Word Embeddings

Deadline: 11:59pm, 31st October

1 Introduction

In modern Natural Language Processing (NLP), distributional semantic models—commonly referred to as word embedding algorithms—play a crucial role in generating meaningful numerical representations for words. The goal of these algorithms is to represent words in a mathematical space such that words with similar meanings are located near each other. Word embeddings can be categorized into two main types: frequency-based and prediction-based.

This task focuses on **frequency-based embeddings**, specifically using methods such as **Co-occurrence Matrix** and **Singular Value Decomposition (SVD)**. These methods offer a statistical approach to word vectorization, capturing the relationships between words in a given context. The task is to generate word embeddings using SVD and evaluate word similarities based on the derived embeddings.

2 Task Description

2.1 Constructing the Co-occurrence Matrix

A simple and effective way to represent words is through a co-occurrence matrix, where each row corresponds to a target word and each column to a context word. The matrix entries indicate how often a target word appears with context words within a specified window size. This method captures the syntactic and semantic relationships between words.

Create a co-occurrence matrix for the given vocabulary of words obtained from the data. Consider the context window size of your choice. [Ignore the class indices in the data and just consider the sentences here.]

2.2 Dimensionality Reduction Using SVD

While the co-occurrence matrix provides valuable insights, it can grow very large and sparse when dealing with large vocabularies. To make the matrix more manageable and extract meaningful word-embeddings, Singular Value Decomposition (SVD) is applied to reduce its dimensionality.

SVD decomposes the matrix M into three matrices:

$$M = U\Sigma V^T$$

where U represents the target word matrix, Σ contains the singular values, and V^T represents the context word matrix. This transformation allows for the generation of dense word embeddings by reducing the dimensionality while preserving the relationships between words.

Perform SVD decomposition of the co-occurrence matrix obtained above to reduce dimensionality. To perform this decomposition, the `svds` function from Python libraries such as `scipy` can be used, or any equivalent method. The resulting U matrix provides low-dimensional embeddings for each word in the vocabulary, effectively capturing the semantic relationships between words in a compact form.

2.3 Evaluating Word Embeddings

After generating word embeddings using SVD, you can measure the similarity between words using techniques like cosine similarity. Calculate the similarity for 5 word pairs of your choice using the obtained embeddings. Ensure that the selected word pairs represent different relationships, such as synonyms, antonyms, and other categories, to provide a comprehensive evaluation.

2.4 Analysis

- **Analyze Similarity Scores**: After obtaining word embeddings using SVD, calculate the similarity scores for selected word pairs. In general, higher scores indicate stronger semantic relationships, while lower scores suggest weaker associations.
- **Context Window Variations**: Experiment with at least two different context window sizes to understand how they impact the embeddings and similarity scores.
- **Compare Results**: Report the similarity scores for the chosen word pairs across different context window sizes and analyze any notable differences.

3 Submission Format

Submit a zip file containing the analysis report and the code, named as `RollNo_A6.zip`, on Moodle.

4 Resources

Refer to these resources for further reference:

1. https://cla2019.github.io/embedmatrix.pdf
2. https://web.stanford.edu/ jurafsky/slp3/6.pdf