Natural Language Processing

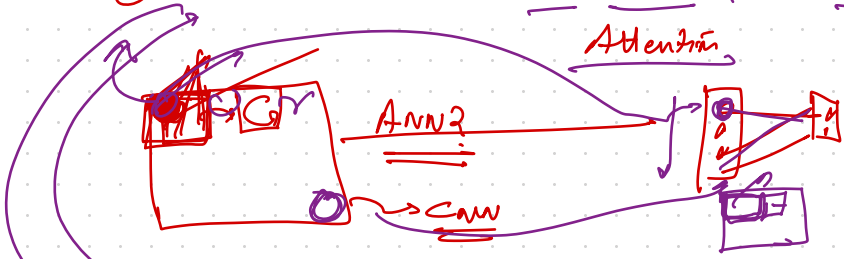
① NLP Problems we can solve.

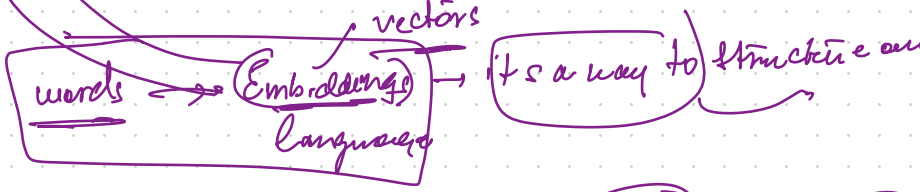
- How do we structure words / sentence

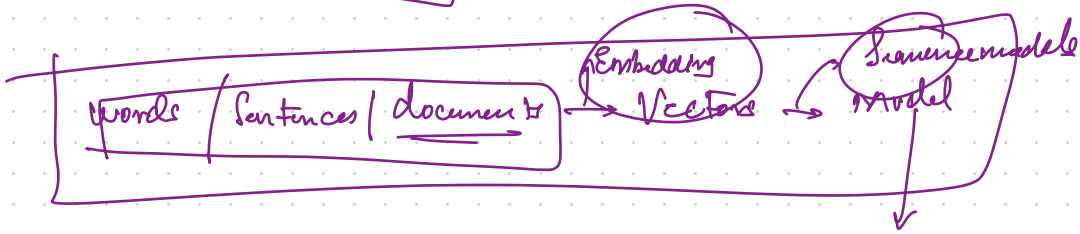
↓

Feature engineering

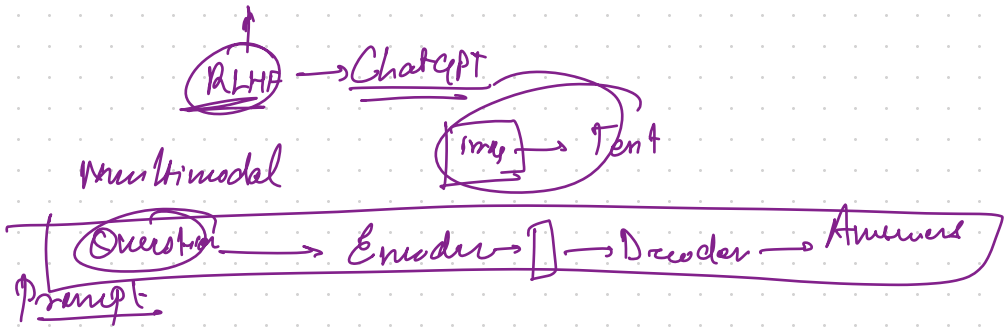
- Sequence models → RNN, LSTM, GRU, Transformer







Large language models



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

- Hinglish - - - - - - - - - - - - -
- Romanized words
- Tujhe main dekh lunga → मैं देख लूंगा तुझे → English
  - Pyinaic translation
  - Phonetics Sounds
- Structured

Use Cases:

Translation
gmail auto type → auto fill
Search engines

Sentiment analysis

Gmail | Spam → Spam, → Ham

Chatbots
Text classification, Sum
NER, information

$y = f(x)$ — f: sentiment, x: input, y: output — Language model.

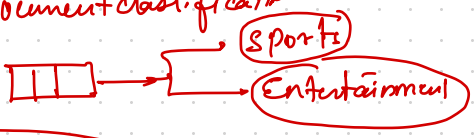
Language model

Sentences are very unclear — Social media, Human Conversations

Preprocessing techniques:

Document classification

Pattern



Special characters @ http....

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

Ticket classifier

Ticket

Sentiment classification

Regex → a precise logic of defining certain patterns using certain rules.

"I am very hungry, I will order pizzas and give it 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 → Todays temp 25°C
→ I am an ------

What's the temp tomorrow

मैं घर जा रहा हूँ → I am going home

Stemming / Lemmatization

Embedding
words → vectors → model

Embedding

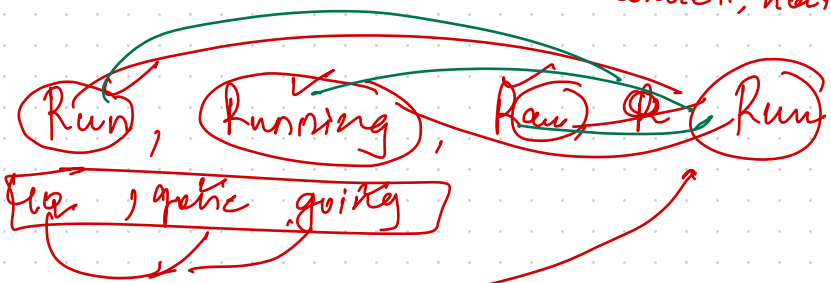
Frequency based
Count of the words in the sentence

Prediction based.
Vectors generated through NN.

Context?

I am going home., home is where heart is

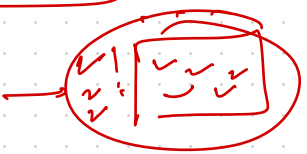
I: 1, am: 1, going: 1 home 2, is: 2
where 1., heart: 1



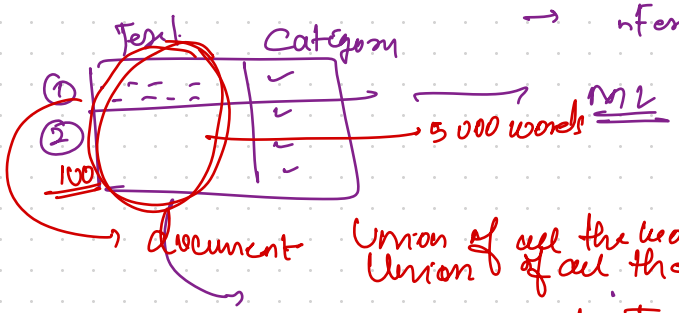
Go

helpful

10:33 → 10:40pm

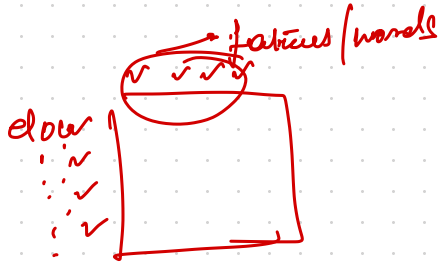


How to represent sentences / documents in a vectorised form using frequency embeddings.

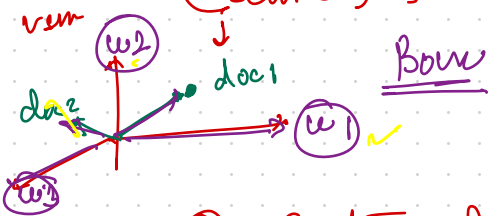


Bag of words model

Document classification



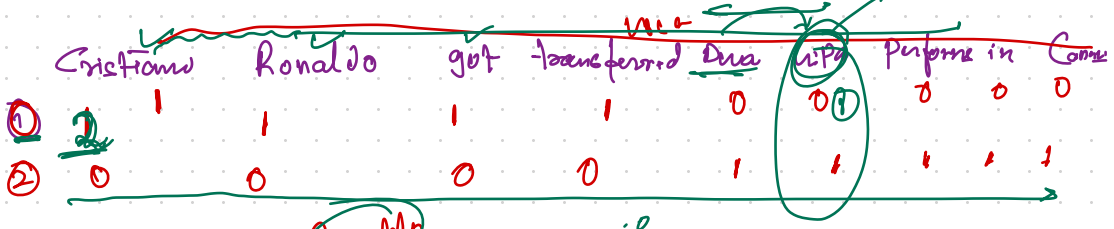
(Document) → outcome

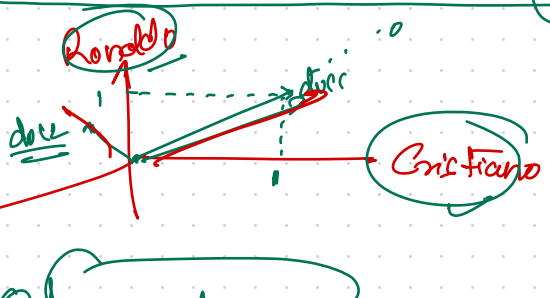


BOW

① Cristiano Ronaldo got transferred to Sports (Cristiano is upset)

② Pwc hopes performs in Cannes.



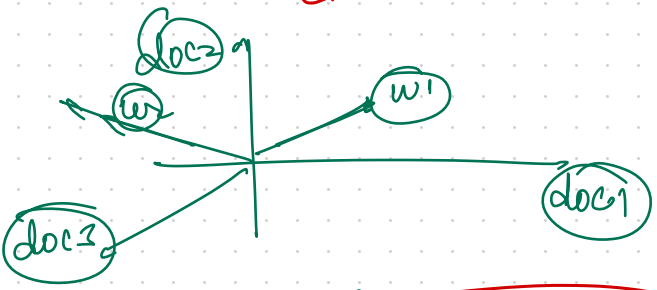


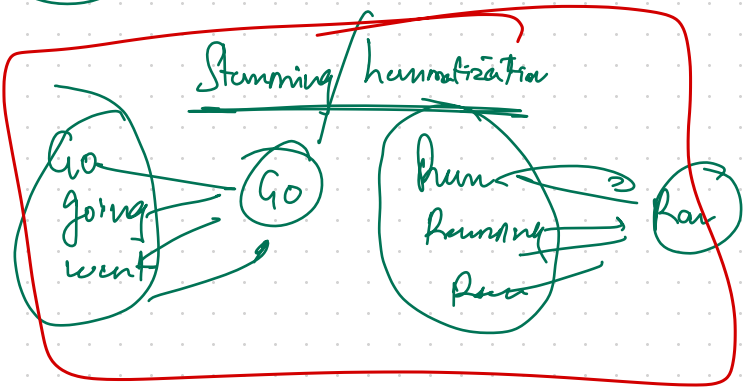
Problem: ① document dimensions

① does not maintain content

→ order not maintained

By definition you and all words are independent





Stemming & Lemmatization

Lemmatization

Porter Stemmer

Running → Run (verb)

Ran

Speeding ✓

Speedfc ✓

Caressed ✓

Going → go

gone

Cry → cri

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

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

Sue my ac is running

Sue my ac is run

Dependency Par

my (run) ions awesome

pos

not

"I am great at public speaking"

Stemming & lemmatization

(word)

Sentence → words

word tokenizer

Sentence tokenisation

document → Sentence

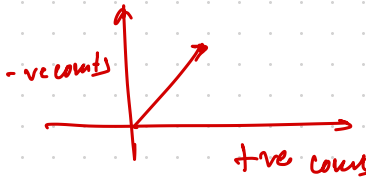
.split(' ')

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

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

title → word tokenisation (' ')

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



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

tweet: The game

tweet (110, 110)

freqs

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

tweet: game is awesome

game awesome

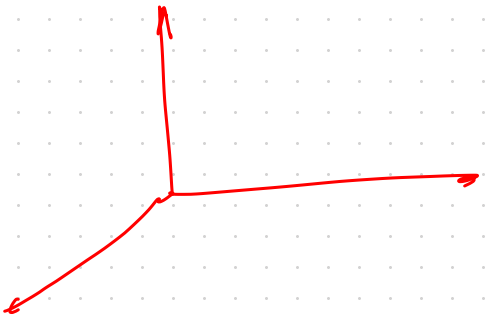
pos = freqs[(game, 1)] = 40 + freqs[(awesome, 1)] → 90

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

pos = 130

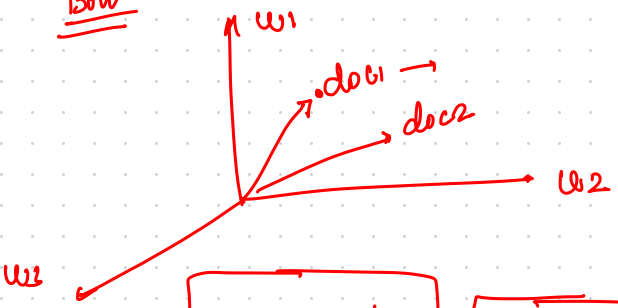
neg = 70

tweet = (1, 130, 70)



NLP → words/docs into vector space models

BoW



Sentence 1 / doc1 : I am happy
Sentence 2 / doc2 : I am sad.

→ Vocabulary

Union of all docs

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

Count vectorizer

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

① doesn't maintain any order

doc1: I am happy today.

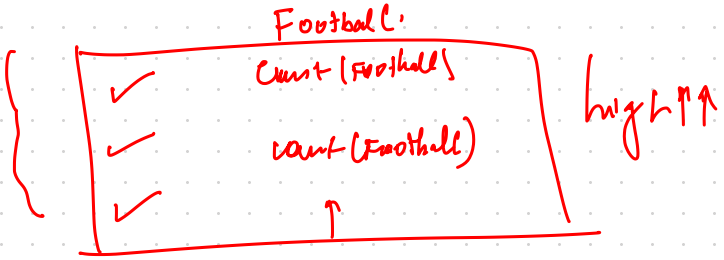
today happy I am

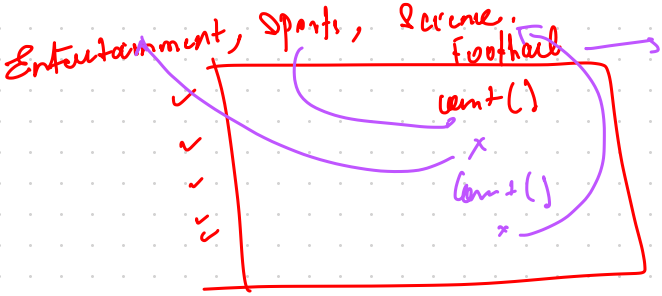
doc1	1	1	1	1	1

Frequency based embeddings don't capture the context

Term Frequency Inverse Document Frequency.

Football





→ Football important word.

tf (Term frequency) x idf

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

w1 w2 w3 ... wn

N = 100

doc1
doc2
doc3
⋮

total frequency of all the tokens in doc1

number of documents in which w2 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

1/3

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

	Ronaldo	Messi	lovely	Rich	famous
doc1	2/3	0			
doc2		0			
doc3		1/1 x log(4/1)			

$\log(4/2)$

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

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

$2/4 \times \text{Log}\left(\frac{4}{2}\right)$ $\quad \text{Log}(1) \neq 0$

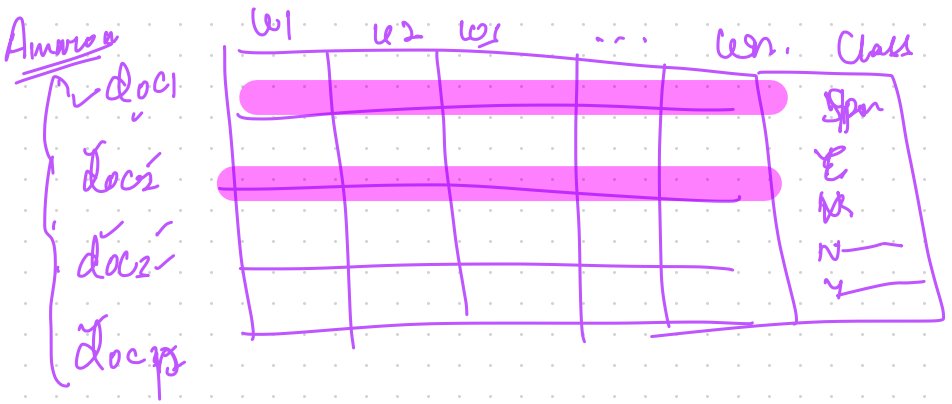
$\log(N/n)$ $\quad n \approx N$, $n \neq N$

TF-IDF (t_i) doc$_j$

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

$N \rightarrow$ total number of docs in corpus.

$n \rightarrow$ total number of docs in which term occurs.



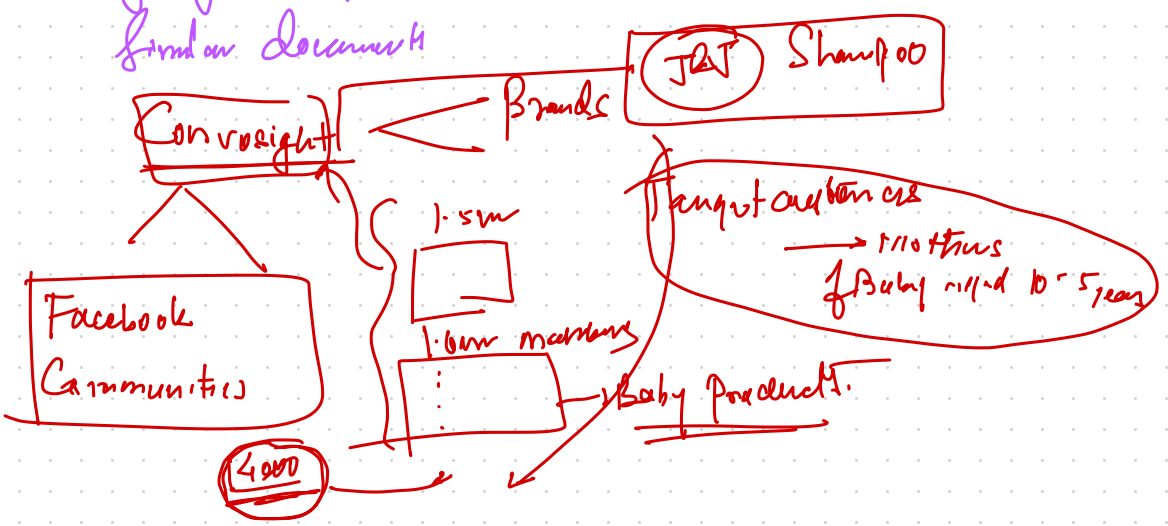
Find out similar documents

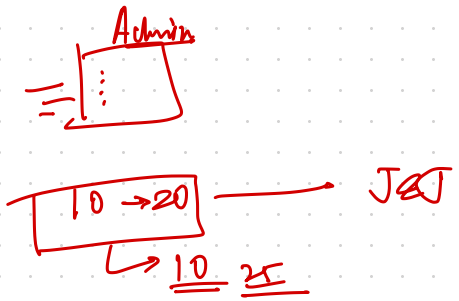
doc1 (Product)

doc3 (Product)

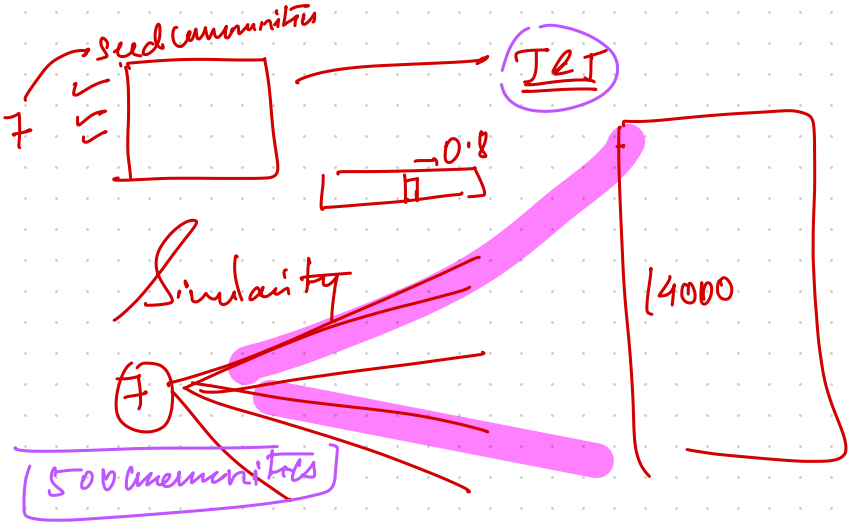
addresses

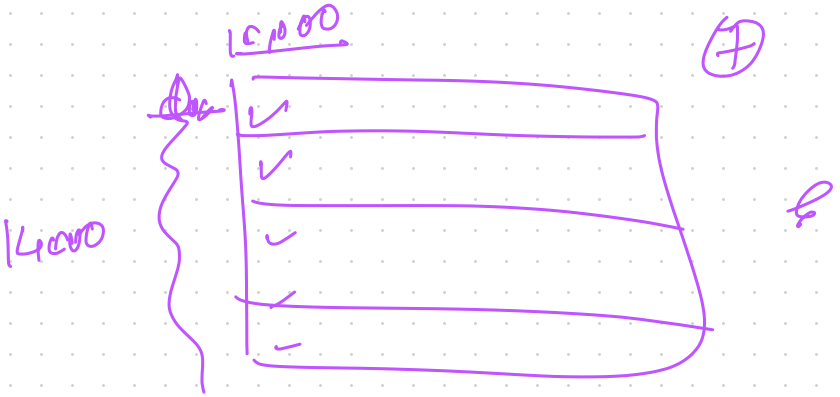
Using document based embedding & confirm similar documents

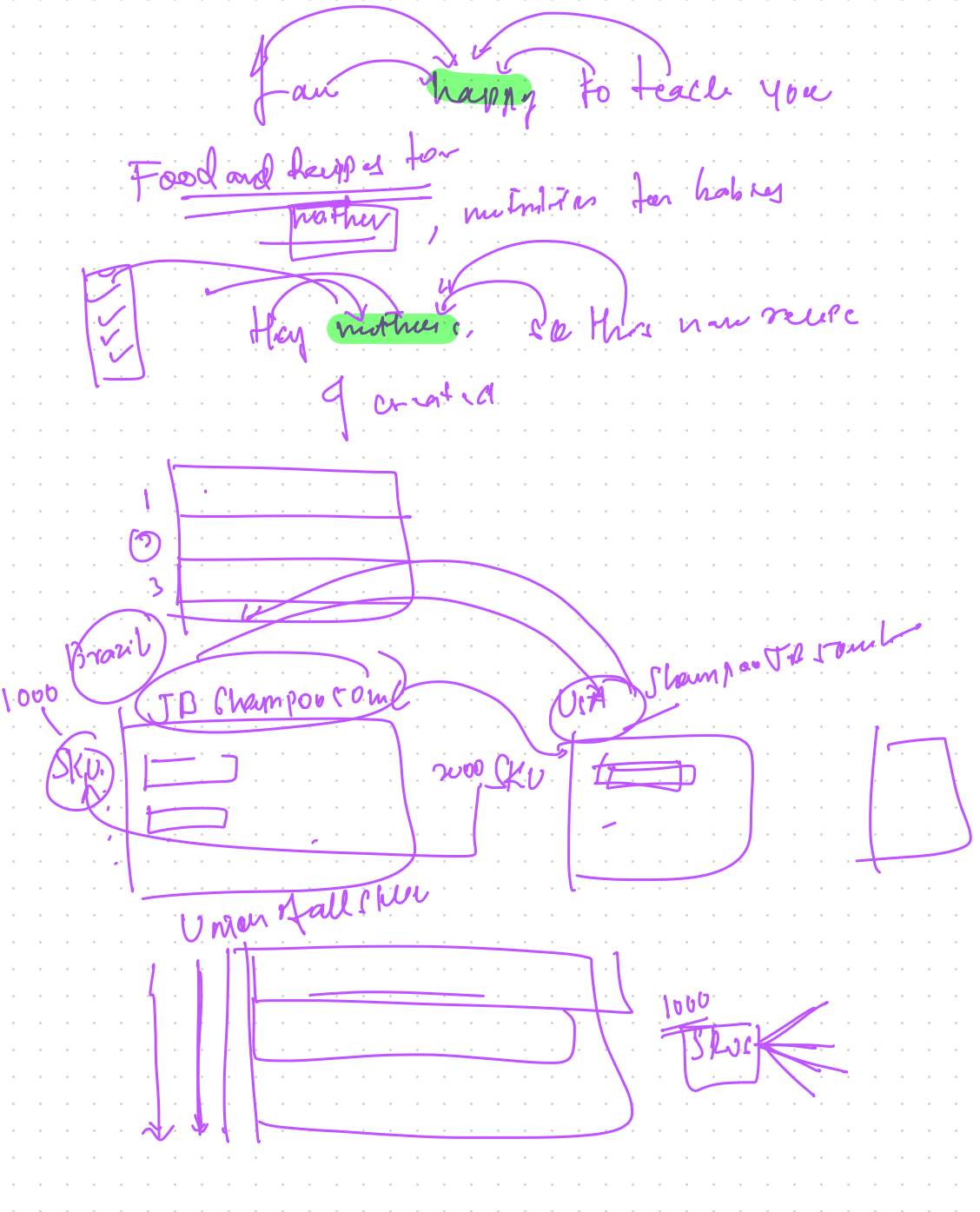


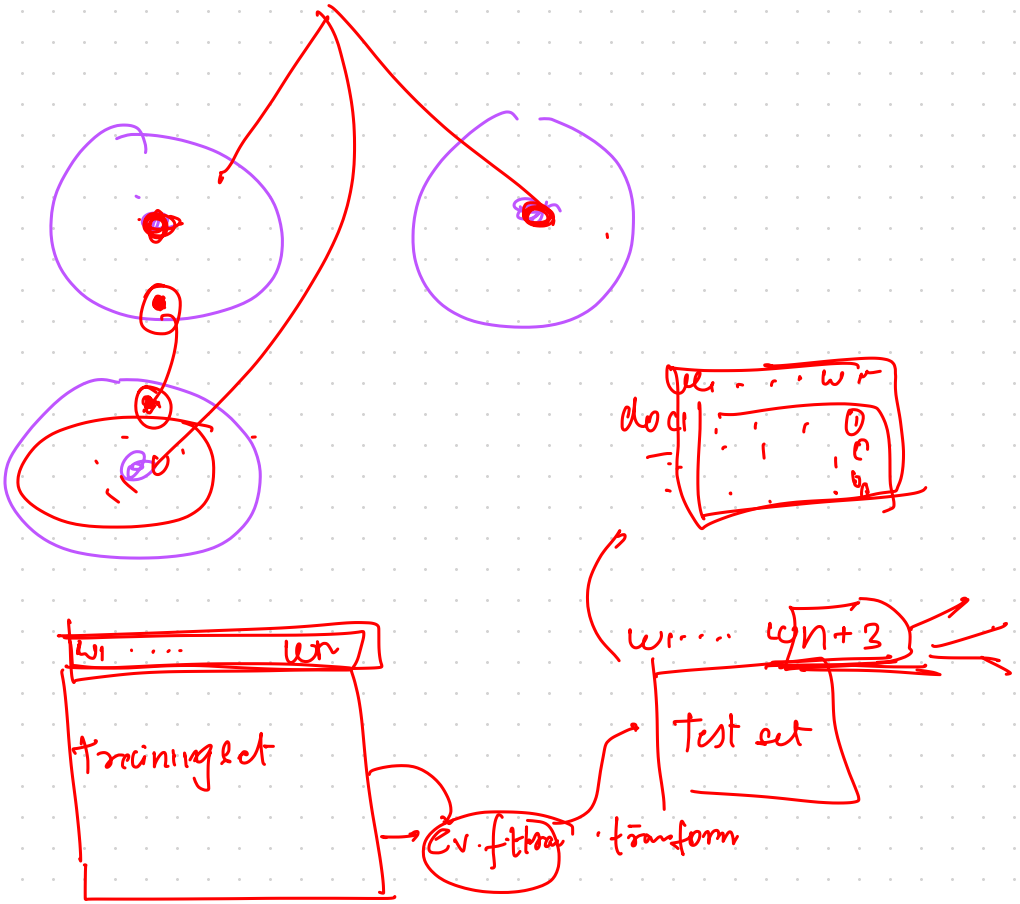


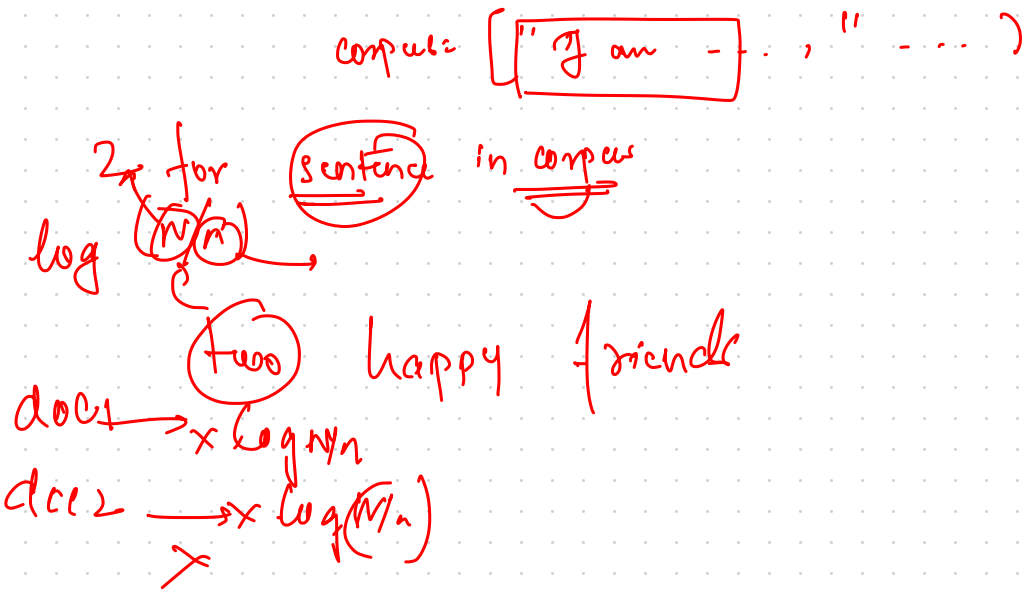
14000 comm
↳ 1.6 billion communications

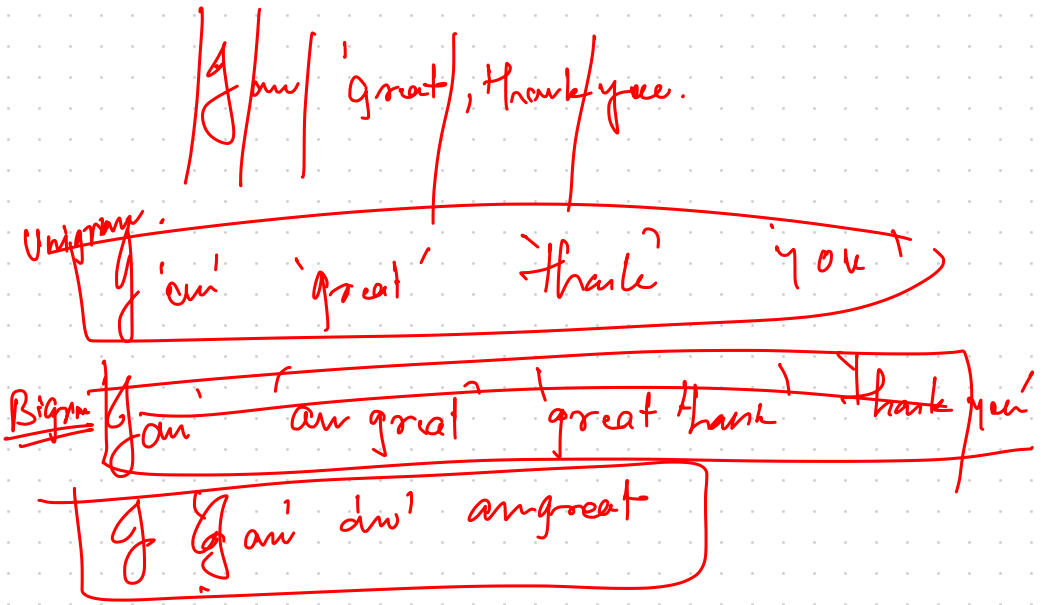





I am happy to teach you
Food and drinks for
mother, nutrition for babies
Hey mothers, See this new recipe
I created
1
2
3
Brazil
1000
SKU
JD Shampoo roml
2000 SKU
UK
Shampoo UK roml
Union all SKU
1000
SKUs

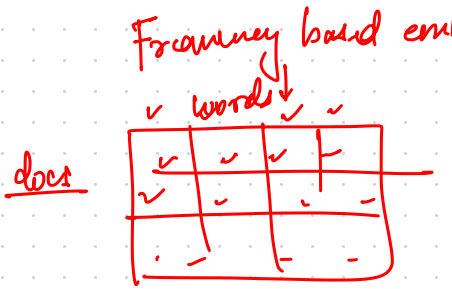

w1 . . . wn
Training set
ev. fit
transform
w1 . . . wn+3
Test set


corpus: ["I am ...", "...
2. for sentence in corpus
log (N/n)
foo
happy friends
doc1 → x log N/n
doc2 → x log(N/n)


I am great, thank you.
Unigram:
I
am
great
thank
you
Bigram:
I am
am great
great thank
thank you
I I am
I am am great

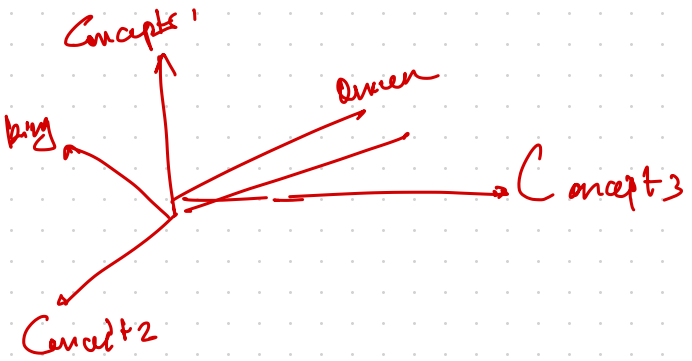
Embedding → word2vec

Frequency based embedding

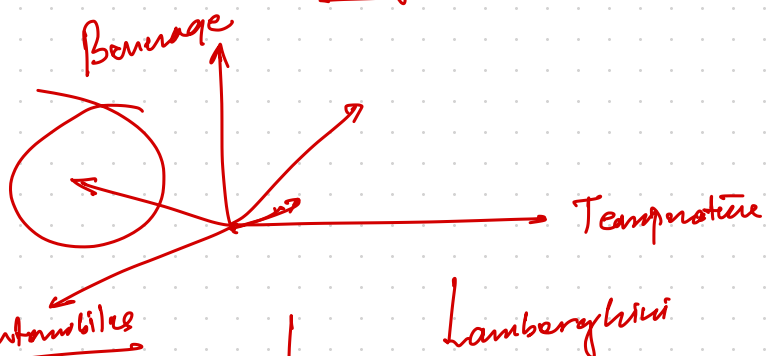


Prediction based Embedding

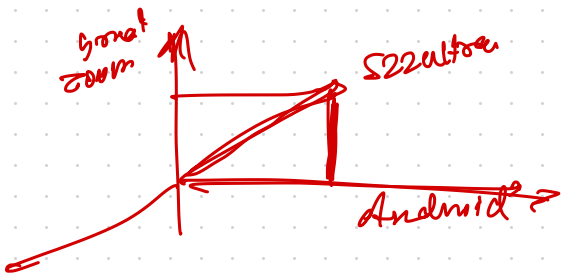
Prediction based curbidding



HotCoffee



Lamborghini



Embeddings are dense representation of concepts

↓

Abstract non-linear combinations of multiple words.

Embeddings (unsupervised)

Word2vec an model

Embeddings SVD

Embedding from language models → (Supervised way)

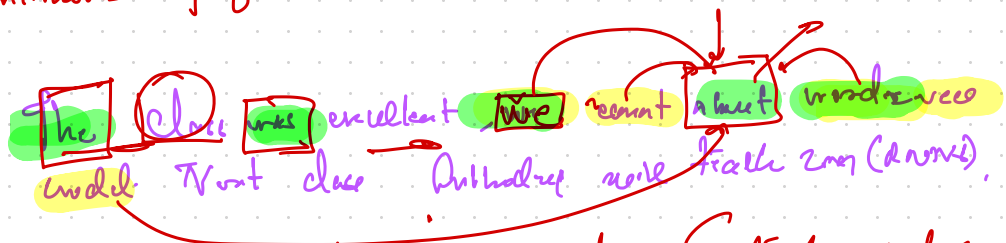
↳ Lan. Lang (RNN, LSTM)

Word2vec

↙ CBOW

Continuous Bag of words

↘ Skip gram model.



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

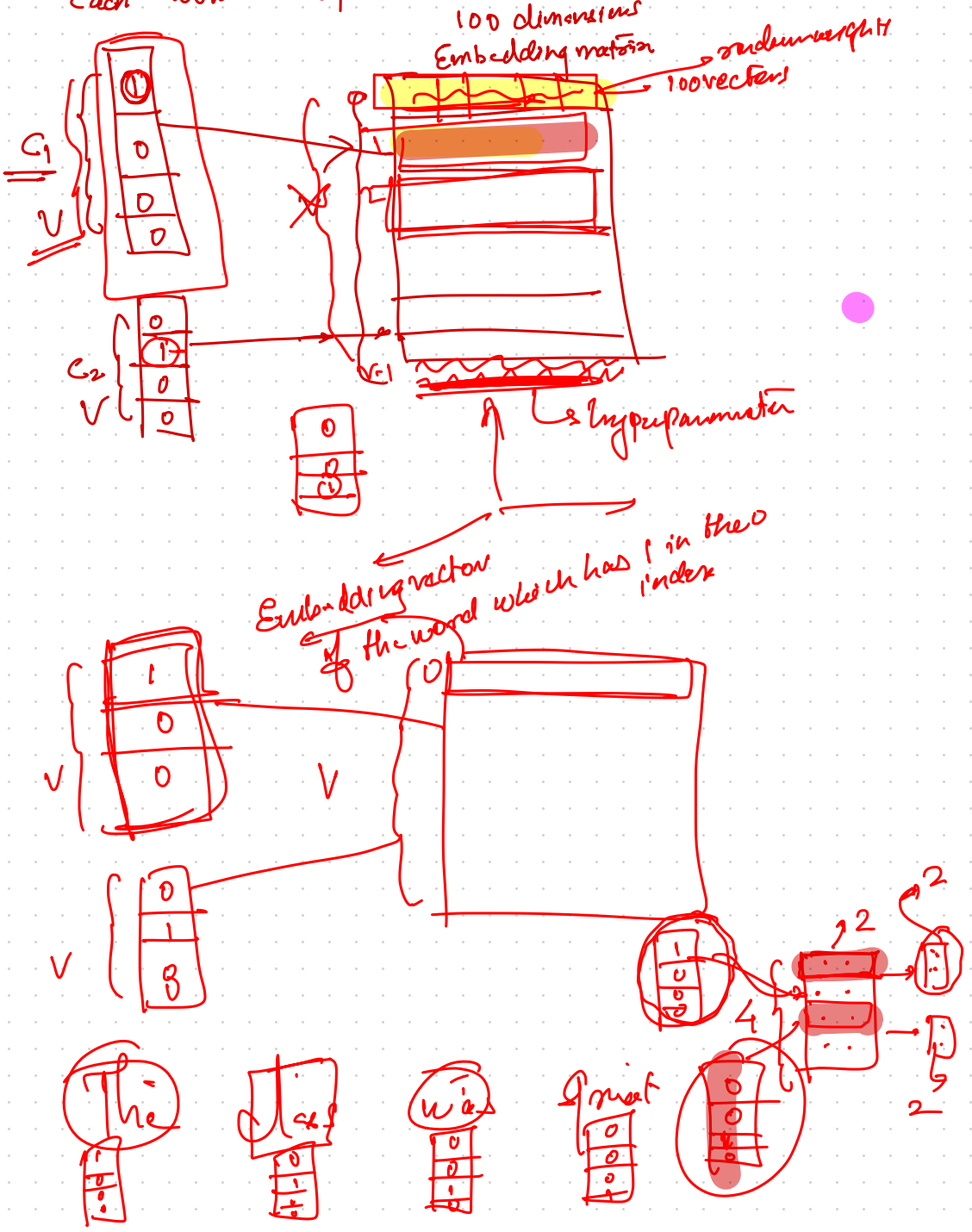
Context window = 1

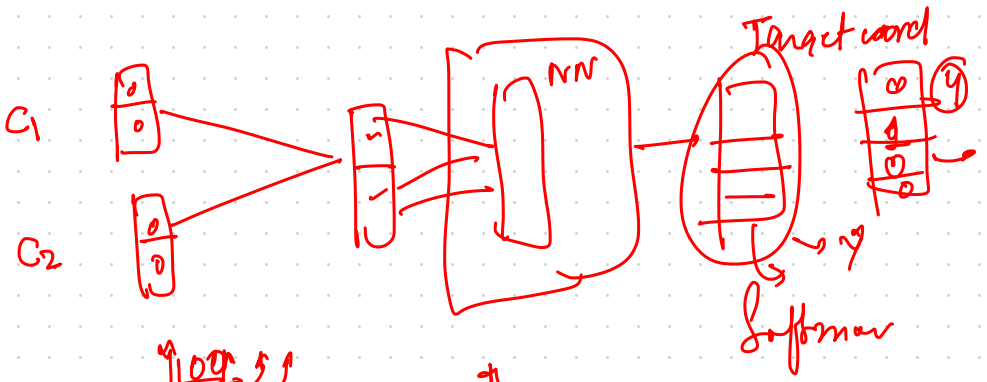
Context window = 2

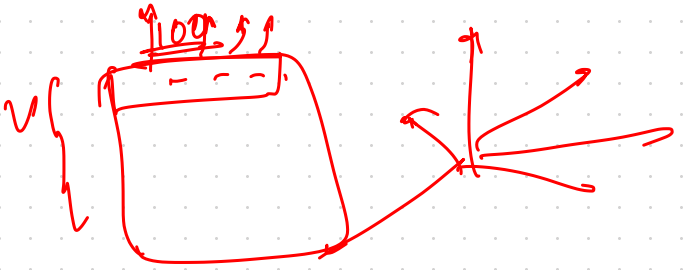
Class

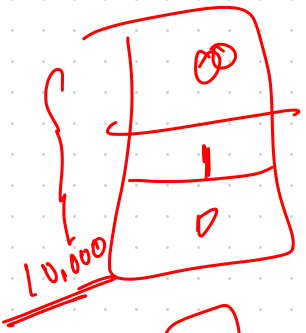
The

Each word is represented as a one hot encoded vector

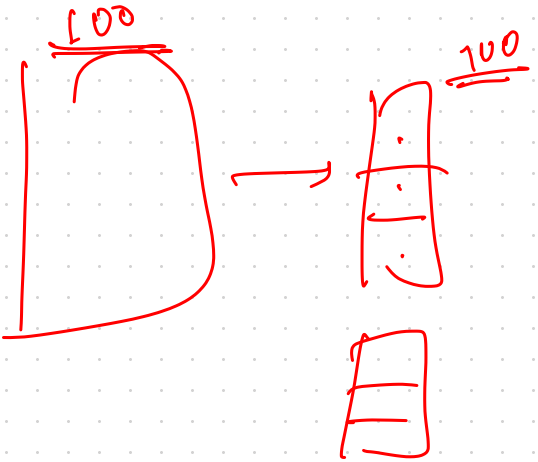



C_1
C_2
NN
Target word
y
Softmax

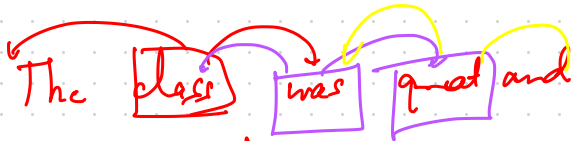

100
V


10,000

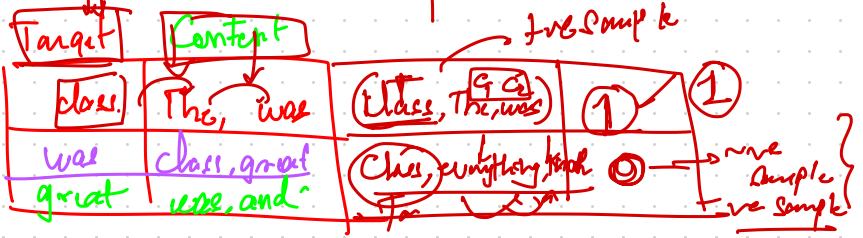

10,000

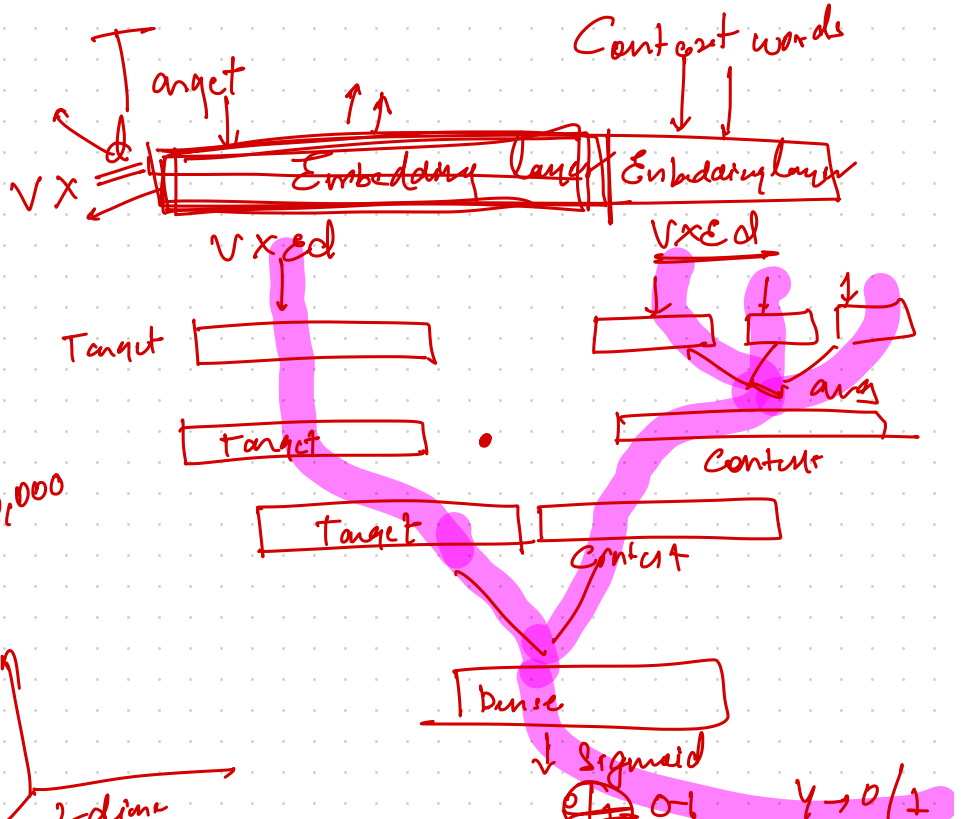

100
100

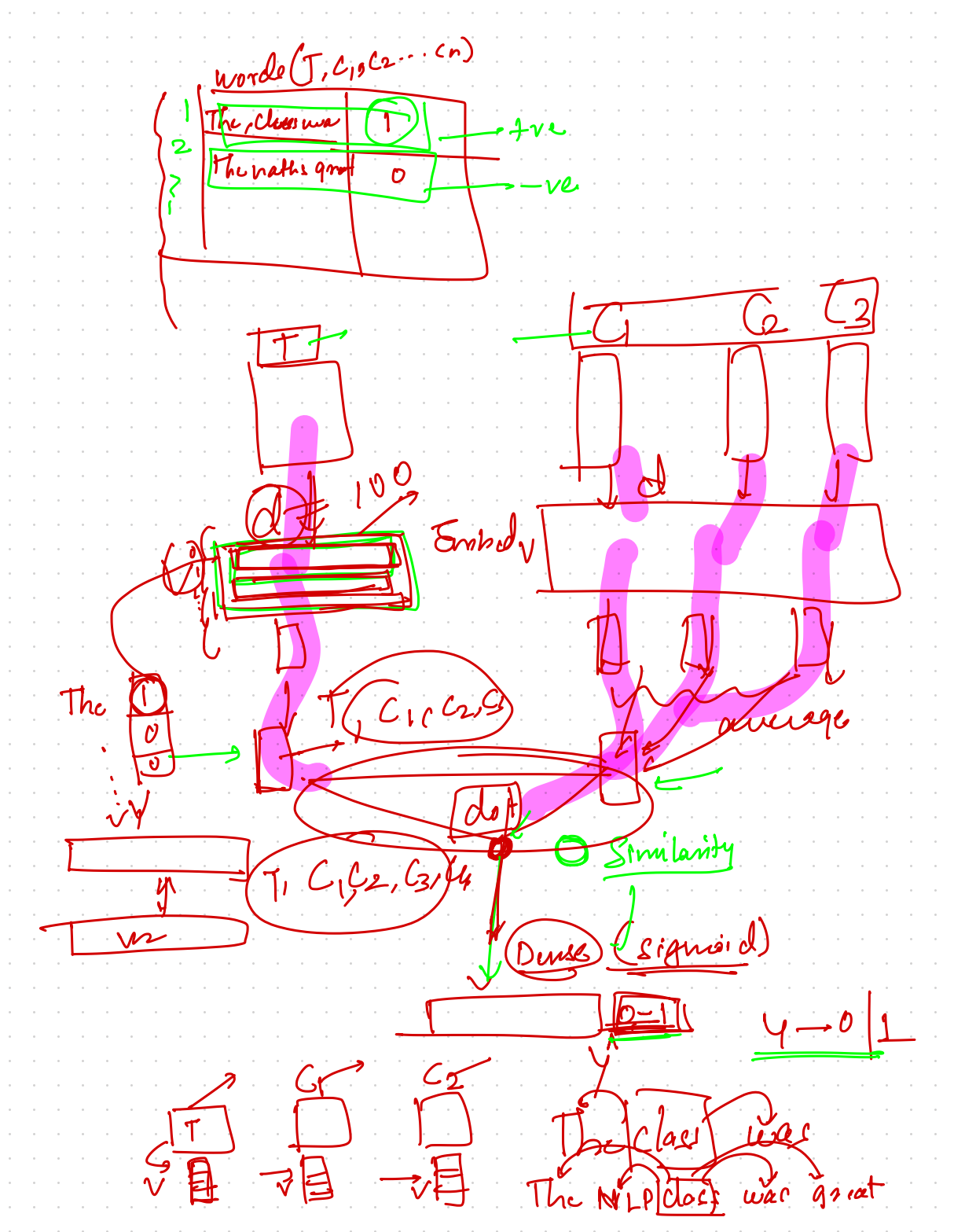
Skip-gram model

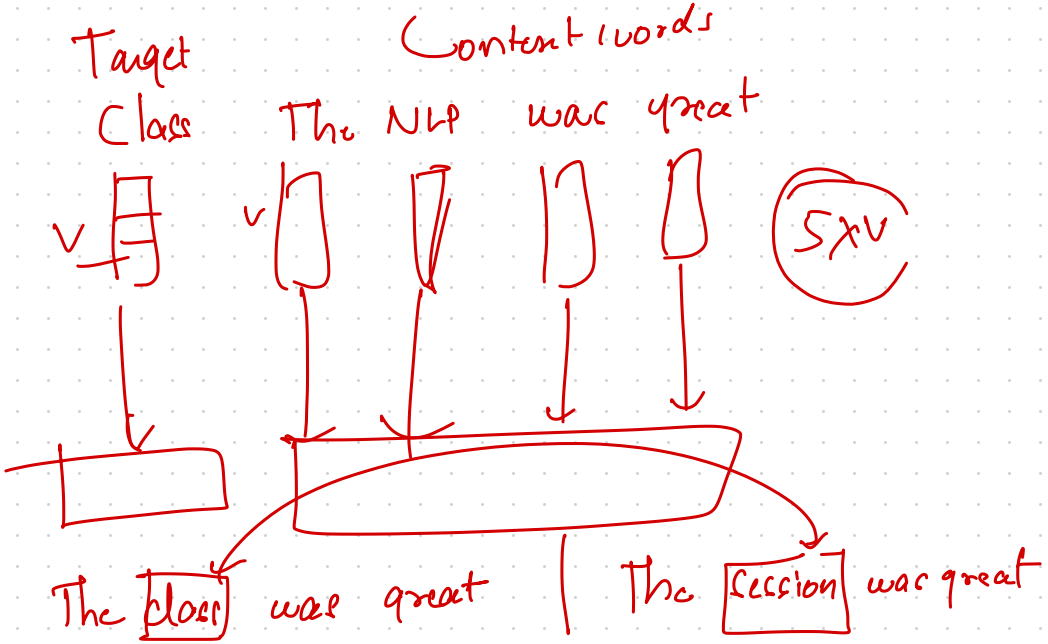


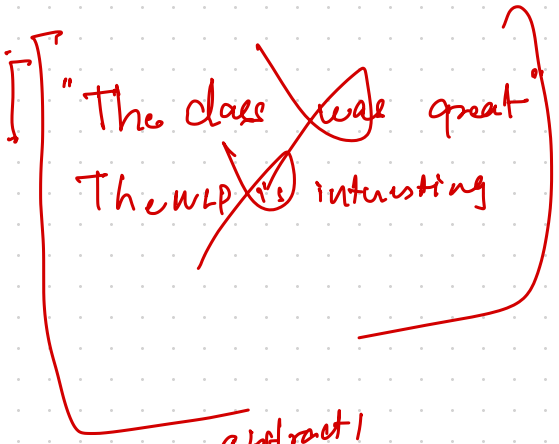
new concept





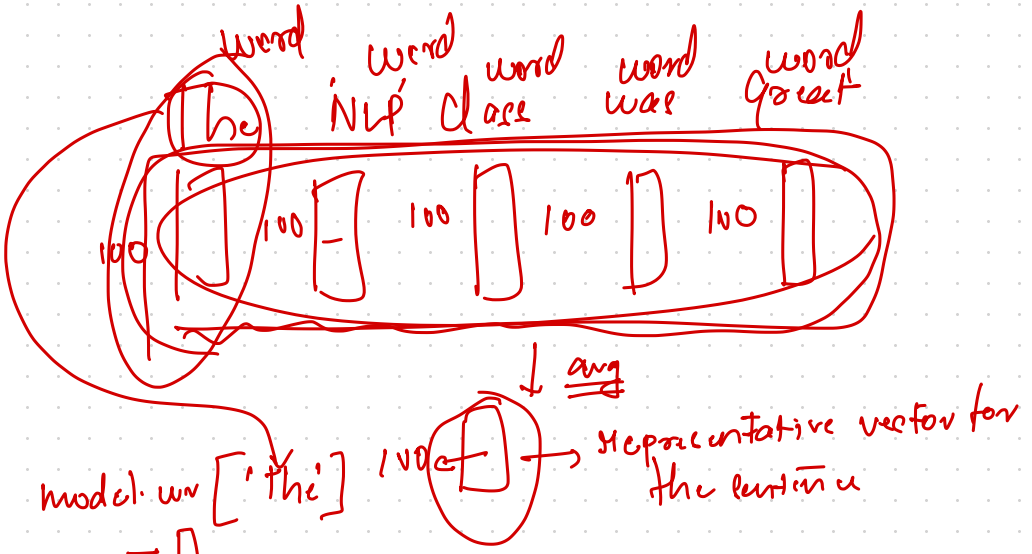

words (T, C1, C2 ... Cn)
The, class was
1
The maths great
0
+ve
-ve
T
C1
C2
C3
100
Embed
The
1
0
0
T, C1, C2, C3
dot
Similarity
T, C1, C2, C3, C4
Dense
(sigmoid)
0-1
y → 0 | 1
T
C1
C2
The NLP class was great

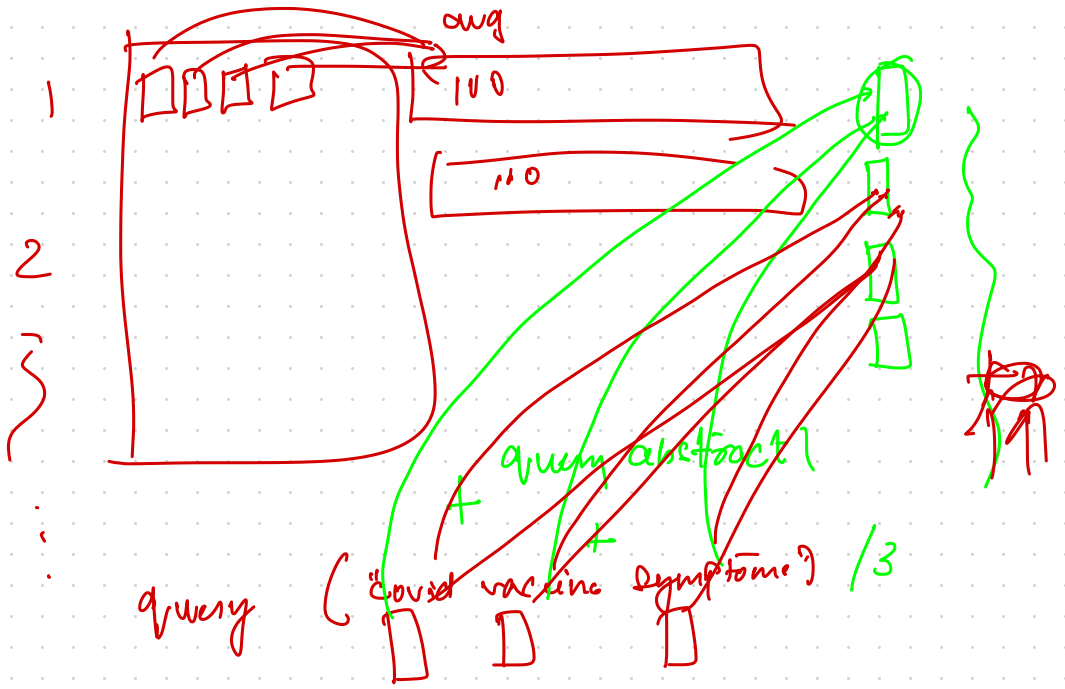


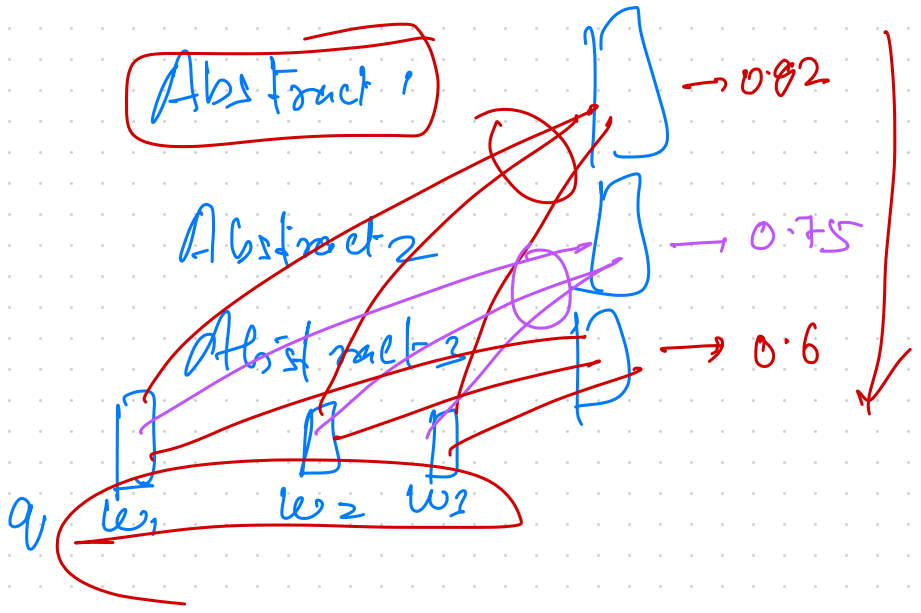


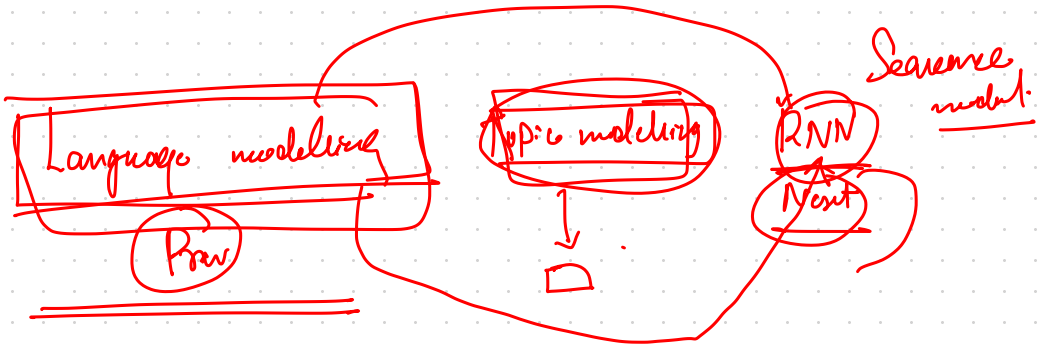
abstract1 abstract2

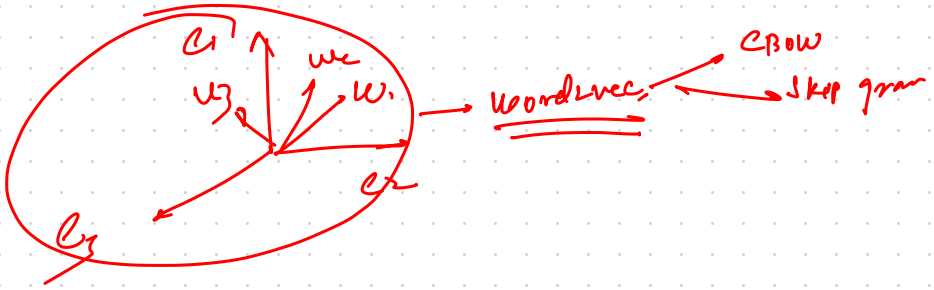
[['the', 'class', 'was', 'great'], ['the', 'NLP', 'is', 'interesting']]


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

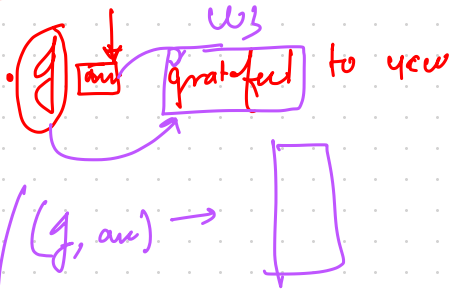

avg
1
2
query abstract 1
query ("covid vaccine symptoms") /3


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



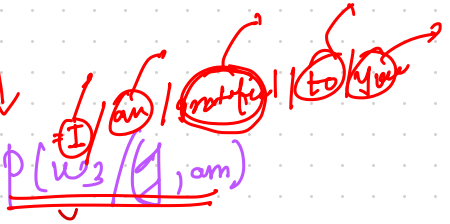


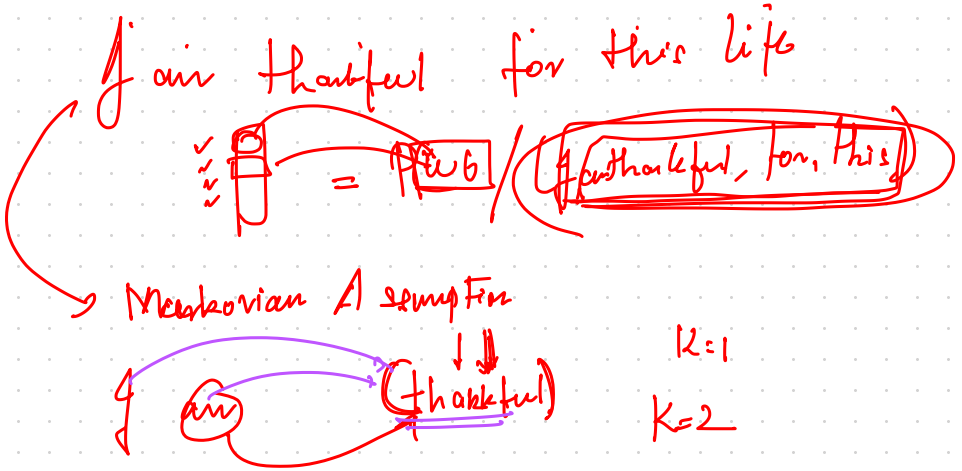
Language model

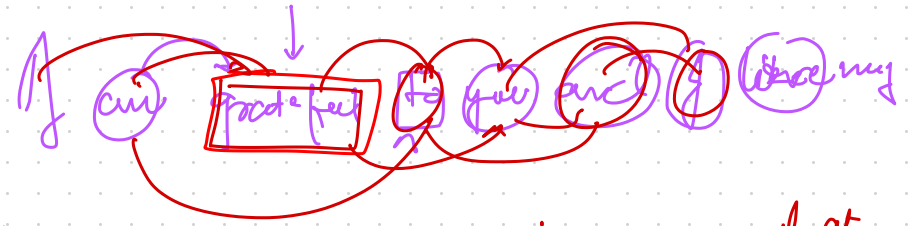


$p(w_3 / (I, am) \rightarrow$

I am

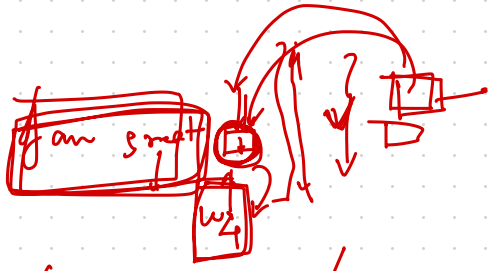






K=1 K=2

a particular word at a position is dependant on previous two words



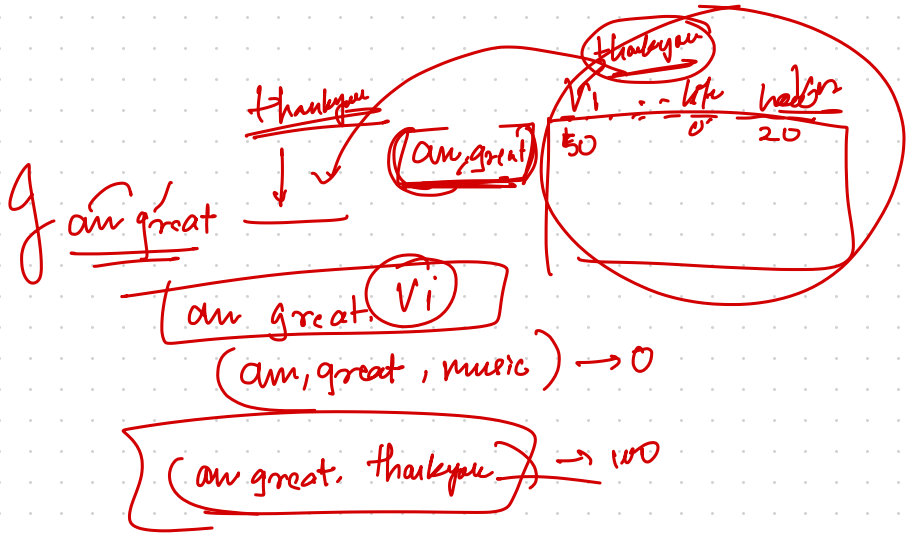
$P(\quad / (I, am, great)$

K=0 $\sim P(\quad)$ Unigram

K=1 $\sim P(w_{4-1}, v_i / great)$ Bigram

K=2 $\sim P(w_4 = v_i / great, am)$ Trigram

K=3 $\sim P(w_4 = v_i / (great, am, I))$ quadgram



I am — 2

am good — 2

good how — 1

how are — 1

are you — 1

you too — 1

I → am → good

I ~~X~~

am ~~X~~

good

how ~~X~~

are ~~X~~

you ~~X~~

too ~~X~~

$P(w_3 = ? / am)$

I am ___ $P(\_ / am)$

$P(w_3 = v_i / (I\ am))$

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

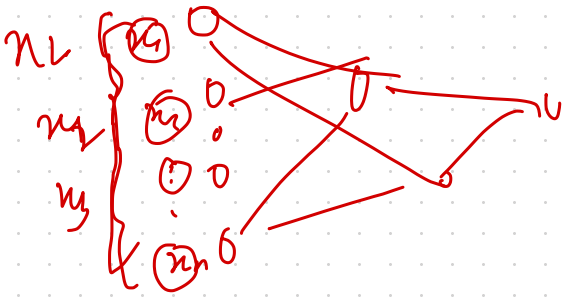
K gram

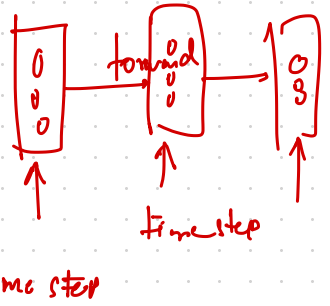
k = language → 100

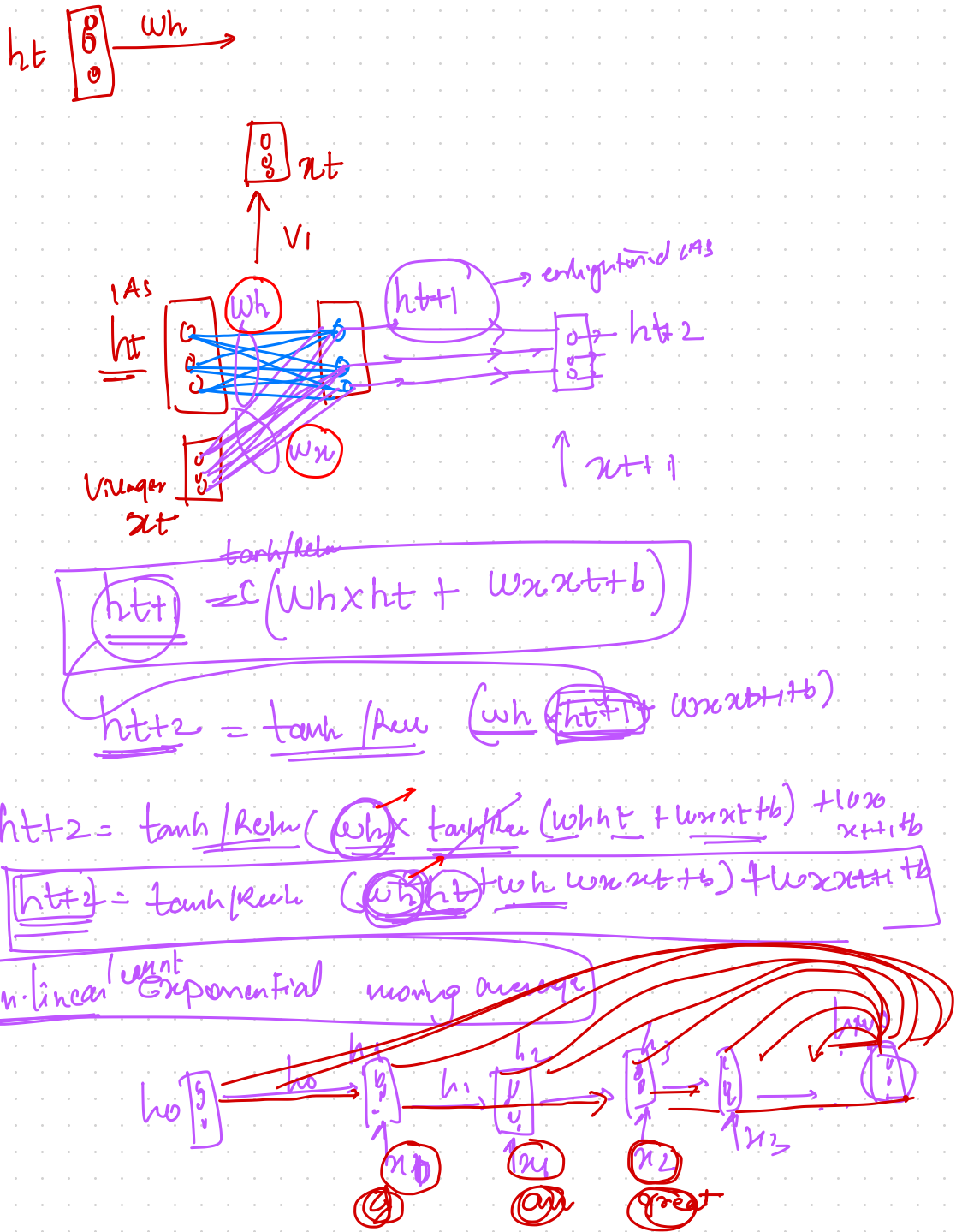
$P(\_ / \ldots\ldots)$

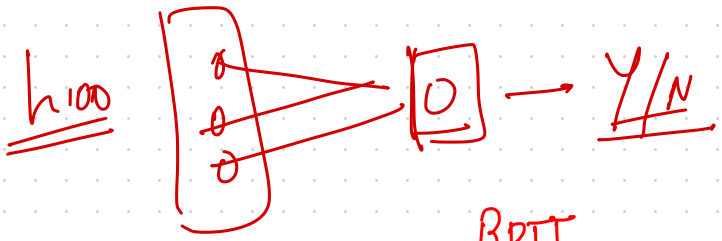
Recurrent Neural Networks

9:51 → ~~9:55~~ 10:00pm

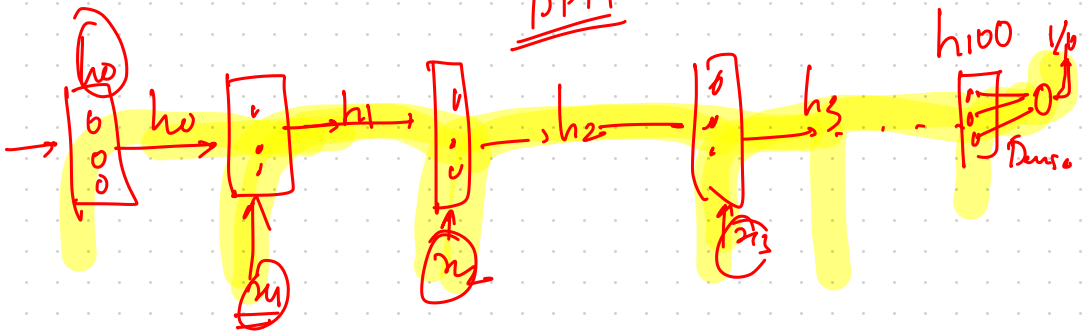




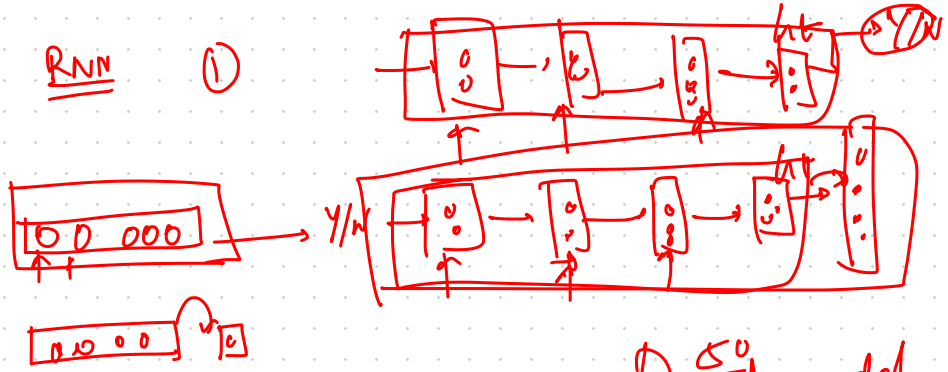

IAS
ht
Wh
Vi
xt
IAS
ht
Wh
ht+1
enlightened IAS
ht+2
Wx
Image
xt
xt+1
tanh/Relu
ht+1 = f(Whxht + Wxxt+b)
ht+2 = tanh/Relu (wh ht+1 + wxxt+1+b)
ht+2 = tanh/Relu (wh x tanh/Relu (Whht + Wxxt+b) + Wx xt+1+b
ht+2 = tanh/Relu (wh2 ht + wh Wxxt+b) + Wxxt+1 +b
Non-linear
Exponential moving average
h0
h1
h2
x0
x1
x2
x3
Cat
Great


h100
Y/N

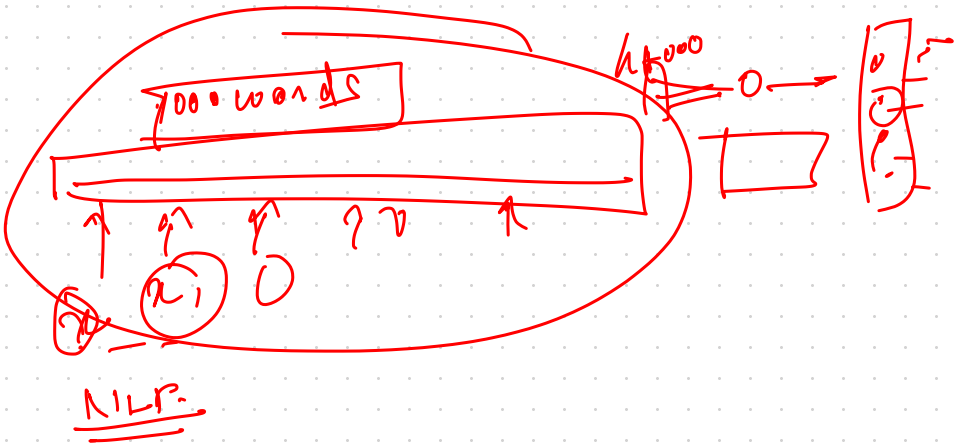
BPTT

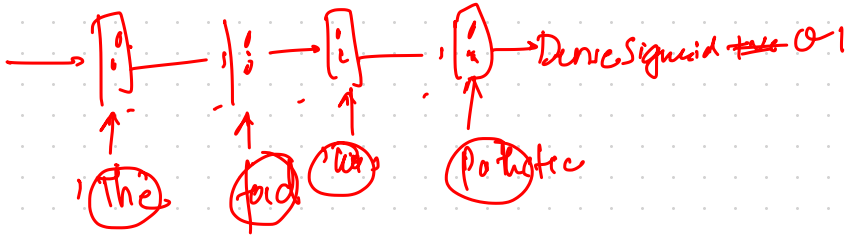

h100
Y/p

RNN ①


Y/N

Decoder model
Print → 1000
Encoder
Source language
Target language


1000 words
NLP.


Dennis Sigueid
The
fox
Politics

1000x10

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

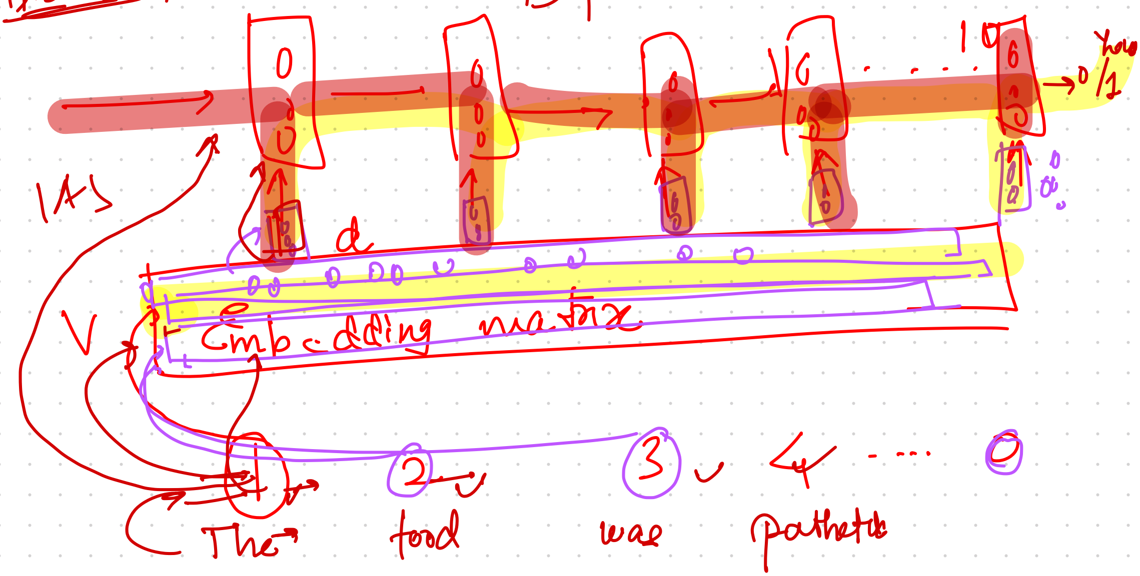
1000

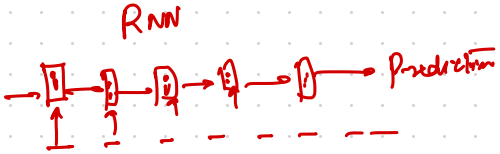
x → y max-length → 10

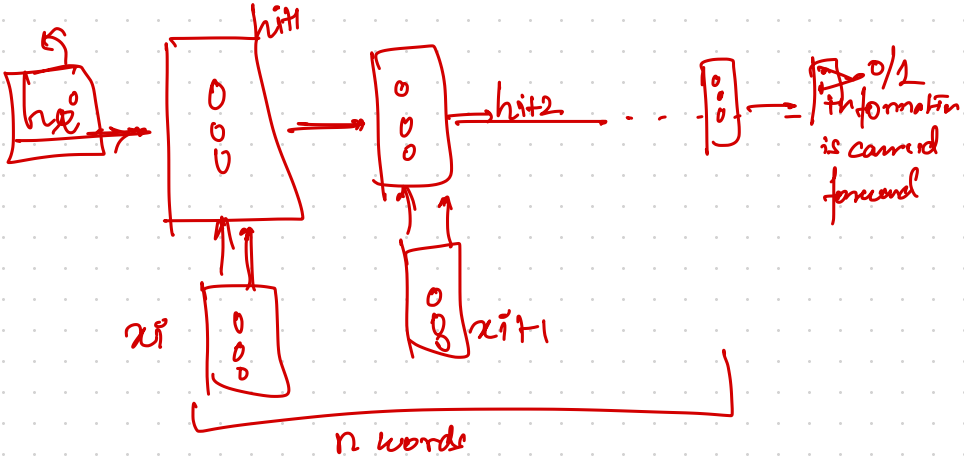
	x	y
1	The food was pathetic	-ve
2	The ambience was excellent, definitely a visit	+ve
3		

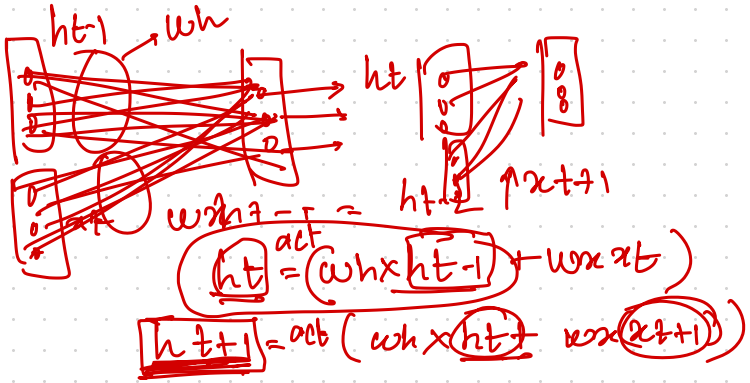
BPTT

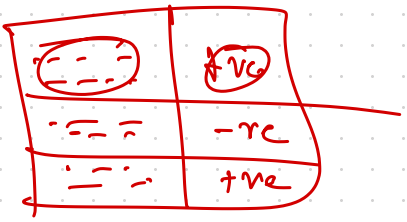
Bi: L case

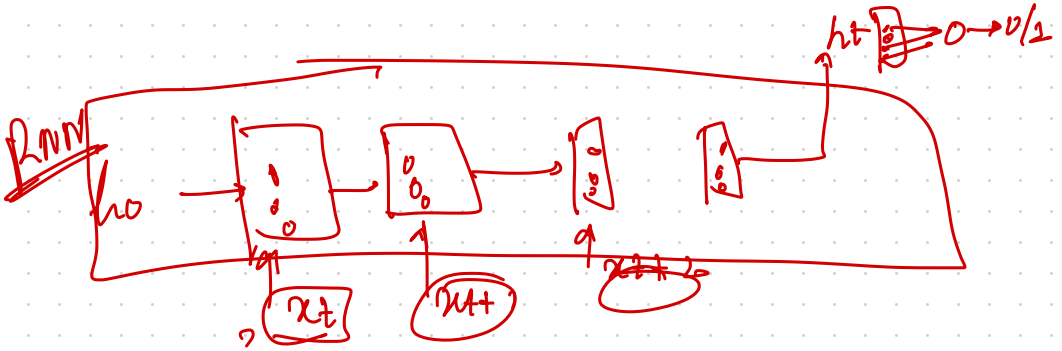


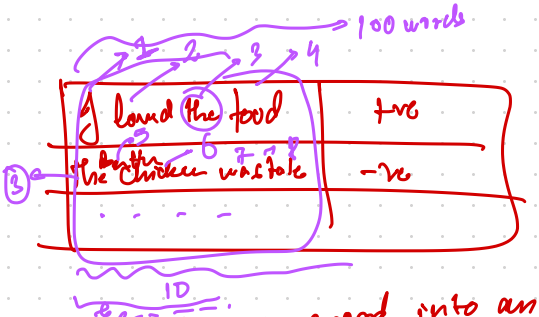

RNN
Prediction


hi
hi+1
hi+2
0/1
information is carried forward
xi
xi+1
n words


ht-1
Wh
ht
xt+1
act
ht = (Whx ht-1 + Wxx xt)
ht+1 = act (Whx ht + Wxx xt+1)

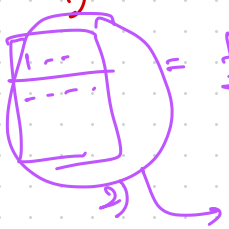


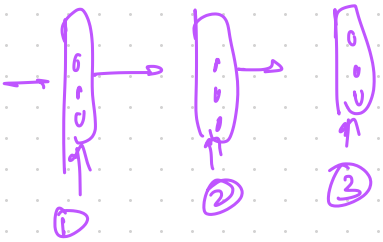


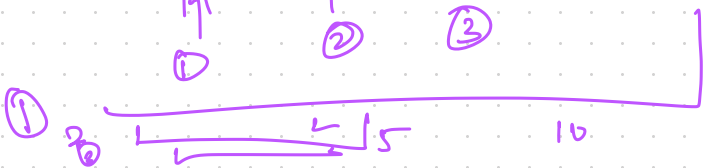


1) We will Convert each word into an integer

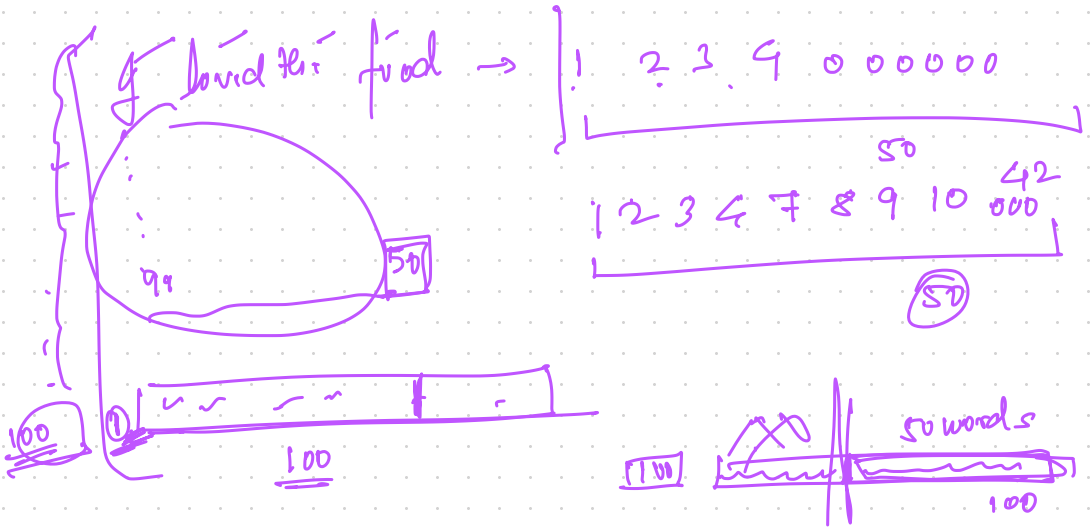
= tokenizer.text_to_sequences (df['text'])





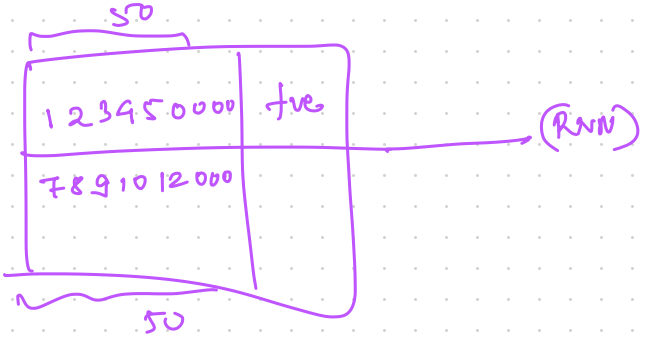


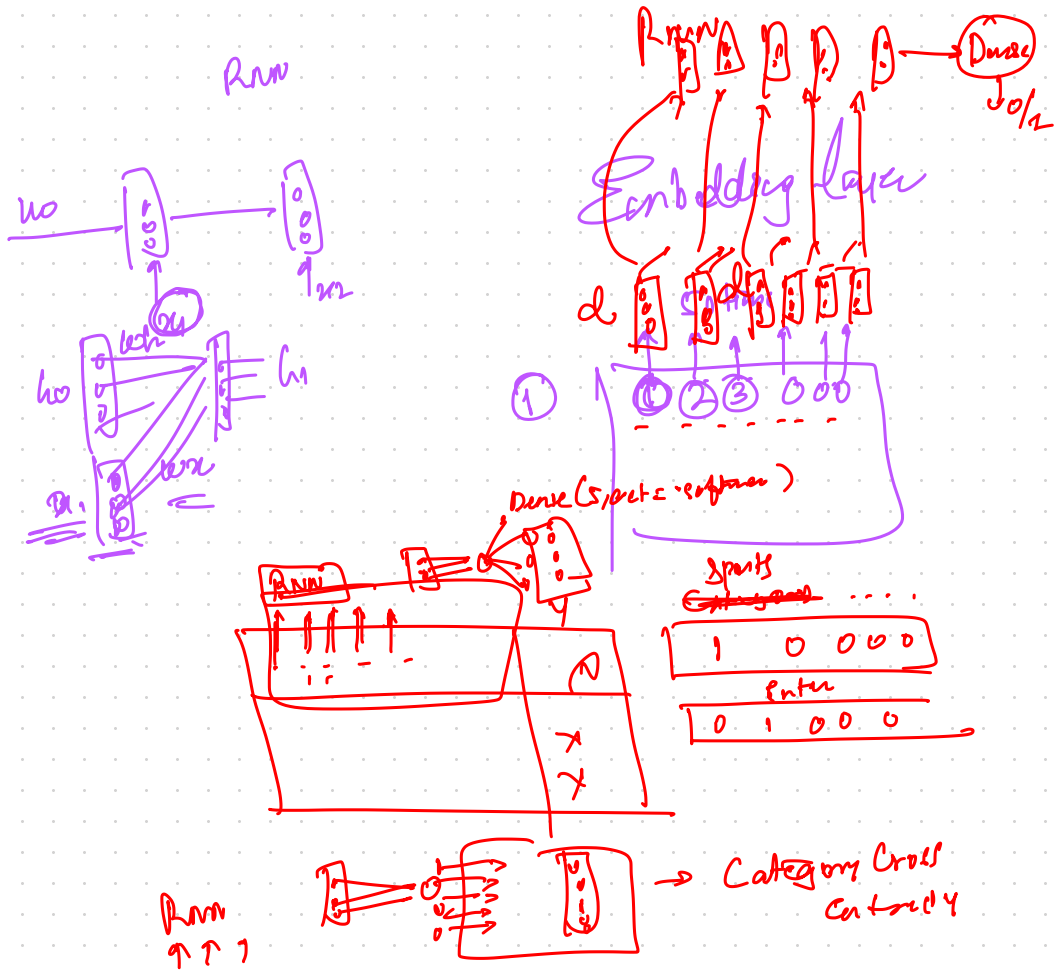
Padding

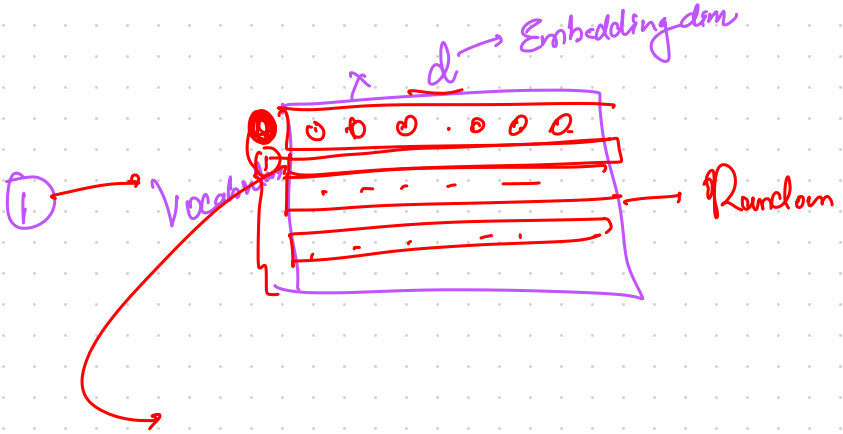


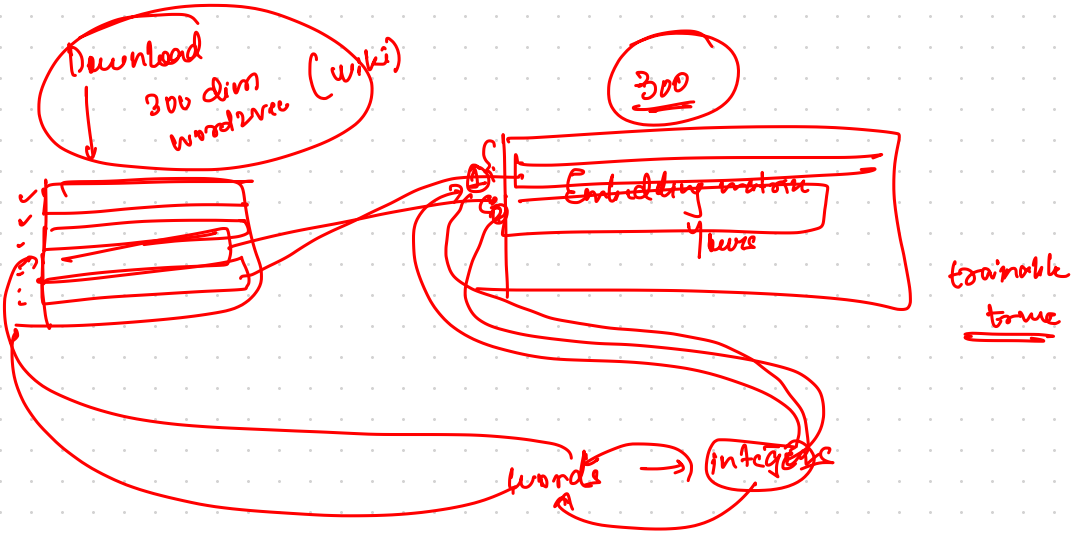
= Pad-sequences (df['integer text'])

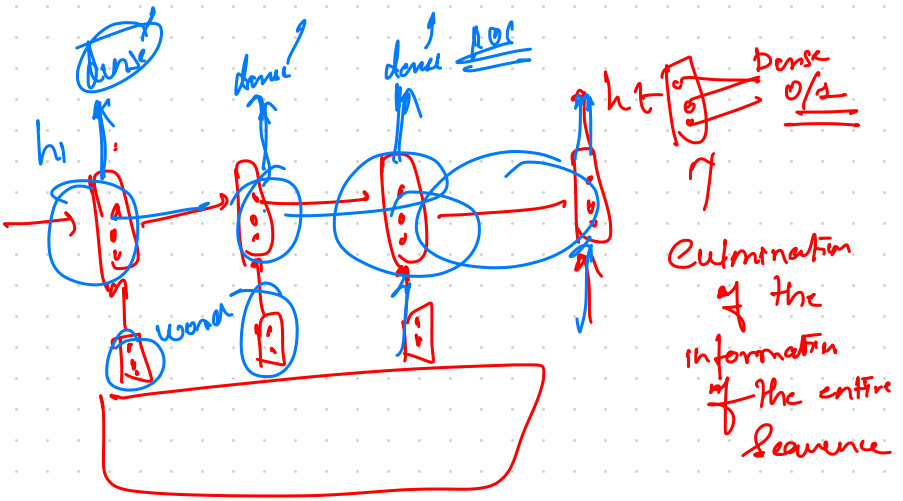
100 X 50

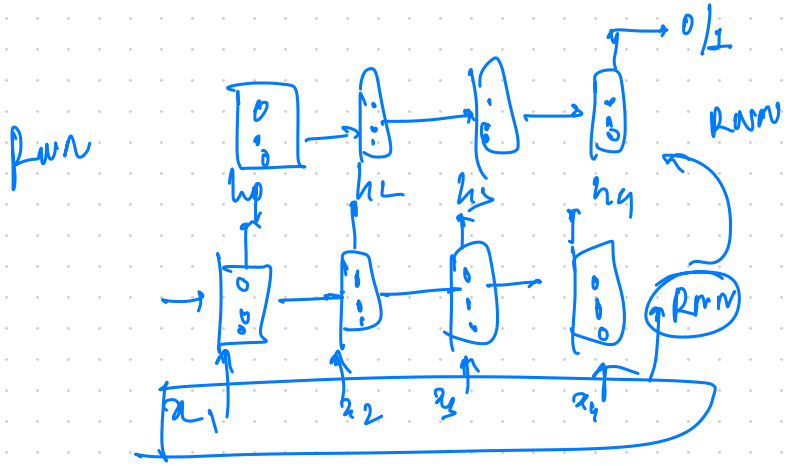



RNN
Embedding layer
Dense
Sparse
Enter
RNN
Category Cross Entropy


d
Embedding dim
1
Vocabulary
Random


Download
300 dim
word2vec
(wiki)
300
Embedding matrix
trainable
true
words
integers

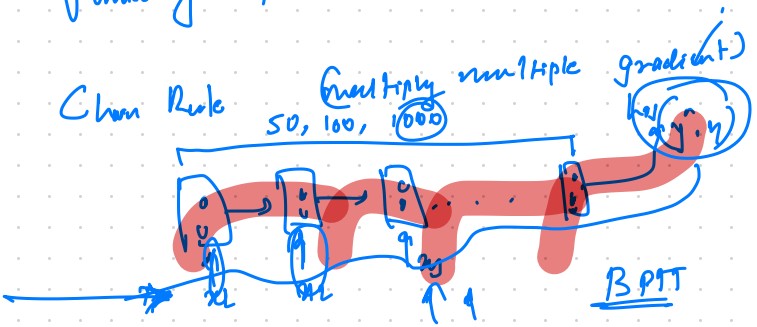

dense
dense
dense
h1
ht
Dense
O/1
word
Culmination
of the
information
of the entire
Sequence


RNN
0/1
RNN
RNN
h_0
h_1
h_2
h_4
x_1
x_2
x_3
x_4

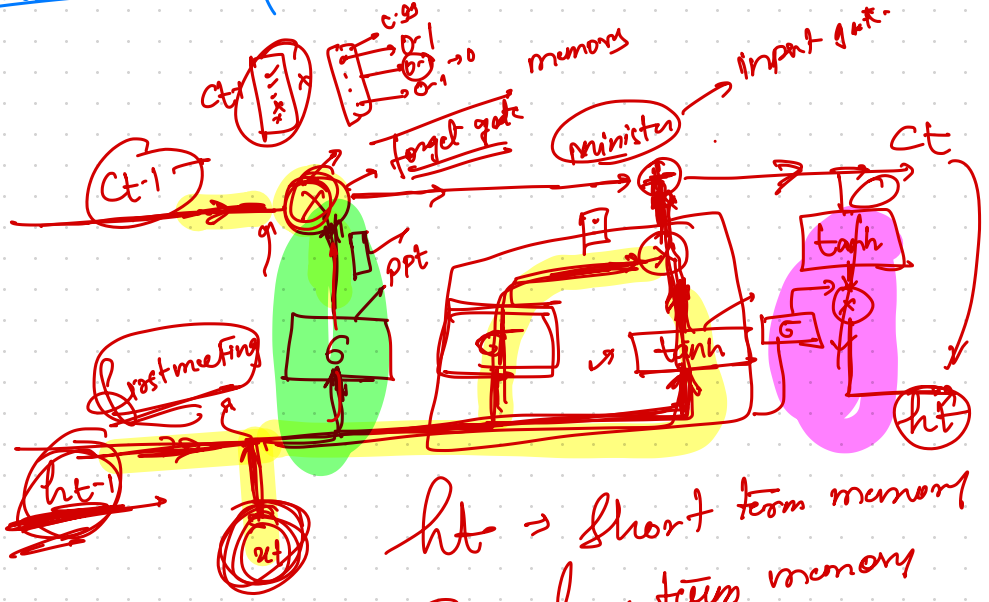
RNN → some Problems

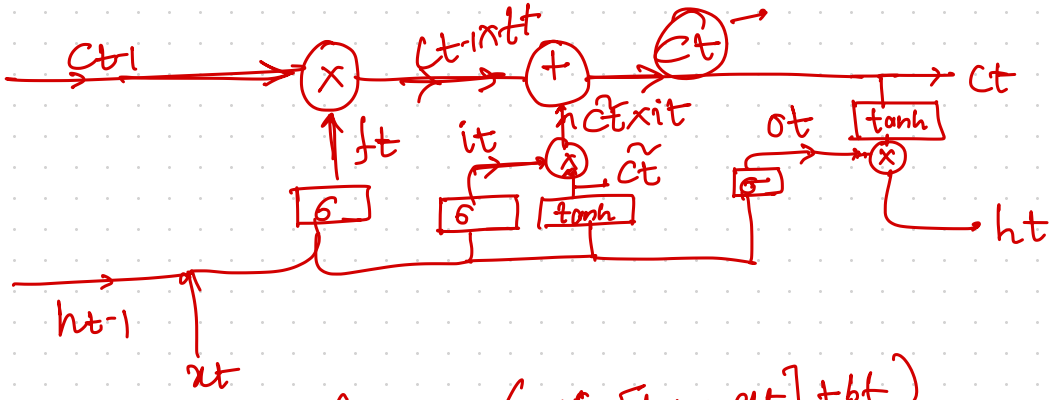
Vanishing Grad

Chain Rule



Solution to RNNs Problem:





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

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

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

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

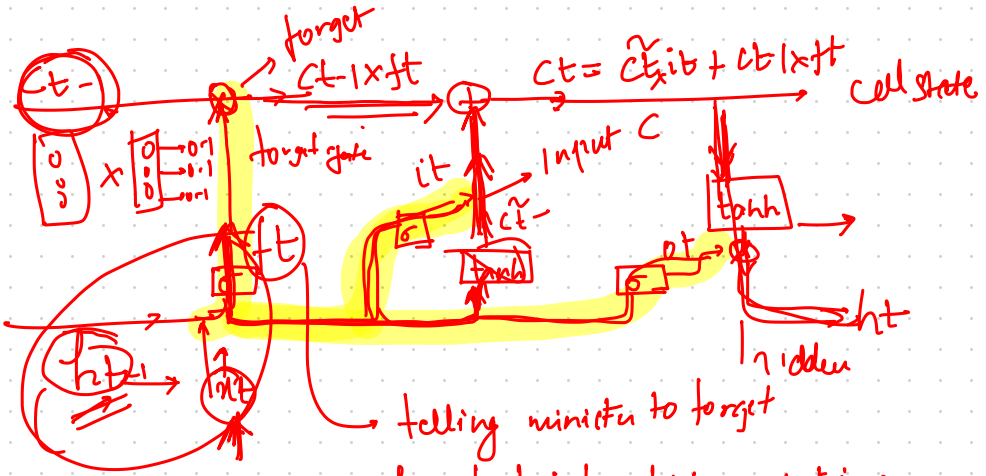
$$h_t = \tanh(C_t) \times o_t$$

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

Long short term memory

RNN → long term dependencies

long term memory
short term memory

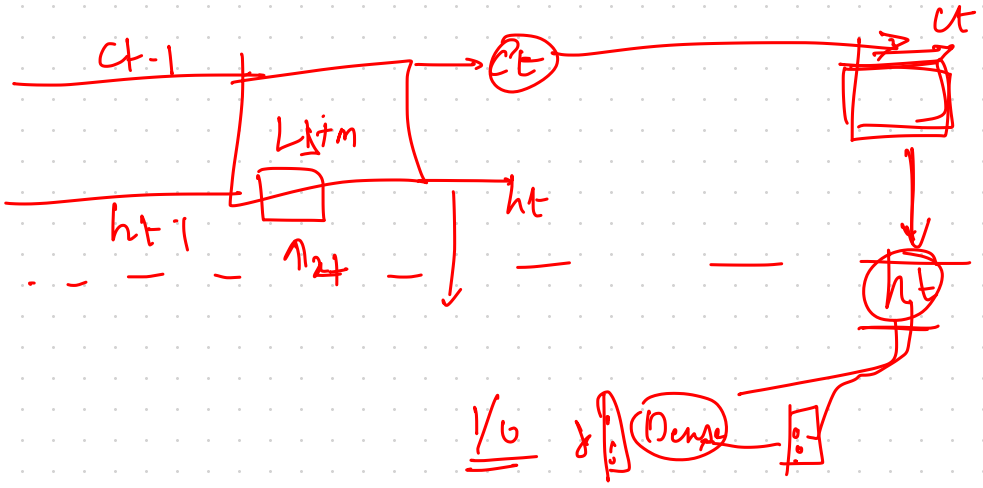


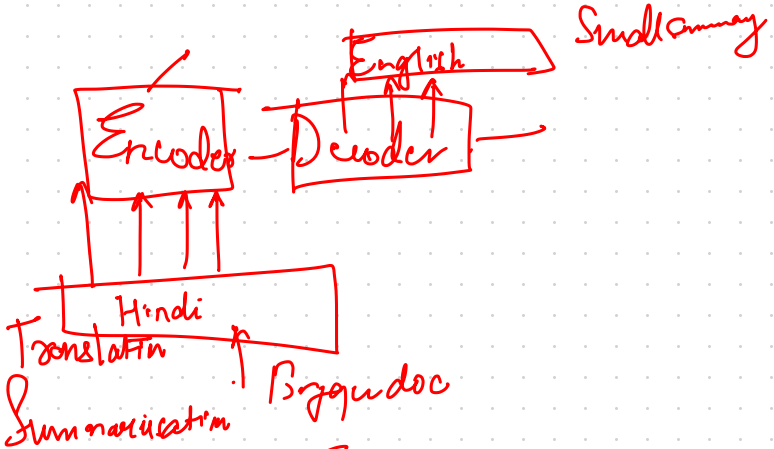
$$i_t = \sigma(W_i [h_{t-1}, x_t] + b_i)$$

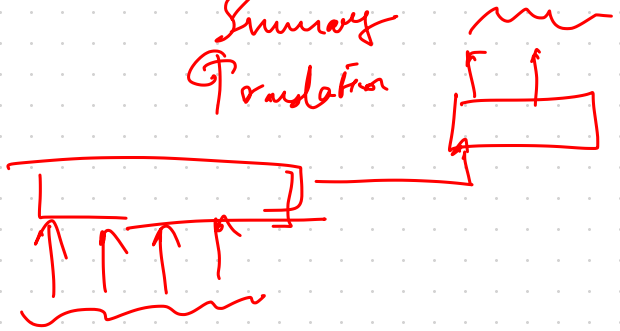
$$o_t = \sigma(W_o [h_{t-1}, x_t] + b_o)$$

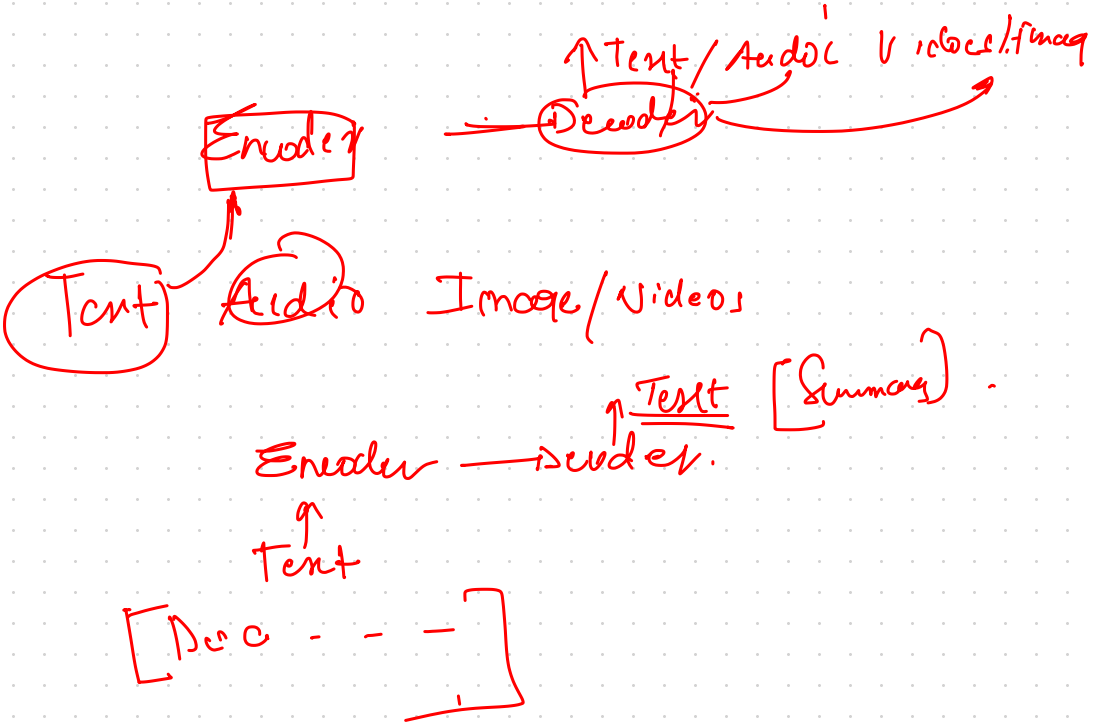
$$h_t = o_t \times \tanh(C_t)$$

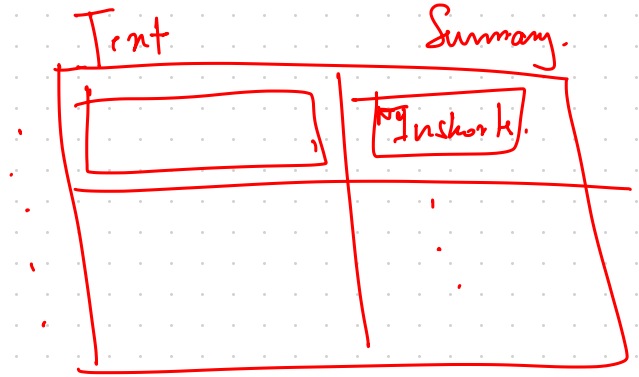
f_t → h_t

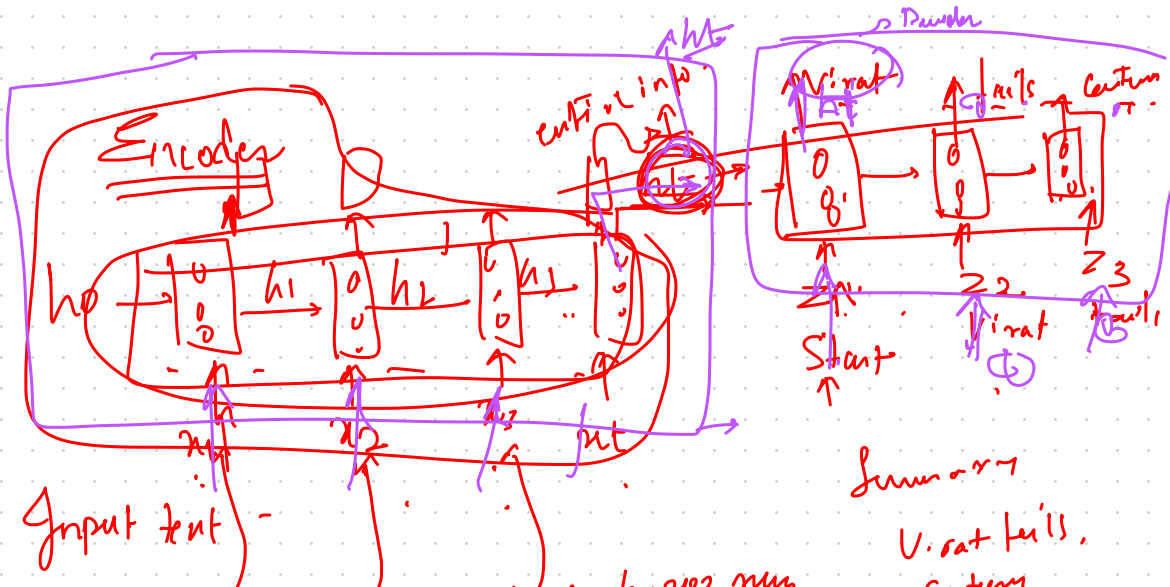

Ct-1
Ct
Lstm
ht-1
ht
ht
1/6
Dense


Small summary
English
Encoder
Decoder
Hindi
Translation
Summarization
Bzqeudoc


Summary
Translation


Text / Audio / Video / Image
Decoder
Encoder
Text
Audio
Image / Videos
Text [Summary]
Encoder
Decoder.
Text
[Doc - - -]


Text
Summary.
Inshort.



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

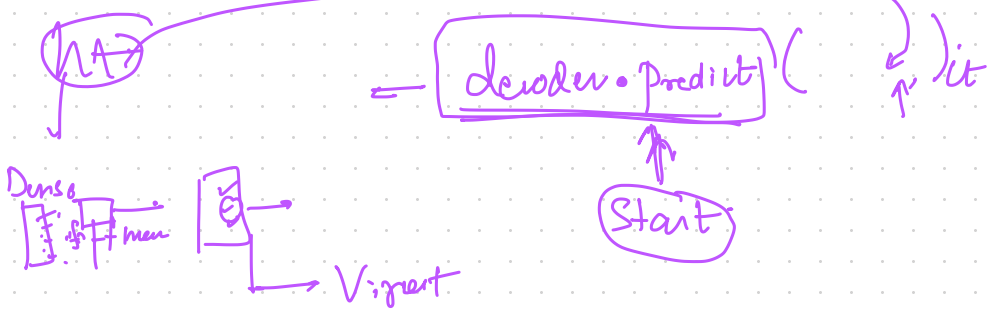
Summary

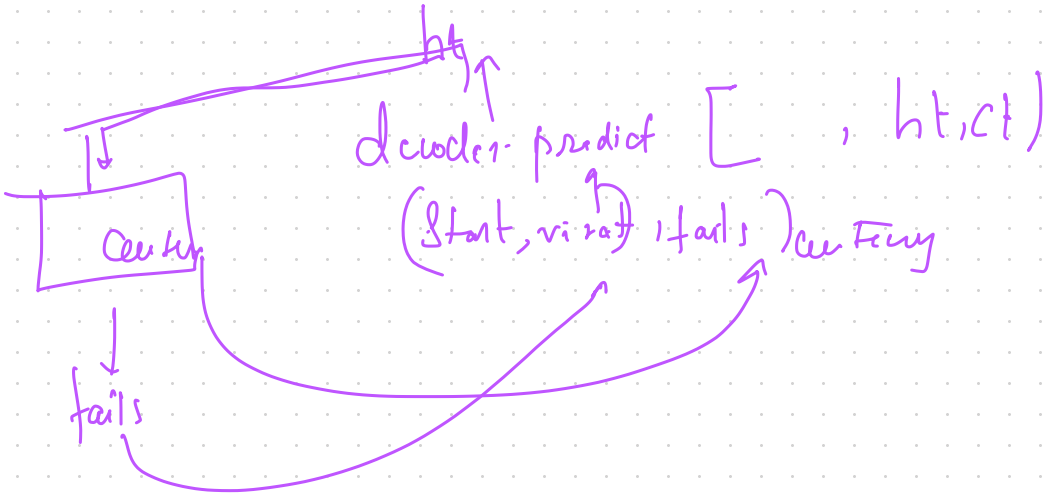
Virat fells, caution, India looses etc.

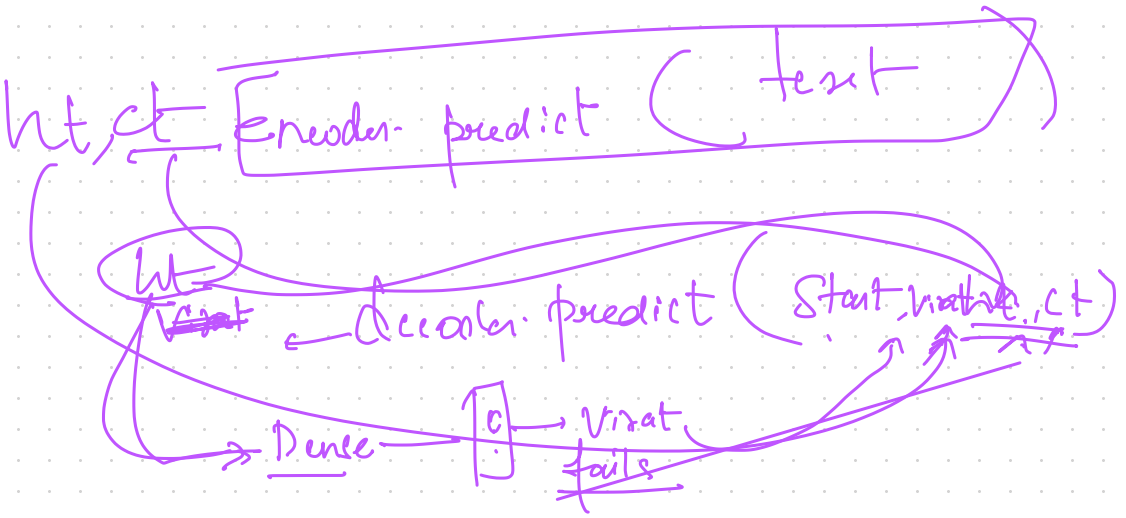
Inference mode (.predict)

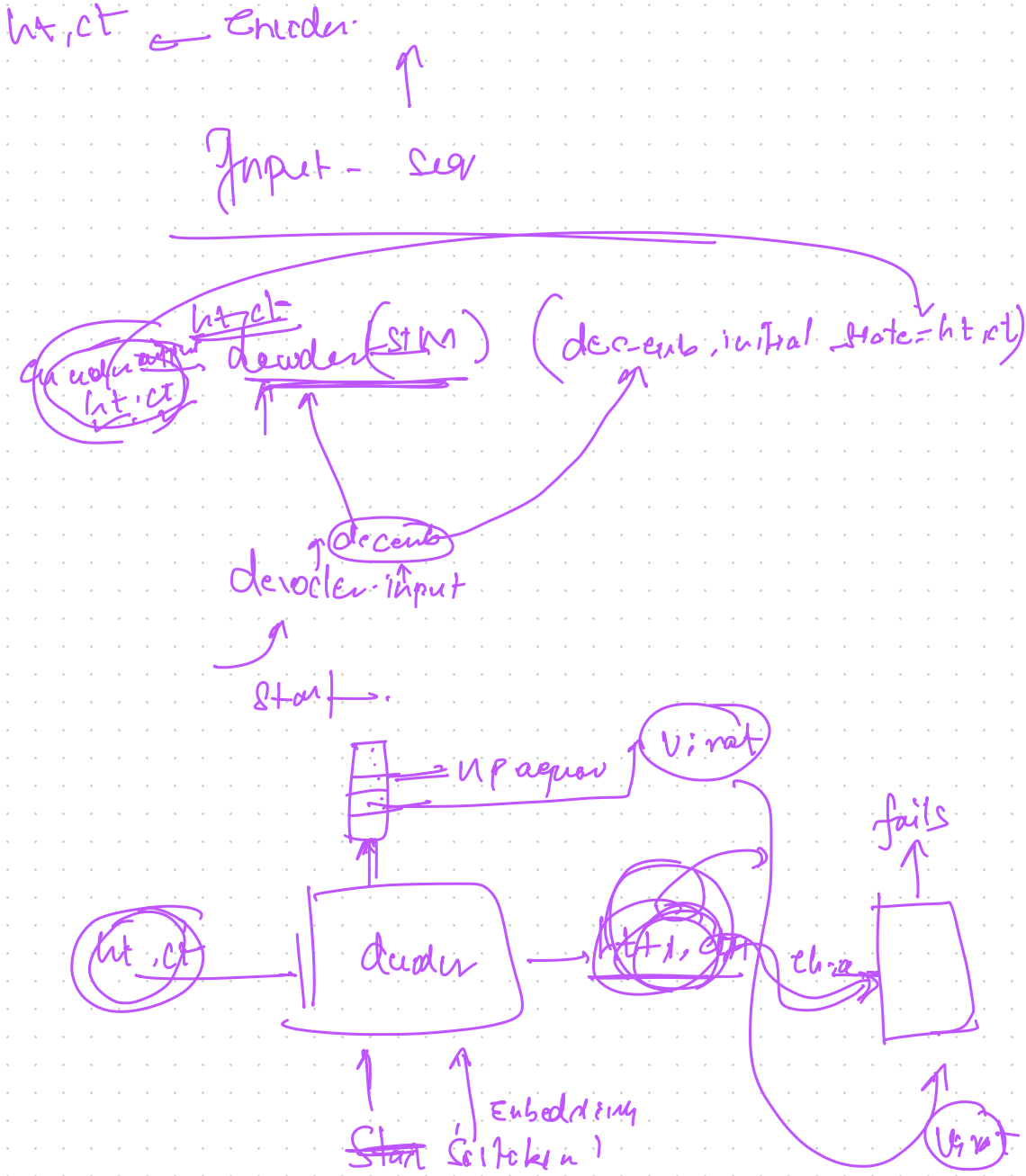
Entire text

h_t = Encoder.Predict (text)




ht
decoder. predict [, ht, ct)
(start, virat, tails) curFeny
tails


ht, ct
Encoder. predict
text
ht
virat
decoder. predict
start, virat, ct
Dense
C
virat
tails


ht, ct
Encoder
Input - Seq
ht, ct
decoder (STM)
(dec-emb, initial state = ht ct)
dec-emb
decoder-input
start
Uparrow
V: mat
fails
ht, ct
decoder
ht+1, ct+1
Embedding
Start Token