

- Aim :

Implement the continuous Bag of words (CBOW) model.  
Stages can be:

- a. Data preparation
- b. Generate Training data
- c. Train model
- d. Output

- Course objectives

1. To be able to apply deep learning algorithms to solve problem of moderate complexity.

- Course outcomes

C01 : Learn and use various deep learning tools and packages

C02 : Build and train deep neural network models for use in various applications.

- Softwares and hardwares requirements

sr.no.	Hardware and software	version / specification
1.	Jupyter notebook	v.7.13-008
2.	Computer / PC	IS version, 64 bits, 8 GB RAM.





26

## Theory

Steps / Algorithm of implementing continuous Bag of word

### Step 1

Import libraries

### Step 2

Tokenize the every word from the paragraph.  
you can call it built tokenization present in a  
Gensim

### Step 3

Fit the data to tokenization

### Step 4

Find total no. of words and total no. of sentences

### Step 5

Generate the pairs of context words and a target words.

### Step 6

Create neural network model

EDUCATIONAL USE

### Step 7

Create vector file or some word for testing

e.g: dimensions = 100

```
vect_file = open('context/gdrive/my drive/vector.txt', 'w')
vect_file.write('{3 {3\n'.format(total_vocab, dimensions))
```

### Step 8

Assign weight to your trained model.

e.g: weights = model.get\_weight()[0]

```
for text in vectorize.word_index.items():
    final_vec = ''.join(map(str, list(weights[i,:])))
    vect_file.write('{3 {3\n'.format(text, final_vec))
```

```
final_vec = ''.join(map(str, list(weights[i,:])))
vect_file.write('{3 {3\n'.format(text, final_vec))
close()
```

### Step 9

Use the vectors created Gensim

e.g. cbow-output

### Step 10

choose the word to get similar type of words:

cbow-output.most-similar(positive=['your word'])

## Word embedding related to NLP

- Word embedding are numerical representation of word in a continuous vector space, capturing their semantic and syntactic meanings.
- This approach allows similar words to have similar vector representations, enhancing understanding in NLP tasks.
- Key method includes Word2vec, which uses the Continuous Bag of word (CBOW) and skip-gram model to predict words based on context or a surrounding words.
- These embedding facilitate various application such as text classification and document clustering by a reducing dimensionality and preserving meaning.

## What are main applications of word embedding in NLP

### 1. Text classification

Used for tasks like sentiment analysis and topic categorization by capturing semantic meaning of words.

## 2. Named Entity Recognition

Helps identify and classify entities such as people and organizations - by understanding word in a relationships.

## 3. Machine Translation

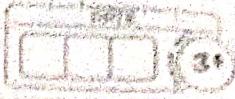
facilitate translation by representing words in a language-agnostic manner, improving semantic understanding

## 4. Question Answering

Improves systems ability to understand context and relationships in posed questions

## 5. Information Retrieval

Enhance search engines by accurately matching queries with relevant documents.



- Conclusion

In this practical, I have learn about the GAN model predicts a target word from its context, and generating word embeddings that capture semantic relationships, useful for tasks like similarity and the clustering.