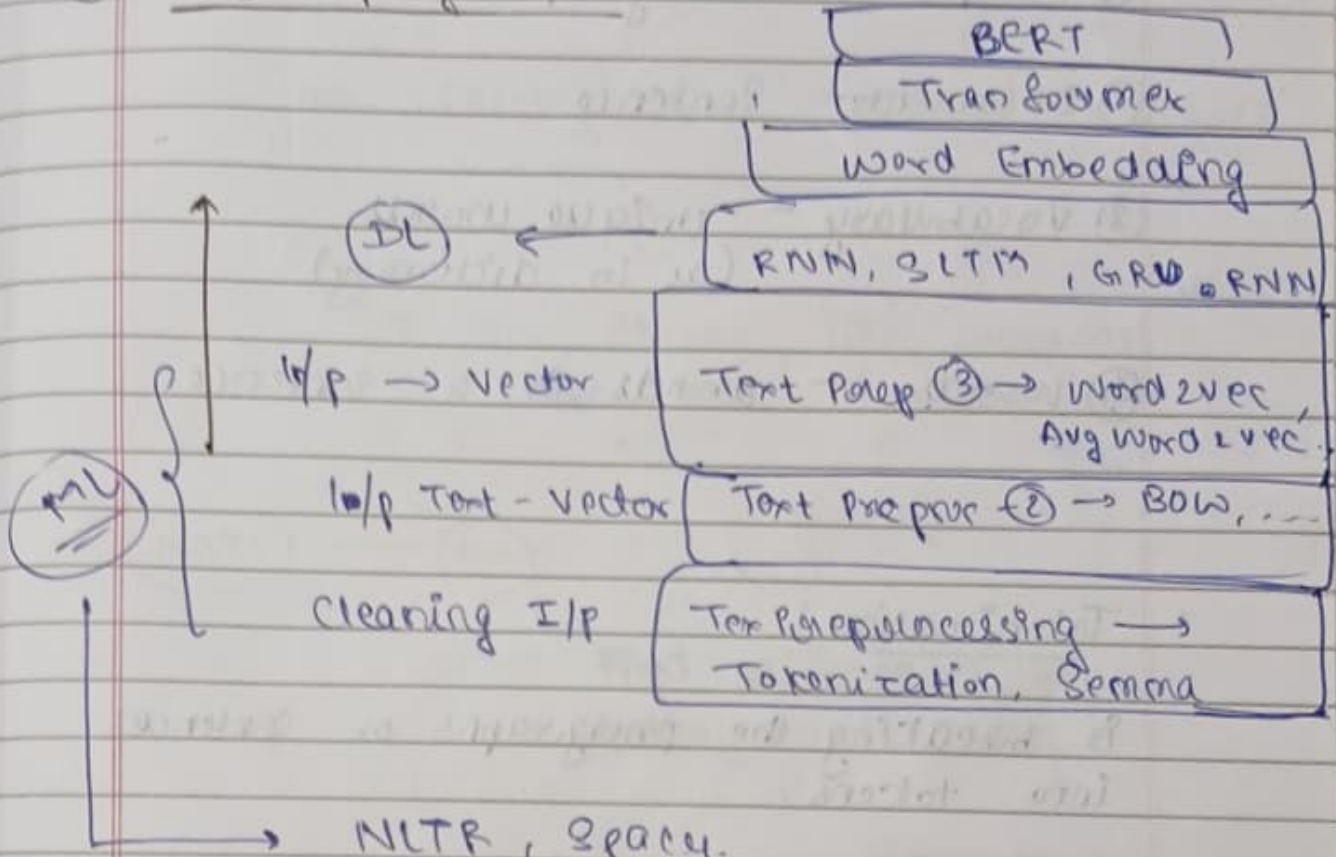
NLP (SOC)

Complete NLP machine learning in one shot.
— Krish Naik (YouTube)

⊕ Road map of NLP :-



→ NLTK, Spacy.

TOKENIZATION IN NLP :-

* Topics

① Corpus – Paragraph

② Documents – Sentence

③ Vocabulary – Unique words (as in dictionary)

④ Words – words in a sentence.

- Tokenization :

B breaking the paragraphs or sentences into tokens.

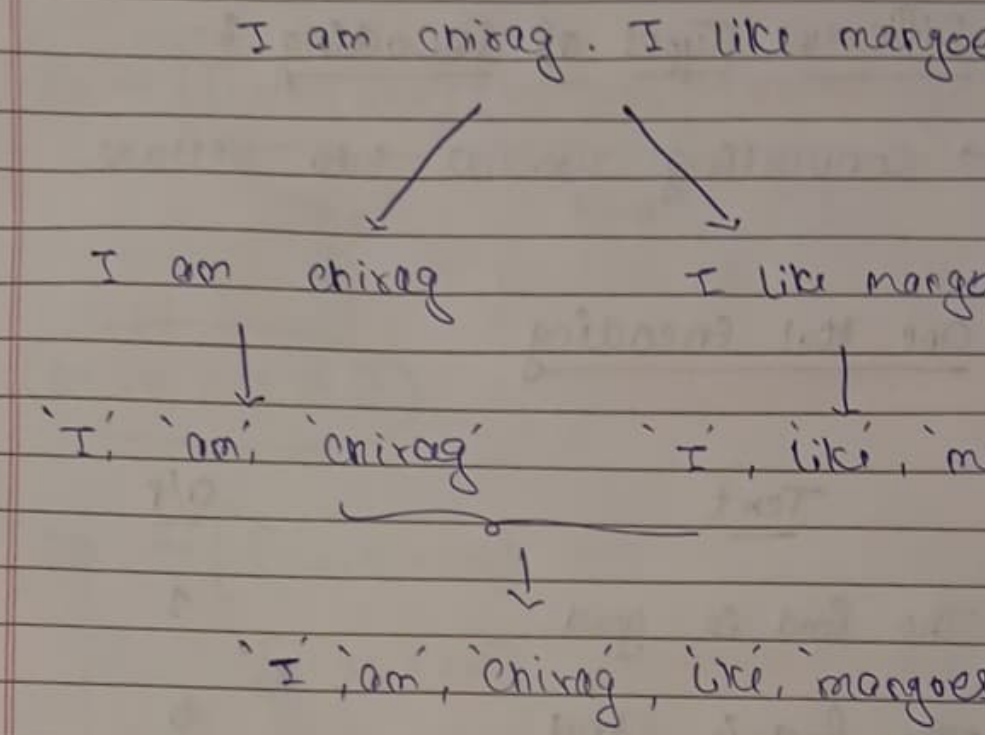
i.e.

{ Paragraph s } → { Sentences } → { words }

↓

{ Unique words }

PTO →



NOTE — HW

→ Find differences b/w Spacey and NLTK

Stemming

→ Process of reducing a word to its word stem that affixes to suffixes or prefixes to the root of words known as lemma.

Different Types of Encoding :-

→ Converting words into vectors.

① One Hot Encoding

	Text	O/P
D1	The food is good	1
D2	The food is bad	0
D3	Pizza is amazing	1

Vocabulary {unique words}

↳ The food is good bad Pizza amazing

⇓

1	0	0	0	0	0	0
0	1	0	0	0	0	0
0	0	1	0	0	0	0
			⋮			

→ Number of unique vocabulary

$V = 7$

∴ We will have 7 vectors to represent

D1 => [[1000000], [0100000], [0010000], [0001000]]

The Food is good.

D1 = 4 x 7

D2 => [[1000000],
[0100000],
[0010000],
[0001000]]

Advantages

→ Easy to implement with python (Sklearn and pandas)

Disadvantages

→ Sparse matrix → Overfitting

⇓

Good accuracy for training but not for new data

→ ML Algorithm → We need fixed size I/p

→ No semantic meaning is getting captured

→ out of Vocabulary → ~~if training data~~ if test data has a new word.

(2) Bag of words :-

→ Sentiment classification, Mail Spam or ham?, etc.

Dataset

	Text	O/p
S1:	He is a good boy	1
S2:	She is a good girl.	1
S3:	Boy and girl are good	1

↓ lowering all words + stopwords

S1 : good boy

S2 : good girl

S3 : Boy girl good

Vocabulary	frequency
good	3
boy	2
girl	2

→ Maximum frequency is put first (i.e. descending order)

→ We select the top 10 ~~if~~ 20 features based on the frequency.

	good	boy	girl
S1 :	[1	1	0]
S2 :	[1	0	1]
S3 :	[1	1	1]

Binary BOW and BOW

{ 1 and 0 only } { Count is updated with frequency in statement }

Advantages

→ Simple and Intuitive

→ Fixed sized I/P - ML algorithm.

Disadvantages

→ Sparse matrix or arrays → overfitting

→ Ordering of the word is getting changed

→ Out of vocabulary (OOV)

→ Semantic meaning is still not getting captured.

(3) N - grams (Not in video)

④ TF - IDF

(Term Frequency - Inverse Document Frequency)

S1: good boy

S2: good girl.

S3: boy girl good.

Term freq (TF) = $\frac{\text{No. of rep of word}}{\text{No. of words in sent.}}$

IDF = $\log_e \left[\frac{\text{No. of sentences}}{\text{No. of sentences containing the word}} \right]$

Term Frequency

	S1	S2	S3
good	1/2	1/2	1/3
boy	1/2	0	1/3
girl	0	1/2	1/3

IDF

words	
good	$\log_e(3/3) = 0$
boy	$\log_e(3/2)$
girl.	$\log_e(3/2)$

Page No.
Date

Final TF - IDF (Product)

	good	boy	girl
S1 :	0	$\frac{\log(3/2)}{2}$	0
S2 :	0	0	$\frac{1}{2}\log_e(\frac{3}{2})$
S3 :	0	$\frac{1}{3}\log_e(\frac{3}{2})$	$\frac{1}{3}\log_e(\frac{3}{2})$

Advantages

→ Intuitive

→ Fixed size → Vocab size

→ Word importance is getting captured.

Disadvantages

→ Sparsity still exists

→ ~~oov~~ OOV.

Word Embedding :-

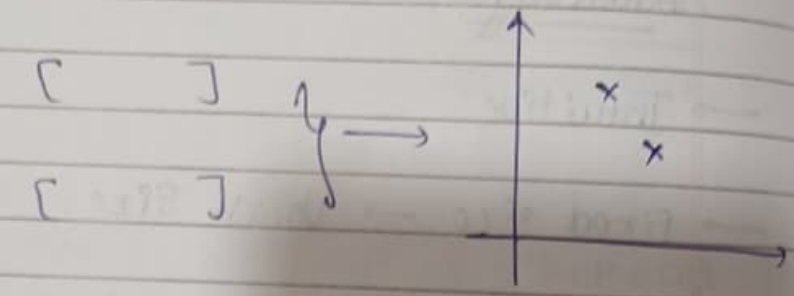
→ Used for the representation of words for text analysis, typically in the form of a real-valued vector that encodes the meaning of the word such that the words that are closer in the vector space are expected to be similar in meaning

↓

[Wikipedia def^n]

Happy = [ ]

Excited = [ ]



⇓

Similar words.

Word Embedding

↓

Count & Freq.

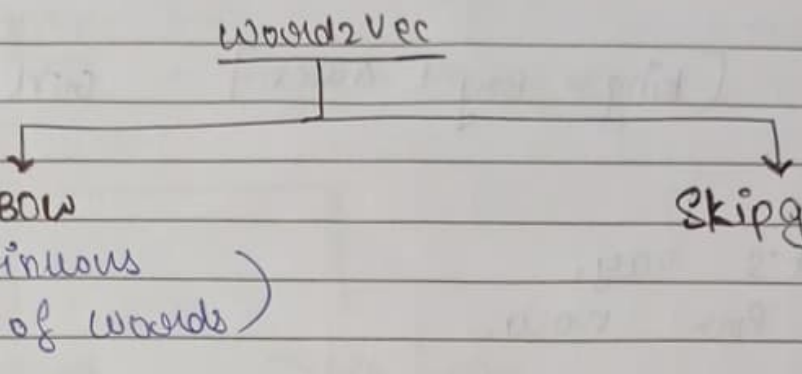
① OHE
② BOW
③ TF-IDF

Deep learning Trained model

⇓

[Word2Vec]

→ Deep learning trained models are very accurate and ~~have very~~ solve the disadvantages



Word2Vec :-

→ Technique for NLP published in 2013 → by Google

→ Uses Neural Network model to learn word associations from a large corpus of text.

→ Each distinct word with a vector.

• Feature representation

→ Vocabulary → unique words

Boy Girl King Queen Apple Mango

Features [300 dim]

Gender
Royal
Age
⋮
n

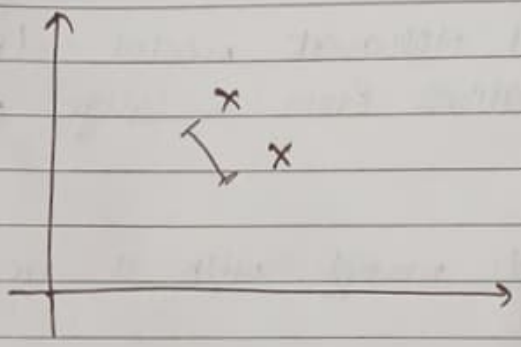
→ Each vector has a 300 dimension

[king - Boy + Queen] = Girl

let's say,
Back now,

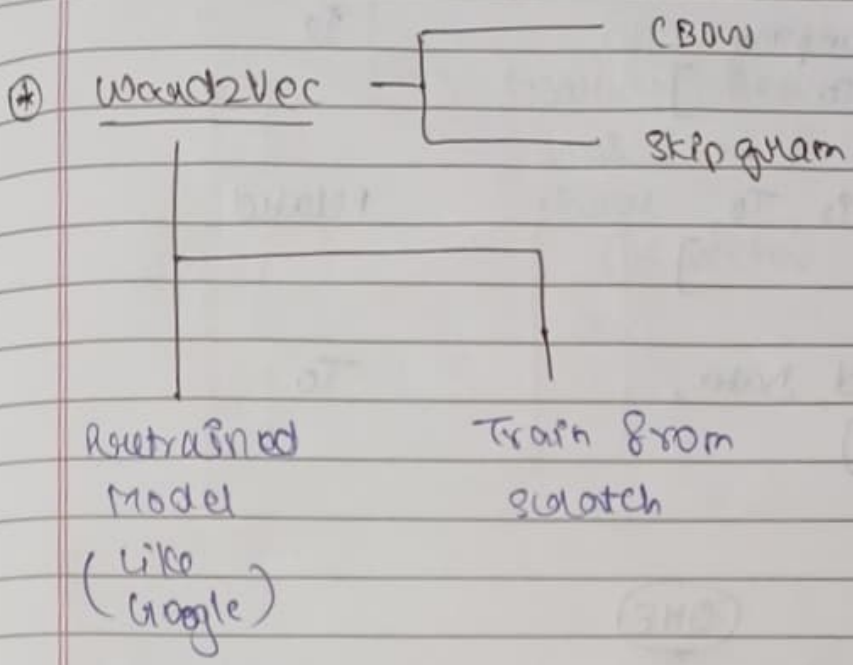
we just have 2 dimensions.

- Cosine Similarity



Distance = 1 - (Cosine Similarity)

~~Cosin~~
~~Cosine Similarity = cos 45°~~



① CBOW [Continuous Bag of Models]

CORPUS ÷ Dataset

["iNeuron Company is related To Data science"]

* Choose a window size

Say [window size = 5] (preferredly odd)

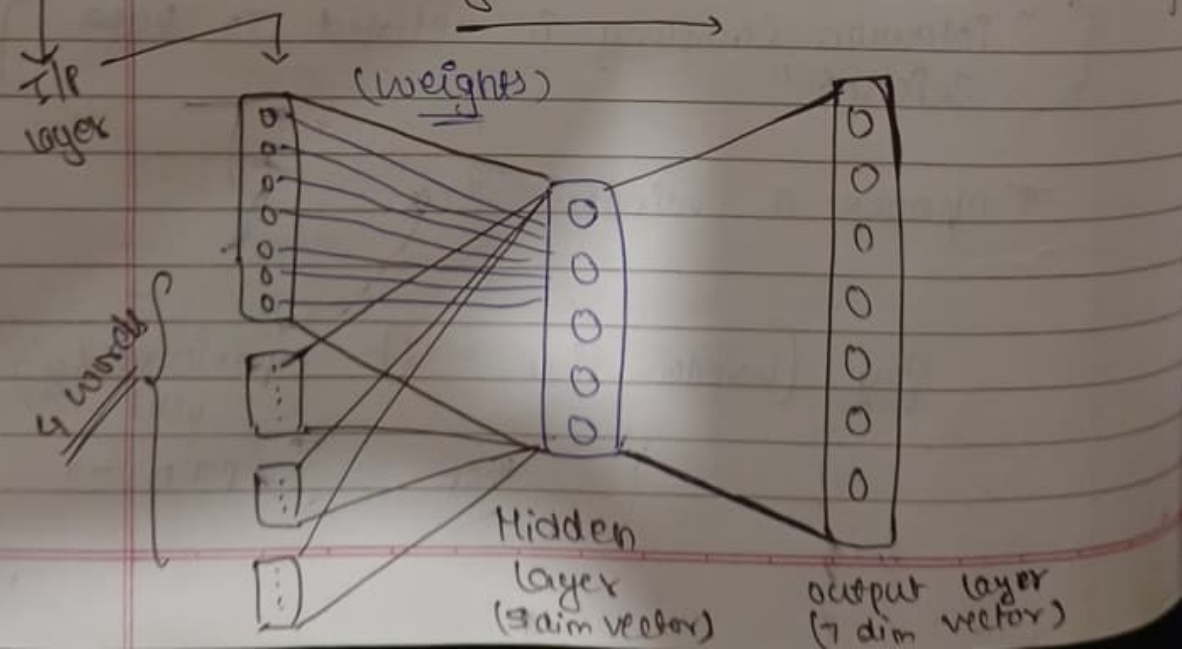
then PTO →

I/P	O/P
→ [INewton, Company Related To]	Is
→ [Company, Is, To, Data]	Related
→ [Is, Related, Data, Science]	To

OHE

INewton	[1 0 0 0 0 0 0]
Company	[0 1 0 0 0 0 0]
Related	[0 0 1 0 0 0 0]
to	[0 0 0 0 1 0 0]

CBOW { Fully connected Neural Network }



→ Window size = 5

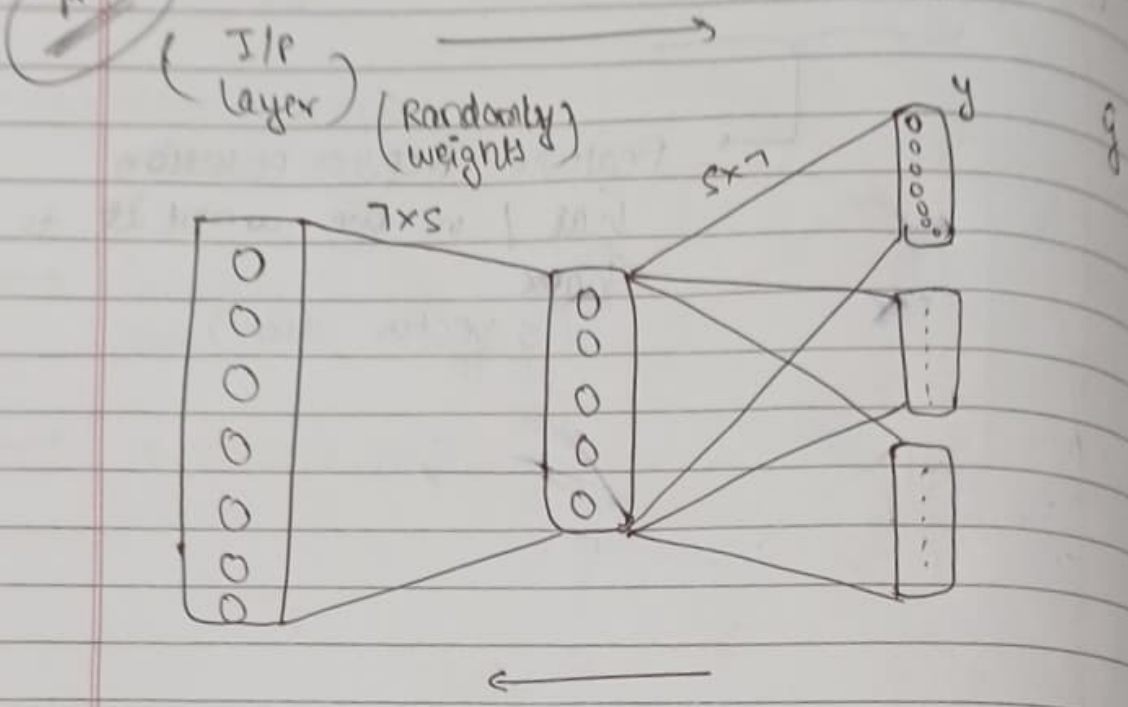
→ Feature Representation has / or we want it to have (5 vector dim)

Skipgram - word2vec

[window size = 5]

O/P	I/P
→ [Newton, Company.. .. Related, To]	IS
→	related
→	To

ANN



* When to apply / use CBOW or Skipgram

{ Small dataset — CBOW
Hugo dataset — Skipgram

* Improving

1. Increase training data
2. Increase the window size.
   ↳ Vector dimension is also increasing.

Google Word2Vec.

3 - billion words — Google News

feature representation of 300 dimension vector

cricket → [. . . . .]

⑦ Avg word2Vec.

[The food is good] → D1