Asian Institute of Technology (AIT)

Artificial Intelligence: Natural Language Understanding (NLU)

Assignment 1 (A1)

Submitted To:

Dr. Chaklam Silpasuwanchai

Assistant Professor

Computer Science and Information Management

Asian Institute of Technology

chaklam@ait.asia

Submitted By:

Tisa Bajracharya

St126686

GitHub Link: https://github.com/bajratisa/NLP_assignment

This project focuses on building a smart search engine that understands the meaning behind words rather than just matching text. By training models to represent words as mathematical vectors, we can calculate how "similar" different pieces of information are based on their context.

1. Scope of Project

The goal of this project is to create a semantic search system. Unlike a basic search that looks for exact words, this system uses "word embeddings" to find contextually related information. It includes training custom models, evaluating them against human intuition, and deploying a web-based search tool.

2. Training Data

We used the Reuters-21578 corpus for training. This is a famous collection of news stories from 1987, commonly used in computer science to teach machines how to process language.

- **Citation:** Lewis, D. D. (1997). Reuters-21578 Text Categorization Test Collection. https://www.nltk.org/book/ch02.html
- **Evaluation Data:** We used the WordSim353 Crowd dataset to check if our models "thought" like humans by comparing our results to human-assigned similarity scores. https://www.kaggle.com/datasets/julianschelb/wordsim353-crowd?resource=download

3. Workflow

The project follows a step-by-step path from raw text to a working website:

1. **Preprocessing:** We cleaned the news text by removing punctuation and making everything lowercase so the computer doesn't get confused.
2. **Training:** we built and trained Skip-gram and Negative Sampling (NEG) models to learn word relationships.
3. **Evaluation:** We tested these models using Mean Squared Error (MSE) and Spearman Correlation to see how accurate they were.
4. **Application:** Finally, we built a web app that takes a user's query and finds the best matching story from the news data using the "Dot Product" math.

4. Libraries Used

- **NumPy:** Used for all the heavy math and handling the word vectors.
- **PyTorch:** The engine used to build and train the neural networks for our models.
- **Scikit-learn:** Specifically used to calculate the similarity between vectors and evaluate errors (MSE).
- **Gensim:** Allowed us to load high-quality, pre-trained word vectors to compare against our custom ones.
- **Plotly:** Used to build the web interface because it allows us to create a website using only Python.
- **Flask:** Flask is the underlying web framework that handles the server logic.

5. Models Used: Pros & Cons

Model	**Advantages**	**Disadvantages**
Skip-gram	Excellent at understanding the context of rare words.	Slower to train because it looks at every single word pair.
Negative Sampling (NEG)	Much faster than Skip-gram because it only looks at a few "wrong" examples at a time.	Can be slightly less accurate if you don't pick enough negative samples.
GloVe (Pre-trained)	Very high accuracy because it was trained on billions of words.	Not "custom" to our specific news data; it’s a general-purpose model.

6. Conclusion

This project demonstrates how we can turn language into math to compare how models "understand" word relationships. By testing our custom **Skip-gram** and **NEG** models against a professional **GloVe** model, we observed that while all models can rank word similarities, their accuracy varies significantly based on training data and algorithms. The final web application provides a clear way to see these differences in real-time, successfully ranking the top 10 most similar words for any given input based on the chosen model's learned logic.