Experiment No : Group 1-②

- **title :**

Perform bag-of-words approach , TF-IDF on data

- **Problem statement**

Perform bag-of-words approach (count occurrence normalized count occurrence) , TF - IDF on the data. create embeddings using Word2vec.

- **course objective :**

To understand bag-of-words using Word 2 vec.

- **course outcome :**

CO3: Design and develop applications on the subjects of their choice.

- **software and hardware requirements :**

Sr. no.	software / hardware requirements	specifications.
1.	Laptop / Desktop	64-bits, 8 GBRam.
2.	Jupyter Notebook	version 7.3.3.

- ## Theory:

What are Word Embedding:

1. It is an approach for representing words and documents. Word embedding word vectorisa numeric vector input that represent a word in lower- dimension space.

2. It allows words with similar meaning to have a similar representations.

3. They can also approximate meaning. A word vector with 50 values can represent 50 unique features.

4. features: anything that relates word to one another. E.g: Age, sports, fitness, Employed etc. each word vector has values corresponding to these features.

Implementation of Word Embedding:

1. Word embeddings are a method of the extracting features out of text so that we can input those features into a machine learning model to work with text data.

2. They try to preserve syntactical and semantic information.

3. The model methods such as Bag of words (BOW), count vectorizer and TFIDF rely on the word count in a sentences but do not save any syntactical or semantic information.

4. In these algorithms, the size of the vector is the number of elements in the vocabulary.

5. we can get as parse matrix if most of the elements are zero.

6. Large input vectors will mean a huge number of weights which will result in high computation required for training. word embeddings give a solution to those problems.

Method Word2vec:

1. Word2vec is one of the most popular technique to learn word embeddings using a two-layer neural network.

2. Its input is a text corpur sand in output is a set of vectors.

3. Word embedding via. word2vec can make a natural language computer-readable, then the further implementation of mathematical operations on words can be used to detect their similarities.

4. A well-trained set of wordvectors will place similar words close to each others in that space.

5. for instance, the words women, men, and human might cluster in one corner, while yellow, red and blue cluster together in another.

6. There are two main training algorithms for word2vec, one is the continuous bag of word ((BOW), another is called skip-gram.

7. The major different between the set methods is that CBOW is using context to predicta-target word while-skip-gram is using a word to predict. a target context.

8. Generally, the skip-grammer method can have a better performance compared with CBOW method, for it can capture two semantics for a single word.

Method Bag of Words:

1. Bag of words is a natural language processing technique of text modelling.

2. In technical terms, we can say that it is a method off feature extraction with text data.

Implementations :

Method Word2vec:

Gensim python library introduction Gensim is an open source python library for naturally language processing and it was developed. and is maintain by the czench natural language processing researcher.

Step implementation of word Embedding with Gensim

1. Install gensim.
2. Download data.
3. Data Preprocessing
4. Gensin word2vec model training.

method Bag of words:

Technique to build Bag of words:

1. count occurrence using count vectorize
2. Normalized count occurrence using vectorizer
3. TF-IDF using vectorizer.
4.

steps in implementation of Bag of word techniques

1. Download data from kaggle.
2. cleansing
3. Data processing
4. Model Building
5. Pipeline.

Page No.
Date

- Conclusion :

In this practical, I have perform bag-of-word approach. TF-IDF on the data. The Bag-of-word TF-IDF and word2vec techniques transform text data into numerical representations, with word2vec providing richer semantic context through learned word embeddings.