LAB Manual

PART A

(PART A : TO BE REFFERED BY STUDENTS)

**Experiment No.06**

**A.1 Aim:**

**Introduction to vector space models**

1. **Constructing Distributional Semantic Model**
2. **Text representation using pre-trained word embedding models.**
3. **Training embedding’s using gensim**

**A.2 Prerequisite:**

1. Understanding of fundamental programming functions/commands and environment of Python.

**A.3 Outcome:**

**After successful completion of this experiment students will be able to**

1. Explore and understand distributional semantic models
2. Importance of word embedding

**A.4 Theory:**

**A.4.1 Distributional Semantic Model**

* Represent the meaning of words as vectors keeping track of the words’ distributional history
* Focus on the notion of semantic similarity, measured with geometrical methods in the space inhabited by the distributional vectors
* Are intended as general-purpose semantic models that are estimated once, and then used for various semantic tasks, and not created ad-hoc for a specific goal.
* It follows that model estimation phase is typically unsupervised
* Pre-process the source corpus
* Collect a co-occurrence matrix (with distributional vectors representing words as rows, and contextual elements of some kind as columns/dimensions)
* Transform the matrix: re-weighting raw frequencies, dimensionality reduction
* Use resulting matrix to compute word-to-word similarity

# **A.4.2 Understanding Model with steps**

#### Step-1: Fetch Data

#### Step-2: Preprocessing

#### Step-3: Count Vector Representation

from sklearn.feature\_extraction.text import CountVectorizer

#### Step-4: Convert into matrix table

import pandas **as** pd

names **=** cv.get\_feature\_names() *# This are the entity names (i.e. keywords)*

#### Step-5: Dimensionality Reduction

#### Step-6: Cosine similarity

from sklearn.metrics.pairwise import cosine similarity

**A.4.3 Word Embedding**

**Word Embedding** is a language modeling technique for mapping words to vectors of real numbers. It represents words or phrases in vector space with several dimensions.

### **What is Word2Vec?**

Word2Vec is a widely used method in natural language processing (NLP) that allows words to be represented as vectors in a continuous vector space. Word2Vec is an effort to map words to high-dimensional vectors to capture the semantic relationships between words, developed by researchers at Google. Words with similar meanings should have similar vector representations, according to the main principle of Word2Vec. Word2Vec utilizes two architectures:

**gensim**:

pip install nltk  
pip install gensim

* **GENSIM:**Gensim is an open-source Python library that uses topic modelling and document similarity modelling to manage and analyse massive amounts of unstructured text data. It is especially well-known for applying topic and vector space modelling algorithms, such as Word2Vec and Latent Dirichlet Allocation (LDA), which are widely used.

from gensim.models import Word2Vec

import gensim

**A.5 Procedure/Algorithm:**

**A.5.1 TASK 1:**

1. Create a new python file.
2. Read input of texts and Pre-process text data
3. Extract features from text data.
4. Python code to find the similarity measures.
5. Word embedding on pre-trained model
6. Training embedding’s using gensim
7. Complete PART B of lab manual.
8. Save and close the file and name it as **EXP5\_ your Roll no.py**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PART B

(PART B : TO BE COMPLETED BY STUDENTS)

***(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case there is no Black board access available)***

|  |  |
| --- | --- |
| Roll No. | Name: |
| Class : | Batch : |
| Date of Experiment: | Date of Submission |
| Grade : |  |

**B.1 Software Code written by student:**

***(Paste your Matlab code completed during the 2 hours of practical in the lab here)***

**B.2 Input and Output:**

***(Paste your program input and output in following format, If there is error then paste the specific error in the output part. In case of error with due permission of the faculty extension can be given to submit the error free code with output in due course of time. Students will be graded accordingly.)***

**B.2.1 TASK 1**

**Input texts: User input data**

**Output:**

1. Create a new python file.
2. Read input of texts and Pre-process text data
3. Extract features from text data.
4. Python code to find the similarity measures.
5. Word embedding on pre-trained model
6. Training embedding’s using gensim

**B.3 Observations and learning:**

***(Students are expected to comment on the output obtained with clear observations and learning for each task/ sub part assigned)***

**B.4 Conclusion:**

*(****Students must write the conclusion as per the attainment of individual outcome listed above and learning/observation noted in section B.3)***

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*