PPMI Matrix and Word Vectors

In the first part of this assignment, I first constructed the co-occurrence matrix, where C[w,c] quantifies the frequency of word w occurring with context c within sentences. After its initialization, this matrix undergoes a two-fold transformation: multiplication by 10, mimicking enhanced visibility of each sentence, and smoothing through the addition of 1 to every cell to mitigate the impact of zero counts.

When the Positive Pointwise Mutual Information (PPMI) matrix is used as a weighted count matrix, it does enhance the representation of the word "dogs" in comparison to the original count matrix. PPMI effectively refines the count matrix by emphasizing meaningful co-occurrences over common or less informative ones. This reweighting process mitigates the influence of frequent but semantically sparse words. PPMI assists by allocating higher weights to word-context pairs that occur more frequently than expected by chance, which signifies a stronger association or relevance between the pair. It also diminishes the weights of pairs that occur together as frequently as or less than random chance. So PPMI in fact facilitates a more nuanced and semantically rich representation of words within the vector space.

The Euclidean distances computed both before and after applying Singular Value Decomposition (SVD) to the PPMI matrix indeed align with the principles of distributional semantics—that words appearing in similar contexts tend to have similar meanings. Prior to SVD, the distances reflect the raw co-occurrence frequencies, with "women" and "men" (similar in context) having a lower distance compared to pairs like "women" and "dogs," which are contextually more distant. This already suggests that similar words cluster closer together, even in the unrefined, high-dimensional space. The euclidean distances are below for reference.

Euclidean distances before SVD	Euclidean distances after SVD
women - men: 31.622776601683793 women - dogs: 53.85164807134504 men - dogs: 38.72983346207417 feed - like: 14.142135623730951 feed - bite: 42.42640687119285 like - bite: 31.622776601683793	women - men: 0.0990214880887122 women - dogs: 0.7150679237473749 men - dogs: 0.6262006606365683 feed - like: 0.26330238890636665 feed - bite: 0.7331034036315125 like - bite: 0.5970953455295674

After applying SVD and dimensionality reduction, the distances between word pairs become significantly smaller, illustrating that the reduced space more efficiently captures and emphasizes the essence of semantic relationships. For instance, the dramatic decrease in distance between "women" and "men" post-SVD to 0.099 emphasizes their close semantic relationship even more clearly. Similarly, verbs related to actions and their contextual alignment ("feed" vs. "like/bite") exhibit distances that resonate with intuitive semantic relationships.

Synonym Test

The objective was to assess the ability of these vector sets to identify synonyms among approximately 900 verb pairs, structured within a test comprising 1,000 multiple-choice questions. Each question was designed to present a target verb alongside four distractors and one true synonym. The evaluation hinged on the accuracy of synonym selection using Euclidean distance and cosine similarity measures.

Results revealed differential performance between the two vector sets. The classic vectors, processed through our own pipeline, achieved a synonym detection accuracy of 67.7% for both Euclidean distance and cosine similarity. In contrast, the word2vec vectors, despite their deep learning origins, exhibited lower accuracy rates at 51.5% for Euclidean distance and 56.6% for cosine similarity.

This outcome suggests that the methodology applied in creating the classic vectors, particularly the incorporation of PPMI weighting and dimensionality reduction via SVD, significantly enhances their semantic discernment capabilities. The performance disparity underscores the effectiveness of these techniques in capturing and leveraging the nuances of semantic relationships within the vector space, an advantage seemingly less pronounced in the pre-trained word2vec vectors despite their deep learning foundation.

Table: Synonym Test Results

Vector Type	Euclidean Distance Accuracy	Cosine Similarity Accuracy
Classic	67.7%	67.7%
Word2Vec	51.5%	56.6%

The reason word2vec didn't do as well as the classic vectors on the synonym task might be because of how and what they were trained on. Since Word2vec models are trained on bigger datasets like Google News, they might be getting a lot of varied information and can understand more relationships between words. But, this might also make them less sharp when it comes to picking out synonyms specifically, because they have a lot of information they are juggling at the same time.

SAT Analogy Test

I started by parsing the SAT analogy questions from 'SAT-package-V3.txt'. This involved filtering out headers and irrelevant information to focus solely on the content of the analogy questions. I identified "stem pairs" and "choice pairs" and noted the correct answer for each. This allowed me to effectively set up each question for evaluation.

To evaluate the analogies, I decided to use vector subtraction to quantify the relationship between words in a pair. My thinking was that the difference between vectors would capture the relational nuances between words. For each question, I calculated the vector difference between the stem pair and compared it with the differences of each choice pair using cosine similarity. This method seemed logical, considering cosine similarity's effectiveness in measuring the orientation and similarity between vectors.

The results were interesting, with an accuracy of about 30% for both Classic and word2vec vectors. This was slightly better than the baseline expectation of 20% accuracy for random guesses and aimed for the assignment. It was intriguing to see that the Classic vectors, despite their traditional foundation, performed comparably to the deep learning-based word2vec vectors. This outcome highlighted that both types of vectors could capture essential semantic relationships to some extent.

However, the results also made it clear that capturing and applying analogical relationships through vector arithmetic is challenging. The accuracies achieved indicate that there's a lot more to understanding language nuances that current vector representations might not fully capture.

Moving forward, exploring more sophisticated models or neural network architectures might offer deeper insights into semantic nuances but this was a nice start.