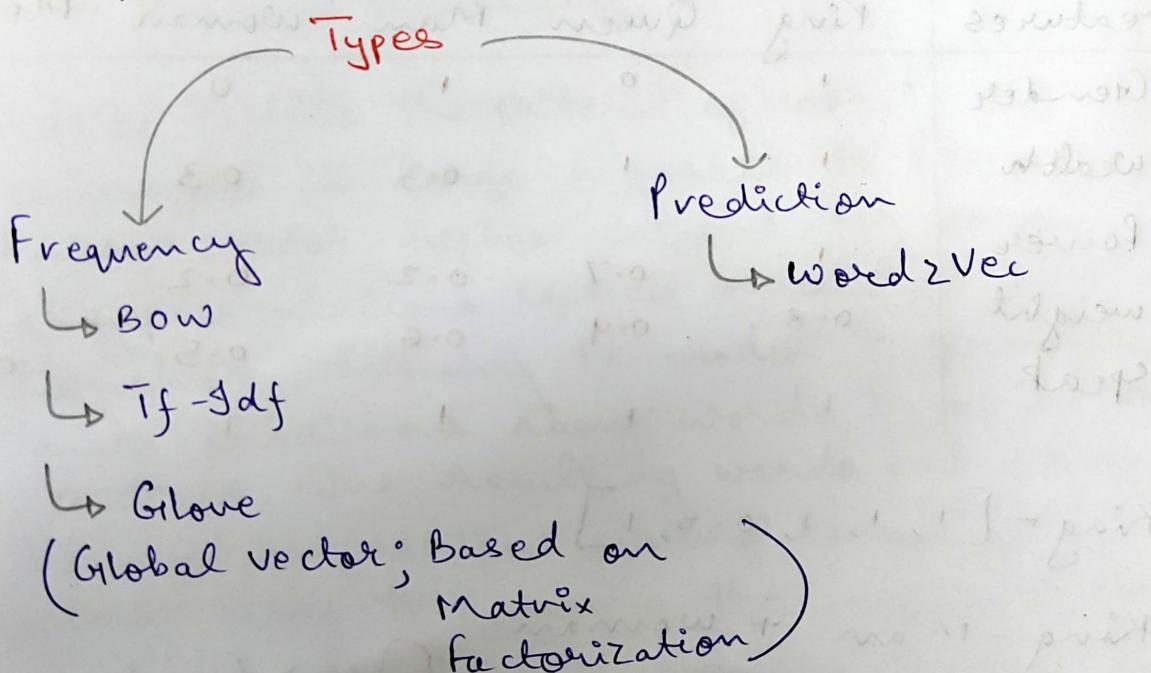


LECTURE-5

WORD-2-VEC

Word Embeddings

→ In NLP, word embedding is a term 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 that vector space are expected to be similar in meaning.



Word - 2 - Vec

DEFINITION

WORD 2 VECTOR

Pros: Carries semantic meaning, low dimension
 therefore, high computation, dense vector
 which means higher no. of non-zero
 values and thus no problem like zero
 everything is random and always talk
 test in math and test always talk
 always 3d or below, no song notes

~~Cons~~

Intuition

Features	King	Queen	Man	Woman	Monkey
Gender	1	0	1	0	1
Wealth	1	1	0.3	0.3	0
Power	1	0.7	0.2	0.2	0
weight	0.8	0.4	0.6	0.5	0.3
speak	1	1	1	1	0

$$\text{King} = [1, 1, 1, 0.8, 1]$$

$$\begin{aligned}
 & \text{King} - \text{Man} + \text{woman} \\
 & 1 - 1 + 0 = 0 \rightarrow \text{Gender} \\
 & 1 - 0.3 + 0.3 = 1 \rightarrow \text{wealth} \\
 & 1 - 0.2 + 0.2 = 1 \rightarrow \text{Power} \\
 & 0.8 - 0.6 + 0.5 = 0.7 \rightarrow \text{weight} \\
 & 1 - 1 + 1 = 1 \rightarrow \text{Speak}
 \end{aligned}$$

NOTE: for a large dataset, the features are made automatically through Neural Networks. So in that case we would not know the features but only the values.

↳ The intuition behind word2vec lies in the ~~data~~ idea that words with similar meanings are often used in similar context. By capturing these contextual relationships, word2vec aims to represent words as vectors in a continuous vector space, where the spatial proximity of vectors reflects semantic similarity.

↳ Word2Vec exploits the pattern of word co-occurrences in large datasets to learn continuous vector representations for words. These representations capture semantic relationships, allowing the model to generalize and make predictions about words based on ~~the~~ their contexts. The resulting word embeddings can be used in various natural language processing tasks to enhance the understanding of semantic relationships b/w ~~two~~ words.

↳ The underlying assumption of word2vec is that two words sharing similar contexts also share a similar meaning and consequently a similar vector representation from the model.

Types of Word2Vec

(a) CBOW (Continuous Bag of words)

Rummy problem → Solve → Vector → By product

→ CBOW, which stands for Continuous Bag of Words, is one of the architectures used in the Word2Vec model for generating word embeddings. CBOW aims to predict a target word based on its context, where the context is defined by the surrounding words in a given text.

Example - If we've a sentence;
"The cat sat on the mat", and the target word is "sat", the context window might be ["The", "cat", "on", "the", "mat"].

After training, the word vectors obtained from CBOW capture semantic relationships b/w words. Words with similar meanings or contextual usage have similar vector representations, allowing the model to generalize and make accurate predictions about words based on their context. CBOW is particularly useful when dealing with well-represented and frequent words in a large corpus.

b) Skip-Grams

Sentence - Watch Campus for data science

i/P	O/P
Campus	watch, for
for	Campus, data
data	for, science.

↳ Skip-Gram is another architecture used in the Word2Vec model for generating word embeddings. In contrast to the Continuous Bag of words (CBOW) model, skip-gram operates in the reverse manner. Instead of predicting the target word based on its context, skip-gram predicts the context words (surrounding words) given a target word.

↳ Skip-gram focuses on predicting the context words given a target word. This allows the model to learn representations that capture the contextual usage and semantic relationships of words. Skip-gram is often considered useful in scenarios where the target words are less frequent or when dealing with long range dependencies in language.

NOTE : Small dataset \rightarrow CBOW
Large dataset \rightarrow Skip Gram.

* How to increase efficiency of Word2Vec?

- ↳ Increasing the training dataset
- ↳ Increase dimension of vectors
- ↳ Increase window size.

not efficient
slow, requires
memory, not
scalable

In basic, word2vec algorithm can handle up to 1000 words per sentence. Below, mentioning few techniques which can help to handle large set of tokens.
1. Using word2vec with pre-trained embeddings for frequent words.
2. Clustering words with topics like news, sports, etc.
3. Using (cluster, pre-trained) above training.
4. Using hierarchical softmax.

It is not possible to use softmax over all words. So, it is better to split a corpus into multiple topic-wise sub-corpus. Then, each topic-wise sub-corpus can have its own word2vec. This allows us to use different softmax for different topics. This is called the parallel softmax method or topic-specific softmax. In this approach, words from one topic are mapped to a vector space specific to that topic. This allows us to handle large number of topics simultaneously.