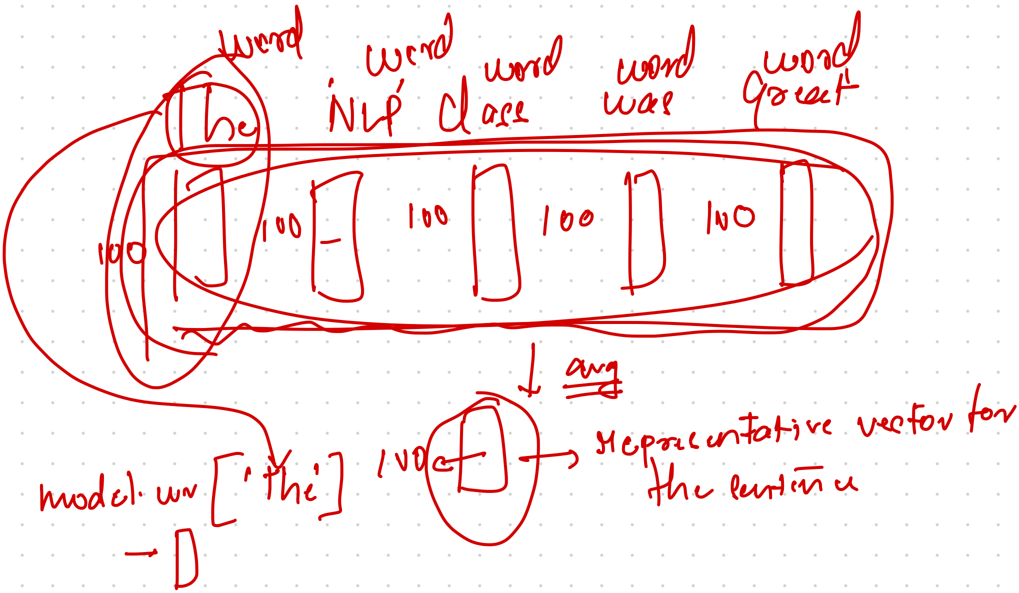

words (T, c1, c2 ... cn)
The, class was
1
The maths great
0
+ve
-ve
T
C1
C2
C3
d
100
Embed
The
1
0
0
T, C1, C2, C3
average
dot
Similarity
T, C1, C2, C3, C4
Dense
(sigmoid)
0-1
y → 0 | 1
y
T
C1
C2
The class was
The NLP class was great

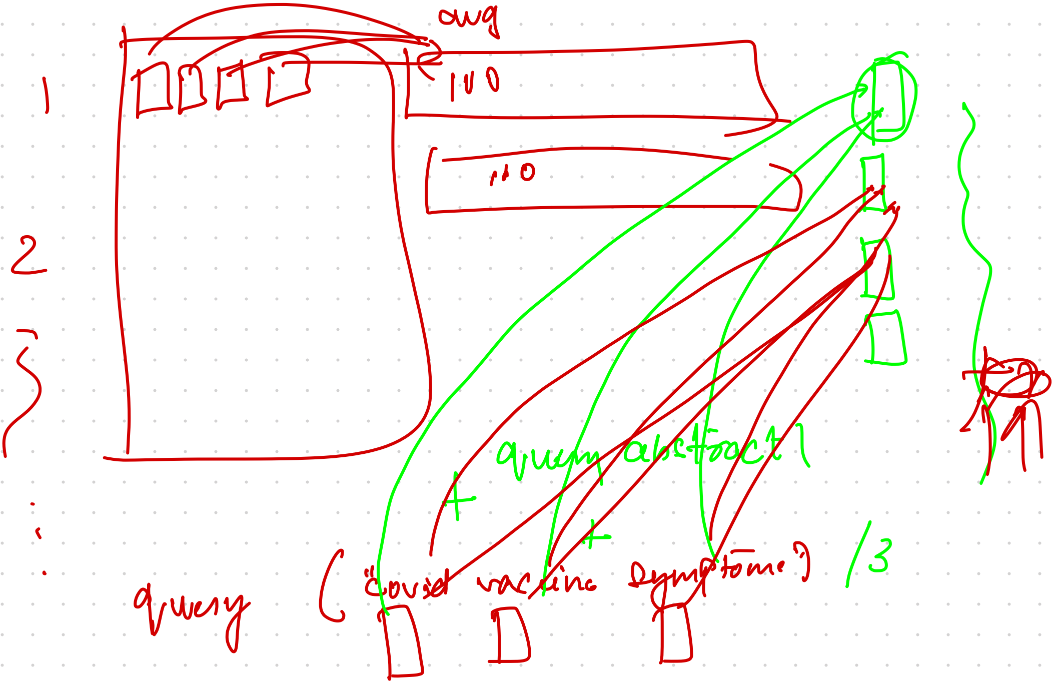


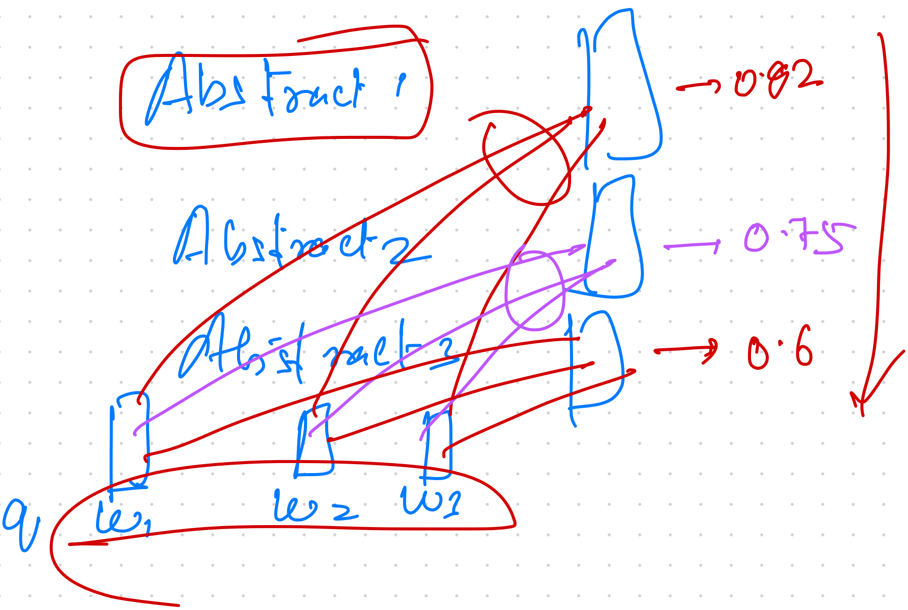
["The class ~~was~~ great"
The NLP ~~is~~ interesting]

abstract1 abstract2

[['the', 'class', 'was', 'great'], ['the', 'NLP', 'is', 'interesting']]


word
word
word
word
word
The
NLP
class
was
great
100
100
100
100
100
100
avg
100
representative vector for the sentence
model.wv['the']


avg
1
100
110
2
query ("covid vaccine symptoms") /3
query abstract]


Abstract 1
Abstract 2
Abstract 3
→ 0.82
→ 0.75
→ 0.6
q
w1
w2
w3

Language modelling

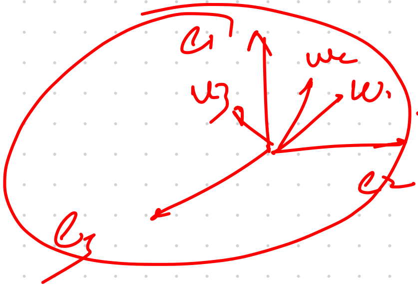
Prev

Topic modelling

RNN

Next

Sequence model.



→ word2vec

CBOW

skip gram

Language model

w3

I am grateful to you

$p(w_3 / (I, am)) \rightarrow$

I am grateful to you

I am

I am grateful to you

I am

$P(w_3 / (I, am))$

I | am | grateful | to | you

I am thankful for this life

$= P(w_6 / (I, am, thankful, for, this))$

Markovian Assumption

I am (thankful)

K=1

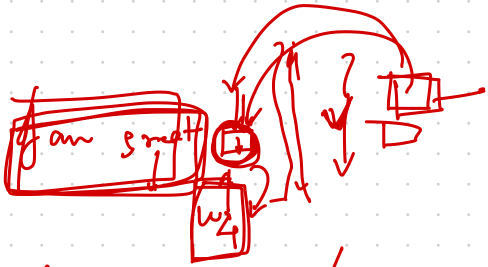
K=2

I am grateful to you and I ~~love~~ my life.

K=1

K=2

a particular word at a position is dependent on prev two words



$p(\_\_\_\_ \mid (I, am, great))$

$K=0$ ~ $p(\_\_\_)$ Unigram

$\boxed{K=1}$ ~ $p(w_4 = v_i \mid great)$ Bigram →

$K=2$ ~ $p(w_4 = v_i \mid great, am)$ Trigram

$K=3$ ~ $p(w_4 = v_i \mid (great, am, I))$ quadgram

thankyou

I am great ____

(am, great)

v_i	...	life	hello
50		0	20

am great v_i

(am, great, music) → 0

(am great, thankyou) → 100

I am   am good   good how   how are   are you   you too

2   2   1   1   1   1

I am

I $\boxed{\text{am}}$ good

v_i

I ×

am ×

good

how ×

are ×

you ×

too ×

$P(w_3 = ? \mid \text{am})$.

I am ___ $P(\_ \mid \text{am})$

$P(w_3 = v_i \mid \text{I am})$

I am good, I love my country. I love India

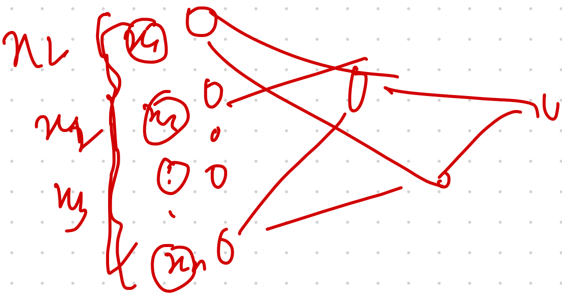
K gram

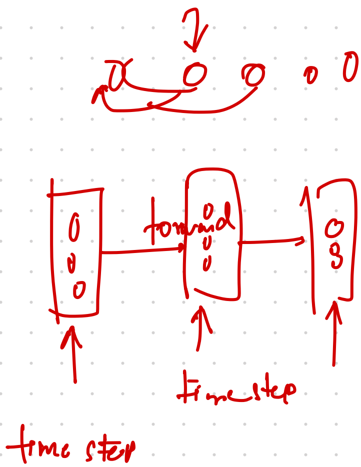
k = large — 100

$P(\downarrow \mid [\ldots\ldots])$

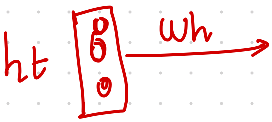
Recurrent Neural Networks

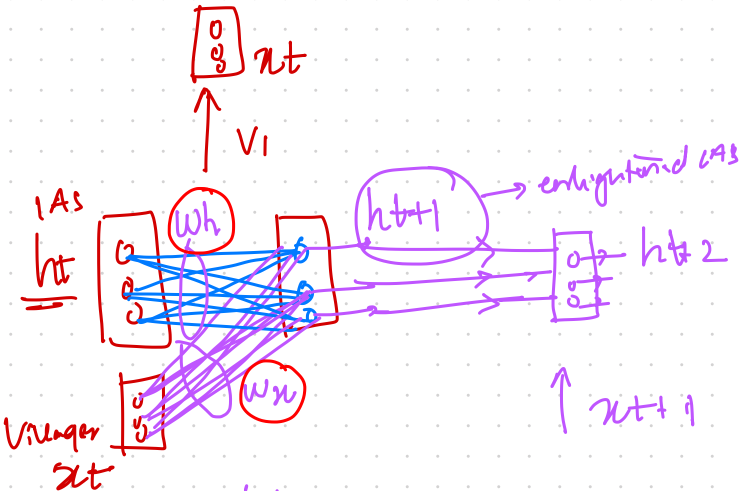
9:51 → ~~9:5~~ 10:00pm





IAS





~~tanh/Relu~~

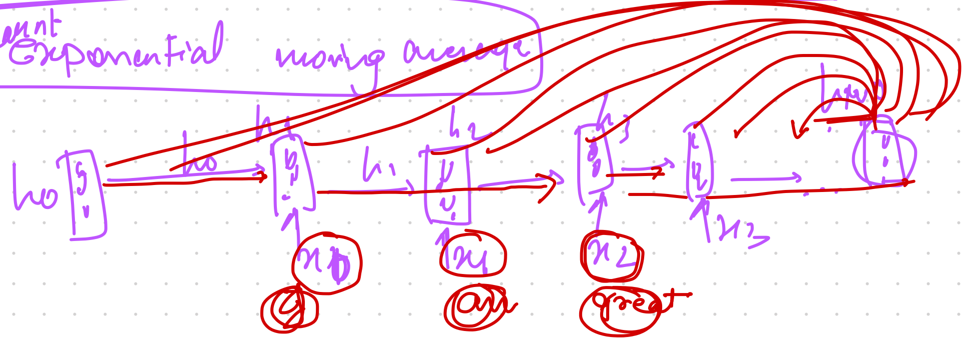
$$\boxed{h_{t+1} = \sigma(W_h \times h_t + W_x x_t + b)}$$

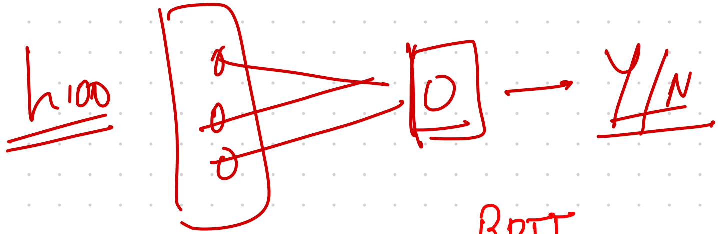
$$h_{t+2} = \tanh/\text{Relu}\,(W_h\, h_{t+1} + W_x x_{t+1} + b)$$

$$h_{t+2} = \tanh/\text{Relu}\,(W_h \times \tanh/\text{Relu}\,(W_h h_t + W_x x_t + b) + W_x x_{t+1} + b)$$

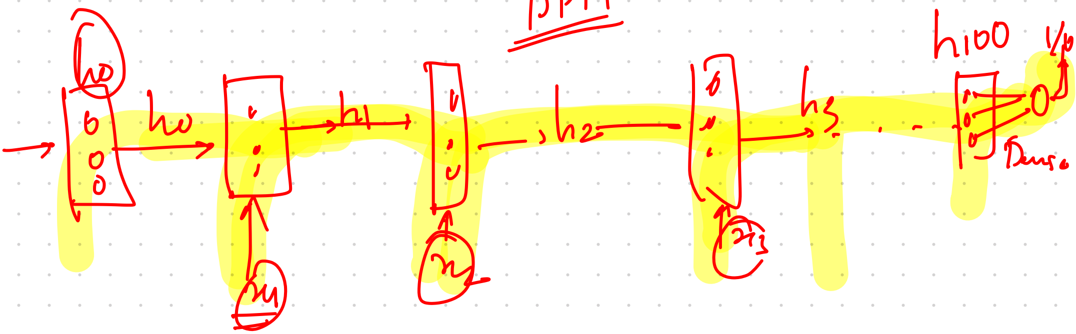
$$\boxed{h_{t+2} = \tanh/\text{Relu}\,(W_h h_t + W_h W_x x_t + b) + W_x x_{t+1} + b}$$

Non-linear learnt exponential moving average

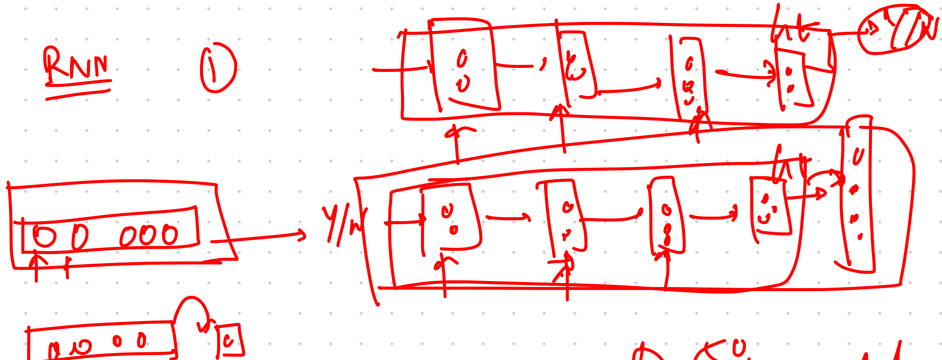


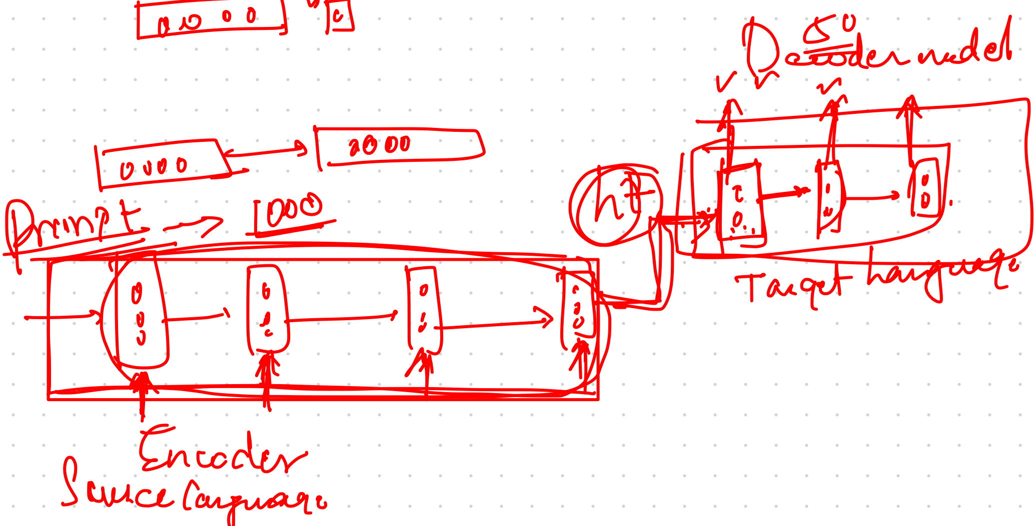

h100
Y/N

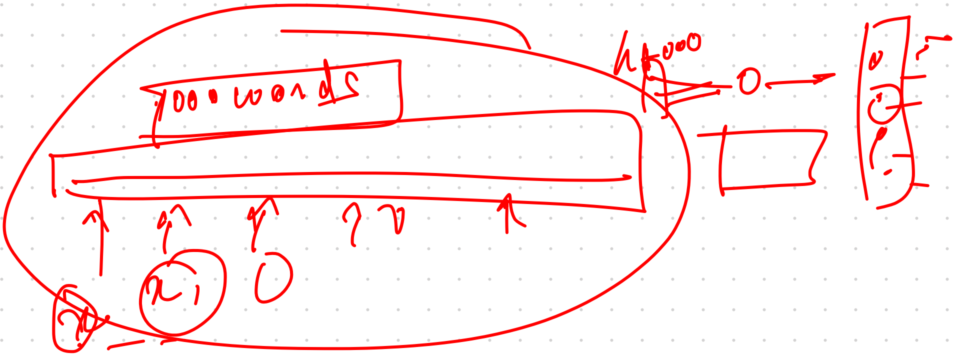
BPTT


h0
h0
h1
h2
h3
h100
Y/N
Dense

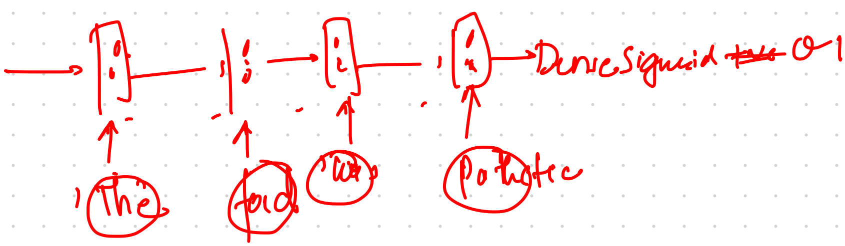
RNN ①


ht
Y/N
ht
Y/N


Decoder model
Print → 1000
ht
Target language
Encoder
Source language



NLP



1000x10

x	y
1 2 3 4 000000	
1 5 3 6 7 8 9 000	
. 000	

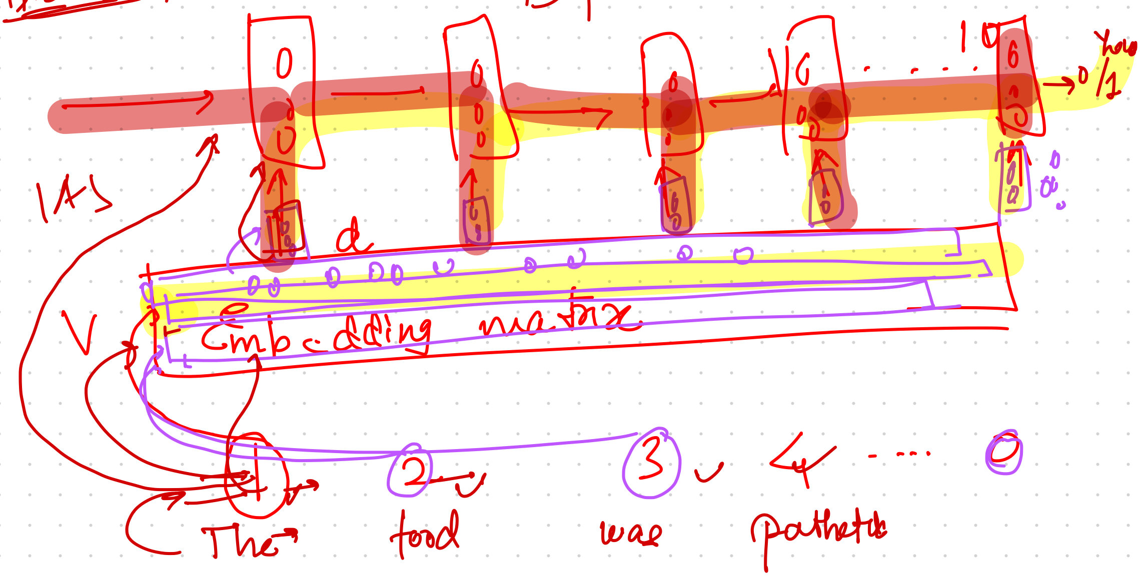
1000

x → max-length → 10

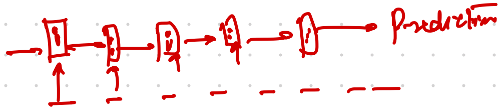
	x	y
1	The food was pathetic	-ve
2	The ambience was excellent, definitely a visit	+ve
3		

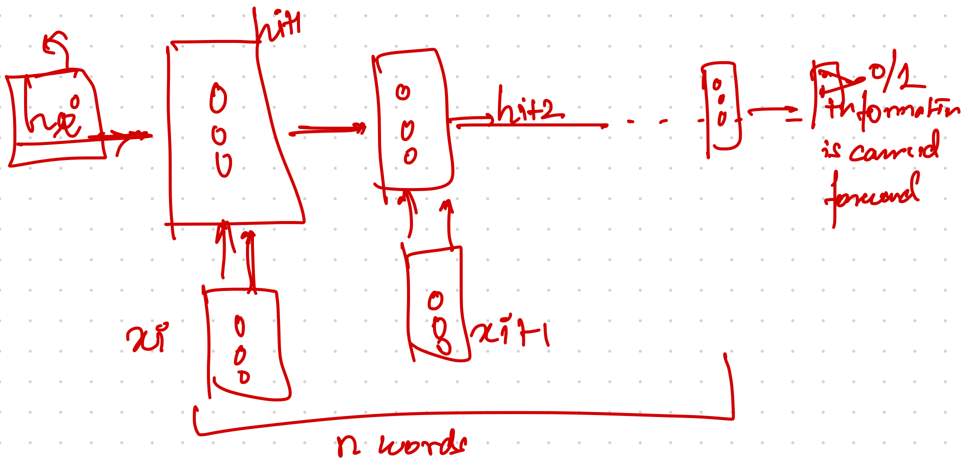
BPTT

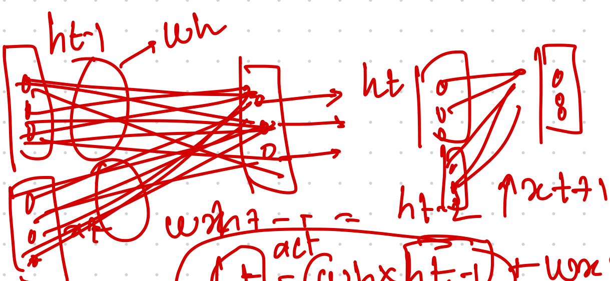
Bi: L case



RNN

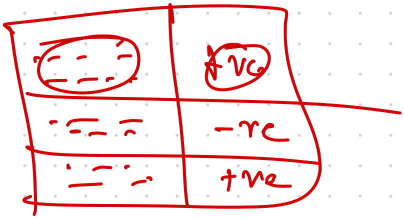


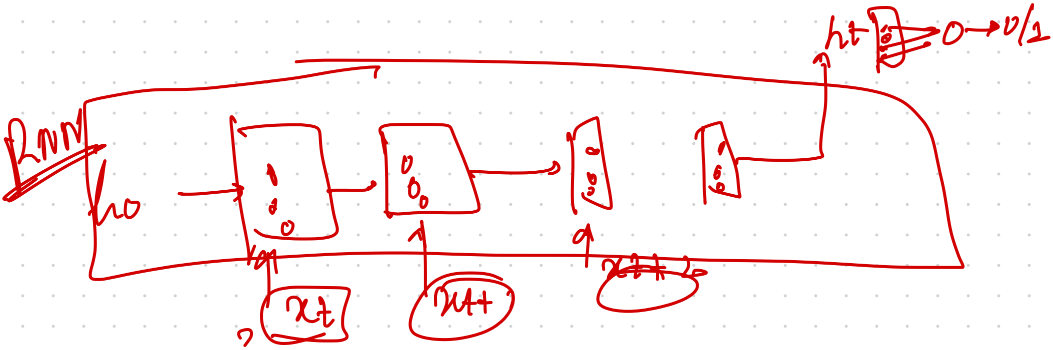


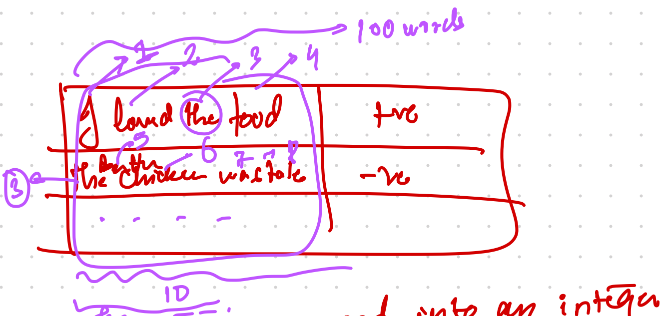


$$h_t = act\left(w_h \times h_{t-1} + w_x x_t\right)$$

$$\boxed{h_{t+1}} = act\left(w_h \times h_t + w_x x_{t+1}\right)$$

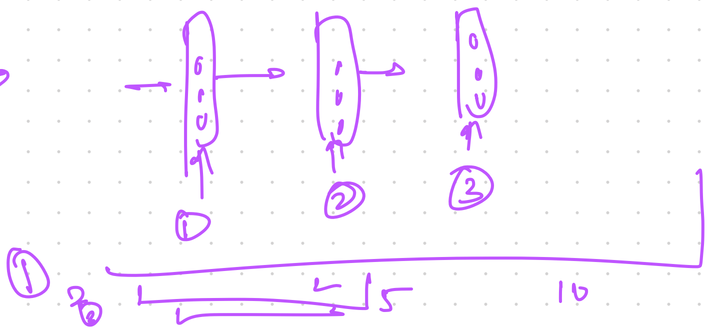






1) we will convert each word into an integer

= tokenizer.text_to_sequences (df['text'])



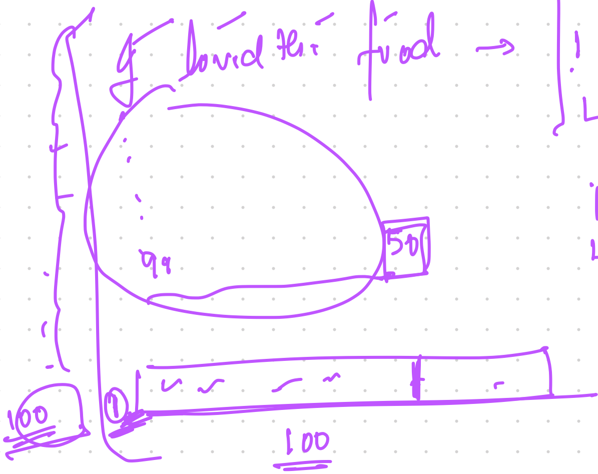
Padding

I loved the food → 1 2 3 4 0 0 0 0 0 0

50

1 2 3 4 7 8 9 10 000 42

50



= Pad-sequences (df['integer text'])

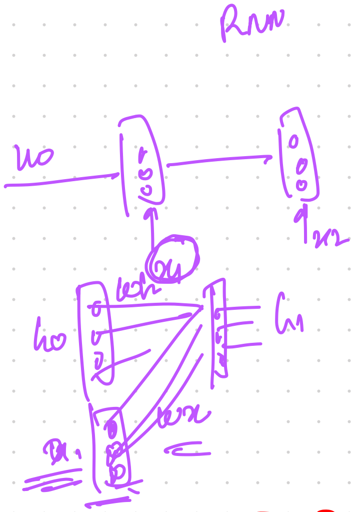
100x 50

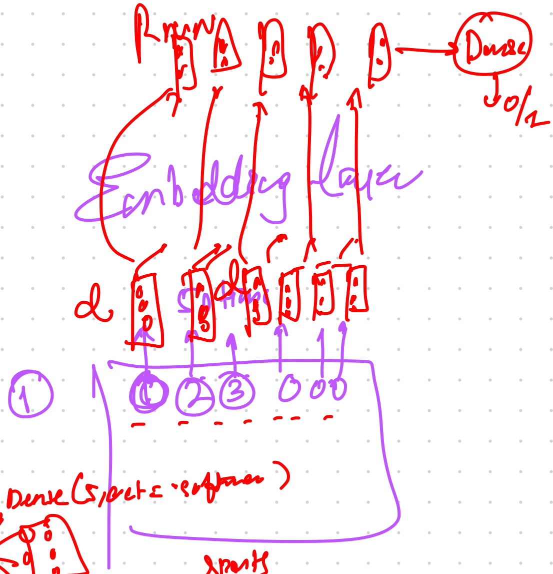
50

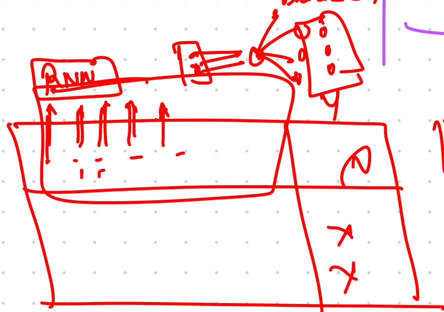
1 2 3 4 5 0 0 0 0	+ve
7 8 9 10 12 000	

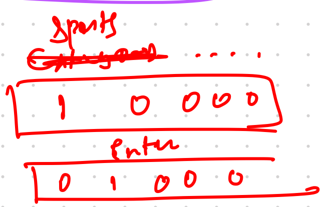
50

→ (RNN)

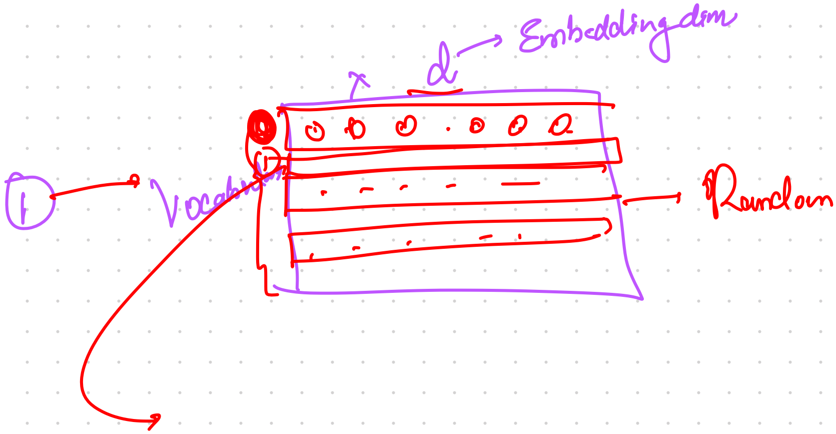

RNN
h0
h1


RNN
Dense
0/1
Embedding layer
d
1
2
3
Dense(5, act = softmax)

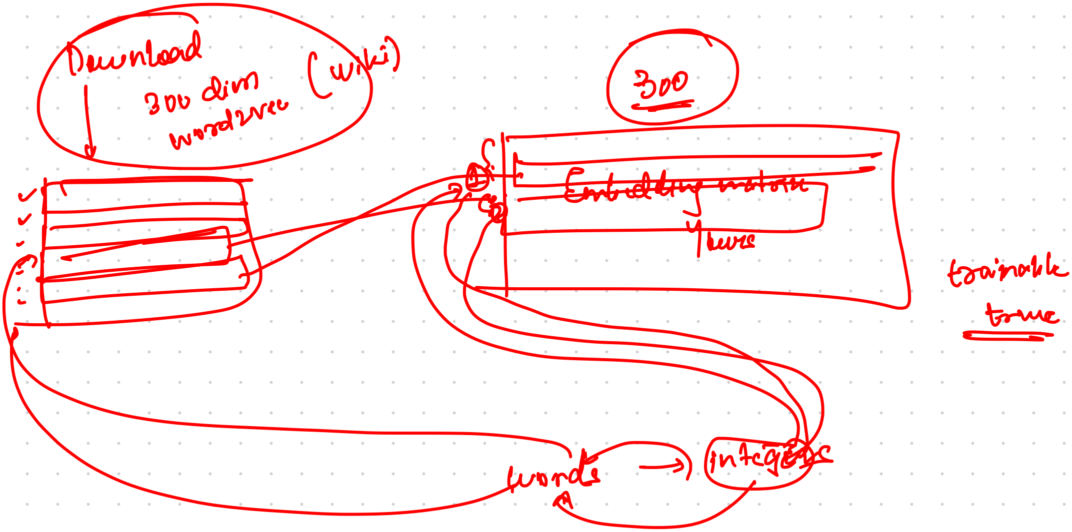

RNN

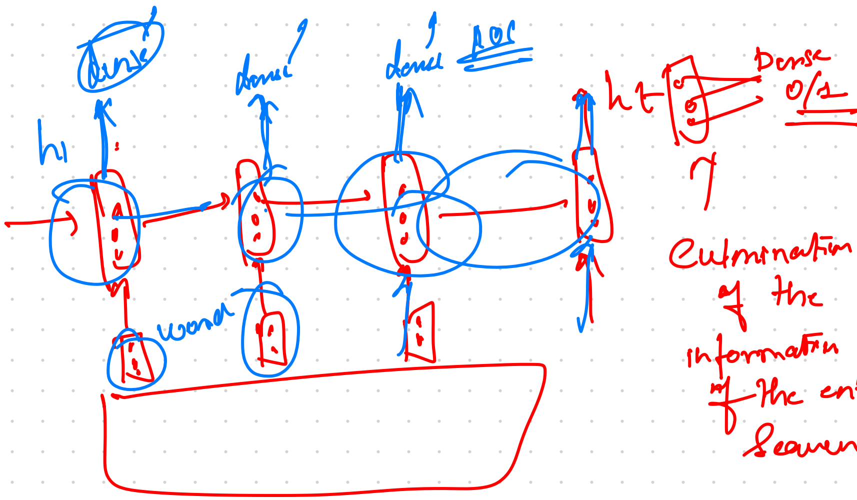

Sports
1 0 0 0 0
Enter
0 1 0 0 0


RNN
Category Cross
Categorical

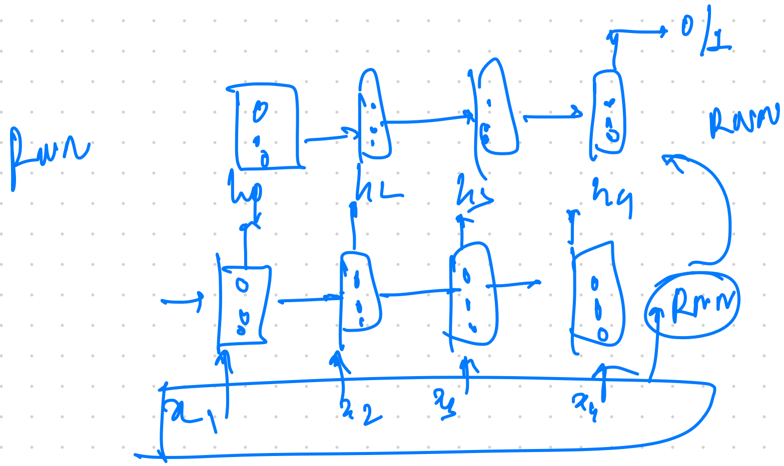


Embedding dim
d
1
Vocab
Random





Culmination
of the
information
of the entire
sequence

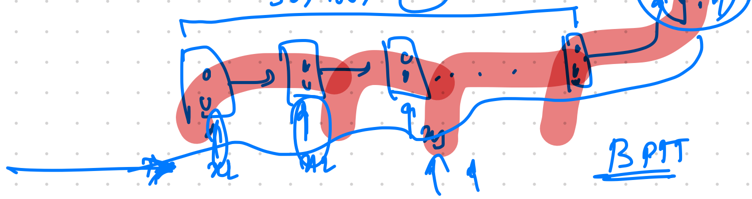

Rwn
0/1
RNN
h0
h1
h3
h4
RNN
x1
x2
x3
x4

RNN → some Problems

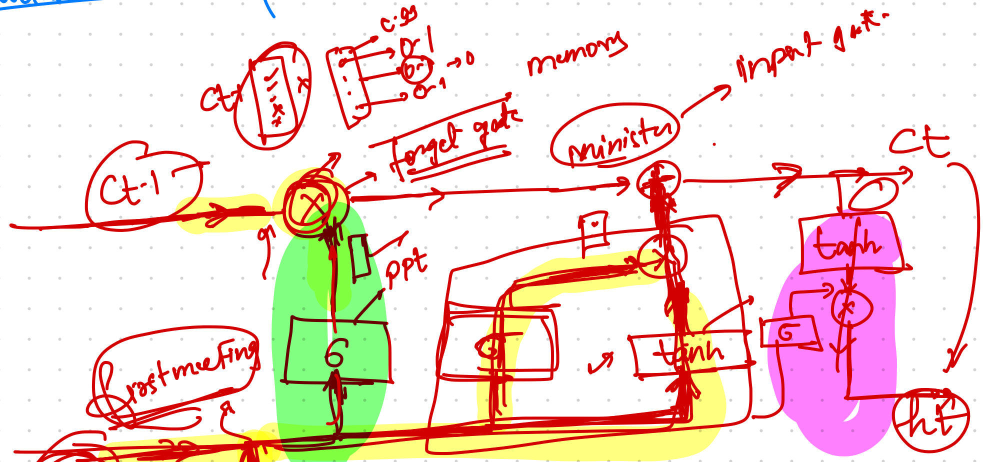
Vanishing Grad

Chain Rule (multiply multiple gradient)

50, 100, 1000



Solution to RNNs Problem:

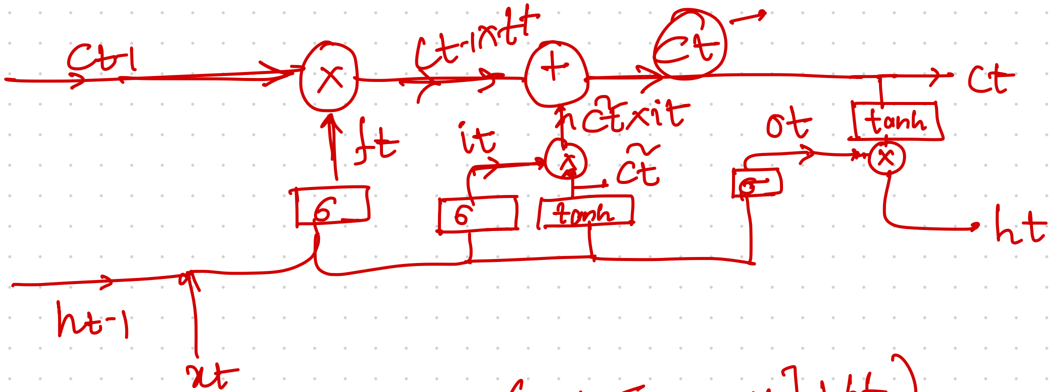


ht → short term memory

ct → long term memory

tanh → word

ht / IAf → memory manager for ct



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

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

$$\tilde{C}_t = \tanh\left(W_C\,[h_{t-1}, x_t] + b_C\right)$$

$$\boxed{C_t = C_{t-1} \times f_t + i_t \times \tilde{C}_t}$$

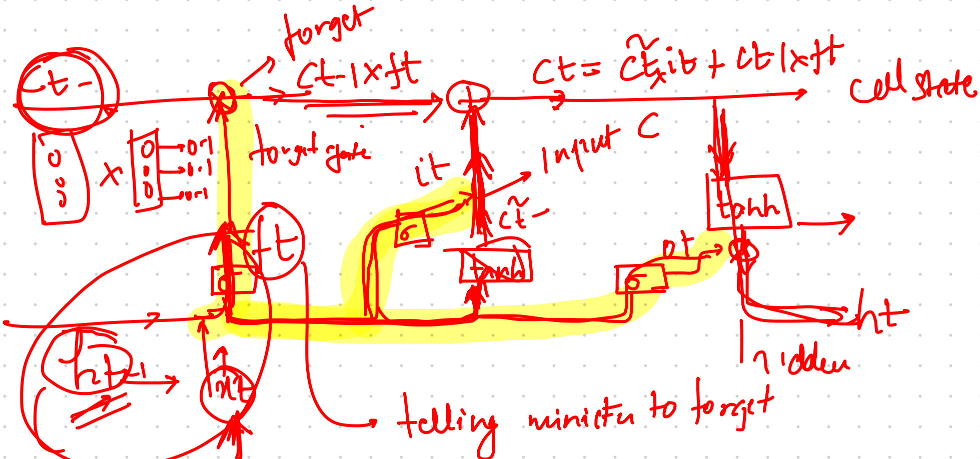
$$h_t = \tanh(C_t) \times o_t$$

$$o_t = \sigma\left[W_o\,[h_{t-1}, x_t] + b_o\right]$$

Long Short term memory

RNN → Long term dependencies

Long term memory
Short term memory

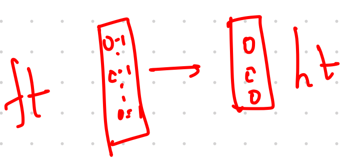


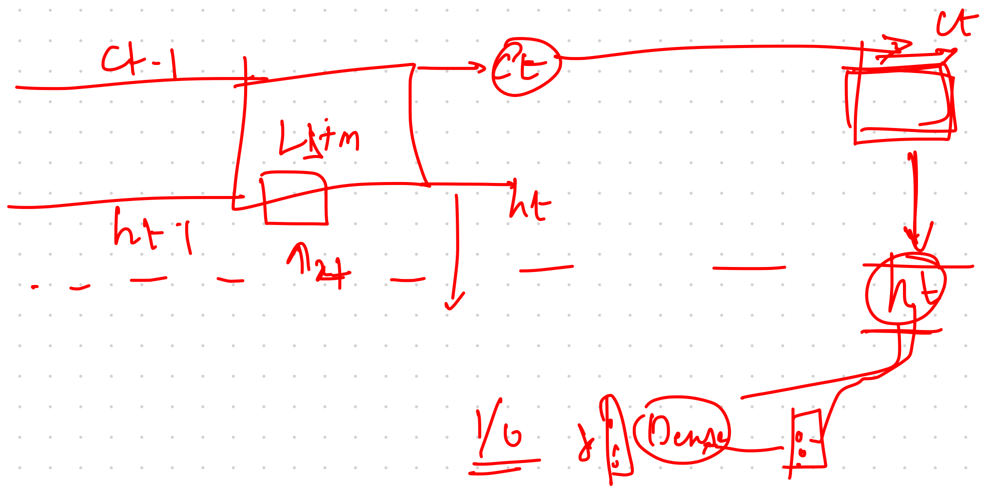
telling minicta to forget some past info which are not imp.

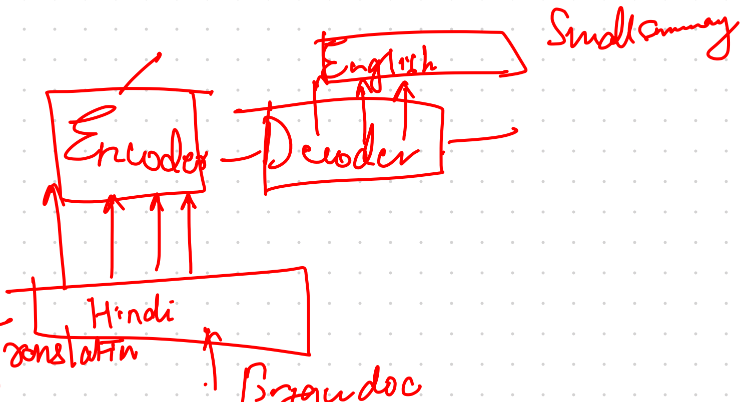
$$\boxed{i_t = \sigma(W_i [h_{t-1}, x_t] + b_i)}$$

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

$$\boxed{h_t = o_t \times \tanh(c_t)}$$

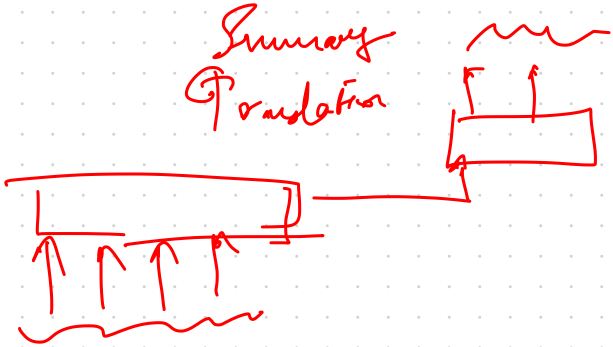


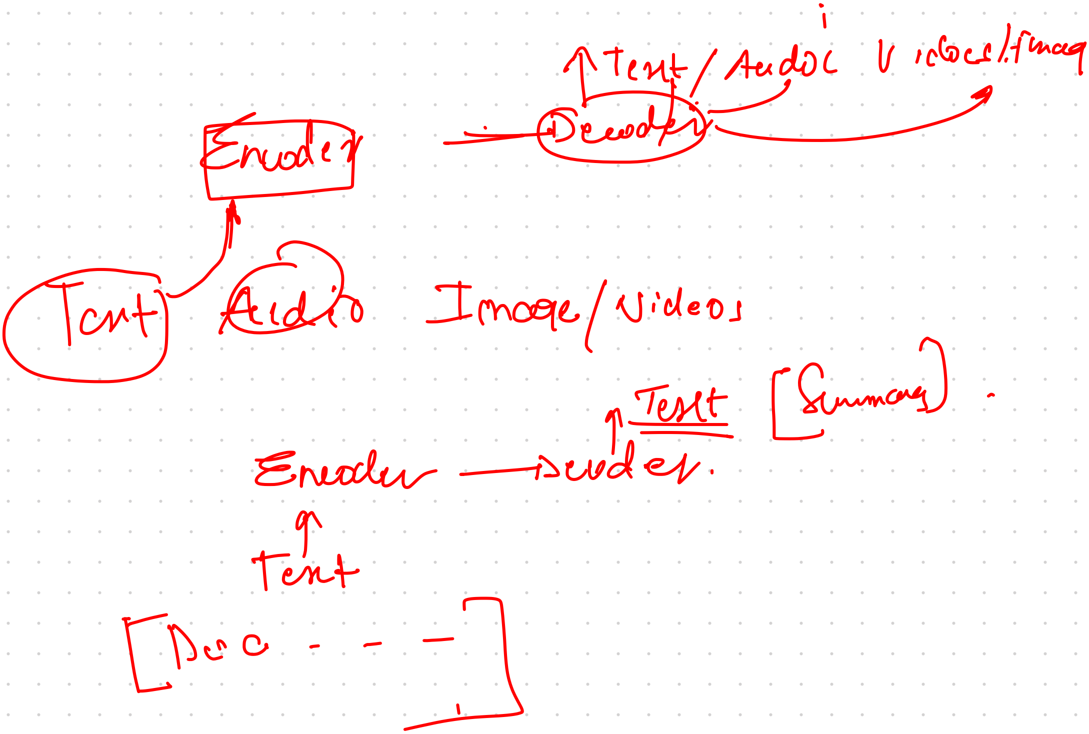




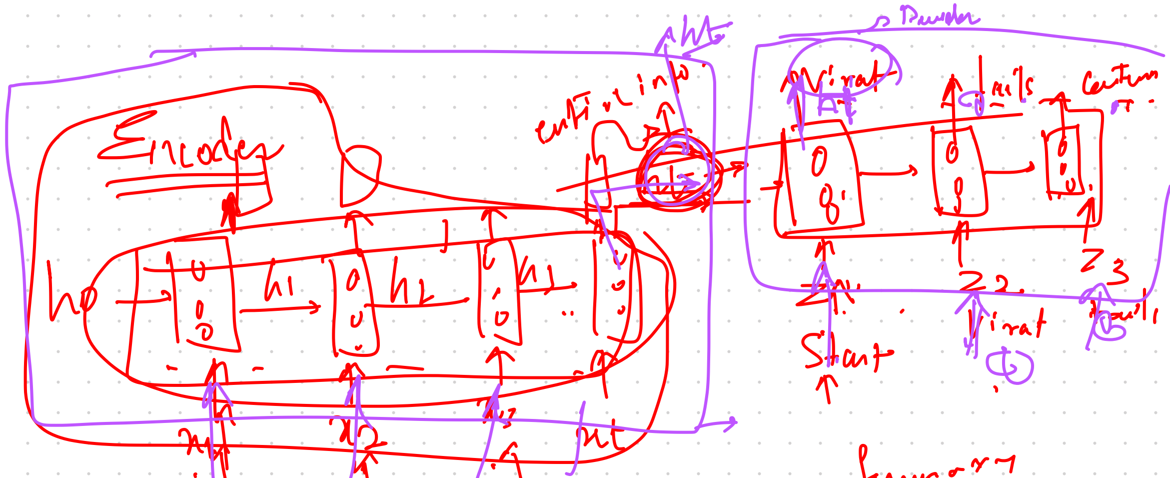
Translation
Summarization

Summary
Translation





Text	Summary.
	Inshort.



Input text -

Australia defeated India by 203 runs
India could not catch up . . . .
Virat kohli scored — — —

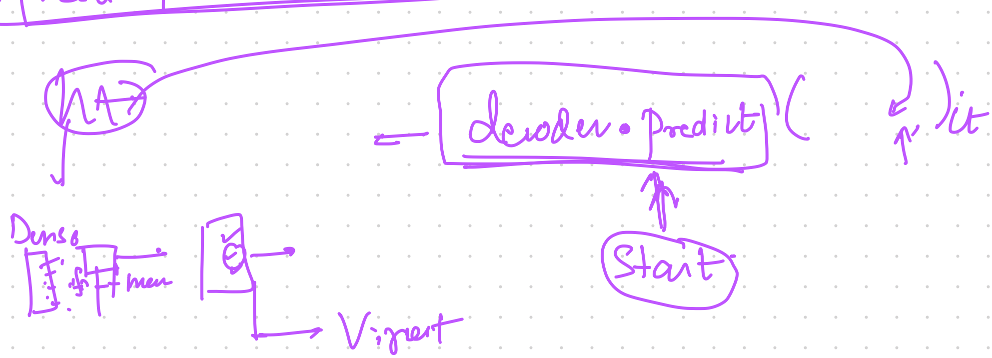
Summary

Virat fails,
centum,
India loses
wic.

Inference mode (.predict)

Entire text

h_t = Encoder.Predict (text)



ht

decoder.predict [, ht, ct)

(Start, virat, fails) ex. Ferry

Center

fails

ht, ct = Encoder.predict (text)

ht

~~virat~~

decoder.predict (Start, virate, ct)

Dense

c

virat

fails

ht, ct ← Encoder

Input - seq

ht, ct

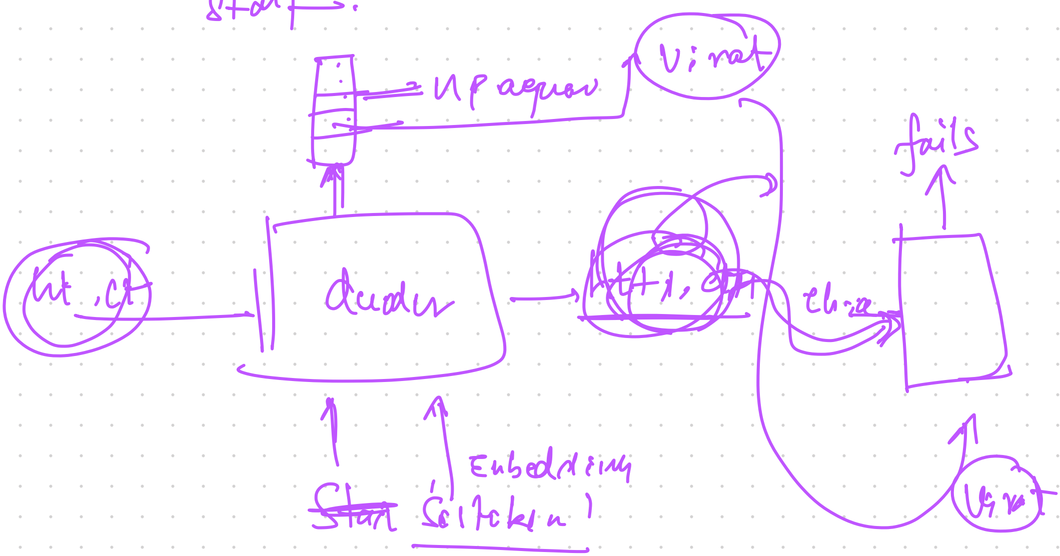
on every run ht, ct

decoder(LSTM) (dec-emb, initial_state=ht, ct)

dec-emb

decoder-input

start.



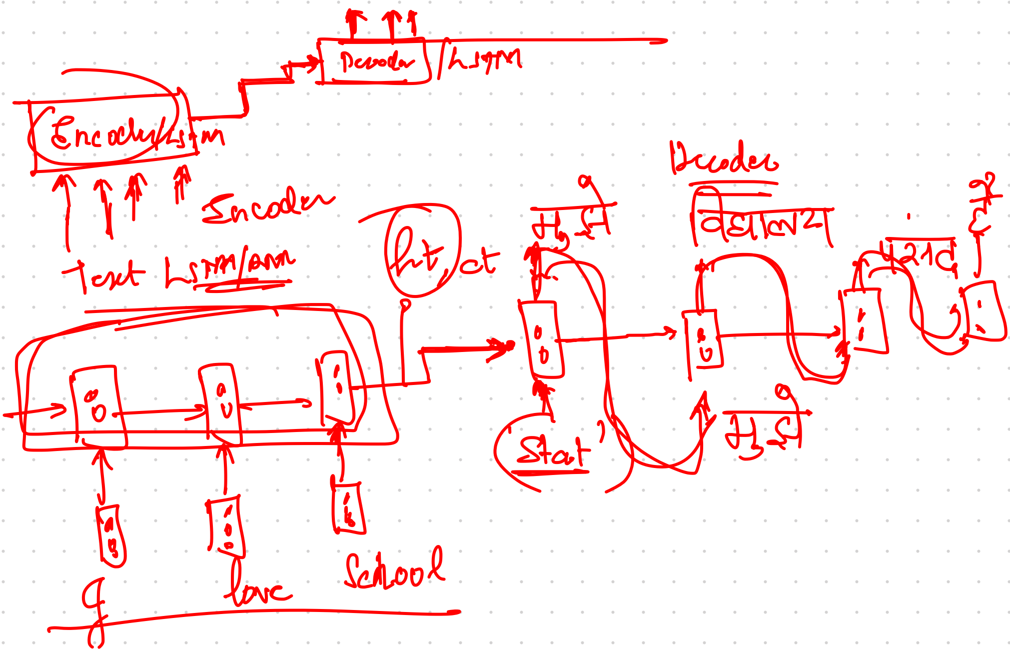
Agenda

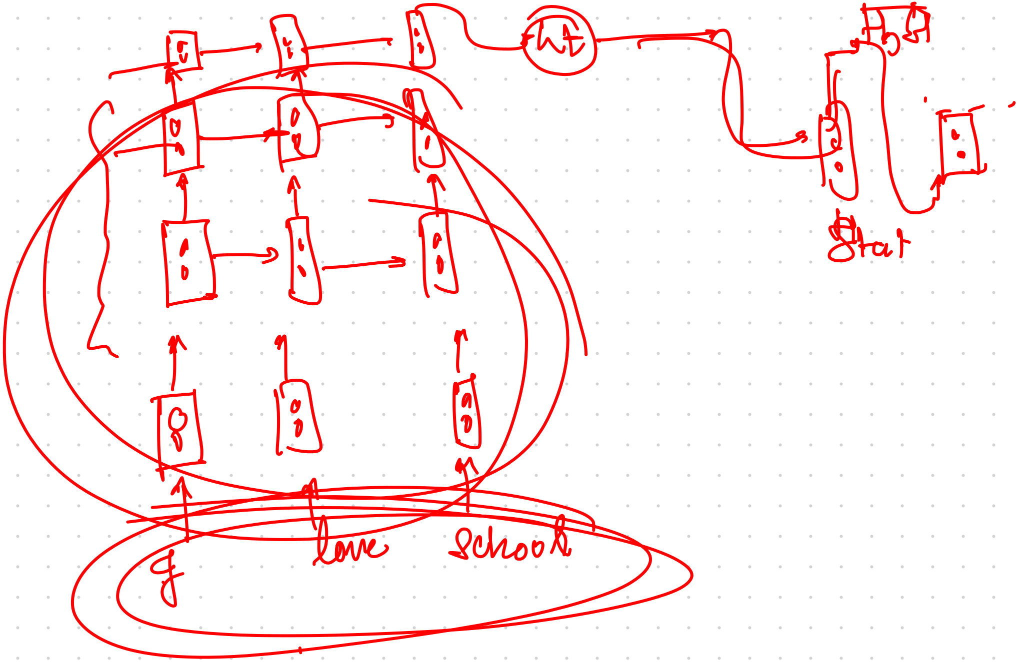
- Encoder-decoder Revisit
- NER

Text Summarization

Encoder → Decoder (Summarised).

Input text





Vocabulary

3000 words

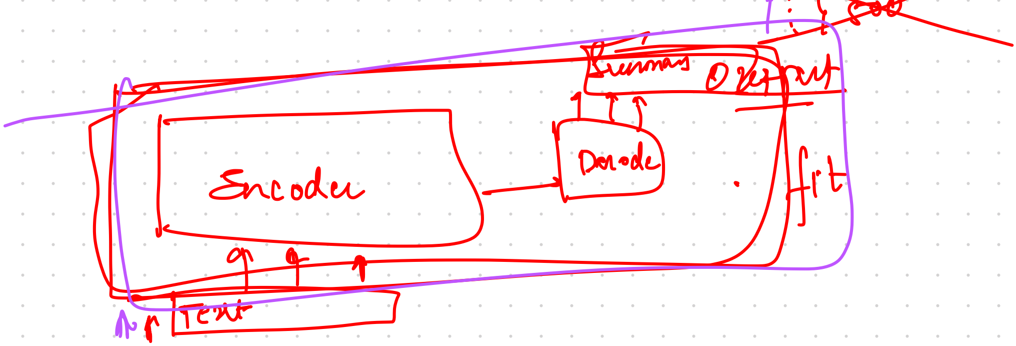
500 words

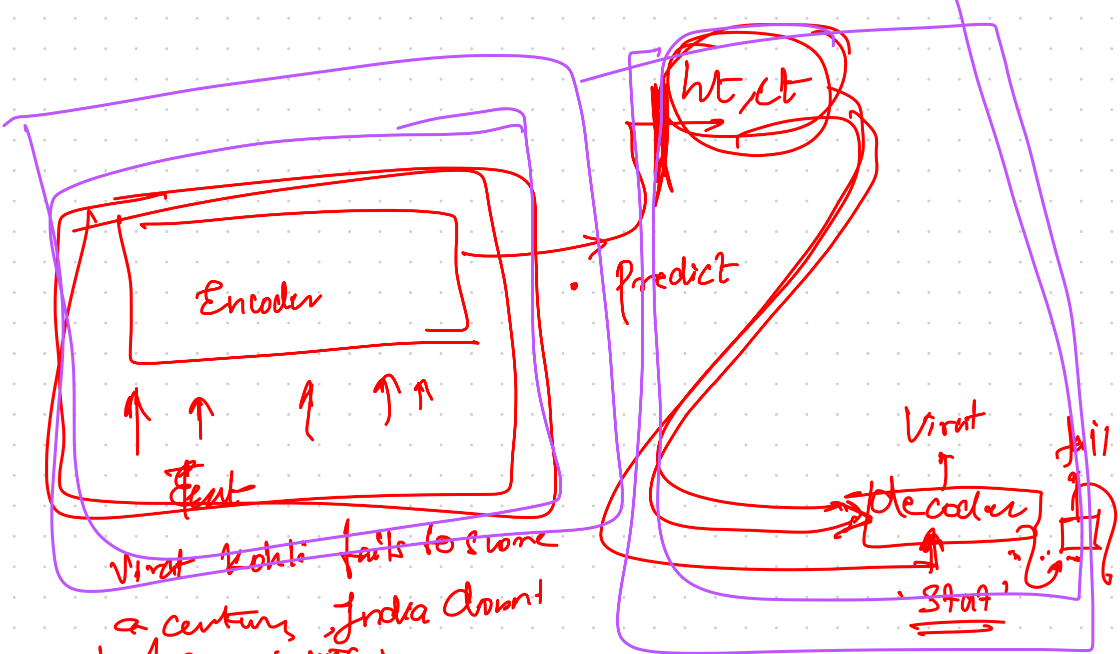
Tokenizer (3000-2500)

Tokenizer(2500 words)

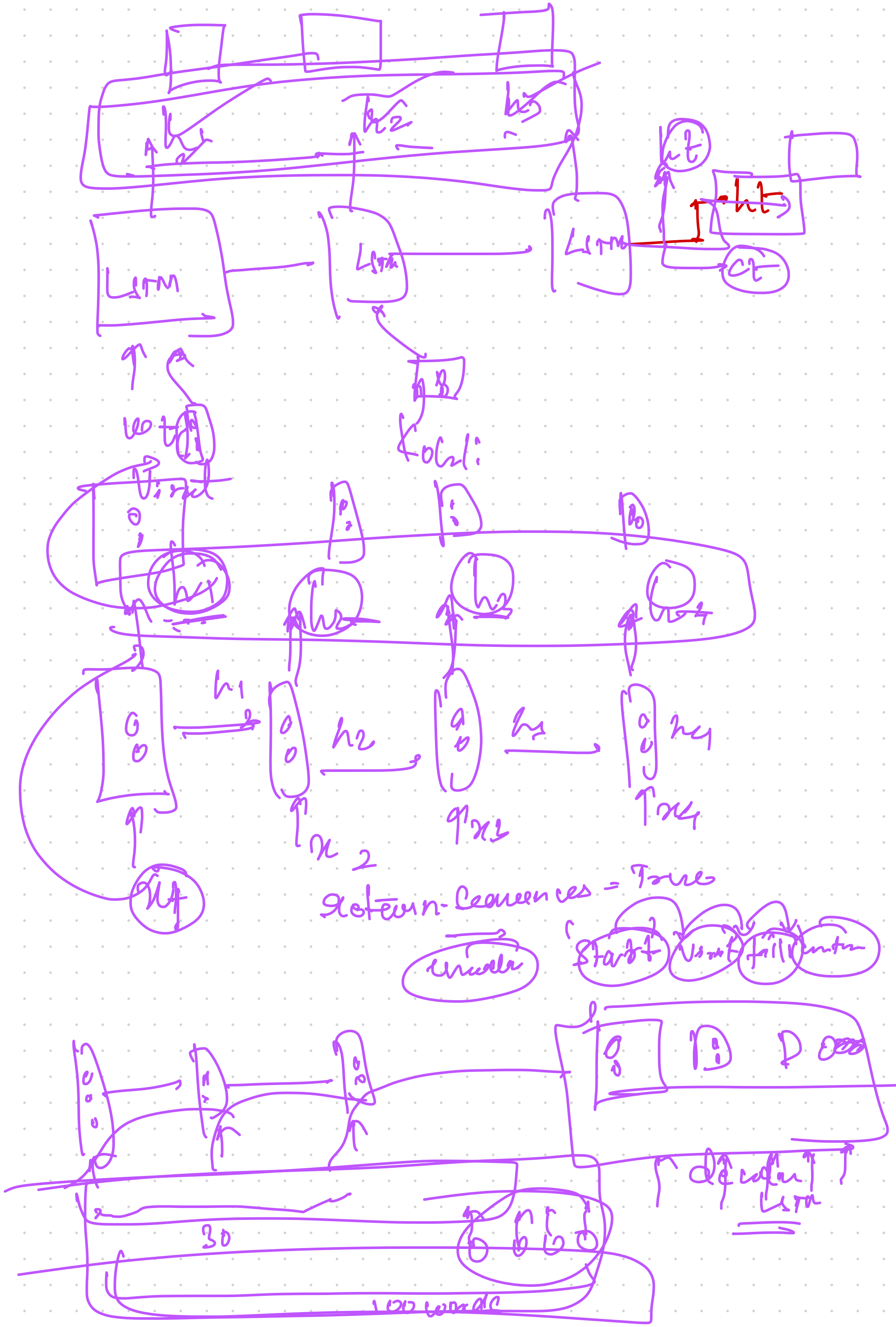
1
2
3
.
.
.
2500

~~500~~

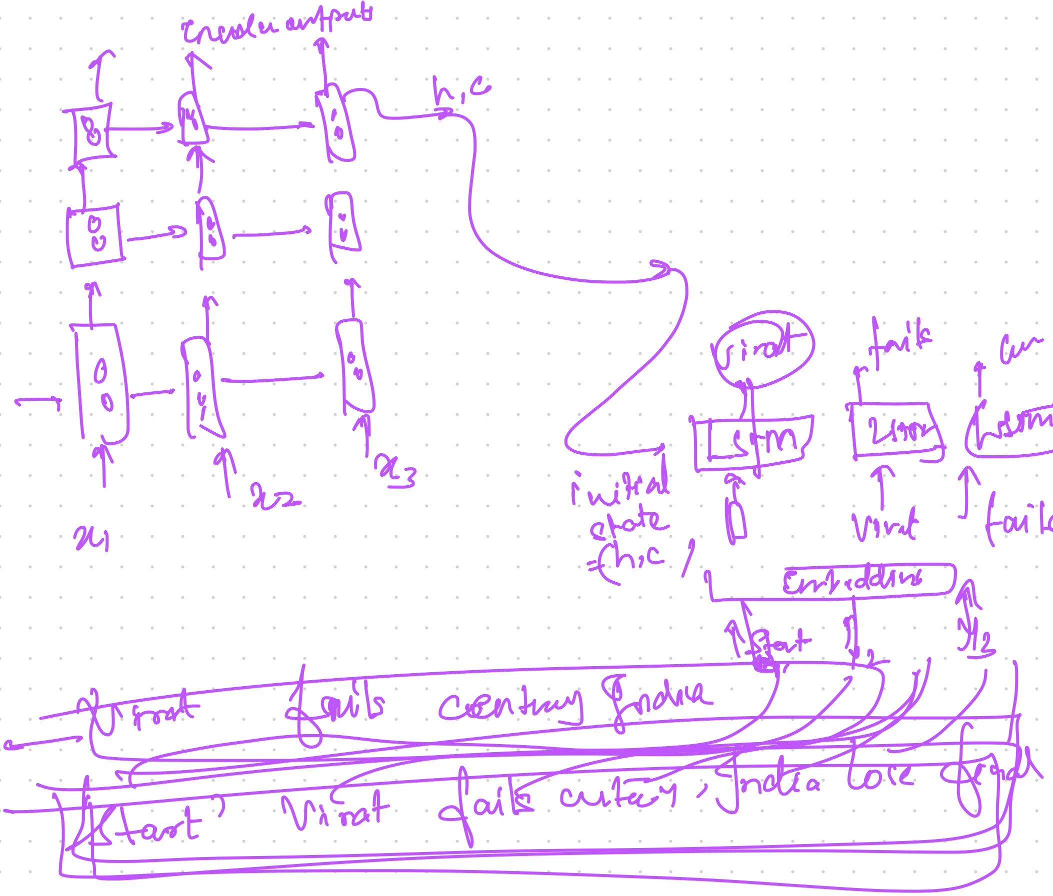




Virat kohli fails to score a century, India doesn't perform in WTC.



LSTM
LSTM
LSTM
h1
h2
ht
ct
ht
h1
h2
h3
x2
return-sequences = True
encoder
start
decoder
LSTM
30
100 words



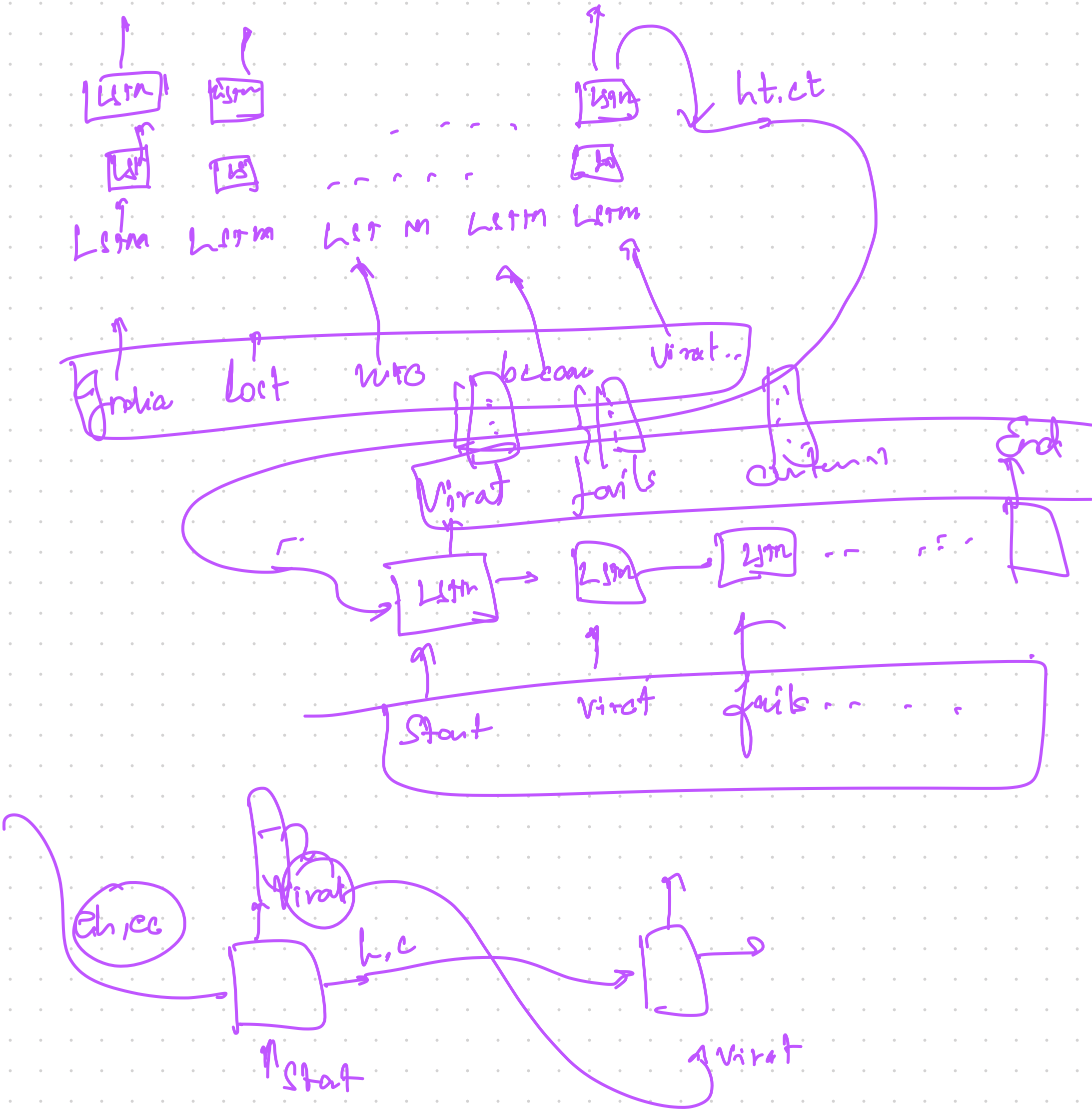
encoder outputs
h, c
x_1
x_2
x_3
initial state (h, c)
virat
fails
LSTM
virat
fails
Embedding
start
virat fails century India
start virat fails century, India lose final

Text

India lost WC because Virat failed Century

Summary

Virat fails century. India lost WC



10:28 pm

NER

NER

The patient Mr Sharma aged 25 years is suffering from high blood pressure has been prescribed clospirin

Name

age

Symptom

Drug

Drug → Side effects

Action

Hey ChatGPT, $\boxed{\text{calculate}}$ $\boxed{2x2}$

equation

Methodologies

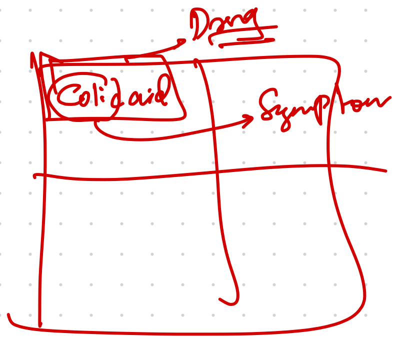
Sentence → Word Entity

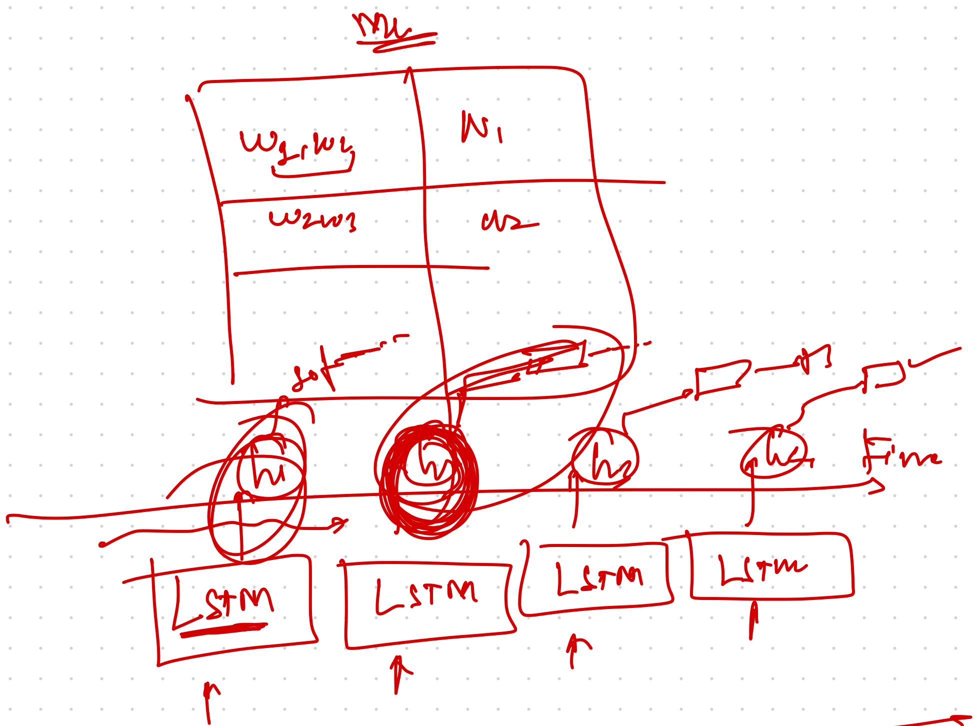
Ram, Disease, Drug..

Word	Entity
W1	✓
W2	–
⋮	–

Train

entity ← f(w)

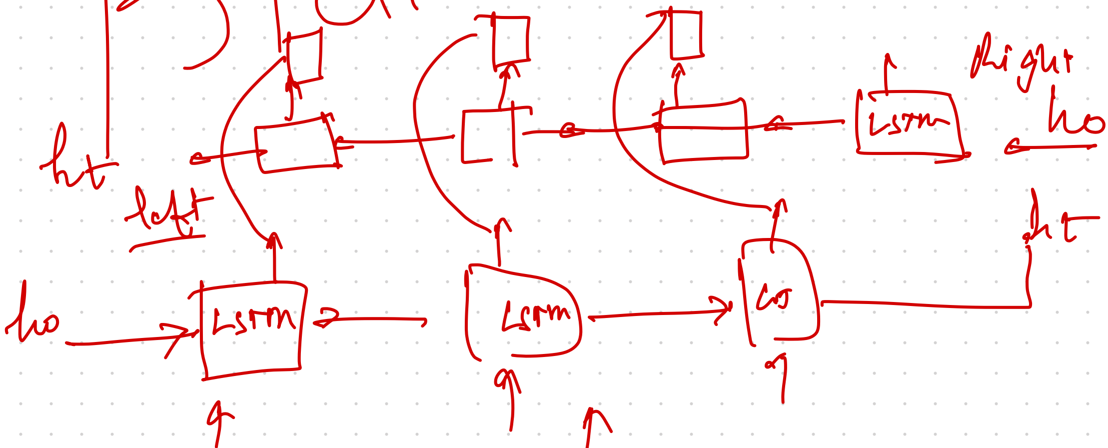


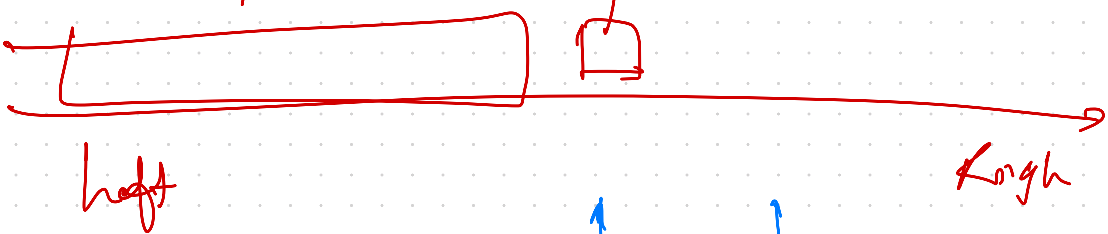


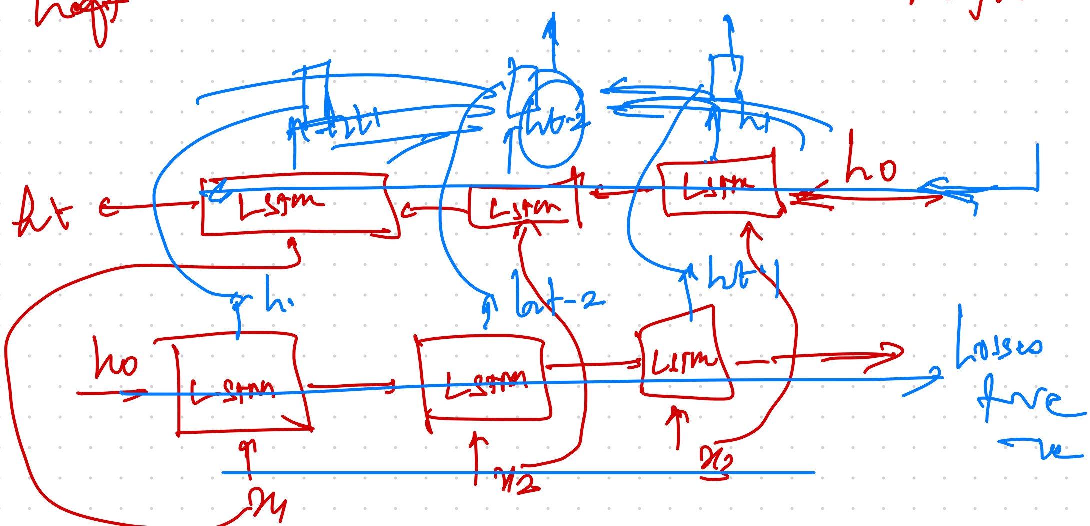
① we are predicting at a word level not at a phrase level

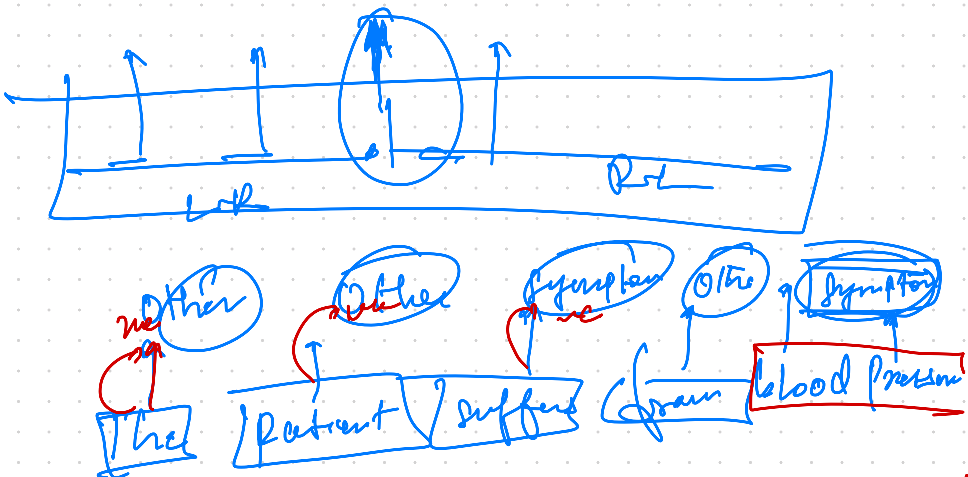
② we are predicting based on the ~~left content~~ previous words

Bidirectional LSTM









Name

symptom

The patient was RD Sharma suffers from heart arterial fibrillation

CRF → Conditional Random Fields

Comparison of some common allocations

Some common applications

Pytorch and tensorflow applications common allocations

Named entity Recognition

1) — — — — — — —

Medical entities

my stomach was bloated and I had vomiting

symptom

Camera 5/5

issue 4/5

VFM 3/5

Structure : Q/A

S22 ultra is a great phone but has heating issues

issues

issues → heating, Battery

good points → camera

Price → high, VFM

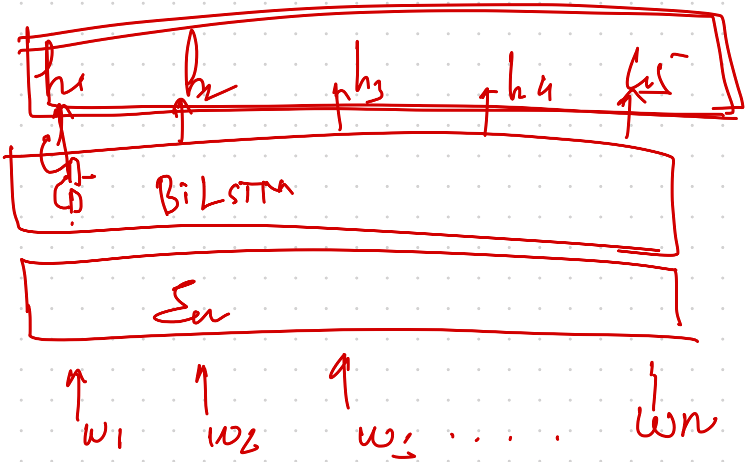
10,000

NER

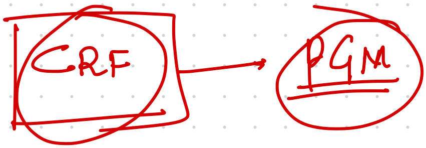
① can learn relationship bidirectionally.

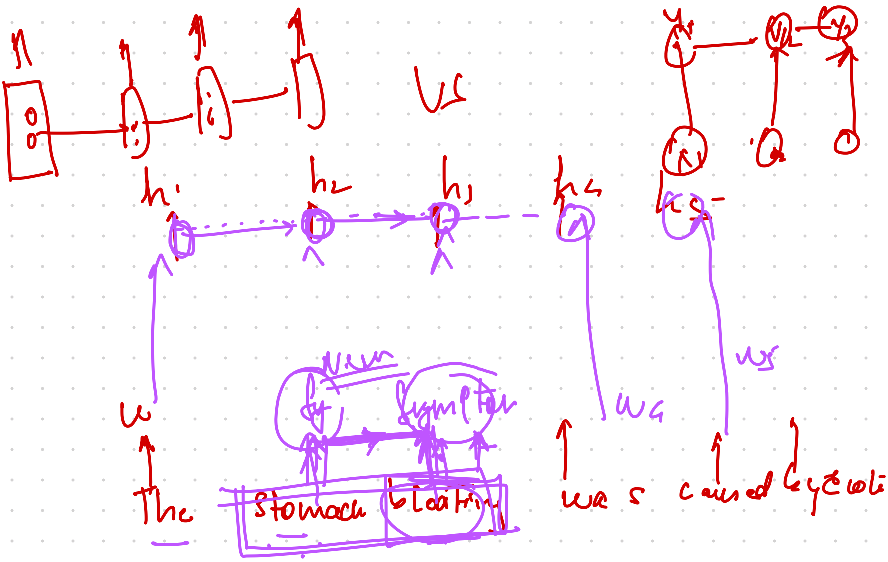
L → R

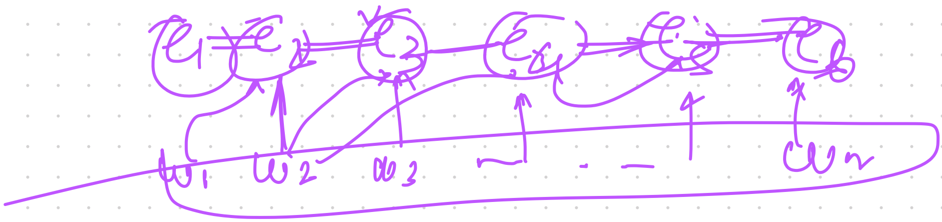
R ← R → BiLSTM

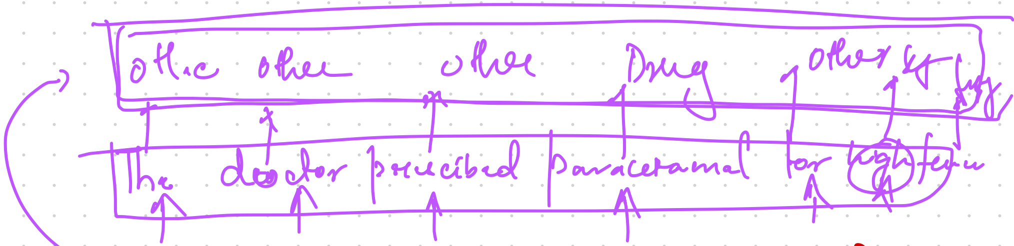

h1 h2 h3 h4 h5
BiLSTM
w1 w2 w3 . . . . wn

1


CRF
PGM


vs
h1 h2 h3 h4 h5
w4
w5
symptom
The stomach bloating was caused by e.coli





$P(y/x)$

$F_1 \rightarrow LR,\ F_2$

$$P_\theta(y/x) = \frac{e^{\left(\sum_j w_j F_j(x,y)\right)}}{\sum_{y'} e^{\left(\sum_j w_j F_j(x,y)\right)}}$$

F_1 → LR
F_2 → DT
F_3 Rule Based
F_4 → ANN

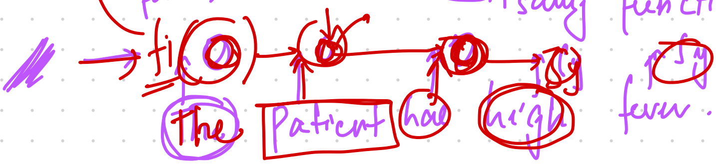
Sequence of words
Sequence of tags

$$F_j(x,y) = \sum_{i=1} f_j(y_{i-1}, y_i, x, i)$$

is the word a drug or not

F_j(the pat... high fever, E, ..., 0, Sy, ...) is any function $f_j(i)$

$f_i(i)$



$f_j(0, 0, \text{Patient})$

Sy Sy Drug

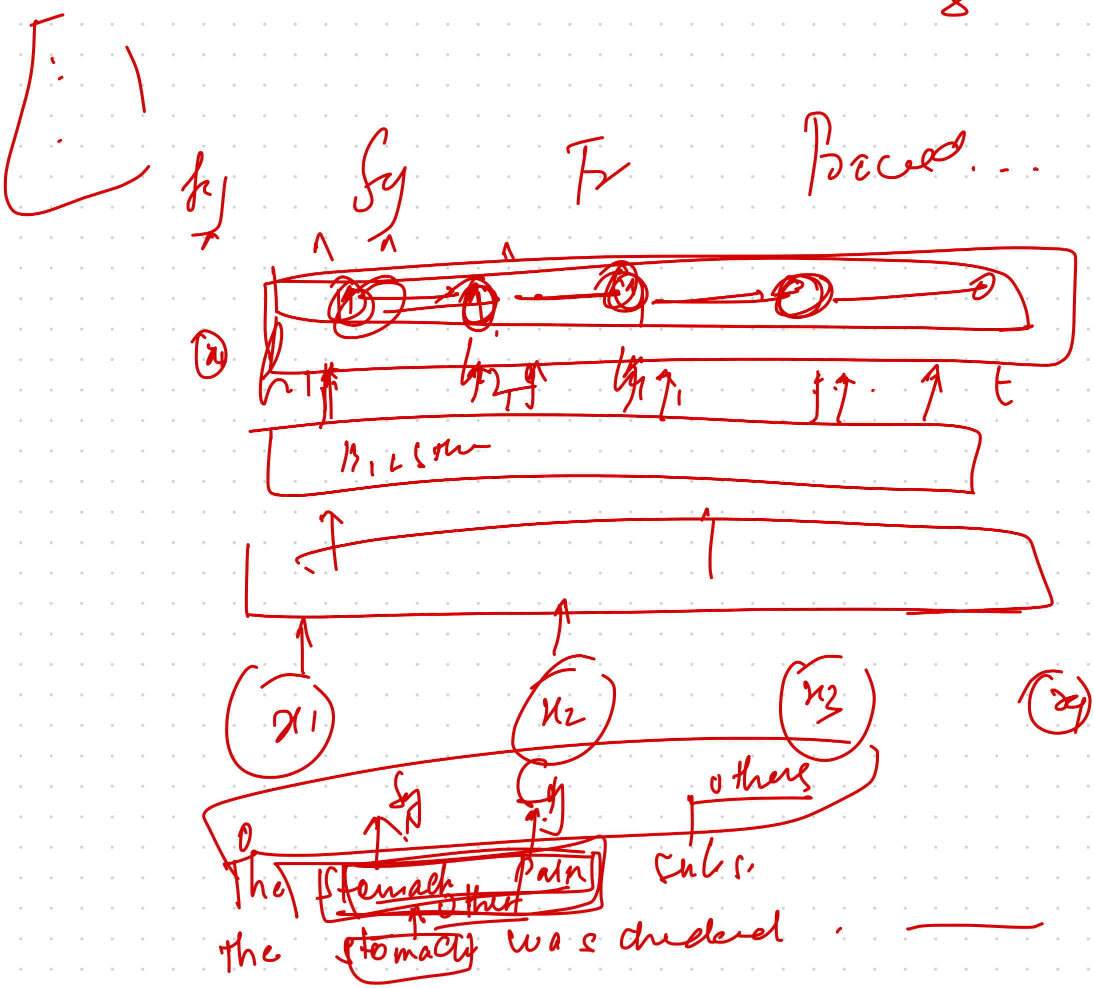
High fever ~~...~~ paracetamol.

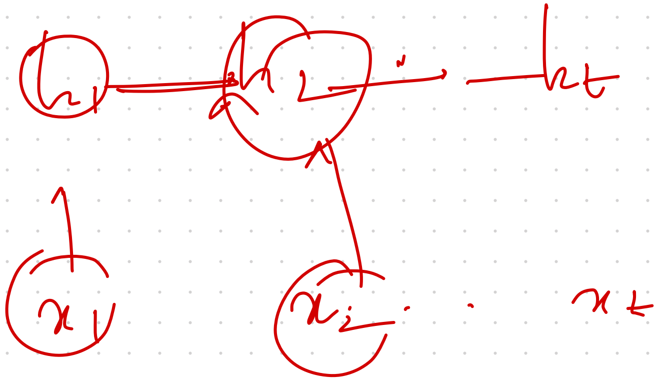
$$P(\boxed{Sy, Sy, Drug} \,/\, \text{High fever paracetamol})$$

Sy Drugs other

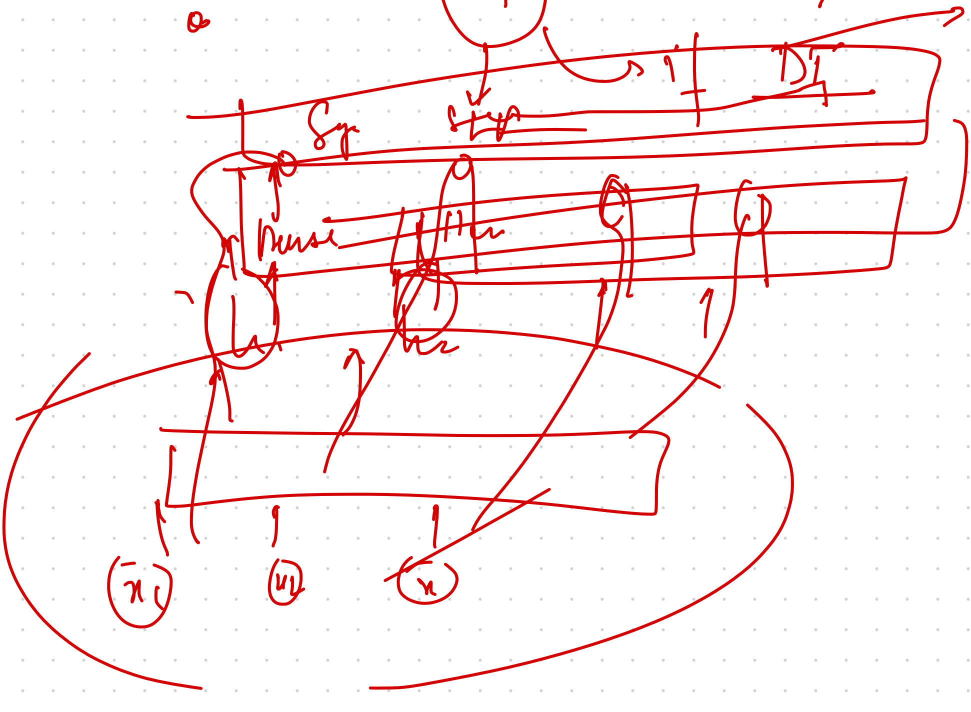
$$\frac{e^{\ldots}}{e(\text{Sy Drug other} / \text{High fever Para}) + e(\text{Sy Other other}) + e(\ldots\ \text{Sy}) + e(\text{Sy Drug})}$$

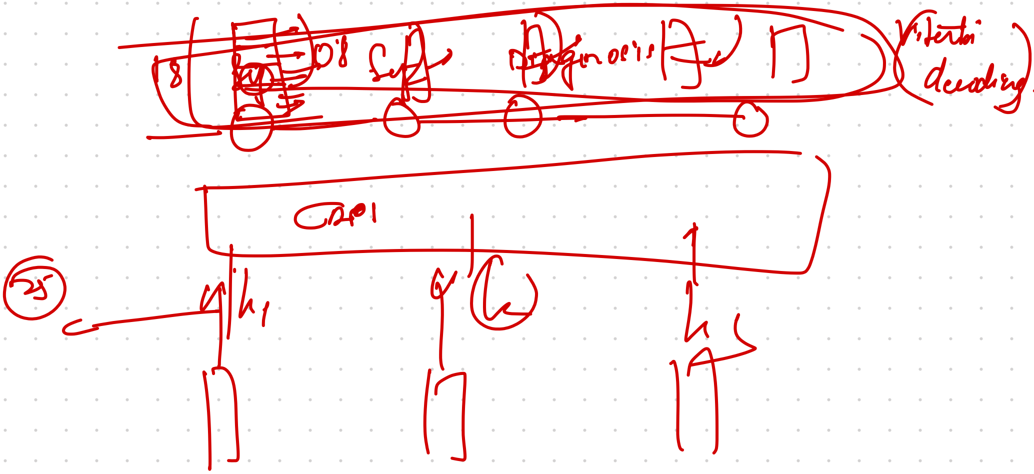
8





$$P_{\theta}(y/x) = \boxed{f_i}\,(y_i, y_{i-1}, x_i)$$





(model. Predict)
[e, sy, sy, O, O, Dxsy]
The stomach pain cubricated with ROLO

Attention

Google → 2017/2018 ⟶ Attention → Transformer

CNN

ChatGPT

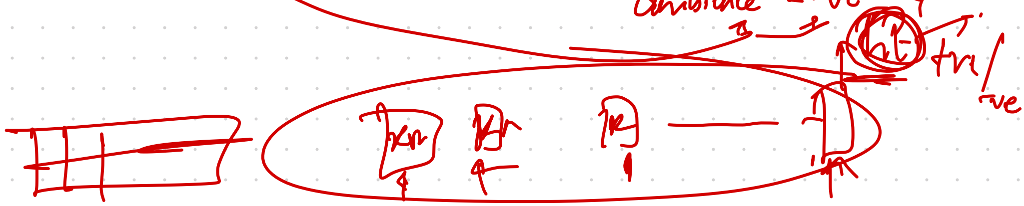
Abstract based Sentiment

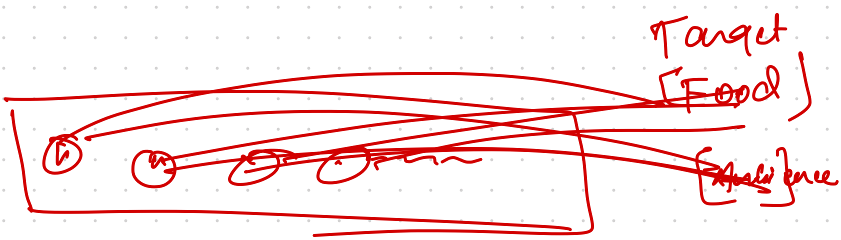
h_1 h_2 . . .

"I liked the food but the ambience was not great" h_{10}

{ "food" : +ve

"ambience" = -ve }





Food target Ambience

+ve

[The food was great but the ambience was bad]

ht

+ve

-ve

ht

I am going to school

स्कूल जा रहा हूँ

attention mechanism

Enhances our learnner learning

ht

I am going to school

start

स्कूल जा रहा हूँ

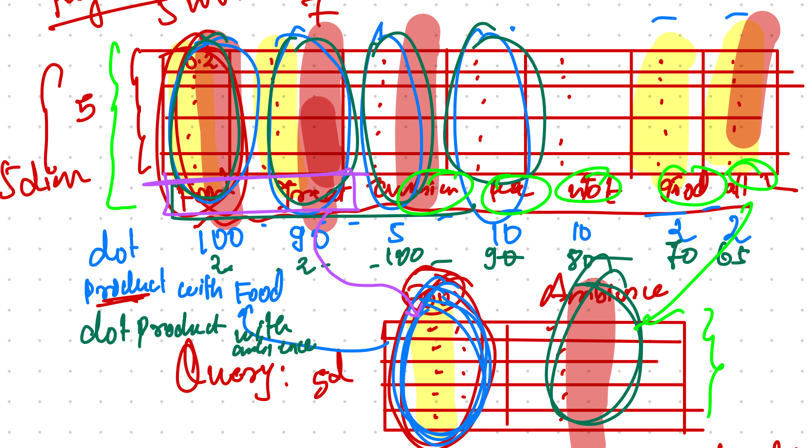
Attention: It is good to have a relationship b/w the output and each of the words

Food +ve
Amb, -ve

Absa: The food was great ambience was not

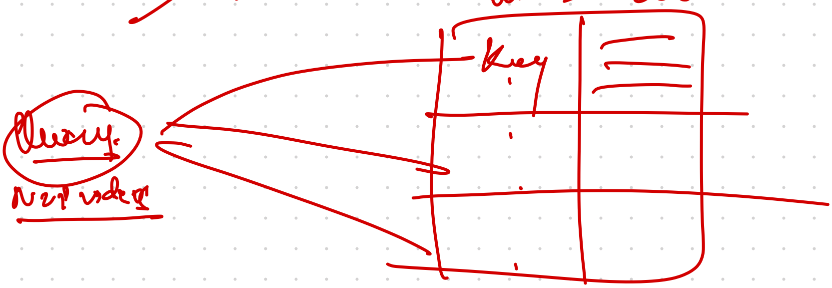
Query, Key, Value

Key matrix
5 words



why are we taking the dot product b/w the words / target

Search engines



Search engine from

competitors