AY: 2025-26

Class:	BE	**Semester:**	VII
Course Code:	CSDOL7011	**Course Name:**	Natural Language Processing

Name of Student:	Konisha Jayesh Thakare
Roll No. :	71
Experiment No.:	8
Title of the Experiment:	Measuring Semantic Similarity Between Sentences using Sentence Transformers
Date of Performance:	02.09.2025
Date of Submission:	09.09.2025

Evaluation

Performance Indicator	Max. Marks	Marks Obtained
Performance	5	
Understanding	5	
Journal work and timely submission	10	
Total	20	

Performance Indicator	Exceed Expectations (EE)	Meet Expectations (ME)	Below Expectations(BE)
Performance	4-5	2-3	1
Understanding	4-5	2-3	1
Journal work and timely submission	8-10	5-8	1-4

Checked by

Name of Faculty : Dr. Tatwadarshi P. Nagarhalli

Signature :

Date :

Aim: To measure semantic similarity between sentences using Sentence Transformers, a modern NLP technique for generating contextual embeddings at the sentence level.

Theory:

Traditional word embeddings such as Word2Vec or GloVe capture semantic meaning of individual words but fail to represent the meaning of an entire sentence. Sentence Transformers address this by generating dense vector representations of sentences, preserving semantic and contextual information.

Key concepts:

- **Sentence Embeddings:** Fixed-size vector representations capturing sentence meaning.
- **Cosine Similarity:** Measures the closeness of two vectors; values range from -1 (opposite meaning) to 1 (identical meaning).
- **Sentence Transformers Library:** Built on top of pre-trained transformer models (e.g., BERT, RoBERTa) optimized for sentence embeddings.

Applications of sentence similarity include:

- Information retrieval
- Semantic search
- Duplicate question detection
- Paraphrase identification

Procedure:

1. Install and import the sentence-transformers library.
2. Prepare a set of sample sentences for comparison.
3. Load a pre-trained Sentence Transformer model.
4. Convert sentences into embeddings.
5. Compute cosine similarity between sentence embeddings.
6. Interpret similarity scores to determine semantic closeness.

Algorithm:

1. Import the SentenceTransformer class and cosine_similarity function.
2. Load a pre-trained model (e.g., all-MiniLM-L6-v2).
3. Encode sentences into vector embeddings.
4. Compute pairwise cosine similarity between sentence vectors.
5. Output similarity scores.

Implementation:

```
from sentence_transformers import SentenceTransformer
from sklearn.metrics.pairwise import cosine_similarity
sentences=["The cat is sitting on the mat.","A dog is lying on the
carpet.","The feline rests on the rug."]
model=SentenceTransformer('all-MiniLM-L6-v2')
embeddings=model.encode(sentences)
similarity_matrix=cosine_similarity(embeddings)
print("Pairwise Cosine Similarity Matrix:\n")
for i,row in enumerate(similarity_matrix):
    for j,sim in enumerate(row):
            print(f"Similarity between sentence {i+1} and sentence {j+1}:
{sim:.3f}")
```

Output:

```
Pairwise Cosine Similarity Matrix:

Similarity between sentence 1 and sentence 1: 1.000
Similarity between sentence 1 and sentence 2: 0.370
Similarity between sentence 1 and sentence 3: 0.496
Similarity between sentence 2 and sentence 1: 0.370
Similarity between sentence 2 and sentence 2: 1.000
Similarity between sentence 2 and sentence 3: 0.496
Similarity between sentence 3 and sentence 1: 0.496
Similarity between sentence 3 and sentence 2: 0.496
Similarity between sentence 3 and sentence 3: 1.000
PS C:\Users\Konisha Thakare\OneDrive\Desktop\Python>
```

Conclusion:

This experiment demonstrated the use of Sentence Transformers to measure semantic similarity between sentences. By generating dense, context-aware embeddings, the model captures the meaning of entire sentences rather than individual words. Cosine similarity between embeddings provides a numerical measure of semantic closeness, enabling applications such as paraphrase detection, semantic search, and information retrieval. The results show that sentences with similar meanings have higher similarity scores, confirming the effectiveness of sentence embeddings in modern NLP tasks.