# Comparing Word-Embeddings: PPMI and Lower-Rank Approximation

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

The meaning of a word can vary across extralinguistic contexts. To capture these phenomena, various types of word embedding have been proposed. In this project, we will use Singular Value Decomposition (SVD) and Positive Pointwise Mutual Information (PPMI) to transform natural language (English) into vector representations and compare their performances on MEN similarity tasks.

## 1 Introduction

Representing words with vectors is critical for NLP researchers to study the semantics and syntactics of natural languages. Word embedding techniques assign a vector representation for each word. There are two traditional methods to construct word embeddings: PPMI and a low-dimensional approximation of the PPMI embeddings learned via singular value decomposition. These methods are "count-based", as opposed to neural-network based models like skip-gram with negative-sampling training method (SGNS). "Count-based" models compute a co-occurrence matrix where the (i, j) entry, $X_{i,j}$ , of co-occurrence matrix X is the number of times word i appeared in the same context window with word j in the corpus. These bodies of work also assume that words occurring together have similar meanings.

### 1.1 Research Questions

In this project, we seek to test the performances of PPMI and SVD embeddings on word similarity tests using human raters' scores as the ground truth. We apply the two methods to three large-scale diachronic datasets (patent data, lyrics and news), each with over 10k unique tokens. To evaluate the embeddings, we select a list of words and extract their nearest neighbors in the embedding space to compare their semantic similarities using human judgement. Theoretically, these neighbors can be viewed as the synonyms for the target word. Our goal is to see which embedding (PPMI or SVD) has a stronger correlation with MEN similarity scores across three domain-specific corpus.

## 2 Data

The first set of data includes over 18,000 lyrics from Spotify and Billboard Top 100 from 1950 to 2020. The second source of data includes 581,684 historical patent data scraped from USPTO API3. Lastly, we use the news dataset from the Corpus of News on the Web (NOW).

We purposefully choose these three corpora to represent the unique challenge when applying general pertained embeddings on domain-specific corpora. Compared to the NOW dataset, patent and lyric data contain industry jargons or slangs/expressions that require cultural knowledge to understand. We hope to contrast SVD and PPMI methods for word representations on both generic news corpus and domain-specific corpus, to draw recommendations on which method would be more appropriate in different thematic contexts.

Calculating PPMI requires a word oc-currence matrix that counts the number of times each pair of unique words appear in a context window.This process is computationally exhausting as the size of the matrix is the squared number of distinct vocabularies. Therefore, due to computational limitations, we sampled a subset of songs/news/patent abstracts from each corpus for co-occurrence matrix and subsequent embeddings. For the Lyrics dataset, we build a model by randomly sampling 1,500 songs. We use the downsampled data to calculate its co-occurrence matrix. For the NOW data, we randomly sampled 100 news for each year.

For all datasets, sentences are stripped of punctuation and broken into tokens. Tokens are lower-cased and stemmed, and stop words are removed. We use the English stop words list and the Snowball Stemmer from Python's NLTK package.

## 3 Methods

### 3.1 Co-occurrence Matrix

We calculate the word co-occurrence matrix by first extracting the unique tokens in the corpus and constructing an empty matrix with the size of the number of unique tokens. We then loop through all sentences in a corpus, sliding a context window through each sentence and counted the number of times a pair of words occurred in the same context window. We iteratively update the matrix as it went through all words in the corpus. The window length is default to 2, meaning only words appearing back to back are counted as co-occurring.

### 3.2 PPMI

PPMI represents the vector embedding for each word containing the positive point-wise mutual information values between the word and a large set of pre-specified context words (Hamilton et. al, 2018). Since raw word frequency is quite skewed and captures less informative words such as "the" and "of", PPMI evaluates whether a context word is particularly informative about the target word. Pointwise mutual information (PMI) ranges from negative infinity to infinity. Negative values represent the co-occurrence is less than we expected by chance, whereas the positive values are more commonly used to evaluate the relatedness between two words. Clipping the PPMI values above zero ensures they remain finite and has been shown to dramatically improve results. We calculated the PPMI matrix from the co-occurrence matrix by taking the log ratio of the probability of word i given word j divided by probability of word i and clipped the value above 0.

$$M_{i,j}^{PPMI} = max\{log(\frac{P(w_i|w_j)}{P(w_i)}), 0\}$$

### 3.3 SVD PPMI

SVD embeddings perform singular value decomposition on PPMI embeddings to extract a low dimensional approximation of the PPMI matrix (Levy et al., 2015).We used truncated SVD that calculates the optimal rank d factorization with respect to L2 loss. We use the truncated SVD to transform PPMI matrix M into the product of three matrices $U \cdot \Sigma \cdot V_t$, where U and V are orthonormal and $\Sigma$ is a diagonal matrix of eigenvalues in decreasing order (Eckart Young, 1936). By keeping only the top d (default to 100) elements of $\Sigma$, we obtain $M_d = U_d \cdot \Sigma_d \cdot V_d^T$. The vector embedding for word wi is computed as the dot product between U and Sigma. We added an eigenvalue weighting parameter $\gamma$ on $\Sigma$ and set it to be 0.5.

$$w_i^{SVD} = (U_d \cdot \Sigma_d^{\gamma})_i$$

A value under 1 has been empirically shown to improve performances on semantic tasks (Turney Pantel, 2010; Bullinaria Levy, 2012). As the dimensionality reduction functions as a form of regularization, we expect the low-dimensional representation of PPMI to be more robust in similarity tasks(Hamilton et.al, 2016).

### 3.4 Word Similarity Evaluation

One way to evaluate the quality of the embeddings is to perform a word similarity test. Usually, this is done by comparing the cosine similarities between a pair of word vectors and the human rated similarity scores. In this project, we use the similarity scores from the MEN dataset that consists of 3,000 word pairs and their human ratings. For each corpus, we first extract word pairs from our tokens that also appear in the MEN dataset. We compute the cosine similarities between the pair of words, which is the normalized dot product between two word vectors.

$$cos(w_i, w_j) = \frac{w_i \cdot w_j}{||w_i||||w_j||}$$

We then calculate the correlations (Spearman's) between human ratings and the cosine similarities computed from embeddings. A higher correlation coefficient implies that the word embedding is able to represent the semantic similarity closer to how human raters do. Therefore, we are able to quantitatively compare how the 6 embeddings across three domain-specific corpora perform in capturing meanings.

## 4 Findings

We see that across all corpora, PPMI embeddings have stronger positive correlations with human ratings than SVD embeddings in the MEN similarity test. The difference between SVD and PPMI is particularly evident in the patent dataset. From Figure 1.and Figure 2, we can see that the distribution of PPMI centering around 0.1 while the distribution of SVD is sparser.

Table 1: Word Similarity across Three Corpus

| Corpus | Correlation of PPMI Similarity | Correlation of SVD similarity ($\mu$m) |
|---|---|---|
| Patent | 0.39(p=3.9e-29) | 0.19(p=1.1e-08) |
| News | 0.19(p=4.8e-58) | 0.14(p=9.9e-30) |
| Lyrics | 0.14(p=4.5e-11) | 0.06(p=0.01) |

## 5 Discussion

A few studies have demonstrated that SVD embeddings perform better than PPMI in word similarity tasks (Levy et al., 2015)). Some reasons include that PPMI vectors tend to generate long and sparse vectors that are not good at capturing synonymy. SVD essentially performs dimension reduction on PPMI embeddings, rendering it theoretically more robust (Hamilton, et al., 2016). However, our experiments on three domain-specific datasets show opposite results with PPMI outperforming SVD. We suspect that while reducing dimensions of the embedding, SVD finds latent concepts in the word space and maps these words to those concepts for a condensed representation. However, these concepts are difficult to interpret as such mapping processes are quite different from the ways humans extract concepts. Therefore, the cosine similarities computed by SVD might be less correlated to human judgement. Radinsky et al. (2011) even go so far as to abandon the notion of word similarity in exchange for a broader goal of word relatedness in semantic tasks. This explanation is also supported by the observation that in linguistically diverse and complex corpora such as the patent dataset, SVD performs even more suboptimally compared to PPMI. This is expected as the word pairs from the MEN dataset are plain and simple English, while the words frequently used in patent data are obscure. Therefore, on average, the cosine similarities of tested words in PPMI are smaller than SVD. A unit change in cosine similarity for PPMI results in greater variation for MEN's similarity scores. Hence, the correlation is stronger for PPMI on domain-specific corpora.

## References

[1] Eckart, C., Young, G. (1936) The approximation of one matrix by another of lower rank. *Psychometrika 1, 211–218.*, https://doi.org/10.1007/BF02288367 .

[2]Hamilton, W. L. Leskovec, J., Jurafsky, D. (2016) Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change.*Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* https://doi.org/10.18653/v1/P16-1141.

[3] Levy, O., Goldberg, Y., Dagan, I. (2015) Improving Distributional Similarity with Lessons Learned from Word Embeddings. *Transactions of the Association for Computational Linguistics. 3. 211-225.*

[4]Radinsky, K. Agichtein, E., & Gabrilovich, E., Markovitch, S. (2011) A Word at a Time: Computing Word Relatedness using Temporal Semantic Analysis. *Proceedings of the 20th International Conference on World Wide Web, WWW 2011. 337-346.*

[5] Zellig H. (1954) Distributional structure. Word, 10(23):146–162.

## A Appendix

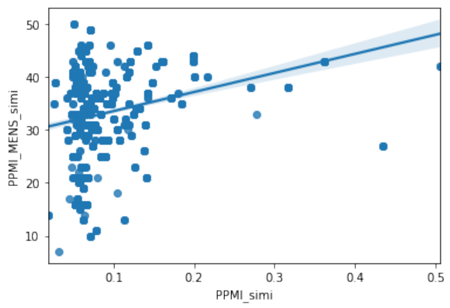

Figure 1: Linear Regression on PPMI and word similarity

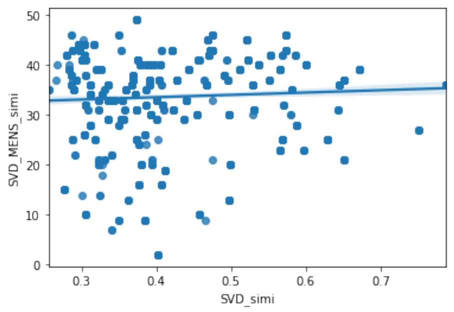

Figure 2: Linear Regression on SVD and word similarity