3-10-2019

Capturing Word Semantics From Co-occurrences Using Dynamic Mutual Information

Yaxin Li
University of Windsor

Capturing Word Semantics From Co-occurrences Using Dynamic Mutual Information

By

Yaxin Li

A Thesis
Submitted to the Faculty of Graduate Studies
through the School of Computer Science
in Partial Fulfillment of the Requirements for
the Degree of Master of Science
at the University of Windsor

Windsor, Ontario, Canada

2019

Capturing Word Semantics From Co-occurrences Using Dynamic Mutual Information

by

Yaxin Li

APPROVED BY:

A. Hussein
Department of Mathematics and Statistics

R. Gras
School of Computer Science

J. Lu, Advisor
School of Computer Science

January 21, 2019

DECLARATION OF ORIGINALITY

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyones copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

ABSTRACT

Semantic relations between words are crucial for information retrieval and natural language processing tasks. Distributional representations are based on word co-occurrence, and have been proven successful. Recent neural network approaches such as Word2vec and Glove are all derived from co-occurrence information. In particular, they are based on Shifted Positive Pointwise Mutual Information (SPPMI). In SPPMI, PMI values are shifted uniformly by a constant, which is typically five. Although SPPMI is effective in practice, it lacks theoretical explanation, and has space for improvement. Intuitively, shifting is to remove co-occurrence pairs that could have co-occurred due to randomness, i.e., the pairs whose expected co-occurrence count is close to its observed appearances. We propose a new shifting scheme, called Dynamic Mutual Information (DMI), where the shifting is based on the variance of co-occurrences and Chebyshev's Inequality. Intuitively, DMI shifts more aggressively for rare word pairs. We demonstrate that DMI outperforms the state-of-the-art SPPMI in a variety of word similarity evaluation tasks.

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to my supervisor Dr. Jianguo Lu for his constant guidance and encouragement during my whole Master's period at the University of Windsor. Without his help, this thesis would not have been possible.

I would also like to express my appreciation to my thesis committee members Dr. Robin Gras and Dr. Abdulkadir Hussein. Thank you all for your valuable suggestions to this thesis.

Last but not least, I want to express my gratitude to my parents and my friends who are always supportive over the past two years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

Semantic relations between words is crucial for information retrieval and natural language processing problems, such as entity extraction [31], word similarity and analogy [18, 13]. Word co-occurrence has been widely used for extracting word semantics and representations.

The most simple word distributional representation is the co-occurrence raw count matrix in HAL [16]. Each row in the co-occurrence matrix is used to represent a word vector. COALS [25] improves HAL model by ignoring the left-right distinction in word pair collection, replacing the raw co-occurrence count with Pearson's Correlation coefficient between two words. More importantly, it removes negative values in the matrix. It outperforms HAL on word similarity tasks and word classification tasks. More generally, more measures of word association can be applied in the matrix, each cell of which is the word association between two words.

The first influential research on word association norms [21] is based on empirical estimates. This study measured 200 words by asking subjects from grade school to college to write down a word after one word is given. As more computational resources are available, we can analyse the word relationships on large corpora, and more measures are proposed to represent word associations, such as Pointwise Mutual Information (PMI) [6], log-likelihood ratio [8] and χ^2 measure [11].

Recent neural network approaches to word representation learning, such as Word2Vec, are all based on word co-occurrences within a text window [19]. Those approaches are also tied to more traditional measurement of co-occurrence, e.g., using pointwise mutual information (PMI) [6], as demonstrated by [14]. Word2vec is implicitly factorizing a shifted

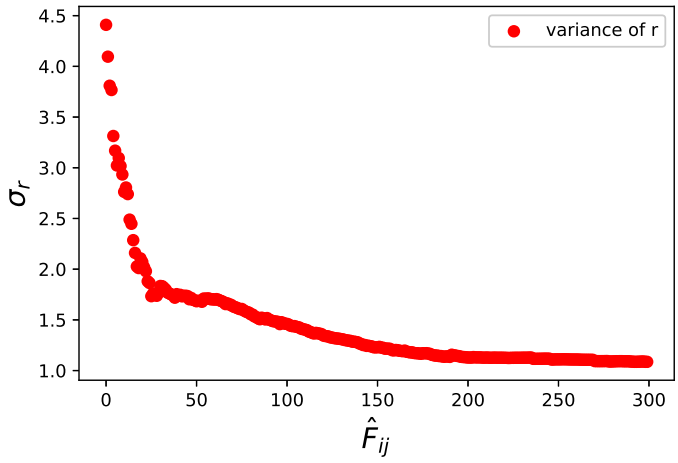


FIGURE 1: The variance of r as the $\hat{F_{ij}}$ increases.

PMI matrix, of which each cell is the PMI of the corresponding word pair shifted by a constant, typically $\log 5$. Moreover, they found that, if all the negative values are nullified, the word representations generated from shifted positive PMI(SPPMI) matrix can also achieve great performance on different tasks.

PMI measures the logarithm of the ratio r between the co-occurrence count and the expected count, assuming that two words i and j are independent. $PMI(i,j) = \log r = \log F_{ij}/\hat{F_{ij}}$. If the co-occurrence count is larger than expected, the ratio r will be larger than 1, thus its $PMI(i,j) > 0$. This means that words i and j co-occur more often than randomness, and they are associated positively. Although $PMI(i,j) > 0$ indicates that x and y associate positively, [6] observed that "genuine" association requires $I(i,j) \gg 0$. $PMI(i,j)$ (the logarithm in this work is based on 2) should be greater than 3 by manually inspecting word pairs and MI values. Associations with MI value less than 3 are found to be generally not interesting. [14] shows a similar value ($PMI = \log_2 5 = 2.3$), they shift all the ratios by 5.

SPPMI nullifies unreliable values, i.e., the values less than $\log 5$ or ratios less than 5, in a PMI matrix. Intuitively, the ratio is unreliable for rare words. However, for popular words, i.e., when f_i and f_j are large, the ratio becomes more reliable. More formally, the variance of the ratio changes over the frequency of the words, which is shown in Figure 1. Thus, the shifting should be dynamic that changes in accordance with the variance of the

ratio.

This paper proposes Dynamic Mutual Information (DMI), where the PMI of each word is dynamically shifted according to the variance of ratio r. We show that DMI outperforms SPPMI on different word similarity test sets, and an improved Steiger's test [24] is used to compute how significant the improvements of our DMI against are SPPMI, and most of the improvements are significant.

CHAPTER 2

Review of The Literature

This chapter reviews related research works about word co-occurrences and different word representation models.

2.1 Different Co-occurrences

Lexical co-occurrence is widely used to construct semantic spaces. Almost all the word representation models are based on word co-occurrences, and they belongs to two main categories: one is the distributional model and the other is the neural network based model. The distributional models [16, 25, 14] are derived from the co-occurrence matrix, and the popular neural network based models also utilize the information of word co-occurrences.

There are different word co-occurrences that are commonly used. The first and simplest is to use a window of length k that moves across the corpus: every two words in this window is considered one co-occurrence count. Another version of this window is that, for every word in the corpus, its neighbor word will receive a weighted count of $k-1$ if they are adjacent, $k-2$ if it is 2 words from the word, and so forth. In Section 3.1, we will discuss how different windows capture the co-occurrence between two words, and prove that these windows are roughly the same. There are also some variations of this kind of window, but they are all based on the same idea that closer words have higher co-occurrence weights.

In distributional models, these word co-occurrences can form a co-occurrence matrix, of which each row and each column represents one word in the vocabulary and each element is the co-occurrence count of two corresponding words. However, it is impossible to observe all the co-occurrence of related word pairs because of Zipf's law, thus the co-

occurrence matrix is extremely sparse. [29] proposed Neural Latent Relational Analysis (NLRA) to relieve the sparseness problem, and can obtain the embeddings of word pairs that do not co-occur in the corpus.

Apart from word pair co-occurrences, k-way word co-occurrences [4], which is generated from random walk are also used. Associated words do not always occur as pairs in the sentences, in natural languages multiple words can be related and co-occurring in the context. By splitting the sentences into word pairs, we lose some information. For example, in the sentence *The University of South California is located in Los Angeles*, we can have one three-way co-occurrence *(University, South, California)*, if it is split into 3 word pairs *(University,South)*, *(South, California)* and *(South, University)*, the information is incomplete.

Moreover, [33] introduced ngram statistics to explore more information in the source text. They use "ngram-ngram" type of co-occurrences instead of "word-word". To solve the problems brought by ngrams, new methods are proposed to build the co-occurrence matrix more efficiently.

2.2 Distributional Models

The most explicit model is Hyperspace Analogue to Language(HAL) [16], which uses word co-occurrence count directly as the word vector. Word co-occurrences are recorded by moving a window over the corpus in one word increment, and for each word pair within a window, their co-occurring counts are weighted inversely proportional to the number of other words separating them. Also, the word pair is direction sensitive, which means word pairs *(x, y)* and *(y, x)* are different. In this way, the co-occurrence matrix is not symmetric and the representation of the ith word is the concatenation of the ith column and the ith row. In the end, they use the Minkowski family of distance metrics to measure the similarity of different words. HAL gives a simple way of finding the word representations and yields good results. However, it suffers from the extremely high frequency of popular words or stop words, because they use raw counts directly.

COALS [25] improved HAL by converting raw counts to word pair correlations, and

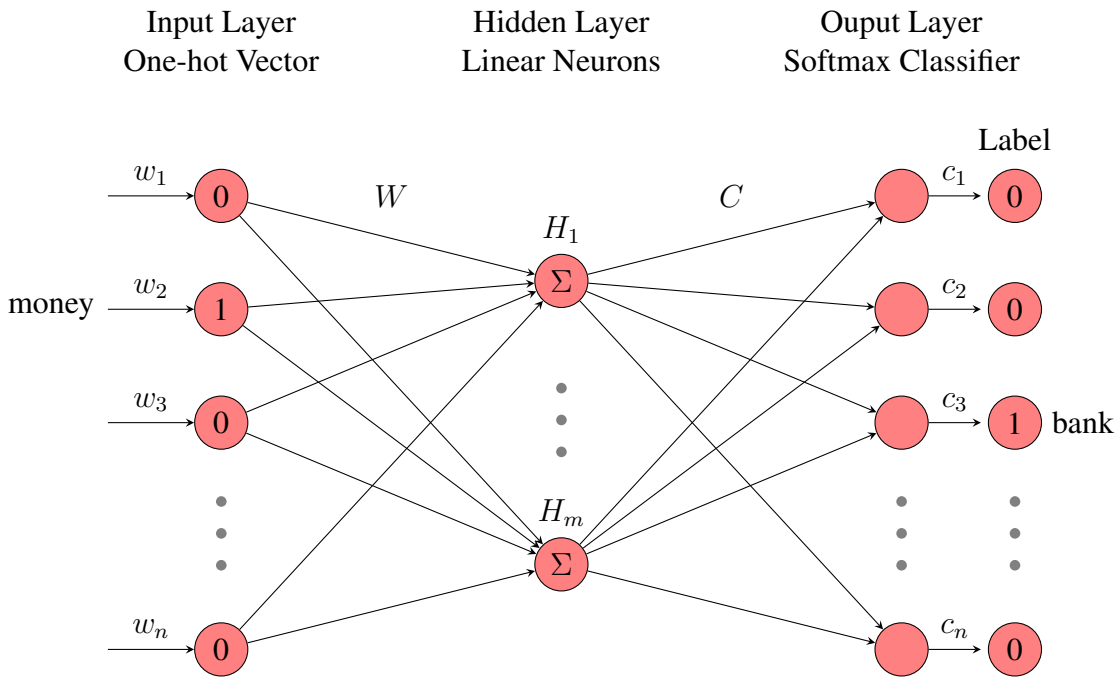


FIGURE 2: The architecture of Word2vec Skip-Gram model, and the current word pair is *(money, bank)*.

solved the problems caused by raw counts. Besides, it used a similar window, but ignores the left-right distinction between word pairs. Another improvement is discarding the negative values, which greatly boosts the performance of word vectors. In the end, each row of the matrix is the representation of the corresponding word, and COALS measures the word similarity with correlations.

In practice, the co-occurrence count in the matrix can be replaced with different measures of word association, such as Pointwise Mutual Information (PMI) [6], log-likelihood ratio [8] and χ^2 measure [11].

2.3 Neural Network Models

Neural network based models are becoming popular in discovering word semantic relations. They use neural network's internal word vectors to represent the word [3, 7], and the most popular models are Word2vec [19, 18] and GloVe [22]. These models are also using word co-occurrence information to create word embedding.

In Word2vec a size k dynamic window is applied, and it contains one center word w_i

and several context words c_j which are within k tokens to each side of the center word. As the window moves, each word can be a center word or context word and the actual window size will be shrunk by a random integer r ($0 \leq r \leq k-1$). Thus, the closer a context word is to the center word, the higher probability the context word can be considered as a word pair with the center word. Note that the window is the same as that used in COALS. Similarly, GloVe also gives higher weights to the closer context word, but a different weighting scheme is applied, less weights are given to the farther context word than that in word2vec window.

Figure 2 shows the simple architecture of Word2vec Skip-Gram model. The input layer is the one-hot vector of the center word "money", which means the element in "bank" position is 1 and all the other elements are 0s. The hidden layer consists of m linear neurons and m is the dimension of word embeddings. In the output layer, it uses softmax classifier to predict the probability of each word being its context, and its ground truth is "bank" in this case. Moreover, Word2vec uses stochastic gradient descent to minimize the loss function. In the end, the input vectors W are used as word representations and output vectors C are the context vector. It is reported that the word vectors from Word2vec have a more promising performance than any other neural network models. Glove is similar to Word2vec, but it takes the global word statistics into account.

Compared with distributional models, neural network based models project the word representations into a very low dimension, usually a few hundred, but the dimension is much higher in distributional models: it is the size of the vocabulary. The low-dimensional dense vectors have its advantages in improving the computational efficiency over distributional models. However, the neural network based models have several hyper parameters and it suffers from parameter tuning. The parameters have to be changed for different corpus, and this process is time-consuming.

2.4 Matrix Factorization and Neural Network Models

[14] related the neural network models with the traditional distributional models. They found the Skip-Gram models with negative sampling in Word2vec is implicitly factorizing

a shifted Pointwise Mutual Information (PMI) matrix, of which each cell is the PMI value between two words shifted by a constant. Furthermore, in order to take the advantage of dense low-dimensional word vectors, [15] proposed to use Singular Value Decomposition(SVD) to find the optimal rank d factorization of the shifted PMI matrix to get the low-dimensional word representations, and gave some suggestions on tuning the hyper parameters in generating word embeddings. [28] discussed further about all kinds of parameters in the matrix factorization, and introduced canonical correlation analysis (CCA) to improve the performance of matrix decompositions. [2] gives a theoretical justification for PMI based models, as well as hyper parameter choices by proposing a new generative model.

CHAPTER 3

Co-occurrence

There are several different methods to calculate the co-occurrences of word pairs [16, 25, 18, 22]. Usually, a window is introduced, and the co-occurrences of every word pair within one window are recorded. As the window moves across the corpus by one token, a co-occurrence matrix is formed to record the co-occurrence values of every two words in the corpus. Each column and each row of the matrix represents a unique word in the corpus, and the value of each cell is the co-occurrence count between two corresponding words.

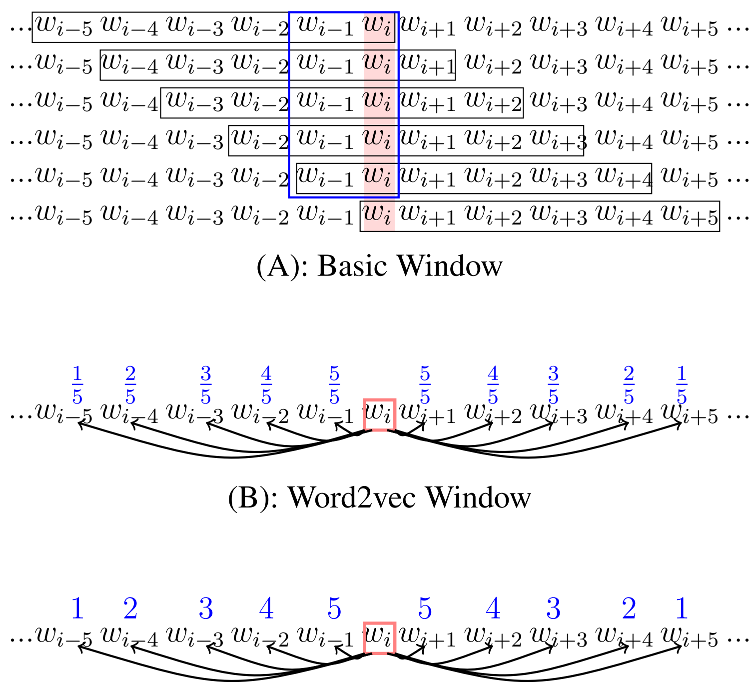
More formally, suppose that the vocabulary (the set of distinct words) is $V = \{w_1, w_2, \ldots, w_{|V|}\}$, and the size of the co-occurrence matrix is $|V| \times |V|$. Given a corpus that consists of a sequence of words $w_{x_1}, w_{x_2}, \ldots, w_{x_n}$, where $x_i \in \{1, 2, \ldots, |V|\}$ and n is the number of tokens in the corpus . Let k be the window size. The notations we are going to use are listed in Table 1, and we will talk about three popular window styles in collecting word pairs.

Corpus	n	Corpus length, i.e., # words in the corpus
	V	Vocabulary size, # unique words in the corpus
	f_i	Frequency of word i in corpus
Pairs	k	Window size
	N	# pairs sampled. $N = (n-k)k(k-1) \approx nk(k-1)$
	F_i	# word pairs whose first word is word w_i. $F_i = f_i k(k-1)$.
	F_j	# word pairs whose second word is word w_j. $F_j = f_j k(k-1)$.
	F_{ij}	Cooccurrence of word pair (w_i, w_j) in sampled word pairs
	$\hat{F}_{ij}$	Estimated cooccurrence count of word i and j if they are independent. $\hat{F}_{ij} = \frac{F_i F_j}{n} = \frac{f_i f_j}{n} k(k-1)$.

TABLE 1: Notations.

how much wood would a woodchuck chuck, if a woodchuck could chuck wood? as much wood as a woodchuck would,if a woodchuck could chuck wood.

TABLE 2: A simple text corpus



(A): Basic Window

(B): Word2vec Window

(C): Weighted Window.

FIGURE 3: Three kinds of co-occurrences. The window size is 6.

3.1 Window Sytles

In this section, we will introduce three most common window styles, and compare the differences between them by analysing their word pair distributions. In order to make each method easier to understand, we will give a sample co-occurrence matrix for each window style using a short text in Table 2.

3.1.1 Basic Window

The basic window is the most straightforward way of collecting word pairs. We move a window of fixed length k across the text, and every pair of two words in a window is

considered as one co-occurrence. As the window moves, a co-occurrence matrix is formed.

Even though it ignores the order and position of words, adjacent words have a higher weight than words lying farther apart. Figure 3(A) shows that, with a window of size 6, adjacent words (w_{i-1}, w_i) has five co-occurrences when the window slides by, while words four positions apart like (w_{i-4}, w_i) has only two co-occurrence.

Some steps of collecting word pairs are shown in 3, and the final co-occurrence matrix is shown in Table 4. Since we ignore the left-right distinction of word pairs, the co-occurrence matrix is symmetric, for example, the co-occurrence count of w_i and w_j is $F_{ij} = 5$, thus the value of i th row and j th column is 5, so is the value of j th row and i th column, and $F_{ji} = 5$.

For the word pair $(a, chuck)$, they first co-occur in the window(size 6) ...*much wood would a woodchuck chuck* ..., when the window moves, the following three windows also contain this pair, so the count increases to 4. Then 3 windows also contain this pair starting at ...*chuck, if a woodchuck could chuck wood*..., next, the pair also appears in another 3 windows starting at ...*would, if a woodchuck could chuck wood*.... In total, $F(a, chuck) = 4 + 3 + 3 = 10$.

Let the corpus length be n, and there are total $n - k$ windows if there are no sentence breaks in the corpus, i.e., the sliding window does not restart for each sentence. Since $n \gg k$, the total number of windows is approximate n. Within each window, there are $\binom{k}{2}$ word pairs collected, and because $F_{ij} = F_{ij}$, we have $\binom{2k}{2}$ more co-occurrence count on the co-occurrence matrix.

3.1.2 Word2vec Window

Word2vec window is a very popular window style in neural network based models. The length of the window varies as it moves over the sentence. In a dynamic window with size k, one word is picked as the center word, and its left and right $k - 1$ tokens are its context words. First, a random integer r between 0 and $k-2$ is generated first $(0 \leq r \leq k-2)$, then the window is shrunken by r, thus the actual window length is $k-r$,$(0 \leq r \leq k-2)$. In other words, for a size k window, the probability of the d th token to the center word sampled as

Current Window	Pairs	Count
how much wood would a woodchuck	(how much)	+1
	(how wood)	+1
	(how would)	+1
	(how a)	+1
	(how woodchuck)	+1
	(much wood)	+1
	(much would)	+1
	8 more pairs...	
much wood would a woodchuck chuck	(much wood)	+1
	(much would)	+1
	(much a)	+1
	(much woodchuck)	+1
	(much chuck)	+1
	(wood would)	+1
	(wood a)	+1
	8 more pairs...	
.....	.....	...

TABLE 3: Steps of collecting word pairs using basic windows

	a	as	chuck	could	how	if	much	wood	woodchuck	would
a	8	9	14	8	1	16	5	12	28	13
as	9	6	5	3	0	2	9	16	7	3
chuck	14	5	2	12	0	9	4	14	16	4
could	8	3	12	0	0	6	2	9	12	2
how	1	0	0	0	0	0	1	1	1	1
if	16	2	9	6	0	0	0	4	16	7
much	5	9	4	2	1	0	0	11	5	3
wood	12	16	14	9	1	4	11	6	12	5
woodchuck	28	7	16	12	1	16	5	12	8	12
would	13	3	4	2	1	7	3	5	12	0

TABLE 4: The co-occurrence matrix using basic window

Center Word	Dynamic Length	Context Word	Word Pair	Count
how	3	much	(how much)	+1
		wood	(how wood)	+1
		would	(how would)	+1
much	5	how	(much how)	+1
		wood	(much wood)	+1
		would	(much would)	+1
		a	(much a)	+1
		woodchuck	(much woodchuck)	+1
		chuck	(much chuck)	+1
...	...	...	...	...

TABLE 5: Steps of collecting word pairs using Word2vec window

the context word is $\frac{k-d}{k-1}$, which can be illustrated in Figure 3 (B). After collecting the pairs in the current window, the window moves forward by one token, so does the center word. For example, the center word in Figure 3 (B) is w_i and the center word of the next window will be w_{i+1}.

Some word collecting steps in Word2vec window are shown in Table 5, and the final co-occurrence matrix is shown in Table 6. The matrix is supposed to be symmetric because as the window moves, the current center word can be the context word of the next several center words, but the matrix is not symmetric due to the dynamically changed window size.

Similar to the basic window style, there are approximate n windows, and within each window, we collect k co-occurrence count on average due to the probability, which means if we run Word2vec window $(k-1)$ times, we will collect the same number of co-occurrence counts as that in basic window.

Word2vec window gives a higher probability to the closer words, and it works fine for the neural network based methods, because the neural network models usually have more than one iteration(epoch), and the corpus will be scanned several times. In this way, even the distant word pairs with lower probabilities can be sampled in one of these iterations.

However, such windows were also applied in some count-based models [15], where the corpus is scanned only once, and if we use dynamic window on a small corpus to count the word pairs, the relations between distant word pairs will be lost, which leads to degraded performance.

	a	as	chuck	could	how	if	much	wood	woodchuck	would
a	2	2	1	0	1	4	2	3	5	2
as	1	1	1	0	0	0	2	3	1	1
chuck	3	1	0	3	0	2	0	3	4	2
could	2	1	3	0	0	2	1	2	2	0
how	0	0	0	0	0	0	1	1	0	1
if	3	1	2	1	0	0	0	2	2	1
much	2	2	2	1	1	0	0	3	3	2
wood	2	3	4	3	1	2	3	2	3	2
woodchuck	5	0	3	2	0	3	1	2	2	3
would	3	1	1	1	0	1	2	2	3	0

TABLE 6: The co-occurrence matrix using word2vec window

3.1.3 Weighted Window

Another window style is widely used in distributional models [16, 25], and here we call it weighted window. It used a length k window, and for every word w_i in the corpus, its neighbor word will receive a weighted count of $k-1$ if they are adjacent, $k-2$ if it is 2 words from w_i, and so forth. The window and the weighting schemes are shown in Figure 3(C). We can see that the weighted window is another version of Word2vec window, and it replaces the probabilities with the real count to avoid the uncertainty. When $k=6$, if we run the Word2vec window 5 times, it should be equal to the weighted window.

Some steps of collecting word pairs are shown in Table 7. After all the pairs are collected, the co-occurrence matrix is in Table 8, where each row represents the center words and the columns are the context words. Note that, the matrix is symmetric.

3.1.4 Relationship Between Different Window Styles

From the section 3.1.2 and 3.1.3, we can see that the Word2vec window and weighted window are the same if we run Word2vec window $k-1$ times. So what is the relationship between basic window and weighted window? Intuitively, we should compare the co-occurrence matrix between these two windows, and the matrices are shown in Table **??** and Table 8 respectively. Their co-occurrence matrices for the small sample corpus are similar to each other, and it is likely that when the corpus grows larger, their co-occurrence matrices are the same.

Center Word	Context Word	Word Pair	Count
how	much	(how much)	+5
	wood	(how wood)	+4
	would	(how would)	+3
	a	(how a)	+2
	woodchuck	(how woodchuck)	+1
much	how	(much how)	+5
	wood	(much wood)	+5
	would	(much would)	+4
	a	(much a)	+3
	woodchuck	(much woodchuck)	+2
	chuck	(much chuck)	+1
...			

TABLE 7: Steps of collecting word pairs using weighted window

	a	as	chuck	could	how	if	much	wood	woodchuck	would
a	8	9	14	8	2	16	6	13	28	14
as	9	6	5	3	0	2	9	16	7	3
chuck	14	5	2	12	0	9	4	14	16	4
could	8	3	12	0	0	6	2	9	12	2
how	2	0	0	0	0	0	5	4	1	3
if	16	2	9	6	0	0	0	4	16	7
much	6	9	4	2	5	0	0	14	5	5
wood	13	16	14	9	4	4	14	6	12	7
woodchuck	28	7	16	12	1	16	5	12	8	12
would	14	3	4	2	3	7	5	7	12	0

TABLE 8: The co-occurrence matrix using weighted window

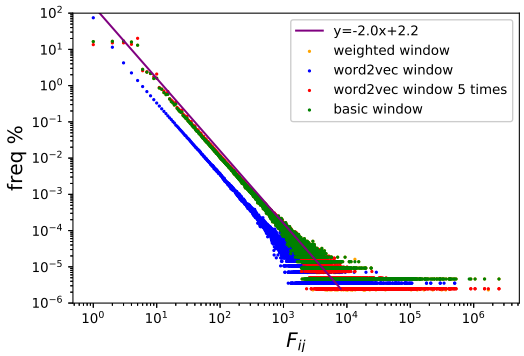
First of all, there will be approximate n windows in both basic window and weighted window styles, and within each window, they will both collect $k(k-1)$ word co-occurrence counts in the matrix, which means the total co-occurrence counts on both the matrices are the same.

Then for the word pair (w_i, w_{i+d}), where $d(-k < d < k)$ is an integer indicating the distance between two words in a window, the co-occurrence count of (w_i, w_{i+d}) in both basic window and weighted window is $k-d$. For example, if $d = 1$, in basic windows, there will be $k - 1$ consecutive windows passing by containing the word pairs, and in weighted windows, the weight given the word pair is also $k - 1$. In conclusion, the co-occurrence matrices for the basic window, the weighted window and Word2vec window(repeated $k-1$ times) are the same.

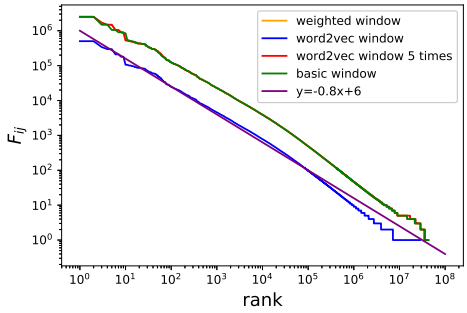
We use a small dataset Wiki-100 to demonstrate our points. Wiki-100 contains paragraphs randomly selected from English Wikipedia(July 2017 dump). All the "\n" or "\r" are removed from the text to meet our assumption. The small corpus contains 18,939,641 tokens in total, after removing the stop words(used in Lucene), there are 12,828,356 tokens left.

Figure 4 shows the distribution of word pair co-occurrences(F_{ij}) on three different data sets, and $k = 6$. In Figure 4(A) (C) (E), x-axis is the word pair co-occurrence F_{ij}, and y-axis is the frequency of F_{ij}. The subplot 4(B) (D) (F) shows the rank of F_{ij}, all the F_{ij}s are ranked from the largest to the smallest, the largest F_{ij} ranked first and when there are several same F_{ij}s, they have different ranks. In the left three plots, we can see that the co-occurrence distributions of the basic window, the weighted window and the Word2vec window repeated 5 times are roughly the same, and the distribution of Word2vec window run only one time is so different from all the others. Similar observations can be found in the right three plots.

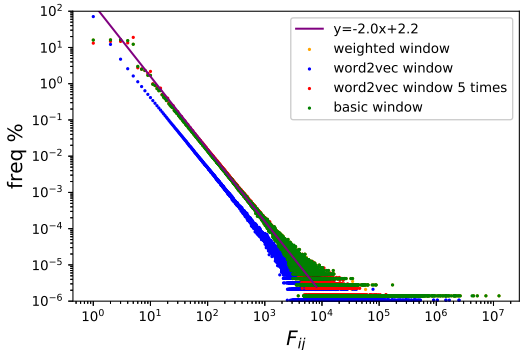
In all the windows except for word2vec window(run one time), when $F_{ij} = 1 \sim 5$, the frequencies of co-occurrence are similar to each other, that can be explained by the window. Suppose we have a sequence of text containing n distinct words, which means there are no words repeated in the corpus, if we use, for example, weighted window to capture the word co-occurrences, the word co-occurrence count should be no larger than



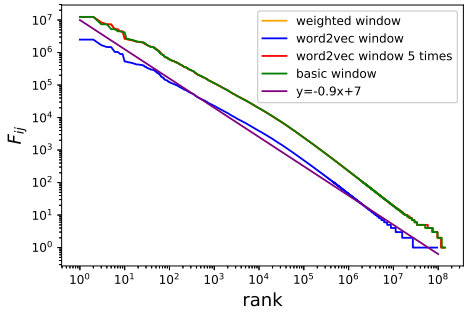
(A): Frequency of F_{ij}, Wiki-100.



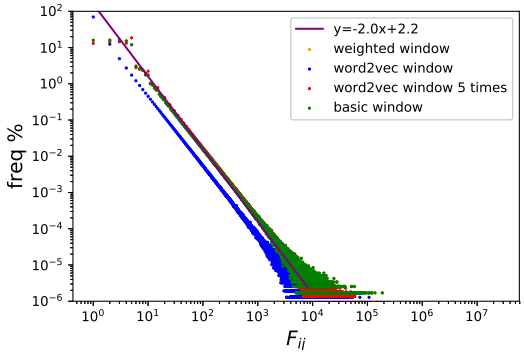
(B): Rank of F_{ij}, Wiki-100.



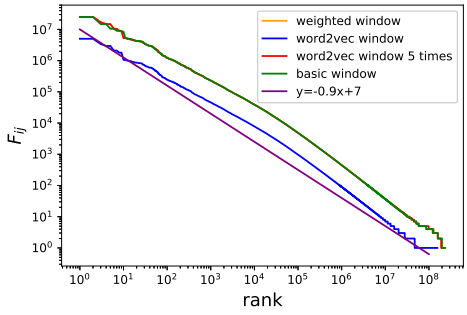
(C): Frequency of F_{ij}, Wiki-500.



(D): Rank of F_{ij}, Wiki-500.

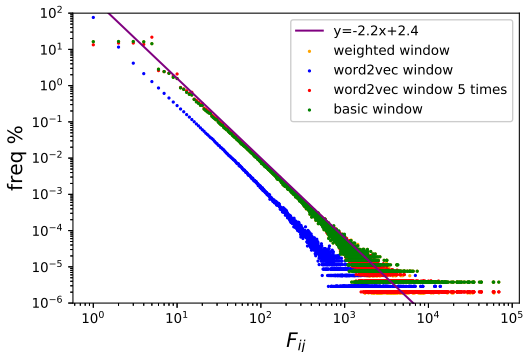


(E): Frequency of F_{ij}, wiki-1000.

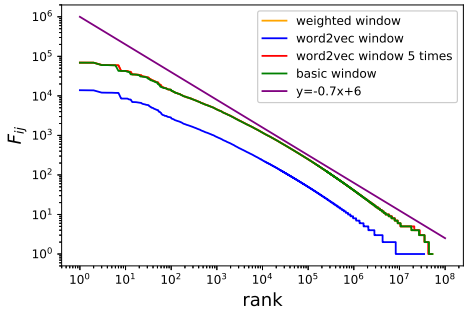


(F): Rank of F_{ij}, Wiki-1000.

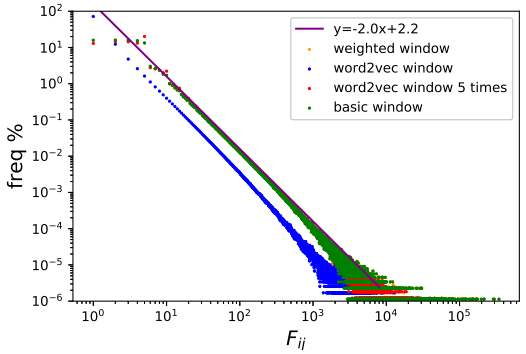
FIGURE 4: Distribution of co-occurrences using different windows on 3 datasets. Stop words are included.



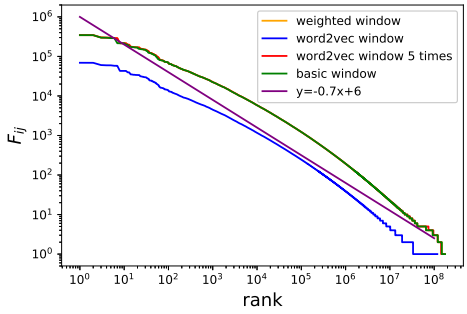
(A): Frequency of F_{ij}, Wiki-100.



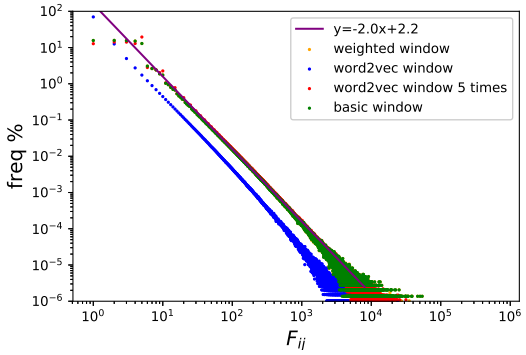
(B): Rank of F_{ij}, Wiki-100.



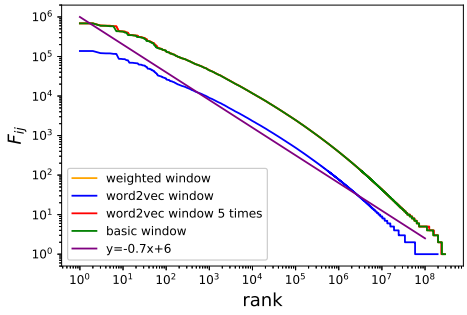
(C): Frequency of F_{ij}, Wiki-500.



(D): Rank of F_{ij}, Wiki-500.



(E): Frequency of F_{ij}, wiki-1000.



(F): Rank of F_{ij}, Wiki-1000.

FIGURE 5: Distribution of co-occurrences using different windows on 3 datasets. Stop words are removed.

$k-1$, and the frequencies of different F_{ij} should be the same. In our experiment, many word frequencies are larger than 1. Thus there are F_{ij}s larger than 5. However, in natural languages, according to Zipf's law, most words are irrelevant, and the chances of them co-occurring more than 5 are extremely low, only the small portion of highly associated word pairs can have F_{ij} larger than 5. Thus, the shape of the line when $F_{ij} = 1 \sim 5$ can be explained.

Table 9, 10 and 11 show the detailed frequencies of word co-occurrence on different datasets. With the basic window, weighted window and the Word2vec window repeated 5 times, we collect 384,850,570, 384,850,610 and 384,848,534 word occurrences respectively, which is roughly $k(k-1) = 30$ times of the length of the corpus and 5 times as that in Word2vec window. The distribution of F_{ij} in weighted window and basic window is the same in general, and the Word2vec window repeated 5 times is not exactly the same but similar to them because of the probabilities. Moreover, the basic window and weighted window collect the same number of unique pairs.

In basic windows, most of the pairs occur 1 to 5 times, taking up to 79.9% among all the collected word pairs, and only 20.1% word pairs occur more than 6 times. It means there are approximately 80% irrelevant word pairs. In other words, those pairs co-occurring less than 5 times in weighted windows are mostly irrelative pairs.

In word2vec windows, 76,961,221 pairs are collected in total, and there are 34,666,638 unique pairs. What is the most different thing from the other two window styles is that around 76.1% pairs only co-occur once. Unlike the other two window styles, if two consecutive words only co-occur in a window once, their co-occurrence would be 1, or if two words co-occur in a window a couple of times, but they are separated by some words, their co-occurrence can still be one or zero because the length of the window is dynamically changed. For example, word w_{i-4} and word w_i are separated by 3 words, when the center word is w_i the length if the current window can be 2, then the co-occurrence of w_{i-4} and w_i is 0. In word2vec window, most irrelevant pairs will co-occur once. However, when we repeat the word2vec window 5 times on the same corpus, the total number of pairs is 5 times of that in word2vec window and we collect more unique pairs, but not as many as in weighted windows.

F_{ij}	Word2vec-5 Count	%	Word2vec Count	%	Weighted Count	%	Basic Count	%
#pairs	384,848,534		76,961,211		384,850,610		384,850,570	
#unique pairs	49,191,978		34,666,638		52,815,864		52,815,864	
1	6,579,534	13.4	26,384,294	76.1	8,614,908	16.3	8,620,542	16.3
2	7,333,837	14.9	4,011,748	11.6	8,722,926	16.5	8,723,964	16.5
3	7,357,958	15.0	1,444,174	4.2	8,713,212	16.4	8,720,802	16.5
4	6,765,531	13.8	743,057	2.1	8,563,100	16.2	8,564,841	16.2
5	10,687,163	21.7	448,149	1.3	7,585,648	14.4	7,589,946	14.4
6	1,277,879	2.6	298,833	0.86	1,505,118	2.8	1,499,370	2.8
7	1,200,338	2.4	212,008	0.61	1,289,290	2.4	1,290,643	2.4
8	1,089,929	2.2	158,127	0.46	1,162,499	2.2	1,155,473	2.2
9	907,100	1.8	122,190	0.35	893,452	1.7	894,365	1.7
10	1,031,031	2.1	97,508	0.28	830,810	1.6	827,584	1.6
11	437,238	0.89	78,589	0.23	457,858	0.87	458,518	0.87
12	397,499	0.81	65,912	0.19	416,664	0.79	415,469	0.79
13	354,053	0.72	55,026	0.16	344,134	0.65	344,641	0.65
14	305,198	0.62	47,033	0.13	304,169	0.57	303,344	0.57
15	322,515	0.66	40,239	0.12	289,154	0.55	289,590	0.55
16	209,521	0.43	34,949	0.10	219,709	0.41	218,739	0.41
17	191,863	0.39	30,394	0.088	187,698	0.36	188,020	0.36
18	172,076	0.35	26,975	0.078	171,501	0.32	170,886	0.32
19	153,665	0.31	23,705	0.068	150,610	0.29	150,879	0.29
20	158,054	0.32	21,033	0.061	150,290	0.28	149,460	0.28

TABLE 9: Frequencies of Word Pairs Using Different Windows. The window size is 6. The data set is Wiki-100.

F_{ij}	Word2vec-5 Count	%	Word2vec Count	%	Weighted Count	%	Basic Count	%
#pairs	1,913,934,197		382,802,641		1,914,005,300		1,914,005,260	
#unique pairs	164,700,978		117,626,027		176,429,876		176,429,876	
1	21268854	12.9	84287775	71.7	27892424	15.8	27,906,332	15.8
2	23684984	14.4	14487468	12.3	28152484	16.0	28,155,136	16.0
3	23636736	14.4	5619888	4.8	27896948	15.8	27,915,854	15.8
4	21684144	13.2	3052189	2.6	27185962	15.4	27,191,947	15.4
5	33174939	20.1	1909129	1.6	23514786	13.3	23,525,899	13.3
6	4549812	2.8	1314720	1.1	5338138	3.0	5,324,328	3.0
7	4287086	2.6	960984	0.8	4568502	2.6	4,572,185	2.6
8	3908159	2.4	734233	0.6	4178551	2.4	4,160,033	2.4
9	3283245	2.0	579771	0.5	3231618	1.8	3,234,217	1.8
10	3656721	2.2	467884	0.4	2999387	1.7	2,991,075	1.7
11	1676934	1.0	386903	0.3	1749500	1.0	1,751,323	1.0
12	1533016	0.9	324428	0.3	1596962	0.9	1,593,749	0.9
13	1373102	0.8	277345	0.2	1343182	0.8	1,344,506	0.8
14	1195105	0.7	238483	0.2	1184341	0.7	1,182,054	0.7
15	1236938	0.8	206919	0.2	1125236	0.6	1,126,375	0.6
16	847375	0.5	181804	0.2	882771	0.5	879,835	0.5
17	776434	0.5	160560	0.1	764742	0.4	765,597	0.4
18	705686	0.4	142994	0.1	703199	0.4	701,440	0.4
19	633445	0.4	128521	0.1	619458	0.4	620,185	0.4
20	644154	0.4	115498	0.1	617535	0.4	615,484	0.3

TABLE 10: Frequencies of Word Pairs Using Different Windows. The window size is 6. The data set is Wiki-500.

F_{ij}	Word2vec-5 Count	%	Word2vec Count	%	Weighted Count	%	Basic Count	%
#pairs	3,843,280,467		768,625,759		3,843,332,150		3,843,332,110	
#unique pairs	272,588,032		195,555,046		291,800,831		291,800,831	
1	34795644	12.8	136823084	70.0	45694370	15.7	45714556	15.7
2	38658403	14.2	24533724	12.5	45984696	15.8	45988192	15.8
3	38507472	14.1	9728170	5.0	45366684	15.5	45394235	15.6
4	35254143	12.9	5365872	2.7	44017908	15.1	44028515	15.1
5	53234219	19.5	3400208	1.7	37698070	12.9	37715357	12.9
6	7666642	2.8	2369222	1.2	8993757	3.1	8973863	3.1
7	7229875	2.7	1743580	0.9	7663400	2.6	7669197	2.6
8	6594292	2.4	1345017	0.7	7066801	2.4	7038696	2.4
9	5562734	2.0	1063819	0.5	5469932	1.9	5473993	1.9
10	6174160	2.3	868215	0.4	5094179	1.7	5081151	1.7
11	2890078	1.1	719325	0.4	3010530	1.0	3013282	1.0
12	2646029	1.0	605803	0.3	2752077	0.9	2747377	0.9
13	2369257	0.9	519019	0.3	2322336	0.8	2324514	0.8
14	2072730	0.8	449801	0.2	2050103	0.7	2046392	0.7
15	2128946	0.8	392697	0.2	1946408	0.7	1948200	0.7
16	1481054	0.5	345995	0.2	1540044	0.5	1535524	0.5
17	1360027	0.5	306726	0.2	1340158	0.5	1341493	0.5
18	1236490	0.5	275517	0.1	1234415	0.4	1231654	0.4
19	1112755	0.4	247714	0.1	1095896	0.4	1097079	0.4
20	1129695	0.4	224069	0.1	1084175	0.4	1081049	0.4

TABLE 11: Frequencies of Word Pairs Using Different Windows. The window size is 6. The data set is Wiki-1000.

Rank	Basic Window		Word2vec window		Weighted window	
	F_{ij}	pair	F_{ij}	pair	F_{ij}	pair
1	69,115	his he	69,480	his he	69,115	his he
2	60,548	s s	60,778	s s	60,548	s s
3	60,062	has been	60,095	has been	60,062	has been
4	59,428	united states	59,433	united states	59,428	united states
5	42,755	new york	42,726	new york	42,755	new york
6	42,683	been have	42,718	been have	42,683	been have
7	40,627	been had	40,633	been had	40,627	been had
8	34,915	u s	34,942	u s	34,915	u s
9	34,684	he also	34,747	he also	34,684	he also
10	34,340	s he	34,385	s he	34,340	s he
11	33,764	his his	33,690	his his	33,764	his his
12	33,371	th century	33,428	th century	33,371	th century
13	31,520	from from	31,476	from from	31,520	from from

TABLE 12: Most frequent word pairs using different windows. The window size is 6. The data set is Wiki-100.

Figure 4(B) (D) (F) shows the ranking of F_{ij}, and the largest F_{ij} ranks the first. If there is a tie, we also give them different rankings. The top 20 largest F_{ij} and most frequent word pairs are listed in Table 12, 13 and 14. It is shown that the most frequent word pairs are similar between 3 different methods and for word2vec window repeated 5 times and weighted window, the most frequent pairs are exactly the same, even the rank.

In the above discussion, we can see that the basic window, weighted window and the word2vec window(repeated $k-1$ times) are the same theoretically, and in distributional models, the basic window is a better choice, because it is much simpler and convenient to analysed statistically. Some distributional models [14, 15] used word2vec windows and only run it once on the corpus. Thus, some word pair information will be lost and lead to degraded performance.

3.2 Expected Co-occurrences

The co-occurrence count of words w_i and w_j is the number of windows that contain both words w_i and w_j. Let F_{ij} denote that co-occurrences and $\hat{F}_{ij}$ be the expected co-occurrences. Our question is, given two words w_i and w_j, what is the expected number of

Rank	Basic Window		Word2vec window		Weighted window	
	F_{ij}	pair	F_{ij}	pair	F_{ij}	pair
1	345,147	his he	345,317	his he	345,147	his he
2	302,362	s s	301,666	s s	302,362	s s
3	297,837	has been	297,780	has been	297,837	has been
4	289,911	united states	289,890	united states	289,911	united states
5	215,331	have been	215,314	have been	215,331	have been
6	214,960	new york	214,985	new york	214,960	new york
7	197,783	been had	197,855	been had	197,783	been had
8	175,313	u s	175,413	u s	175,313	u s
9	171,971	he also	171,837	he also	171,971	he also
10	170,510	his his	170,519	his his	170,510	his his
11	168,960	th century	168,944	th century	168,960	th century
12	168,713	s he	168,879	s he	168,713	s he
13	157,422	from from	157,572	from from	157,422	from from

TABLE 13: Most frequent word pairs using different windows. The window size is 6. The data set is Wiki-500.

Rank	Basic Window		Word2vec window		Weighted window	
	F_{ij}	pair	F_{ij}	pair	F_{ij}	pair
1	688,261	his he	688,131	his he	688,261	his he
2	608,818	s s	608,645	s s	608,818	s s
3	600,074	has been	599,959	has been	600,074	has been
4	582,584	united states	582,606	united states	582,584	united states
5	435,134	have been	435,295	have been	435,134	have been
6	428,427	new york	428,548	new york	428,427	new york
7	402,518	been had	402,517	been had	402,518	been had
8	347,405	u s	347,372	u s	347,405	u s
9	346,707	he also	346,673	he also	346,707	he also
10	340,712	s he	341,014	s he	340,712	s he
11	337,702	th century	337,683	th century	337,702	th century
12	335,264	his his	335,275	his his	335,264	his his
13	313,374	from from	313,391	from from	313,374	from from

TABLE 14: Most frequent word pairs using different windows. The window size is 6. The data set is Wiki-1000.

co-occurrences F_{ij}? We can answer this question from two perspectives, the first is from the perspective of windows and another is from pairs. Let f_i denote the occurrences of word w_i in the corpus, and we will talk about the expected co-occurrence from different perspectives.

3.2.1 From the Perspective of Windows

If we use the basic window, this problem can be modelled as the Capture-Recapture problem: given n windows, and the order of words in the window is ignored. We use word w_i to capture (approximately) $f_i k$ windows. It is multiplied by k because each occurrence in the corpus will occur in k sliding windows. It is an approximation because there are cases for multiple occurrences of a word in one window. When the word is not a very popular one, and considering that the window size is small, we can neglect the multiple occurrences for the sake of simplicity. Next, we use w_j to capture windows under the condition that windows containing w_i are marked. After the capture, one position in each window is taken and there are 4 tokens left. Thus, among the rest (k-1) tokens in each window, we can recapture $f_j(k-1)$ windows containing w_j. Among those recaptured $f_j(k-1)$ windows, there are $\hat{F}_{ij}$ windows containing w_i. When words w_i and w_j are independent, the total number of windows can be estimated by

$$n = \frac{f_i k \times f_j(k-1)}{F_{ij}} = \frac{f_i f_j}{\hat{F}_{ij}} k(k-1) \tag{1}$$

In other words, the expected count is

$$\hat{F}_{ij} = \frac{f_i f_j}{n} k(k-1) \tag{2}$$

3.2.2 From the Perspective of Word Pairs

The expected co-occurrences between w_i and w_j can also be derived from word pairs we collected. Suppose we have a word pair (X, Y), where X and Y are two random variables, representing the words of each row and each column respectively. If w_i and w_j are

independent, the expected co-occurrence count for (w_i, w_j) is

$$\hat{F}_{ij} = P(X = w_i)P(Y = w_j) \times N \tag{3}$$

where $P(X = w_i)$ and $P(Y = w_j)$ are the chance of $X = w_i$ and $Y = w_j$ respectively, N is the total number of co-occurrences.

In the symmetric co-occurrence matrix, the sum of all the cells is the total number of co-occurrence counts N. Since in each window we collect $k(k-1)$ word co-occurrence counts, and there are n windows, the total co-occurrences are

$$N = nk(k-1) \tag{4}$$

The sum of i th row F_i^x is the total number of word pairs containing w_i. For each w_i in the basic window, we will collect $k-1$ word pairs containing w_i, and there will be k windows passing by the word w_i, thus F_i^x is:

$$F_i^x = \sum_{j=1}^{|V|} F_{ij} = f_i k(k-1) \tag{5}$$

Thus, the chance of $X = w_i$ is

$$P(X = w_i) = \frac{F_i^x}{N} = \frac{f_i}{n} \tag{6}$$

Similarly, we can have the change of $Y = w_j$

$$P(Y = w_j) = \frac{F_j^y}{N} = \frac{f_j}{n} \tag{7}$$

where F_j^y is the total number of word pairs containing w_j, and is also the sum of j th column.

With Equation 3,6,7, we have the expected co-occurrence of w_i and w_j

$$\hat{F}_{ij} = \frac{f_i f_j}{n} k(k-1) \tag{8}$$

In this section, we derive the expected co-occurrence $\hat{F}_{ij}$ from two different perspectives: windows and word pairs. The Equation 2 and 8 shows the $\hat{F}_{ij}$ values from different perspectives, and they are the same. Therefore, we can see the advantage of using basic windows; it is much easier to understand and calculate $\hat{F}_{ij}$.

3.3 F_{ij} and $\hat{F}_{ij}$

In this section, we will talk about the mean, variance and rse of F_{ij} when $\hat{F}_{ij}$ is in different ranges and the mean, variance and rse of $\hat{F}_{ij}$ when F_{ij} is of different values.

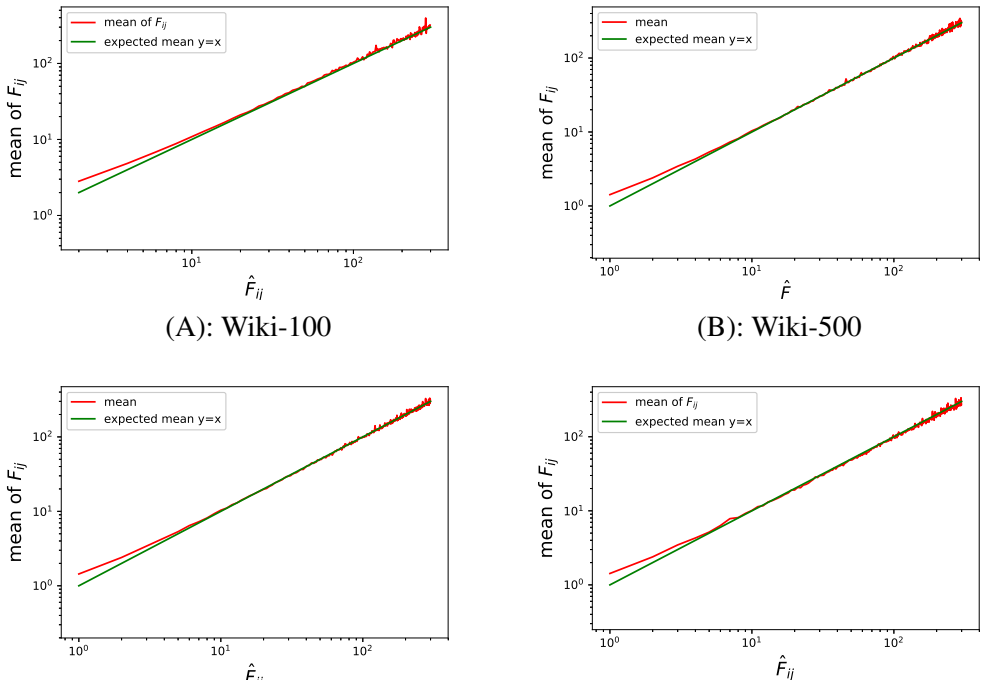
3.3.1 Mean, variance and rse of F_{ij}

When the $\hat{F}_{ij}$s of word pairs are in a certain range, their F_{ij}s can be very different: larger or smaller than $\hat{F}_{ij}$ or even be 0, because when $F_{ij} = 0$ for a word pair, their $\hat{F}_{ij}$ can never be zero. It means even two words never co-occur in a window; their expected co-occurrence is still larger than 0. It is impossible to get all the word pair combinations and their $\hat{F}_{ij}$ and F_{ij}, thus we randomly select two words, and the probability of selecting the word is proportional to its word frequency in the corpus. We select over 7 million random pairs to analyse the mean, variance and rse of F_{ij} when $\hat{F}_{ij}$ is between 0 and 300, which are shown in Figure 6, 7 and 8 respectively. The rse of F_{ij} is calculated as:

$$rse(F_{ij}) = \frac{\sqrt{var(F_{ij})}}{E(F_{ij})} \tag{9}$$

where $var(F_{ij})$ is the variance of F_{ij} and $E(F_{ij})$ is the mean of F_{ij}.

In Figure 6, we can see that the mean of F_{ij} is roughly equal to its corresponding mean of $\hat{F}_{ij}$. Figure 7 shows the variance of F_{ij} increases as the $\hat{F}_{ij}$ gets larger, and that is because of the increasing mean of F_{ij}. Thus, we use rse to measure the variance of F_{ij}, which is



(A): Wiki-100 (B): Wiki-500

(C): Wiki-1000 (D): Reuters

FIGURE 6: Mean of F_{ij} when $\hat{F}_{ij}$ is between 0 and 300.

depicted in Figure 8. The rse of F_{ij} decreases with the growth of $\hat{F}_{ij}$, and it means as the $\hat{F}_{ij}$ increases, the F_{ij} is getting more stable. In other words, when the word frequencies are larger, the estimated co-occurrence is closer to its real value.

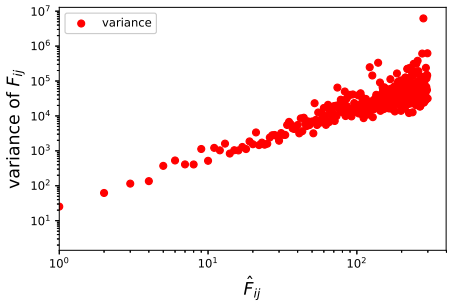
3.3.2 Mean, variance and rse of $\hat{F}_{ij}$

As in Section 3.3.1, we will show the mean, variance and rse of $\hat{F}_{ij}$, which is shown in Figure 9, 10 and 11 and x-axis is changed to F_{ij}.

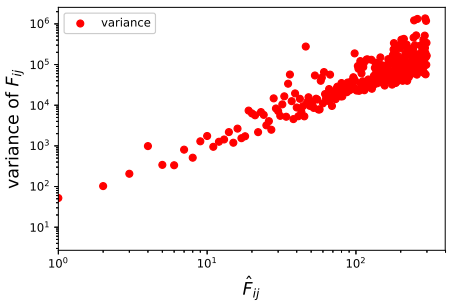
Figure 9 shows that the mean of $\hat{F}_{ij}$ is slightly smaller than F_{ij} as F_{ij} increases. This can be explained by how we estimate the word co-occurrences. It is easy to have

$$\sum_{i=1}^{|V|}\sum_{j=1}^{|V|} F_{ij} = \sum_{i=1}^{|V|}\sum_{j=1}^{|V|} \hat{F}_{ij} \tag{10}$$

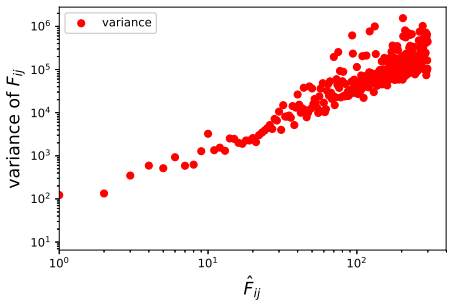
As what we talk about before, $\hat{F}_{ij}$ can never be 0 while a lot of F_{ij}s are 0, thus for the



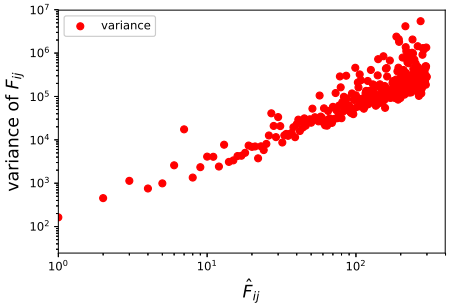
(A): Wiki-100



(B): Wiki-500

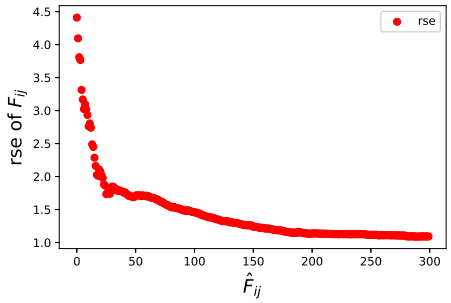


(C): Wiki-1000

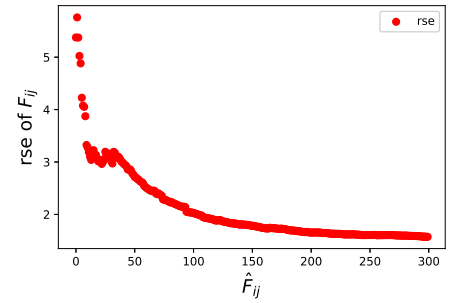


(D): Reuters

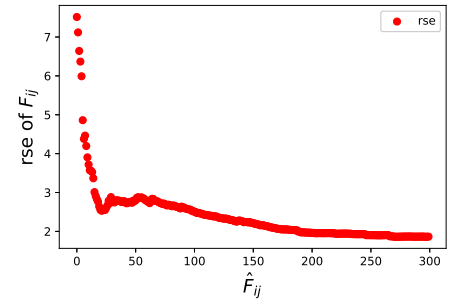
FIGURE 7: Variance of F_{ij} when $\hat{F}_{ij}$ is between 0 and 300.



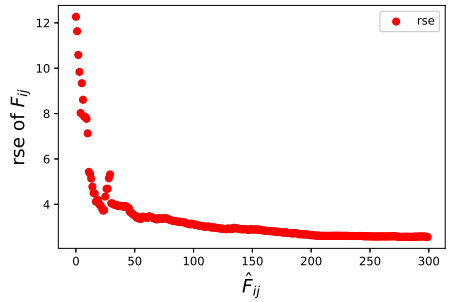
(A): Wiki-100



(B): Wiki-500

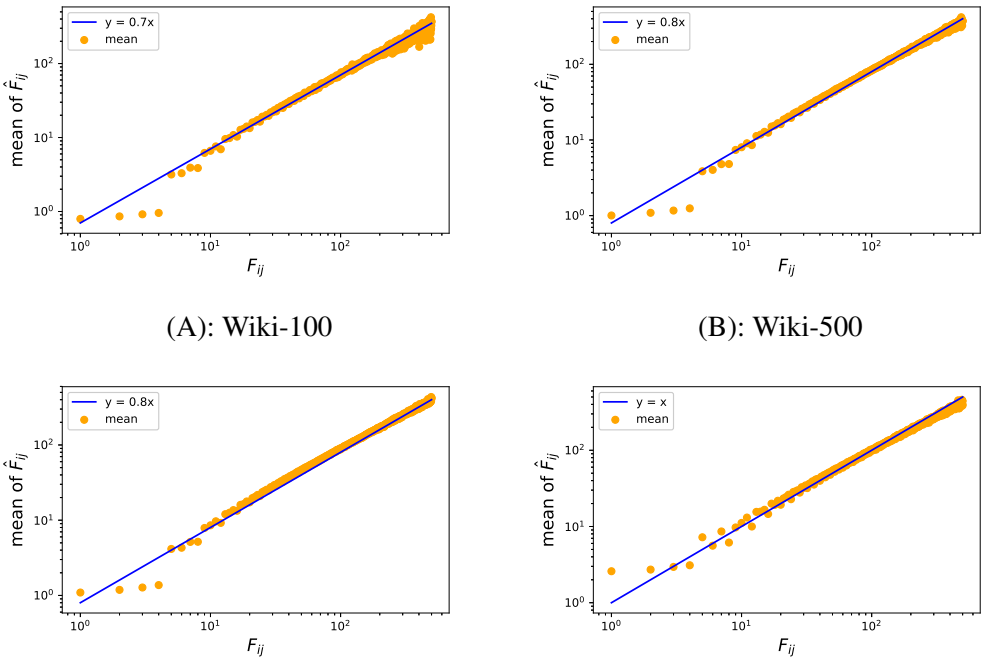


(C): Wiki-1000



(D): Reuters

FIGURE 8: Rse of F_{ij} when $\hat{F}_{ij}$ is between 0 and 300.



(A): Wiki-100

(B): Wiki-500

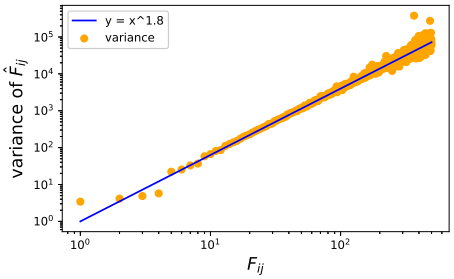
(C): Wiki-1000

(D): Reuters

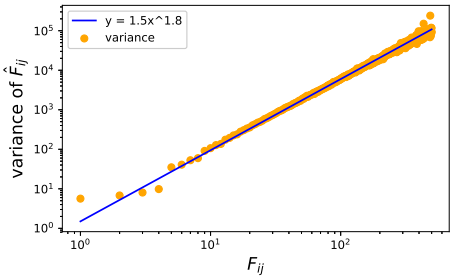
FIGURE 9: Mean of $\hat{F}_{ij}$ when F_{ij} is between 0 and 500.

non-zero F_{ij}, the estimation will be smaller than it should be.

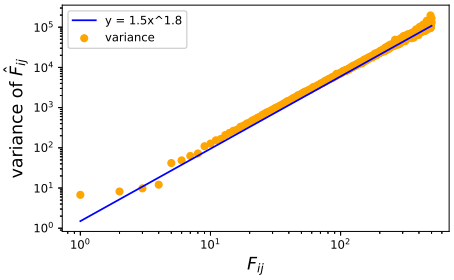
The variance of $\hat{F}_{ij}$ is shown in Figure 10, and it increases with the growth of F_{ij}, because the mean of $\hat{F}_{ij}$ is also getting larger. In the same way, we use rse to remove the influence of mean, and to see the variance of $\hat{F}_{ij}$. The rse of $\hat{F}_{ij}$ drops very quickly as F_{ij} increases and becomes stable when F_{ij} is large. It means the estimation is getting closer to its ground truth when F_{ij} is large enough.



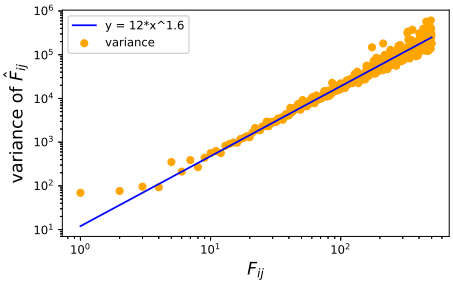
(A): Wiki-100



(B): Wiki-500

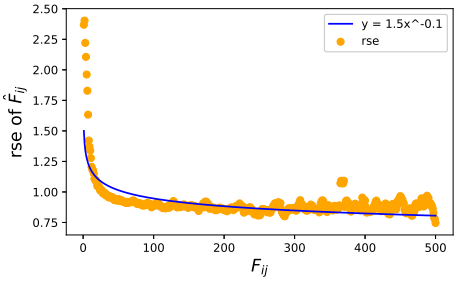


(C): Wiki-1000

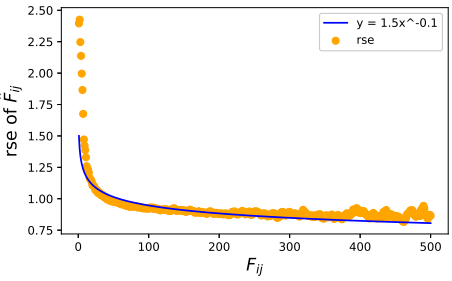


(D): Reuters

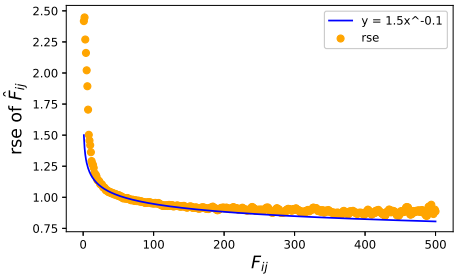
FIGURE 10: Variance of $\hat{F}_{ij}$ when F_{ij} is between 0 and 500.



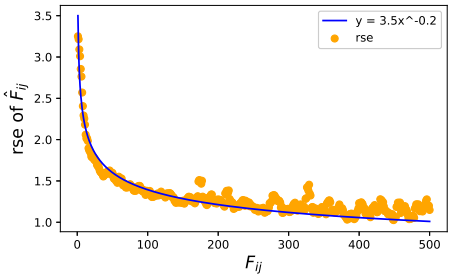
(A): Wiki-100



(B): Wiki-500



(C): Wiki-1000



(D): Reuters

FIGURE 11: Rse of $\hat{F}_{ij}$ when F_{ij} is between 0 and 500.

CHAPTER 4

Pointwise Mutual Information (PMI)

4.1 PMI and $\hat{F}_{ij}$

The Pointwise Mutual Information(PMI) was proposed to measure the word associations by church1990word, and the PMI is defined as

$$PMI = \log \frac{P(X,Y)}{P(X)P(Y)} \tag{11}$$

where $P(X,Y)$ is the joint probability of the variable X and Y, $P(X)$ and $P(Y)$ are the chance of X and Y respectively.

For the PMI of the word pair (w_i, w_j) can be defined as:

$$PMI_{ij} = \log \frac{P(X = w_i, Y = w_j)}{P(X = w_i)P(Y = w_j)} \tag{12}$$

where $P(X = w_i)$ and $P(Y = w_j)$ are already explained in Section 3.2.2 and $P(X = w_i, Y = w_j)$ is the probability of w_i and w_j co-occurring, and we have

$$P(X = w_i, Y = w_j) = \frac{F_{ij}}{N} \tag{13}$$

With Equation 3, we have

$$P(X = w_i)P(Y = w_j) = \frac{\hat{F}_{ij}}{N} \tag{14}$$

and with Equation 13 and 14, the PMI can be rewritten as

$$PMI_{ij} = \log \frac{F_{ij}}{\hat{F}_{ij}} \tag{15}$$

$$= \log \frac{F_{ij} n}{f_i f_j k(k-1)} \tag{16}$$

The PMI between two words is the (logarithm) ratio of its co-occurrence and expected co-occurrence. The co-occurrence count in the matrix can be replaced with PMI values between word pairs, and each row of the PMI matrix can be used as the word vector.

4.2 Shifted Positive PMI (SPPMI)

4.2.1 Shifted PMI

[14] found that the skip-gram model with negative sampling in word2vec is implicitly factorizing a matrix, of which each cell is the shifted PMI (SPMI) of the corresponding word pair, and the shifted PMI can be defined as

$$SPMI_{ij} = PMI_{ij} - \log s = \log \frac{F_{ij}}{s\hat{F}_{ij}} \tag{17}$$

where s is a constant(typically 5).

With all the SPMI value between all the pairs, a SPMI matrix is formed.

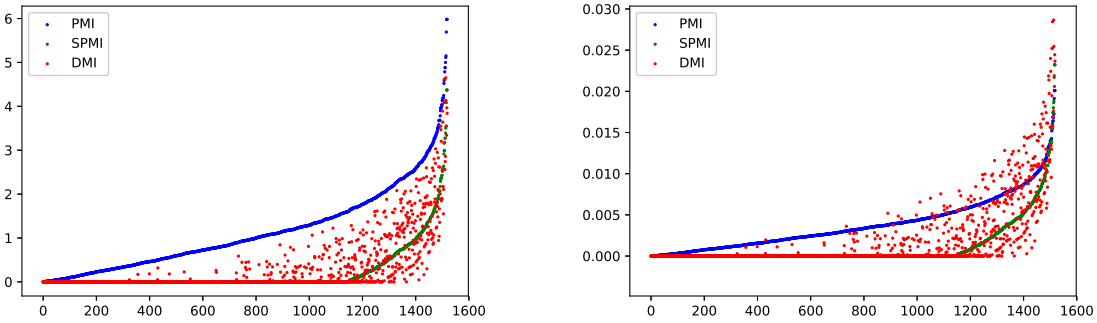
In Word2vec, each word w_i is represented by a word vector $\overrightarrow{w_i}$ and a context vector $\overrightarrow{c_i}$. It has been proved that

$$SPMI_{ij} = \overrightarrow{w_i} \cdot \overrightarrow{c_j} \tag{18}$$

and

$$SPMI = W \cdot C^T \tag{19}$$

where W is the matrix of all the word vectors and C is the matrix of all the context vectors.



(A): the original vectors (B): the normalized vector

FIGURE 12: Vector of word "percent" from PMI, SPMI and DMI matrix.

Word2vec is implicitly factorizing the SPMI matrix to get the word vectors for all the words, that is matrix W.

4.2.2 SPPMI

We can use each row in the SPMI matrix to represent one word directly without factorization, and as [25] and [14] suggested, if all the negative values are removed, great performances can also be achieved using this word representations. Then we change SPMI into SPPMI(Shifted Positive Pointwise Mutual Information).

$$SPPMI_{ij} = \begin{cases} 0, & \text{if } \frac{F_{ij}}{\widehat{F}_{ij}} < s; \\ \log\left(\frac{F_{ij}}{\widehat{F}_{ij}}\right) - \log s, & \text{otherwise.} \end{cases} \tag{20}$$

Usually, s is set to be 5, SPPMI simply removes all the pairs whose F_{ij} is not 5 times larger than its $\widehat{F}_{ij}$, that is nullify the PMIs smaller than $\log 5 \approx 1.6$. There are similar observations in [6], they found the word pairs whose $PMI < 3$ (the logarithm is based on 2, and $\log_2 5 \approx 2.3$) are not interesting. The difference between using SPPMI and directly using PMI can be demonstrated in Figure 12(A), which shows the vector of word "percent" in SPPMI and PMI matrix. SPPMI(green) shifts all the values of PMI(blue) downwards by a constant ($\log 5 = 1.6$).

Intuitively, SPPMI removes the noises in the matrix, that is, removing all the unreliable

	a	b	c	d	e	f
a	1	-2	-1	-3	1	0
b	-2	-1	-2	-3	2	1
c	-1	-2	0	2	0	1
d	-3	-3	2	-1	2	0
e	-1	2	0	2	0	1
f	0	1	1	0	1	0

TABLE 15: A samll PMI matrix.

word pairs whose ratio $(\frac{F_{ij}}{\widehat{F}_{ij}})$ is less than 5. However, the ratio r is more unreliable for rare words, for popular words, i.e., when f_i and f_j are large, the ratio becomes more reliable, that is, the variance of the ratio changes over the frequency of word pairs, which is shown in Figure 1. Therefore, s should be dynamically changed according to the reliability of $\widehat{F}_{ij}$.

4.2.3 Why Positive

[14] suggested, if all the negative values in the matrix are removed, the performance of word vectors on the word similarity tasks will be improved. Why do the negative relations degrade the performance?

It is because the negative relations are not transitive. Suppose Table 15 is a very small PMI matrix, and we have the word vectors for a and b. We can see from the matrix that a and b have negative relations, and they both have negative relations with c and d. If we do not remove the negative values in the matrix, the cosine similarity of the word a and b is positive, which disagrees with the fact that a and b are negatively related. It is mainly due to the negative values. The third element and the fourth element in both vectors are negative, if they are multiplied, it would be a large positive value, and the cosine similarity is a positive value.

Here comes the question: if a is irrelevant to c and b is also irrelevant to c, does that mean a and b are similar? Obviously not. In natural languages, most words are irrelevant to each other, for example, the words “tiger”, “car” and “next”, the first two words are both irrelevant to the third word, but they are not related neither. Of course, there are cases that two related words are both irrelevant to another word, but the circumstances that three

words are all irrelevant to each other are more common.

It means the negative relations are not transitive, but positive relations are. To remove the influences of the negative relations, nullifying the negative values is necessary.

4.3 Singular Value Decomposition (SVD)

Though high dimensional word vectors work well, it is always better to have vectors with lower dimension, because it is faster and easier to generalize. A common way to factorize the SPMI matrix is to do the Singular Value Decomposition(SVD) [12], which finds the optimal rank d factorization according to L_2 loss.

SVD factorize the $m \times n$ matrix M into the product of three matrices $U\Sigma V^T$, where U and V are unitary matrices, and their shapes are $m \times m$ and $n \times n$ respectively. Σ is an $m \times n$ diagonal matrix of eigenvalues in decreasing order. To reduce the dimension and keep most information at the same time, we only keep the top d elements in Σ, thus we have

$$M_d = U_d \Sigma_d V_d^T \tag{21}$$

If we use SVD to factorize the SPMI matrix, $U_d \cdot \Sigma_d$ can be replaced with W, where W is the word vector matrix, and use V_d to represent context vector matrix C, we have

$$W = U_d \Sigma_d \tag{22}$$

$$C = V_d^T \tag{23}$$

Each row of W represent the word vector, and the dimension of the vector is d. Each row of C is the context vector for each word and the dimension is also d.

According to the experiments in [15], if the SPMI matrix is factorized more symmetric,

the quality of word vectors will be better. In this way, we use

$$W = U_d \Sigma_d^{\alpha} \tag{24}$$

$$C = \Sigma_d^{(1-\alpha)} V_d^T \tag{25}$$

where α is usually set to 0.5.

The performance of word vectors from matrix factorization should be similar to that from Word2vec theoretically, but in our experiment, the SVD vector dose not have much advantage over Word2vec, and it is not scalable. When the corpus is very large, it is time-consuming to perform SVD on the co-occurrence matrix, though the matrix is a sparse one.

CHAPTER 5

Dynamic Mutual Information (DMI)

To improve the SPPMI, we propose Dynamic Mutual Information, which dynamically shifts the PMI values according to the variance of ratios, and preserves only the reliable values. Let $r = \frac{F_{ij}}{\hat{F}_{ij}}$, and σ_r^2 be the variance of r, the estimation for r is useful only when its variance σ is within certain range. Applying Chebyshev's Inequality, we have

$$P(|r - \hat{r}| \geq c\sigma) \leq \frac{1}{c^2}. \tag{26}$$

This means the probability that the value of r falls outside the interval $(\hat{r} - c\sigma, \hat{r} + c\sigma)$ does not exceed $\frac{1}{c^2}$. For example, if $c = \sqrt{2}$, the probability of $r \leq \hat{r} - \sqrt{2}\sigma$ or $r \geq \hat{r} + \sqrt{2}\sigma$ is less than $\frac{1}{2}$. Also, the $\hat{r} = 1$, because the F_{ij} is expected to be equal to $\hat{F}_{ij}$.

Hence, we derive DMI as follows:

$$DMI_{ij} = \begin{cases} 0, & \text{if } \frac{F_{ij}}{\widehat{F}_{ij}} < \sqrt{2}\sigma_r + 1; \\ \log\left(\frac{F_{ij}}{\widehat{F}_{ij}}\right) - log(\sqrt{2}\sigma_r + 1), & \text{otherwise.} \end{cases} \tag{27}$$

In our DMI, we preserve all the PMIs larger than $\sqrt{2}\sigma_r + 1$. In the following sections, we will talk about how to get the variance of r.

5.1 Variance of r

The variance of r is:

$$\sigma_r^2 = var\left(\frac{F_{ij}}{\widehat{F_{ij}}}\right) = \frac{var(F_{ij})}{\mathbb{E}(\widehat{F}_{ij})^2} \tag{28}$$

Let σ_F^2 be the variance of F_{ij}, we can have

$$\sigma_r = \frac{\sigma_F}{\mathbb{E}(\widehat{F}_{ij})} \tag{29}$$

If we can get the value of σ_F, then we can calculate σ_r. However, the distribution of F_{ij} is pretty complicated, which is shown in Figure 13. The distribution of F_{ij} is similar to power law distribution when $\hat{F}_{ij}$ is small in a log-log plot, but when $\hat{F}_{ij}$ gets larger, the distribution of F_{ij} is more like log-normal distribution. It is very difficult to calculate σ_F nor σ_r.

However, after examining several data sets, we find that the value of σ_r can be roughly approximated using the function $y = \frac{a}{\sqrt{F_{ij}}}$, as shown in Figure 16. Each dot in Figure 16 represents the variance of r when $\hat{F} = 0-300$. There are 7 million word pairs are collected for each plot. For two words in a word pair, they are randomly selected from the corpus, and the probability of each word being sampled is proportional to its word frequency in the corpus, which means frequent words are more likely to be selected than rare words and some F_{ij}s can be 0. The window size is 5.

5.2 DMI

It is time-consuming to get the plots such as Figure 16 and very difficult to get enough samples when $\hat{F}_{ij}$ is very large. Thus it is much easier to use an approximated function if the variance of r can be represented by a general function on different datasets.

Our DMI can be rewritten as:

$$DMI(w_i, w_j) = \begin{cases} 0, & \text{if } \frac{F_{ij}}{\widehat{F}_{ij}} < \frac{\sqrt{2}a}{\sqrt{F_{ij}}}+1; \\ \log\left(\frac{F_{ij}}{\widehat{F}_{ij}} / \left(\frac{\sqrt{2}a}{\sqrt{F_{ij}}} + 1\right)\right), & \text{otherwise.} \end{cases} \tag{30}$$

where $a = 6 \sim 19$ for different data sets. In our experiment, when a is around 10, we can see a significant improvement over SPPMI, if not the best.

Intuitively, DMI throws away the co-occurrence ratio r when it is not 'large enough'. In this sense, it is similar to SPPMI where all the values are shifted by a constant. What

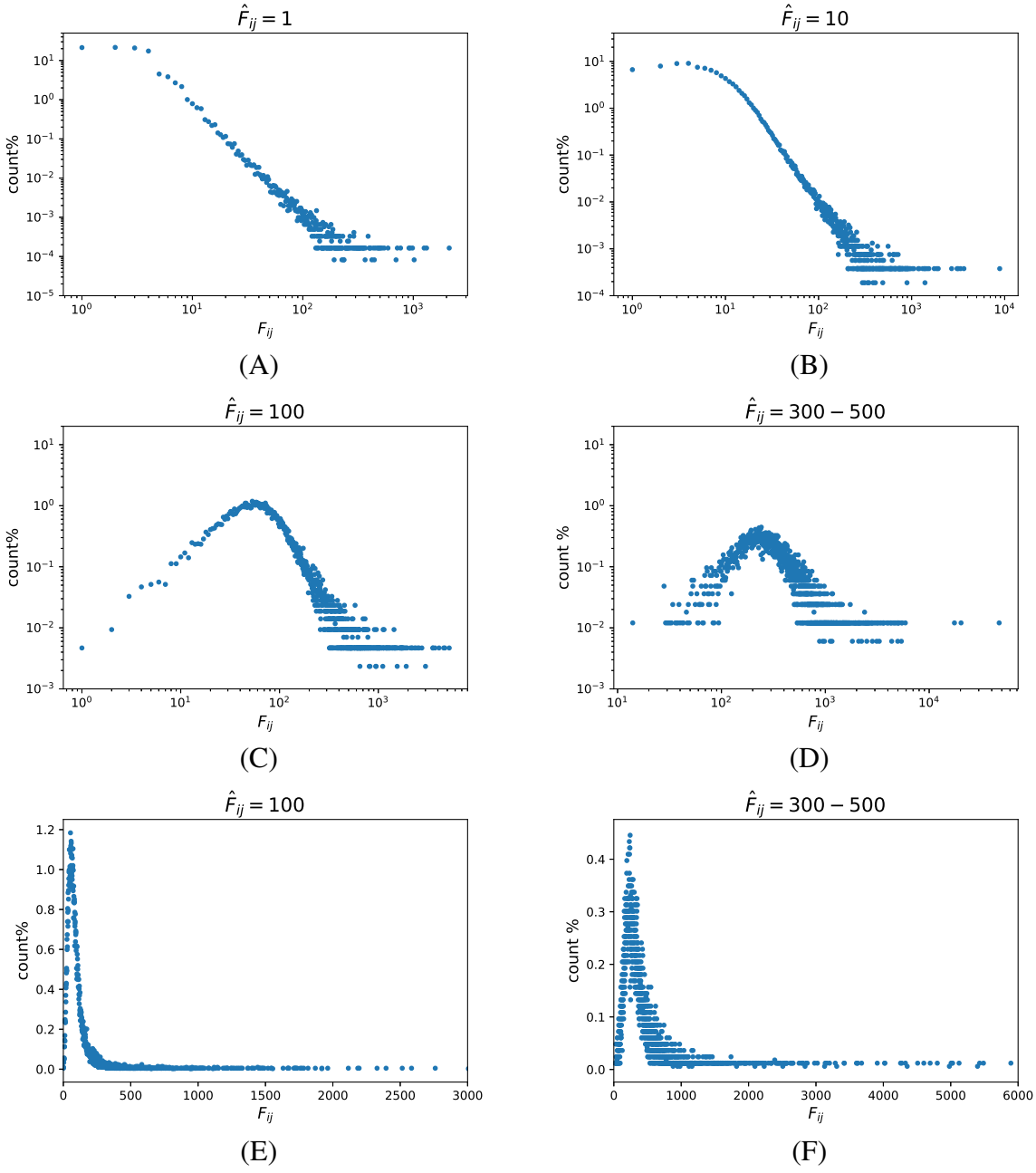


FIGURE 13: The distribution of F_{ij} when $\hat{F}_{ij} = 1, 10, 100 \; and \; 300 - 500$. (E) and (F) are another versions of (C) and (D) without loglog plot. The data set is Wiki-100.

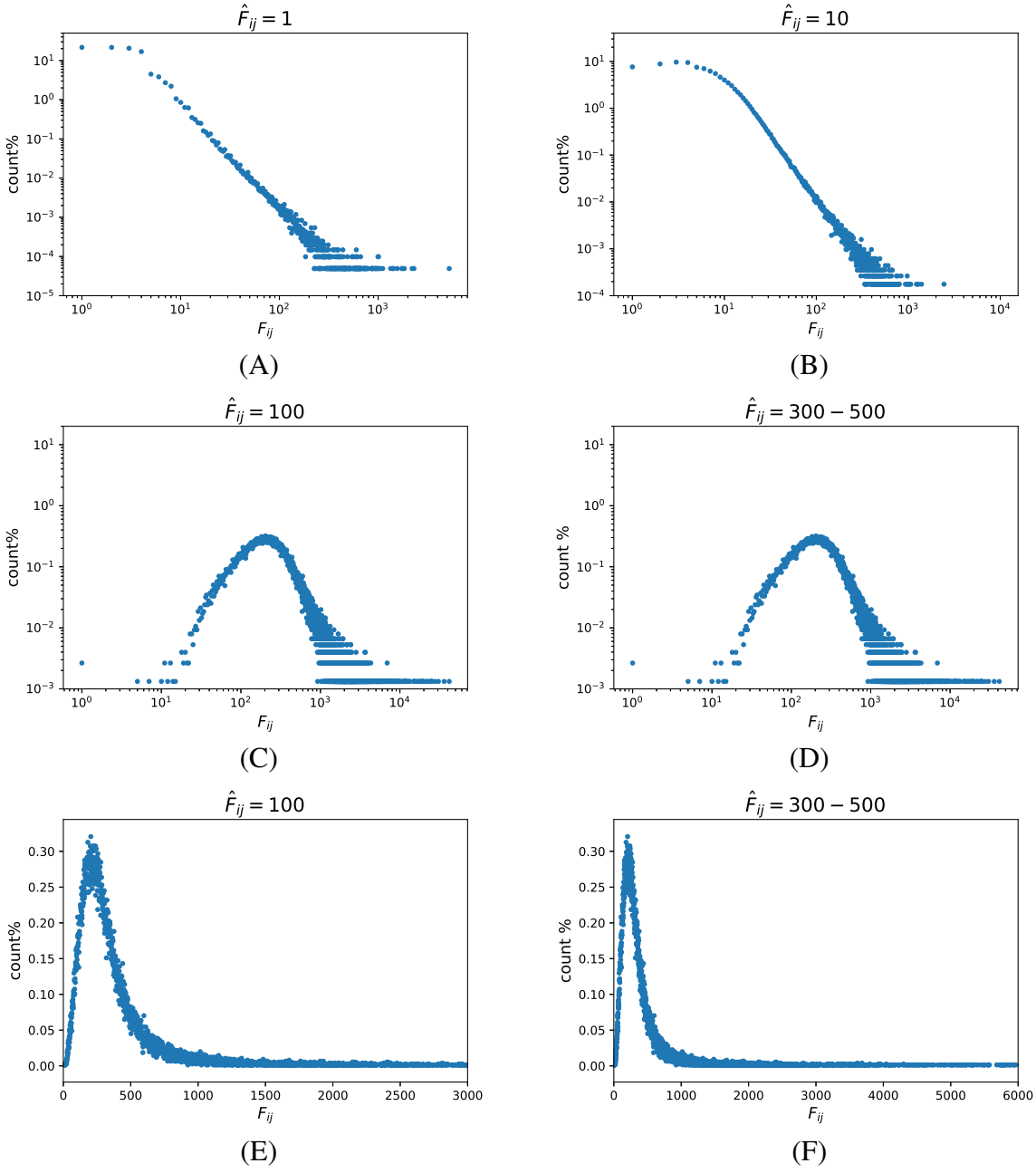


FIGURE 14: The distribution of F_{ij} when $\hat{F}_{ij} = 1, 10, 100 \; and \; 300 - 500$. (E) and (F) are another versions of (C) and (D) without loglog plot. The data set is Wiki-500.

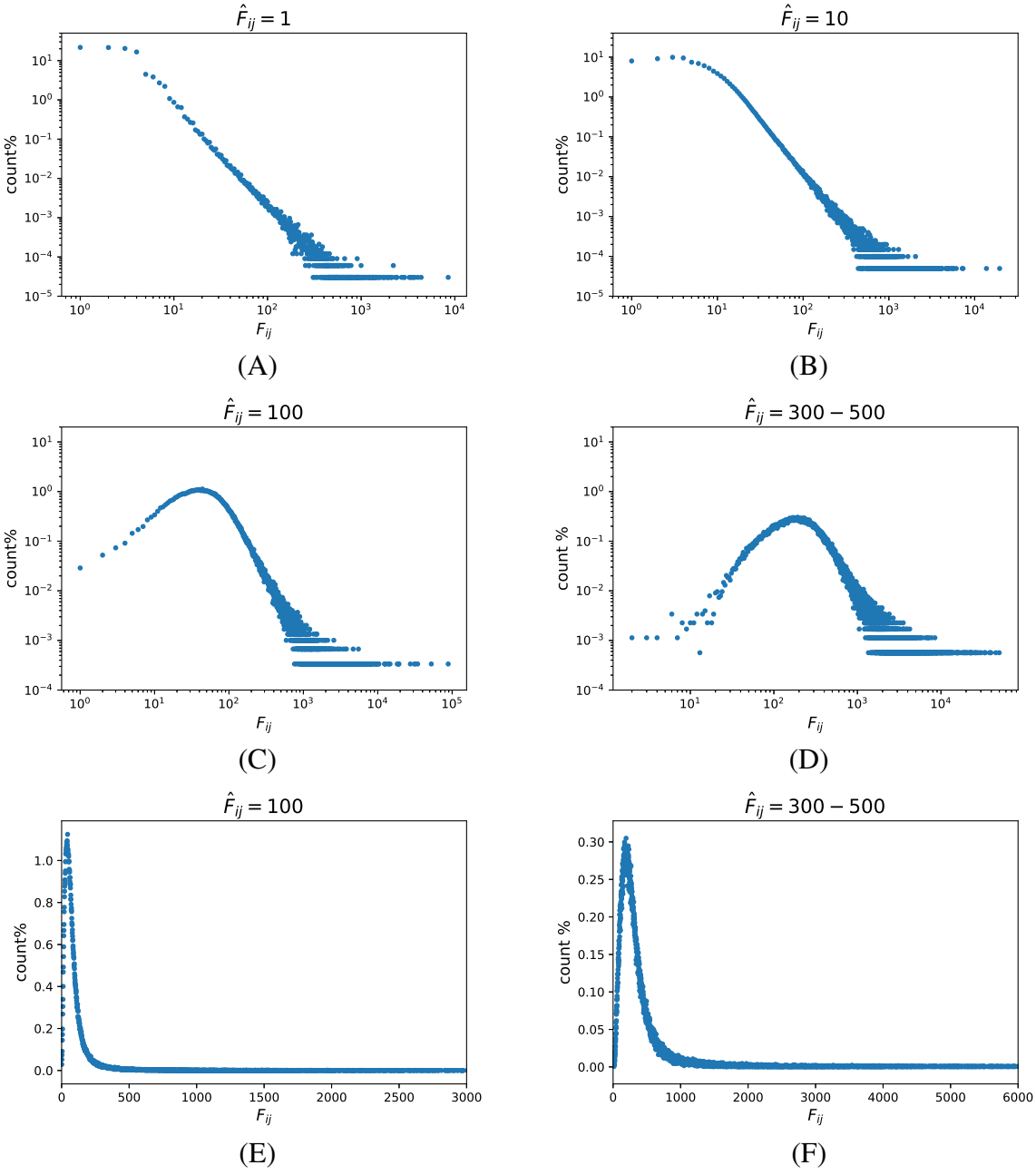
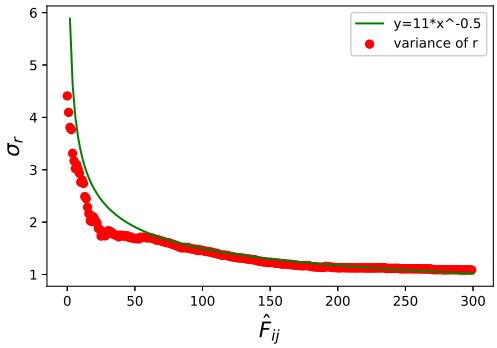
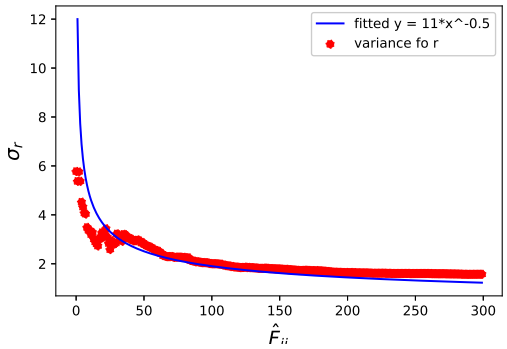


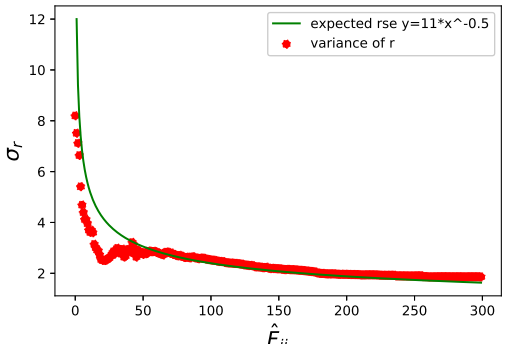
FIGURE 15: The distribution of F_{ij} when $\hat{F}_{ij} = 1, 10, 100 \; and \; 300 - 500$. (E) and (F) are another versions of (C) and (D) without loglog plot. The data set is Wiki-1000.



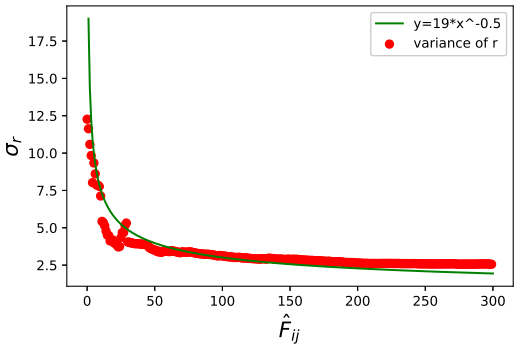
(A): Wiki-100



(B): Wiki-500



(C): Wiki-1000



(D): Reuters

FIGURE 16: σ_r against $\hat{F}_{ij}$ on different corpura.

is different from SPPMI is that DMI no longer shifts by a constant; instead, it shifts by a dynamic value that is dictated by $\sqrt{2}a/\sqrt{F_{ij}}$. When F_{ij} is small, the shift is larger than 5; with the growth of F_{ij}, the shift diminishes. Note that F_{ij} can be very small. In our experiments in the wiki100 data set, the smallest F_{ij} is 1 hence the shift can be much larger than 5.

The differences between DMI, SPPMI and PMI are shown in Figure 12, which is the vector of word "percent". SPPMI shifts PMI by a constant, but the shifting in our DMI is dynamic. Subplot (B) shows the vector values after normalization, large values are enlarged and some values in the DMI vector is even larger than that in original PMI vector, which means important features are emphasized in DMI.

CHAPTER 6

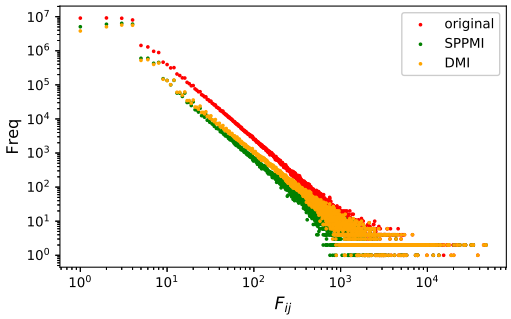
Shifting Schemes

Since DMI and SPPMI both shift the PMI matrix, in this chapter, we will talk about the differences in their shifting schemes.

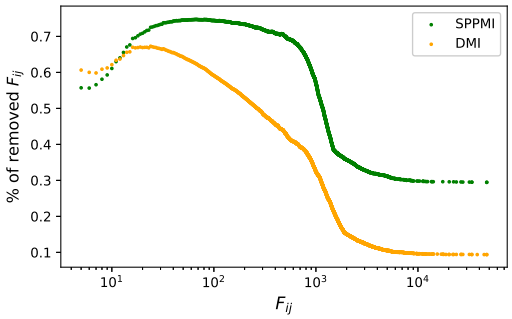
6.1 Word Co-occurrence Distributions

DMI shifts more aggressively than SPPMI when F_{ij} small, which is shown in Figure 17. In the left four subplots (A), (C), (E) and (G), the x-axis represents F_{ij}, and the y-axis is the frequency of F_{ij}, and the red dots represent the original distribution of f before shifting. When the PMI is shifted, all the negative values are abandoned. The green dots represent the distribution of F_{ij} after using the SPPMI shifting scheme and shifting value is 5. The blue dots represent the DMI shifting scheme, but the $a = 11$. As F_{ij} grows, SPPMI abandons more and more pairs than DMI, thus in the left four subplots, the yellow dots are below the green dots when F_{ij} is small, and then the yellow dots are in between the red and green dots, which means DMI shifts very aggressively at first, and as F_{ij} grows, it preserves most of the pairs.

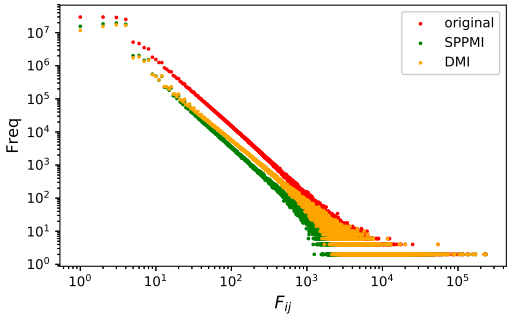
Subplots (B), (D), (F) and (H) show the percentage of removed word pairs in SPPMI and DMI. The SPPMI line increases first and then drops, this strange shape can be explained by the average value of PMI, which is demonstrated in Figure 18. When F_{ij} is less than 100, the average value of PMI keeps dropping, which means more and more word pairs will be eliminated because SPPMI shifts all the PMIs less than $\log 5$. When $F_{ij} > 100$, the average PMI value increases, and SPPMI removes fewer and fewer word pairs. However, because DMI shifts the word pairs dynamically, the DMI line in 17 (B) keeps dropping.



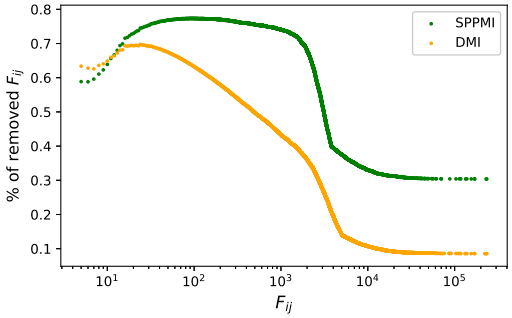
(A): Distribution of F_{ij}.Wiki-100.



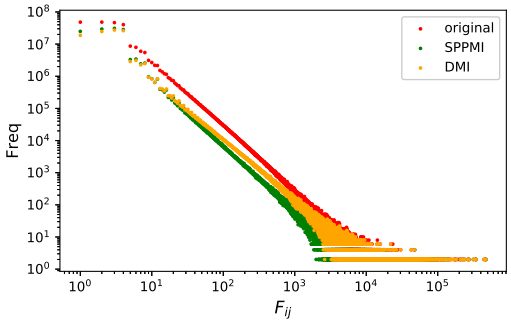
(B): Percentage of F_{ij} removed.Wiki-100.



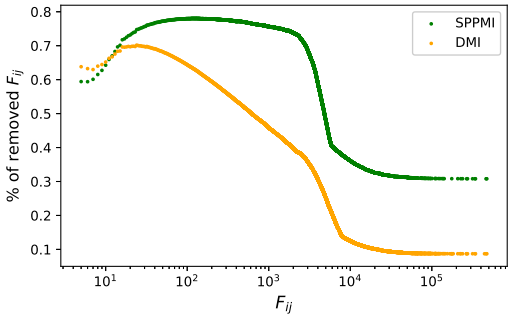
(C): Distribution of F_{ij}.Wiki-500.



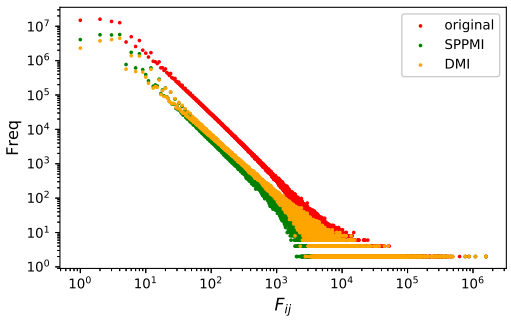
(D): Percentage of F_{ij} removed.Wiki-500.



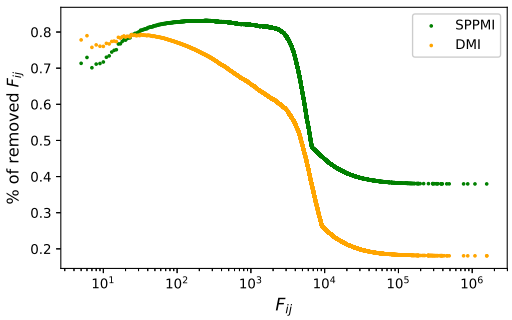
(E): Distribution of F_{ij}.Wiki-1000.



(F): Percentage of F_{ij} removed.Wiki-1000.

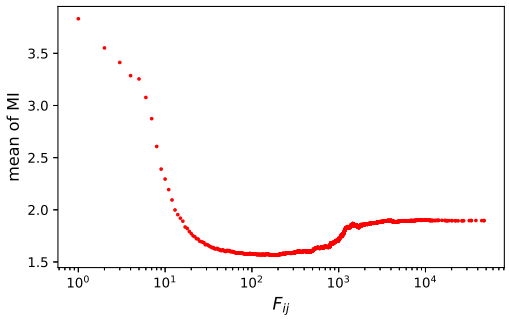


(G): Distribution of F_{ij}.Reuters.

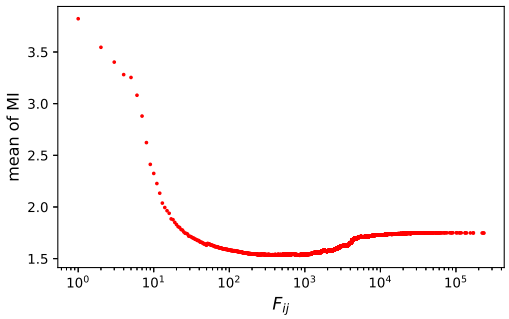


(H): Percentage of F_{ij} removed.Reuters.

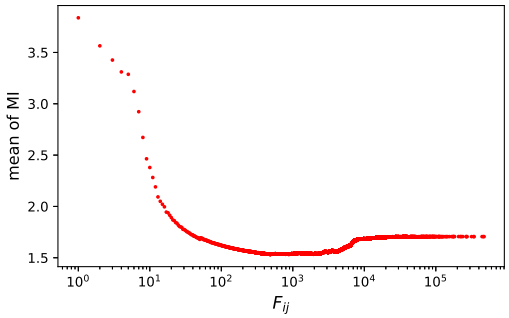
FIGURE 17: Impact of shifting schemes on co-occurrence distribution.



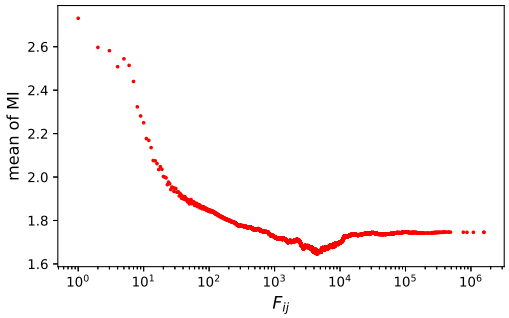
(A): Wiki-100



(B): Wiki-500



(C): Wiki-1000



(D): Reuters

FIGURE 18: Mean of r as F_{ij} increases on different data sets.

F_{ij}	Origin	SPPMI	%	DMI	%
1	9,081,348	5,068,686	55%	3,806,804	42%
2	9,178,167	5,995,860	65%	5,041,020	55%
3	8,996,974	6,379,270	71%	5,677,474	63%
4	8,011,184	6,045,970	75%	5,601,061	70%
10-50	2,694,971	813,692	30%	1,052,846	39%
100-500	178,677	47,624	27%	83,878	47%
> 1000	7,212	2,684	37%	5,123	71%
total	37,882,362	26,806,543	71%	23,431,274	62%

TABLE 16: Frequency of F_{ij} after Shifting on Wiki-100.

All the figures are plotted based on the data set Wiki-100.

The detailed statistics of the frequency of F_{ij} on Wiki-100 are shown in Table 16. Each row represents the frequency of F_{ij} in the original PPMI, SPPMI and our DMI. As the F_{ij} gets larger, DMI preserve more and more word pairs, while SPPMI abandons more pairs. After different shifting schemes are applied, SPPMI removes 45% word pairs when $F_{ij} = 1$, while DMI removes more than half of the pairs. When $F_{ij} > 1000$, about 71% pairs are preserved in DMI, but SPPMI only keeps 37%. In total, our DMI keeps less unique pairs than SPPMI, but we keep more frequent pairs than in SPPMI.

6.2 Word Pair Selection

Since DMI and SPPMI use different shifting schemes, the MI values of the word pairs are also varying. We divide all the word pairs into different groups according to their co-occurrence count. There are 47 distinct word pairs that co-occur more than 10,000 times and 79 word pairs whose F_{ij}s are between 5,000 and 6,000. All the word pairs are listed in Table 17 and 18 respectively. We also list some word pairs with different co-occurrence ranges, but the number of word pairs is too large to list them all, thus we randomly select 100 word pairs from each group, and they are shown in Table 19, 20, 21, 22, 23, 24.

We can see from all these tables that SPPMI and DMI both will preserve highly associated word pairs such as *(francisco san)* and *(lectures university)*. Meanwhile, for the random word pairs, they remove them all, such as *(more while)* and *(he times)*. However,

there are differences in selecting word pairs in both models. When the co-occurrences are not large, some word pairs preserved in SPPMI will be removed in DMI, and as F_{ij} gets larger, the word pairs kept in DMI are eliminated by SPPMI. All such pairs are listed in Table

In order to see the differences between SPPMI and DMI in selecting pairs, we examined all the different pairs that SPPMI and DMI preserved after shifting, that is the pairs only removed by SPPMI or the pairs only removed by DMI, and we use set D to denote all these pairs. When F_{ij} is small, the DMI shifts more aggressively than SPPMI, thus some pairs are preserved in SPPMI but removed in DMI. While F_{ij} gets larger, SPPMI shifts more pairs than DMI.

We randomly select 10 pairs from D in different F_{ij} range and the pairs are listed in Table , and . When F_{ij} is between 1 and 5, two words in each word pairs seems not associated, such as *(surname reunion)* and *(high kot)* but SPPMI preserves these pairs, while DMI remove them all. These word pairs are composed of two irrelevant words with medium word frequency, thus their $\hat{F}_{ij}$ is not large. Therefore, it and can get a PMI larger than 5, though its F_{ij} is small.

When F_{ij} is larger, there are a lot of highly associated pairs removed by SPPMI, like *(occurred within)* and *(games tournament)* while DMI preserves them all.

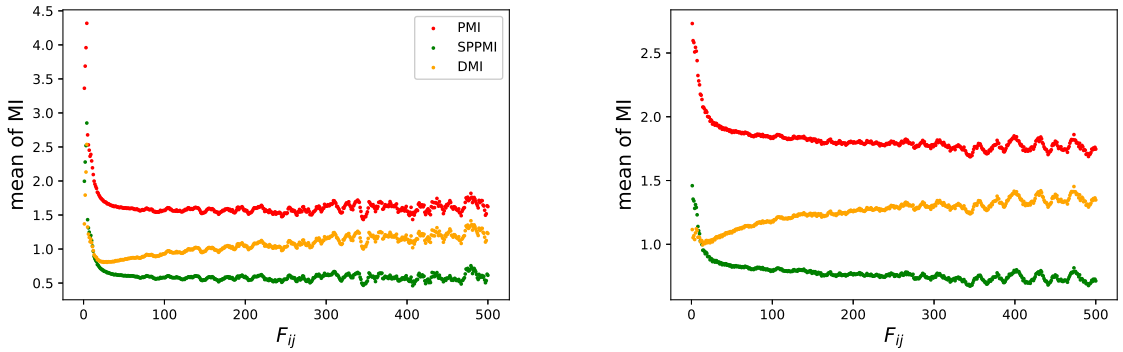
However, when F_{ij} is very lager (in Table 27), for example, larger than 5,000, the shifting value in DMI is extremely small, and some word pairs that are not highly related are also preserved, but the number of such pairs is very small. In this way, our DMI has more advantage over SPPMI in preserving associated pairs.

6.3 Values of Mutual Informations

Moreover, the shifting schemes also have a different impact on the average value of MIs. Figure 19 shows the mean of PMI, SPPMI and DMI when $F_{ij} = 1 - 500$ in dataset Wiki-100. With the growth of F_{ij}, the PMI value drops very quickly at first and keeps steady when F_{ij} is over 100. This shape can be explained by the frequency of the words. When F_{ij} is small, the rare word pairs usually consist of two words with low word frequency, and

Word pair	F_{ij}	PPMI	SPPMI	DMI	Word pair	F_{ij}	PPMI	SPPMI	DMI
(been has)	47,299	3.070	1.461	3.012	(he which)	14,193	0.212	0	0.108
(states united)	47,182	4.813	3.204	4.755	(her she)	14,135	2.054	0.444	1.950
(he his)	44,544	0.846	0	0.786	(career his)	13,788	2.042	0.432	1.937
(new york)	33,812	4.182	2.573	4.114	(had who)	13,091	1.601	0	1.493
(been have)	33,635	3.083	1.474	3.015	(from were)	12,821	0.213	0	0.104
(been had)	31,920	2.684	1.075	2.615	(s women)	12,428	2.094	0.485	1.984
(s u)	27,543	2.664	1.055	2.589	(father his)	12,063	2.570	0.961	2.458
(also he)	26,582	0.990	0	0.913	(also has)	11,993	1.171	0	1.059
(century th)	26,394	4.543	2.933	4.466	(early s)	11,811	1.339	0	1.226
(more than)	23,778	3.806	2.196	3.725	(after his)	11,560	0.670	0	0.556
(he where)	22,350	1.817	0.208	1.734	(may refer)	11,256	4.519	2.910	4.404
(had he)	21,976	0.883	0	0.798	(his wife)	11,250	2.863	1.254	2.748
(he s)	21,589	0	0	0	(he played)	11,180	1.592	0	1.476
(high school)	20,266	3.786	2.176	3.699	(also known)	11,178	2.116	0.507	2.001
(from he)	19,727	0	0	0	(had which)	11,125	0.953	0	0.837
(s s)	19,049	0	0	0	(one s)	11,028	0.179	0	0.062
(first his)	17,908	0.938	0	0.845	(became he)	10,748	1.286	0	1.168
(has he)	17,603	0.653	0	0.559	(his own)	10,368	2.262	0.652	2.141
(war world)	17,508	3.728	2.118	3.634	(first he)	10,313	0.144	0	0.024
(from s)	17,165	0	0	0	(were which)	10,159	0.544	0	0.423
(he when)	16,917	1.180	0	1.085	(during s)	10,116	0.575	0	0.453
(from his)	16,702	0.051	0	0	(from from)	10,062	0	0	0
(first s)	15,477	0.414	0	0.315	(death his)	10,026	2.170	0.560	2.048
(his s)	15,199	0	0	0					

TABLE 17: PPMI, SPPMI and DMI of word pairs whose $F_{ij} > 10,000$. Dataset: Wiki-100.



(A): Wiki-100 (B): Reuters

FIGURE 19: Mean of different MIs when $F_{ij} = 1 - 500$.

Word pair	F_{ij}	PPMI	SPPMI	DMI	Word pair	F_{ij}	PPMI	SPPMI	DMI
(from her)	5,999	0.0693	0	0	(national team)	5,429	2.563	0.954	2.401
(air force)	5,980	4.734	3.124	4.578	(between two)	5,417	1.684	0.074	1.521
(football league)	5,977	3.730	2.121	3.575	(income median)	5,362	6.594	4.984	6.430
(he worked)	5,966	1.827	0.218	1.672	(he season)	5,357	0.556	0	0.392
(minister prime)	5,920	6.083	4.474	5.927	(she where)	5,356	1.702	0.093	1.539
(during world)	5,915	2.066	0.456	1.909	(age from)	5,343	1.284	0	1.120
(same year)	5,893	2.868	1.258	2.71157	(after from)	5,306	0	0	0
(his who)	5,883	0.057	0	0	(career he)	5,291	0.842	0	0.678
(he university)	5,870	0.602	0	0.445	(from one)	5,286	0	0	0
(census population)	5,853	4.608	2.999	4.451	(best known)	5,281	3.108	1.498	2.943
(her her)	5,846	1.140	0	0.983	(away from)	5,273	2.153	0.543	1.988
(band s)	5,840	1.106	0	0.949	(his until)	5,259	0.977	0	0.812
(father s)	5,788	1.458	0	1.300	(until when)	5,258	2.279	0.670	2.115
(his one)	5,777	0	0	0	(s when)	5,252	0	0	0
(city s)	5,746	0.321	0	0.162	(after war)	5,252	1.806	0.197	1.641
(he received)	5,745	1.319	0	1.161	(cup world)	5,246	3.492	1.882	3.327
(has s)	5,744	0	0	0	(jersey new)	5,223	4.015	2.406	3.850
(death s)	5,734	1.233	0	1.074	(first from)	5,223	0	0	0
(made were)	5,715	1.192	0	1.033	(first one)	5,205	0.567	0	0.402
(had she)	5,659	0.840	0	0.680	(average size)	5,191	5.508	3.899	5.342
(from new)	5,656	0	0	0	(olympics summer)	5,178	5.976	4.366	5.810
(he two)	5,640	0	0	0	(also which)	5,165	0.103	0	0
(he her)	5,616	0	0	0	(three years)	5,127	2.124	0.515	1.957
(francisco san)	5,606	6.180	4.571	6.020	(same time)	5,124	2.498	0.889	2.331
(been since)	5,597	2.210	0.601	2.050	(his mother)	5,119	2.210	0.601	2.043
(album released)	5,590	3.171	1.561	3.010	(during he)	5,116	0.030	0	0
(de la)	5,579	4.266	2.657	4.106	(company s)	5,109	0.670	0	0.503
(during time)	5,570	1.696	0.086	1.535	(africa south)	5,107	4.368	2.759	4.201
(people s)	5,554	0.621	0	0.460	(he moved)	5,105	1.452	0	1.285
(did he)	5,537	1.279	0	1.118	(his known)	5,104	0.672	0	0.505
(old year)	5,524	3.089	1.479	2.927	(e i)	5,074	3.135	1.526	2.968
(he new)	5,507	0	0	0	(civil war)	5,054	4.199	2.589	4.031
(over years)	5,504	2.098	0.488	1.936	(he returned)	5,035	1.679	0.0698	1.511
(each other)	5,486	2.498	0.888	2.336	(born life)	5,028	2.787	1.177	2.619
(been he)	5,485	0	0	0	(known well)	5,016	2.543	0.933	2.374
(film s)	5,479	0.439	0	0.277	(life personal)	5,013	4.258	2.648	4.089
(she when)	5,444	1.360	0	1.198	(appointed he)	5,008	1.868	0.258	1.699
(he joined)	5,444	1.749	0.140	1.587	(its own)	5,006	2.446	0.836	2.277
(made up)	5,441	2.292	0.682	2.130	(first season)	5,006	1.492	0	1.323
(population were)	5,433	1.738	0.129	1.576					

TABLE 18: PPMI, SPPMI and DMI of word pairs whose $5,000 < F_{ij} < 6,000$. Dataset: Wiki-100.

Word pair	F_{ij}	PPMI	SPPMI	DMI	Word pair	F_{ij}	PPMI	SPPMI	DMI
(church st)	1,875	2.589	0.980	2.327	(population size)	1,241	3.278	1.669	2.964
(c he)	1,872	0.181	0	0	(he times)	1,233	0.008	0	0
(has made)	1,867	0.383	0	0.120	(political s)	1,228	0	0	0
(after second)	1,809	0.655	0	0.388	(her role)	1,223	1.430	0	1.114
(two weeks)	1,784	2.816	1.207	2.548	(follow up)	1,223	3.485	1.875	3.169
(according s)	1,748	0.136	0	0	(countries from)	1,219	0.847	0	0.531
(state york)	1,739	1.252	0	0.981	(he tells)	1,214	1.546	0	1.229
(street th)	1,700	1.567	0	1.293	(episode s)	1,199	0.425	0	0.106
(now s)	1,678	0	0	0	(played two)	1,197	0.651	0	0.332
(also member)	1,674	0.829	0	0.553	(do he)	1,182	0.075	0	0
(international s)	1,631	0	0	0	(also received)	1,171	0.631	0	0.309
(out spread)	1,628	3.718	2.108	3.438	(although were)	1,162	0.453	0	0.130
(has new)	1,601	0	0	0	(first out)	1,160	0	0	0
(care health)	1,559	4.872	3.263	4.588	(award received)	1,158	2.759	1.150	2.436
(goals scored)	1,555	5.164	3.555	4.879	(area part)	1,152	0.582	0	0.258
(t wasn)	1,547	5.836	4.227	5.551	(most some)	1,154	0.735	0	0.411
(b c)	1,523	2.876	1.267	2.589	(historic listed)	1,154	4.523	2.914	4.199
(published were)	1,512	0.760	0	0.471	(round third)	1,144	3.001	1.392	2.676
(s served)	1,506	0	0	0	(books published)	1,138	3.431	1.821	3.105
(c c)	1,503	2.708	1.098	2.419	(civil during)	1,132	2.126	0.517	1.800
(played team)	1,502	1.664	0.054	1.374	(band members)	1,126	2.381	0.772	2.054
(education school)	1,494	1.148	0	0.858	(around he)	1,114	0	0	0
(located near)	1,464	2.761	1.151	2.468	(division league)	1,112	1.577	0	1.248
(first goal)	1,463	2.070	0.461	1.778	(most notable)	1,110	2.689	1.080	2.360
(post war)	1,460	2.704	1.094	2.411	(had them)	1,108	0.169	0	0
(one over)	1,456	0.089	0	0	(earth s)	1,101	0.909	0	0.578
(his led)	1,449	0.428	0	0.134	(he j)	1,100	0.508	0	0.177
(coast east)	1,442	3.485	1.876	3.191	(been more)	1,100	0.079	0	0
(he minister)	1,425	0.571	0	0.275	(county seat)	1,095	3.195	1.586	2.864
(association s)	1,407	0.202	0	0	(new who)	1,093	0	0	0
(each s)	1,406	0	0	0	(males over)	1,093	2.881	1.272	2.549
(body his)	1,396	0.841	0	0.542	(called were)	1,091	0	0	0
(he once)	1,394	0.457	0	0.159	(moved s)	1,087	0	0	0
(changed its)	1,394	2.181	0.571	1.882	(he post)	1,084	0.239	0	0
(film which)	1,384	0	0	0	(all games)	1,072	1.206	0	0.871
(also team)	1,375	0.107	0	0	(eight years)	1,071	2.250	0.640	1.915
(from president)	1,370	0.103	0	0	(playing role)	1,063	3.343	1.734	3.008
(award from)	1,365	0.347	0	0.045	(council national)	1,056	1.231	0	0.895
(competed summer)	1,353	4.745	3.135	4.442	(include which)	1,054	0.426	0	0.089
(between which)	1,306	0	0	0	(his united)	1,049	0	0	0
(during she)	1,303	0	0	0	(her where)	1,045	0.038	0	0
(p s)	1,302	0.433	0	0.126	(chart uk)	1,038	4.459	2.849	4.120
(from game)	1,276	0	0	0	(s uk)	1,037	0.282	0	0
(husband s)	1,267	1.038	0	0.727	(one week)	1,035	1.626	0.016	1.286
(june released)	1,266	2.005	0.396	1.694	(short story)	1,032	3.362	1.752	3.022
(all members)	1,265	1.337	0	1.025	(have than)	1,030	0.252	0	0
(his said)	1,263	0.170	0	0	(against war)	1,026	1.257	0	0.916
(been some)	1,263	0.386	0	0.074	(his order)	1,024	0	0	0
(one top)	1,253	0.363	0	0.050	(five s)	1,023	0	0	0
(also while)	1,243	0	0	0	(career has)	1,005	0.159	0	0

TABLE 19: PPMI, SPPMI and DMI of 100 randomly selected word pairs whose $1,000 < F_{ij} < 2,000$. Dataset: Wiki-100.

Word pair	F_{ij}	PPMI	SPPMI	DMI	Word pair	F_{ij}	PPMI	SPPMI	DMI
(from highway)	599	0.437	0	0.011	(asked his)	549	0.691	0	0.249
(festival held)	598	2.544	0.935	2.118	(from studies)	548	0.067	0	0
(august june)	597	1.527	0	1.100	(champion he)	548	0.412	0	0
(ran which)	595	1.068	0	0.640	(awards year)	548	1.471	0	1.029
(more while)	595	0.059	0	0	(february new)	547	0.193	0	0
(history series)	593	0.779	0	0.351	(done have)	546	1.724	0.114	1.282
(january march)	592	1.420	0	0.992	(brought his)	546	0.250	0	0
(formed new)	592	0.858	0	0.430	(founded who)	539	0.658	0	0.214
(league play)	589	0.978	0	0.549	(case his)	539	0	0	0
(formed part)	588	1.690	0.081	1.261	(get him)	538	1.781	0.172	1.336
(t want)	587	3.827	2.218	3.398	(blue s)	538	0	0	0
(doubles s)	587	1.520	0	1.091	(assistant professor)	538	4.159	2.550	3.714
(from spain)	584	0.663	0	0.232	(one said)	537	0.179	0	0
(his list)	583	0	0	0	(has line)	537	0	0	0
(technology university)	582	2.033	0.424	1.602	(cancer died)	536	3.697	2.088	3.252
(eastern europe)	582	3.267	1.658	2.836	(county washington)	534	1.522	0	1.076
(s x)	581	0	0	0	(engineering university)	533	2.321	0.712	1.875
(e romanized)	581	4.273	2.664	3.842	(from official)	532	0	0	0
(his official)	579	0.052	0	0	(host show)	530	3.172	1.562	2.724
(also public)	578	0	0	0	(haven new)	530	3.196	1.586	2.748
(point where)	577	1.300	0	0.867	(acting his)	530	0.757	0	0.309
(comedy series)	577	2.901	1.292	2.469	(its site)	527	0.551	0	0.102
(children young)	577	2.354	0.744	1.921	(after king)	527	0.348	0	0
(remains s)	574	0.173	0	0	(captured were)	524	1.302	0	0.852
(against out)	574	0.443	0	0.009	(over presided)	523	4.570	2.960	4.120
(being who)	572	0	0	0	(he numerous)	523	0.094	0	0
(who who)	571	0	0	0	(most years)	520	0	0	0
(miles north)	571	2.633	1.024	2.199	(admiral rear)	519	6.133	4.523	5.681
(change s)	571	0	0	0	(late mid)	517	2.950	1.341	2.498
(however would)	570	0.336	0	0	(championship final)	515	1.433	0	0.979
(city residing)	570	3.597	1.987	3.162	(general hospital)	515	2.451	0.842	1.998
(birth her)	568	2.048	0.439	1.613	(states when)	513	0	0	0
(final third)	567	1.798	0.189	1.363	(cup match)	513	2.652	1.043	2.199
(over river)	565	0.722	0	0.286	(old time)	512	0.481	0	0.027
(companies other)	565	1.622	0.012	1.186	(goals league)	512	2.605	0.996	2.151
(australian he)	565	0	0	0	(chinese from)	512	0.078	0	0
(had held)	564	0	0	0	(its water)	511	0.252	0	0
(fourth he)	564	0	0	0	(government officials)	510	3.177	1.567	2.722
(can time)	562	0	0	0	(go out)	510	1.638	0.028	1.183
(december march)	560	0.762	0	0.324	(occupied were)	509	1.490	0	1.035
(county state)	560	0.542	0	0.105	(front line)	508	2.364	0.754	1.908
(day last)	559	1.561	0	1.123	(second while)	507	0.233	0	0
(century eighteenth)	556	5.318	3.709	4.879	(building office)	507	2.060	0.451	1.605
(ended when)	554	1.709	0.100	1.270	(husband together)	506	3.515	1.906	3.059
(goal one)	553	1.201	0	0.761	(bar he)	505	0.589	0	0.133
(army served)	553	1.232	0	0.792	(california his)	504	0	0	0
(games points)	552	2.515	0.906	2.075	(between divided)	503	2.345	0.735	1.888
(up year)	551	0	0	0	(team tournament)	502	1.932	0.322	1.474
(plot s)	551	0.227	0	0	(she under)	502	0	0	0
(deal signed)	551	3.820	2.210	3.379	(film who)	502	0	0	0

TABLE 20: PPMI, SPPMI and DMI of 100 randomly selected word pairs whose $500 < F_{ij} < 600$. Dataset: Wiki-100.

Word pair	F_{ij}	PPMI	SPPMI	DMI	Word pair	F_{ij}	PPMI	SPPMI	DMI
(continues route)	198	3.086	1.477	2.432	(community largest)	126	1.250	0	0.481
(apparent s)	197	0.540	0	0	(ny state)	125	1.487	0	0.716
(news sports)	196	2.496	0.887	1.839	(has secondary)	125	0	0	0
(award supporting)	193	2.905	1.296	2.245	(hit its)	124	0	0	0
(order when)	189	0	0	0	(includes research)	123	1.358	0	0.583
(man may)	189	0	0	0	(forward were)	123	0	0	0
(concept his)	188	0	0	0	(birth given)	123	2.352	0.742	1.576
(book three)	184	0	0	0	(release when)	121	0	0	0
(bus transport)	178	3.823	2.214	3.143	(even perhaps)	121	2.586	0.976	1.805
(continued th)	173	0.173	0	0	(founded since)	120	0.172	0	0
(native two)	172	0.076	0	0	(fantasy game)	120	2.410	0.800	1.628
(he relations)	172	0	0	0	(attacked she)	120	0.706	0	0
(school summer)	169	0.113	0	0	(own style)	119	0.965	0	0.181
(games tournament)	169	1.583	0	0.890	(after organization)	119	0	0	0
(computer used)	167	1.211	0	0.515	(advisory national)	119	2.048	0.439	1.263
(challenge league)	162	2.037	0.427	1.333	(airplay chart)	118	5.377	3.768	4.590
(cult following)	160	3.102	1.492	2.395	(from hungarian)	117	0	0	0
(policy state)	159	0.882	0	0.173	(establishment first)	117	0.434	0	0
(majority senate)	157	3.295	1.686	2.583	(colleagues her)	117	1.801	0.191	1.012
(kilometres south)	154	3.014	1.405	2.297	(expanded further)	116	2.528	0.918	1.736
(appointed captain)	154	2.359	0.749	1.642	(his strategy)	115	0.025	0	0
(group six)	153	0.322	0	0	(ethnic population)	114	2.162	0.552	1.365
(formal more)	153	1.673	0.063	0.954	(episodes ten)	114	2.742	1.133	1.946
(another just)	152	0.640	0	0	(critical one)	114	0	0	0
(example how)	150	1.746	0.137	1.023	(award team)	113	0	0	0
(he honors)	148	0.220	0	0	(most numerous)	112	0.421	0	0
(held members)	146	0.339	0	0	(historical were)	112	0	0	0
(after revealed)	146	0.471	0	0	(french guiana)	112	5.190	3.581	4.389
(force out)	145	0.034	0	0	(due postponed)	112	4.099	2.489	3.298
(result war)	143	0.473	0	0	(island road)	111	0.617	0	0
(its schedule)	143	1.143	0	0.408	(deaths resulted)	111	4.559	2.949	3.755
(closed july)	142	1.479	0	0.741	(living standard)	109	1.808	0.199	0.999
(key public)	140	1.497	0	0.756	(knee left)	109	3.264	1.655	2.456
(band uk)	139	1.079	0	0.336	(death personal)	109	0.961	0	0.152
(appear only)	137	1.017	0	0.270	(held society)	108	0.502	0	0
(prep school)	136	3.688	2.078	2.939	(has organisation)	108	0.468	0	0
(father years)	136	0	0	0	(spread were)	107	0	0	0
(local tax)	135	1.935	0.326	1.185	(from mill)	107	0	0	0
(club clubs)	134	1.623	0.014	0.870	(produced would)	106	0	0	0
(returned top)	133	0.961	0	0.206	(equation s)	106	0.247	0	0
(however order)	133	0	0	0	(produce two)	105	0.211	0	0
(career french)	132	0.115	0	0	(direct would)	105	0.664	0	0
(history middle)	131	0.592	0	0	(breaking news)	105	3.469	1.860	2.650
(along areas)	130	0.911	0	0.150	(fork south)	104	3.106	1.496	2.284
(campaigns s)	128	0.100	0	0	(each later)	104	0	0	0
(announced released)	128	0.386	0	0	(century series)	104	0	0	0
(second signed)	127	0.280	0	0	(playing where)	103	0	0	0
(between society)	127	0	0	0	(allow other)	103	0.290	0	0
(operate services)	126	2.949	1.339	2.179	(lectures university)	102	2.375	0.765	1.547
(late took)	126	0.436	0	0	(about usually)	101	0.106	0	0

TABLE 21: PPMI, SPPMI and DMI of 100 randomly selected word pairs whose $100 < F_{ij} < 200$. Dataset: Wiki-100.

Word pair	F_{ij}	PPMI	SPPMI	DMI	Word pair	F_{ij}	PPMI	SPPMI	DMI
(solo successful)	59	2.208	0.598	1.217	(assignment s)	55	0	0	0
(part ten)	59	0	0	0	(serie side)	54	2.903	1.293	1.884
(number study)	59	0	0	0	(petition presented)	54	4.015	2.406	2.997
(he senators)	59	0	0	0	(part systems)	54	0	0	0
(first serbia)	59	0.398	0	0	(p produced)	54	0.521	0	0
(early queensland)	59	0.969	0	0	(mines which)	54	0.055	0	0
(different programs)	59	0.881	0	0	(lands over)	54	0.578	0	0
(born founder)	59	0.462	0	0	(films language)	54	1.003	0	0
(analysis first)	59	0	0	0	(features season)	54	0	0	0
(abuse allegations)	59	5.257	3.648	4.267	(crustaceans small)	54	4.339	2.729	3.320
(price share)	58	3.008	1.398	2.012	(both lack)	54	0.431	0	0
(old spanish)	58	0.508	0	0	(assembly called)	54	0.177	0	0
(next points)	58	0.579	0	0	(roads roads)	53	3.391	1.781	2.366
(he prepare)	58	0	0	0	(one organizers)	53	1.901	0.291	0.876
(few women)	58	0.094	0	0	(miami where)	53	0.878	0	0
(early writer)	58	0.177	0	0	(included news)	53	0.467	0	0
(controlled radio)	58	1.990	0.380	0.994	(exhibited salon)	53	6.142	4.533	5.118
(all steel)	58	0.053	0	0	(entered industry)	53	1.565	0	0.541
(newer were)	57	0.744	0	0	(divided over)	53	0	0	0
(media what)	57	0.372	0	0	(born hunter)	53	1.272	0	0.247
(from hampton)	57	0.423	0	0	(along villages)	53	1.205	0	0.180
(campaign supported)	57	1.794	0.184	0.792	(support working)	52	0.289	0	0
(archaeology university)	57	2.451	0.842	1.450	(released small)	52	0	0	0
(announced working)	57	0.557	0	0	(merely who)	52	0.918	0	0
(angels from)	57	0	0	0	(likely two)	52	0	0	0
(administrative province)	57	1.611	0.001	0.609	(increase other)	52	0	0	0
(white work)	56	0	0	0	(included place)	52	0	0	0
(towards women)	56	0.745	0	0	(his tenor)	52	0.603	0	0
(owner private)	56	1.339	0	0.333	(games north)	52	0	0	0
(second students)	56	0	0	0	(force upon)	52	0.492	0	0
(pass she)	56	0	0	0	(demonstrating pattern)	52	5.655	4.046	4.624
(opened united)	56	0	0	0	(defeat round)	52	1.554	0	0.524
(known pro)	56	0.301	0	0	(composer first)	52	0	0	0
(joined senior)	56	0.912	0	0	(competing mainly)	52	3.257	1.648	2.226
(i includes)	56	0	0	0	(coastal located)	52	1.437	0	0.406
(codes which)	56	0.578	0	0	(billion revenues)	52	4.926	3.317	3.896
(cambridge research)	56	1.876	0.266	0.869	(grants received)	52	1.591	0	0.560
(been energy)	56	0	0	0	(attend church)	52	1.492	0	0.461
(became hits)	56	0.824	0	0	(also mills)	52	0.211	0	0
(australia used)	56	0	0	0	(shirt wearing)	51	5.370	3.761	4.333
(alongside series)	56	0.644	0	0	(occurred within)	51	1.100	0	0.063
(against operation)	56	0.023	0	0	(laws those)	51	1.241	0	0.204
(officers service)	55	0.896	0	0	(coming team)	51	0.311	0	0
(he presidents)	55	0	0	0	(capacity generating)	51	4.367	2.758	3.330
(district last)	55	0	0	0	(between less)	51	0	0	0
(department st)	55	0	0	0	(benjamin first)	51	0.317	0	0
(defined under)	55	0.410	0	0	(any research)	51	0	0	0
(church five)	55	0	0	0	(allows you)	51	1.551	0	0.514
(championship japan)	55	0.838	0	0	(all colors)	51	0.797	0	0
(bc one)	55	0	0	0	(all cannot)	51	0	0	0

TABLE 22: PPMI, SPPMI and DMI of 100 randomly selected word pairs whose $50 < F_{ij} < 60$. Dataset: Wiki-100.

Word pair	F_{ij}	PPMI	SPPMI	DMI	Word pair	F_{ij}	PPMI	SPPMI	DMI
(wales water)	19	0.001	0	0	(family quarters)	13	0.466	0	0
(carbon reduce)	19	3.090	1.480	1.708	(fails pay)	13	3.245	1.636	1.718
(allows protocol)	19	3.334	1.725	1.952	(exposed some)	13	0	0	0
(ye zilayi)	18	7.179	5.570	5.777	(dynasty mc)	13	3.776	2.167	2.249
(skill training)	18	2.253	0.643	0.851	(downstream km)	13	2.256	0.647	0.729
(rock surrounding)	18	0.730	0	0	(donated organization)	13	1.520	0	0
(preschool serves)	18	4.663	3.053	3.260	(defeated major)	13	0	0	0
(player self)	18	0	0	0	(decided surrender)	13	2.048	0.439	0.521
(either programs)	18	0.367	0	0	(chrzanw district)	13	3.789	2.179	2.261
(california instead)	18	0	0	0	(available saloon)	13	2.975	1.365	1.448
(rolle s)	17	1.155	0	0	(abuse scandals)	13	5.763	4.153	4.235
(reduced world)	17	0	0	0	(satellite years)	12	0	0	0
(hamilton lord)	17	1.789	0.180	0.366	(right smith)	12	0	0	0
(continued paper)	17	0.399	0	0	(regions written)	12	0.157	0	0
(contacts him)	17	0.979	0	0	(pond woods)	12	3.838	2.229	2.279
(cemetery century)	17	0	0	0	(kirovohrad zirka)	12	13.148	11.539	11.590
(body regulating)	17	3.438	1.829	2.014	(japanese return)	12	0	0	0
(resulting under)	16	0	0	0	(group womens)	12	1.041	0	0
(porno s)	16	1.439	0	0	(gang kids)	12	3.084	1.474	1.525
(numerous scene)	16	0.929	0	0	(ferry victoria)	12	2.265	0.655	0.706
(escort had)	16	0	0	0	(females transportation)	12	1.255	0	0
(could prevail)	16	3.580	1.971	2.133	(expeditionthe research)	12	5.805	4.196	4.247
(city safe)	16	0	0	0	(county fariman)	12	5.118	3.509	3.559
(called dedicated)	16	0	0	0	(centre parts)	12	0	0	0
(attack henry)	16	0.285	0	0	(basic shared)	12	1.698	0.089	0.139
(adjusting difficulty)	16	6.051	4.441	4.604	(affiliated cma)	12	6.231	4.622	4.672
(variant version)	15	1.606	0	0.134	(taylor walter)	11	1.983	0.373	0.389
(requested troops)	15	2.363	0.753	0.891	(studio write)	11	0.795	0	0
(mustered united)	15	2.035	0.426	0.563	(similar supports)	11	1.042	0	0
(had judaism)	15	0	0	0	(most tally)	11	1.380	0	0
(curve known)	15	0.553	0	0	(leicestershire northamptonshire)	11	6.547	4.938	4.954
(bulls during)	15	0.482	0	0	(innings many)	11	0	0	0
(based oklahoma)	15	0.168	0	0	(however throw)	11	0.374	0	0
(access served)	15	0	0	0	(however masses)	11	0.915	0	0
(respectively students)	14	0.039	0	0	(his ramblings)	11	2.889	1.280	1.296
(person uses)	14	0.446	0	0	(gladiators gloster)	11	9.072	7.463	7.479
(julia old)	14	1.377	0	0	(george shortly)	11	0.055	0	0
(events sponsors)	14	2.205	0.596	0.707	(fox provided)	11	0.623	0	0
(estimates since)	14	0.638	0	0	(dark official)	11	0.147	0	0
(due ontario)	14	0	0	0	(collaboration established)	11	0.432	0	0
(community gulf)	14	0.593	0	0	(castle transformed)	11	2.265	0.656	0.672
(both designation)	14	0.302	0	0	(breguet louis)	11	5.397	3.787	3.803
(queens rockaway)	13	6.481	4.872	4.954	(before colorado)	11	0	0	0
(my practice)	13	0.035	0	0	(bedford general)	11	1.275	0	0
(manitoba were)	13	0	0	0	(authority marine)	11	0.844	0	0
(like mole)	13	2.161	0.552	0.634	(association socialist)	11	0.606	0	0
(kadima party)	13	4.916	3.306	3.388	(assistant chicago)	11	0.456	0	0
(internationally widely)	13	2.325	0.715	0.798	(along ceremony)	11	0	0	0
(home idea)	13	0	0	0	(airport band)	11	0	0	0
(heard july)	13	0	0	0	(afghanistan current)	11	1.095	0	0

TABLE 23: PPMI, SPPMI and DMI of 100 randomly selected word pairs whose $10 < F_{ij} < 20$. Dataset: Wiki-100.

Word pair	F_{ij}	PPMI	SPPMI	DMI	Word pair	F_{ij}	PPMI	SPPMI	DMI
(noordam ray)	4	7.664	6.055	5.649	(jr rachette)	2	6.775	5.166	4.453
(naturalists other)	4	1.373	0	0	(jimmy use)	2	0	0	0
(member qurays)	4	5.625	4.016	3.611	(jersey text)	2	0	0	0
(lighted rings)	4	5.017	3.407	3.002	(surname reunion)	2	1.768	0.159	0
(irbene naming)	4	8.888	7.279	6.874	(herself soo)	2	2.660	1.050	0.338
(impact seeing)	4	1.187	0	0	(excavations may)	2	0	0	0
(hereford tomb)	4	4.299	2.689	2.284	(eps individually)	2	4.293	2.684	1.972
(held sloot)	4	4.595	2.985	2.580	(enhanced levels)	2	0.934	0	0
(geological weakness)	4	4.284	2.675	2.269	(done wishes)	2	0.928	0	0
(fr leibesbungen)	4	8.844	7.235	6.829	(dniester located)	2	2.416	0.807	0.095
(forced renounce)	4	3.655	2.045	1.640	(companyj edition)	2	6.348	4.739	4.026
(detention missions)	4	2.799	1.190	0.784	(chatterjee others)	2	2.734	1.125	0.413
(copious creative)	4	4.714	3.104	2.699	(cervera ibn)	2	5.739	4.130	3.417
(conquered gymnastics)	4	4.028	2.419	2.013	(candidate institute)	2	0	0	0
(cineplex total)	4	4.575	2.965	2.560	(boyd forced)	2	1.280	0	0
(cilangkap south)	4	5.141	3.532	3.126	(bits graph)	2	2.943	1.334	0.622
(churrigueresque gothic)	4	7.506	5.897	5.491	(binds tunnels)	2	4.367	2.757	2.045
(blatter resigned)	4	5.632	4.022	3.617	(barashashi talendranath)	2	14.065	12.455	11.743
(biological values)	4	1.554	0	0	(auditions ensemble)	2	3.453	1.844	1.132
(believed shenandoah)	4	3.139	1.529	1.124	(assigned paratrooper)	2	4.337	2.727	2.015
(alternately mcgregor)	4	6.655	5.045	4.640	(antoniaceae genera)	2	8.459	6.849	6.137
(abc ny)	4	1.869	0.259	0	(aimery politically)	2	7.737	6.127	5.415
(use verona)	3	0.756	0	0	(trastevere two)	1	1.542	0	0
(movement sachlichkeit)	3	6.448	4.838	4.307	(other zpass)	1	0.925	0	0
(magnetism ross)	3	4.715	3.106	2.574	(letter shipped)	1	0.671	0	0
(leaves rematch)	3	2.494	0.884	0.353	(katmandoules roads)	1	6.396	4.787	3.757
(job serviceman)	3	4.410	2.801	2.269	(jr previous)	1	0	0	0
(hold peabody)	3	2.743	1.134	0.603	(involuntarily robbie)	1	6.233	4.623	3.594
(high kot)	3	1.627	0.018	0	(intense late)	1	0	0	0
(generosity segerstroms)	3	10.686	9.076	8.545	(holdings penydarren)	1	6.864	5.255	4.225
(end safh)	3	4.619	3.010	2.479	(her sarala)	1	1.533	0	0
(durocher managed)	3	4.177	2.567	2.036	(headmaster pereira)	1	4.280	2.671	1.641
(credibility threats)	3	4.187	2.578	2.046	(greece plague)	1	1.573	0	0
(commercial conversely)	3	1.926	0.316	0	(fields pharmacy)	1	1.131	0	0
(cds traced)	3	3.839	2.230	1.699	(divisional separate)	1	0.501	0	0
(brooke matt)	3	3.062	1.452	0.921	(content granules)	1	2.881	1.271	0.242
(avoids occupation)	3	3.733	2.123	1.592	(concinnatus sionensi)	1	13.371	11.762	10.732
(auxiliary jay)	3	2.567	0.958	0.426	(composite tapestries)	1	4.586	2.976	1.947
(authorities memorial)	3	0.067	0	0	(collaborated kids)	1	0.989	0	0
(asturias herbrugerarturo)	3	10.063	8.454	7.923	(coal mayor)	1	0	0	0
(ajna published)	3	4.912	3.303	2.771	(champions friendly)	1	0	0	0
(agreed almanacs)	3	5.040	3.431	2.899	(catus which)	1	1.288	0	0
(addiction bias)	3	4.008	2.398	1.867	(brainerd scandal)	1	4.921	3.312	2.282
(rural tomb)	2	0.285	0	0	(bounding tribune)	1	5.512	3.903	2.873
(producer standing)	2	0	0	0	(bankeraceae species)	1	4.486	2.876	1.847
(process stirling)	2	0.837	0	0	(council suribachi)	1	2.409	0.800	0
(plot ritesh)	2	5.095	3.485	2.773	(awada general)	1	4.185	2.576	1.546
(nursing utility)	2	2.538	0.929	0.216	(arial fonts)	1	8.535	6.926	5.896
(nim princess)	2	4.913	3.303	2.591	(akash his)	1	0	0	0
(lift wheelchairs)	2	4.965	3.356	2.644	(added locally)	1	0	0	0

TABLE 24: PPMI, SPPMI and DMI of 100 randomly selected word pairs whose $0 < F_{ij} < 5$. Dataset: Wiki-100.

F_{ij} Range	Word pair	F_{ij}	PMI	SPPMI	DMI
0-5	(council suribachi)	1	2.409	0.800	0
	(surname reunion)	2	1.768	0.159	0
	(high kot)	3	1.627	0.018	0
	(commercial conversely)	3	1.926	0.316	0
	(abc ny)	4	1.869	0.259	0
10-20	(variant version)	15	1.606	0	0.134
50-60	(allows you)	51	1.551	0	0.514
	(laws those)	51	1.241	0	0.204
	(occurred within)	51	1.100	0	0.063
	(grants received)	52	1.591	0	0.560
	(attend church)	52	1.492	0	0.461
	(coastal located)	52	1.437	0	0.406
	(defeat round)	52	1.554	0	0.524
	(born hunter)	53	1.272	0	0.247
	(along villages)	53	1.205	0	0.180
	(entered industry)	53	1.565	0	0.541
	(owner private)	56	1.339	0	0.333
100-200	(death personal)	109	0.961	0	0.152
	(own style)	119	0.965	0	0.181
	(includes research)	123	1.358	0	0.583
	(community largest)	126	1.250	0	0.481
	(ny state)	125	1.487	0	0.716
	(along areas)	130	0.911	0	0.150
	(returned top)	133	0.961	0	0.206
	(its schedule)	143	1.143	0	0.408
	(closed july)	142	1.479	0	0.741
	(key public)	140	1.497	0	0.756
	(band uk)	139	1.079	0	0.336
	(appear only)	137	1.017	0	0.270
	(policy state)	159	0.882	0	0.173
	(games tournament)	169	1.583	0	0.890
	(computer used)	167	1.211	0	0.515
500-600	(bar he)	505	0.589	0	0.133
	(occupied were)	509	1.490	0	1.035
	(old time)	512	0.481	0	0.027
	(championship final)	515	1.433	0	0.979
	(captured were)	524	1.302	0	0.852
	(acting his)	530	0.757	0	0.309
	(its site)	527	0.551	0	0.102
	(county washington)	534	1.522	0	1.076
	(founded who)	539	0.658	0	0.214
	(awards year)	548	1.471	0	1.029
	(asked his)	549	0.691	0	0.249
	(bar he)	505	0.589	0	0.133
	(california his)	504	0	0	0
	(occupied were)	509	1.490	0	1.035
	(second while)	507	0.233	0	0
	(over river)	565	0.722	0	0.286
	(against out)	574	0.443	0	0.009

TABLE 25: Word pairs in Table 17, 18, 19, 20, 21, 22, 23, 24 when only DMI=0 or only SPPMI=0. Part I.

F_{ij} Range	Word pair	F_{ij}	PMI	SPPMI	DMI
500-600	(point where)	577	1.300	0	0.867
	(doubles s)	587	1.520	0	1.091
	(from spain)	584	0.663	0	0.232
	(history series)	593	0.779	0	0.351
	(january march)	592	1.420	0	0.992
	(formed new)	592	0.858	0	0.430
	(league play)	589	0.978	0	0.549
	(from highway)	599	0.437	0	0.011
	(august june)	597	1.527	0	1.100
	(ran which)	595	1.068	0	0.640
1,000-2,000	(against war)	1,026	1.257	0	0.916
	(council national)	1,056	1.231	0	0.895
	(include which)	1,054	0.426	0	0.089
	(all games)	1,072	1.206	0	0.871
	(earth s)	1,101	0.909	0	0.578
	(he j)	1,100	0.508	0	0.177
	(division league)	1,112	1.577	0	1.248
	(area part)	1,152	0.582	0	0.258
	(most some)	1,154	0.735	0	0.411
	(also received)	1,171	0.631	0	0.309
	(although were)	1,162	0.453	0	0.130
	(countries from)	1,219	0.847	0	0.531
	(he tells)	1,214	1.546	0	1.229
	(episode s)	1,199	0.425	0	0.106
	(played two)	1,197	0.651	0	0.332
	(her role)	1,223	1.430	0	1.114
	(been some)	1,263	0.386	0	0.074
	(one top)	1,253	0.363	0	0.050
	(all members)	1,265	1.337	0	1.025
	(husband s)	1,267	1.038	0	0.727
	(p s)	1,302	0.433	0	0.126
	(award from)	1,365	0.347	0	0.045
	(body his)	1,396	0.841	0	0.542
	(he once)	1,394	0.457	0	0.159
	(he minister)	1,425	0.571	0	0.275
	(his led)	1,449	0.428	0	0.134
	(education school)	1,494	1.148	0	0.858
	(published were)	1,512	0.760	0	0.471
	(also member)	1,674	0.829	0	0.553
	(state york)	1,739	1.252	0	0.981
	(street th)	1,700	1.567	0	1.293
	(has made)	1,867	0.383	0	0.120
	(after second)	1,809	0.655	0	0.388

TABLE 26: Word pairs in Table 17, 18, 19, 20, 21, 22, 23, 24 when only DMI=0 or only SPPMI=0. Part II.

F_{ij} Range	Word pair	F_{ij}	PMI	SPPMI	DMI
5,000-6,000	(first season)	5,006	1.492	0	1.323
	(his known)	5,104	0.672	0	0.505
	(he moved)	5,105	1.452	0	1.285
	(company s)	5,109	0.670	0	0.503
	(first one)	5,205	0.567	0	0.402
	(his until)	5,259	0.977	0	0.812
	(career he)	5,291	0.842	0	0.678
	(age from)	5,343	1.284	0	1.120
	(he season)	5,357	0.556	0	0.392
	(film s)	5,479	0.439	0	0.277
	(she when)	5,444	1.360	0	1.198
	(people s)	5,554	0.621	0	0.460
	(did he)	5,537	1.279	0	1.118
	(death s)	5,734	1.233	0	1.074
	(made were)	5,715	1.192	0	1.033
	(had she)	5,659	0.840	0	0.680
	(city s)	5,746	0.321	0	0.162
	(he received)	5,745	1.319	0	1.161
	(her her)	5,846	1.140	0	0.983
	(band s)	5,840	1.106	0	0.949
	(father s)	5,788	1.458	0	1.300
	(he university)	5,870	0.602	0	0.445
>10,000	(first he)	10,313	0.144	0	0.024
	(were which)	10,159	0.544	0	0.423
	(during s)	10,116	0.575	0	0.453
	(had which)	11,125	0.953	0	0.837
	(one s)	11,028	0.179	0	0.062
	(became he)	10,748	1.286	0	1.168
	(he played)	11,180	1.592	0	1.476
	(after his)	11,560	0.670	0	0.556
	(also has)	11,993	1.171	0	1.059
	(from were)	12,821	0.213	0	0.104
	(had who)	13,091	1.601	0	1.493
	(he which)	14,193	0.212	0	0.108
	(first s)	15,477	0.414	0	0.315
	(first s)	15,477	0.414	0	0.315
	(has he)	17,603	0.653	0	0.559
	(first his)	17,908	0.938	0	0.845
	(had he)	21,976	0.883	0	0.798
	(also he)	26,582	0.990	0	0.913
	(he his)	44,544	0.846	0	0.786

TABLE 27: Word pairs in Table 17, 18, 19, 20, 21, 22, 23, 24 when only DMI=0 or only SPPMI=0. Part III.

according to Equation 2, $\hat{F}_{ij}$ can be very small. For instance, if the word frequency of two words in Wiki-100 are 3 and 5 respectively, $n = 1.2 \times 10^7$ in Wiki-100, and the $\hat{F}_{ij}$ is less than 10^{-4}. Even though the F_{ij} is 1 or 2, the PMI can be very large. As the F_{ij} increases, the word frequencies of word pairs also becomes larger, thus $\hat{F}_{ij}$ can not be extremely small and the ratio between F_{ij} and $\hat{F}_{ij}$ gets stable.

As we set $s = 5$ in SPPMI, all the SPPMI values are $\log 5$ smaller than PMI values when F_{ij} is the same. The shape of SPPMI line is exactly the same as the PMI line. The line of DMI is very different from the other two. DMI $<$ SPPMI only when F_{ij} is very small, and it becomes larger with the growth of F_{ij}, giving larger weights on frequent word pairs. Our DMI stresses the importance of frequent word pairs after shifting, because it shifts the PMI according to the variance of r, and when F_{ij} gets larger, the value of r is more reliable, thus DMI gives a gentle shifting when F_{ij} is large.

6.4 Word Vectors

The word vectors in PMI, SPMI and DMI are also different.Figure 12 shows the values of MIs of the word "*percent*" in Wiki-100. Only positive PMI values are kept, and the values are sorted in ascending order. When applying SPPMI on this vector, the blue line is shifted downwards by $\log 5$, and all the PMI values smaller than $\log 5$ are turned to 0. However, DMI shifts the PMI values dynamically and there are no specific lines, the red dots spread under the PMI line.

After the vector is normalized, the shapes of PMI line and SPPMI line do not change, but the distribution of DMI dots is different. Some red dots are above the PMI line, which means the DMI is larger than the original PMI, and the largest DMI value is larger than the largest PMI value. It is because the normalization emphasizes the large values in DMI. DMI shifts more aggressively than SPPMI, and removes more small values, but at the same time reserve the large values by giving them a gentle shift. However, SPPMI shifts all the values by $\log 5$. After the normalization, the influence of the small number of large values becomes even larger, which leads to the DMI values larger than the original PMI values.

CHAPTER 7

Experiments

In our experiments, we evaluate our word representations on word similarity tasks and word analogy tasks on corpora of different size. In word similarity tasks, we use 6 word similarity test sets and test the significance of our improvements. In word analogy tasks, the word vectors are tested on two test sets and two different methods of discovering the analogy relations are used.

7.1 Data Sets

The data sets we use can be divided into two different categories: one is the corpus used to generate the co-occurrence matrices and another is the test sets we use to evaluate our methods.

7.1.1 Corpus

We use English Wikipedia(July 2017 dump), Reuters and Text8 to generate the co-occurrence matrices. The size of original English Wikipedia dump data set is about 58.6 GB, and we only use the plain text for creating the co-occurrence matrices. The plain text is about 11.0 GB. In our experiment, we create another 3 smaller data sets of different size from Wikipedia by randomly selecting paragraphs from the original corpus. They are Wiki-100, Wiki-500 and Wiki-1000, and the size is 100 MB, 500 MB and 1 GB respectively. For the Reuters data set, we also use plain text (the title, the headline and content information of news) to generate the matrix. The size is about 1.2 GB. Text8 is a widely used data set to measure the performance of the word similarity and analogy tasks, so in our experiment, it

is also considered.

In order to improve the efficiency of our experiment, all the stop words(defined in Lucene) are removed, and punctuations are filtered, all the text are lowercased. The statistics are listed in Table 28.

	#Tokens($\times 10^6$)	#$\|\|$Voc$\|\|$ ($\times 10^4$)	#Unique Pairs($\times 10^6$)
Text8	12.2	25.4	32.7
Wiki-100	12.8	47.5	43.9
Wiki-500	63.8	130.5	147.3
Wiki-1000	128.1	202.7	244.0
Reuters	129.7	36.6	96.9

TABLE 28: Corpora Statistics

7.1.2 Test Data Sets

Word Similarity Test Sets In word similarity tasks, we use five word similarity test sets, each test set contains a list of word pairs, and each pair has a manually labelled similarity score. WS353[10] data set was released in 2002, the word pair similarity scores are given by more than 10 near-native English speakers. In our experiment, WS353 is partitioned into two data sets, WordSim353 relatedness and WordSim353 similarity [1, 32]. MEN[5] and MTurk [23] contains 3,000 and 287 word pairs respectively, which are selected from Wikipedia. The human judgments are obtained by crowdsourcing using Amazon Mechanical Turk. The Rare Words data set has 2034 word pairs, selected from the pairs with low co-occurrences in Wikipedia, rated within [0,10]. The human similarity judgments are also given by crowdsourcing. RG is the earliest word similarity test set, only containing 65 word pairs and the judgments are made by 51 subjects according to a scale from 0 to 4. The statistics of test sets are summarized in Table 29.

Word Analogy Test Sets In word analogy tasks, we use two popular test sets Google [18] and MSR [20]. The word analogy tasks present the question: "a is to $a*$ as b is to $b*$" where $b*$ is hidden and have to be guessed from the whole vocabulary using the word representations. A sample of the test sets is shown in Table 30, the first 3 words are given, and we have to guess the forth word.

Data set	Word pairs	References
WS353	353	[10]
WS353-relatedness	252	[1, 32]
WS353-similarity	203	[1, 32]
MEN	3,000	[5]
MTurk	287	[23]
Rare Words	2,034	[17]
RG	65	[26]

TABLE 29: Test Sets Statistics

a	$a*$	b	$b*$
baghdad	iraq	bangkok	thailand
boy	girl	father	mother
brother	sister	uncle	aunt
think	thinking	scream	screaming
albania	albanian	italy	italian

TABLE 30: A sample of word analogy test set.

The Google analogy data set contains 19,544 questions, and about half of the set is syntactic analogies, such as "*walk* is to *walking* as *drink* is to *drinking*"; another half is of a more semantic nature, such as "*beijing* is to *china* as *athens* to *greece*". The MSR test set contains 8,000 syntactic questions. In our experiment, we filtered the questions involving words that are out of the vocabulary.

7.2 Word Similarity Tasks

We compare the PPMI, SPPMI, Word2vec with our DMI on the corpora of different size. With the word representations, we first calculate cosine similarity of every test pairs in the test data set, then calculate the Spearman's correlation between the human labelled scores and cosine similarities given by our word vectors. An example is given in Table 31. Suppose the test set contains 5 test pairs:

- For each pair, a human labelled similarity score is given in "Labelled Score" column.
- Sort the score from largest to smallest, thus each pair has a rank ("Labelled Rank"

Word1	Word2	Labeled Score	Labeled Rank	Cosine Similarity	Rank
media	radio	7.42	1	0.03	1.5
television	radio	6.77	2	0.025	3
train	car	6.31	3	0.03	1.5
bread	butter	6.19	4	0.01	5
plane	car	5.77	5	0.02	4

TABLE 31: An example of Spearman's correlation.

column) according to its given score.

- Then with word vectors, we can also have the pairs' cosine similarities ("Cosine Similarity" column) and the rank ("Rank" column).

- In the end, we calculate the Pearson's correlation between the "Labeled Rank" column and the "Rank" column.

The Spearman's correlation, in this case is 0.718. The higher the correlation value is, the better performance the representations get. Moreover, if one word in the test sets does not occur in the corpus, we remove that test pair.

7.2.1 Choosing Parameters for SPPMI

The shifting value s in SPPMI is a hyperparameter, it depends on the size of the corpora. In fact, s comes from a hyperparameter in Word2vec called "negative sample", and s is the number of negative samples. [18] suggested that if the corpus is not large, s can be set to 2 to 5. Otherwise, it can be set to 5 to 25. Therefore, 5 is a decent option.

However, we still tried different s on different corpora and test sets. Figure 20, 21 and 20 shows the influence of s on the performance of SPPMI on the Wiki-100, Wiki-1000 and Reuters respectively. The blue line represents the performance of SPPMI and the red line is the performance of PPMI, which is basically SPPMI when $s = 1$. It can be seen that in most cases, as s increases larger than 5, the performance of SPPMI keeps stable or drops a little bit. Overall, $s = 5$ is a decent choice.

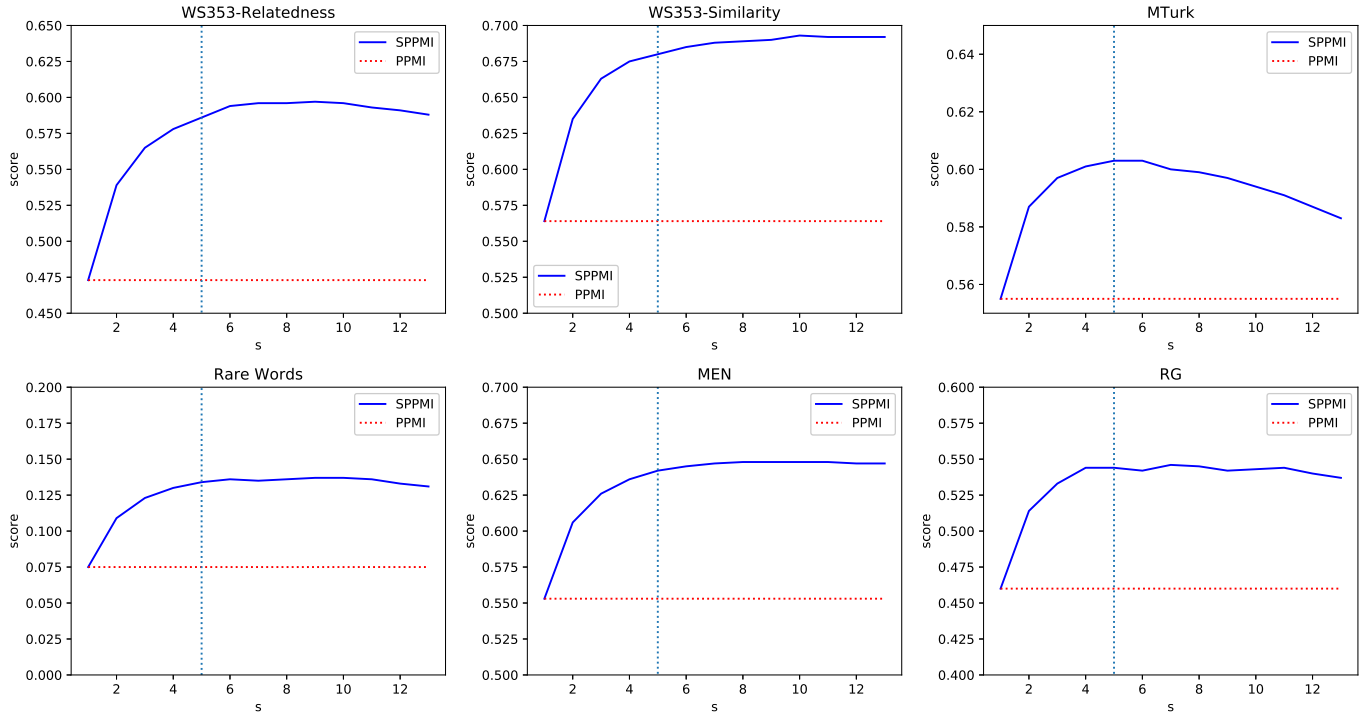


FIGURE 20: The influence of different shifting values in SPPMI on Wiki-100.

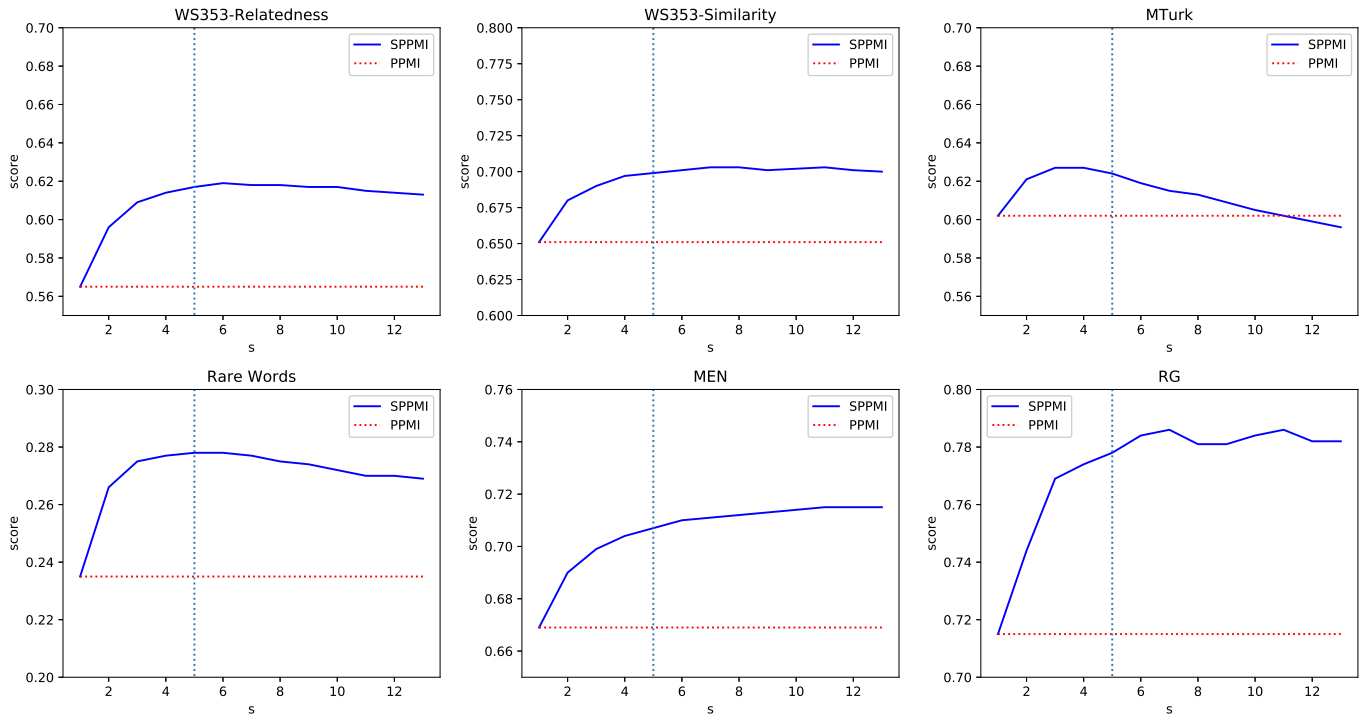


FIGURE 21: The influence of different shifting values in SPPMI on Wiki-1000.

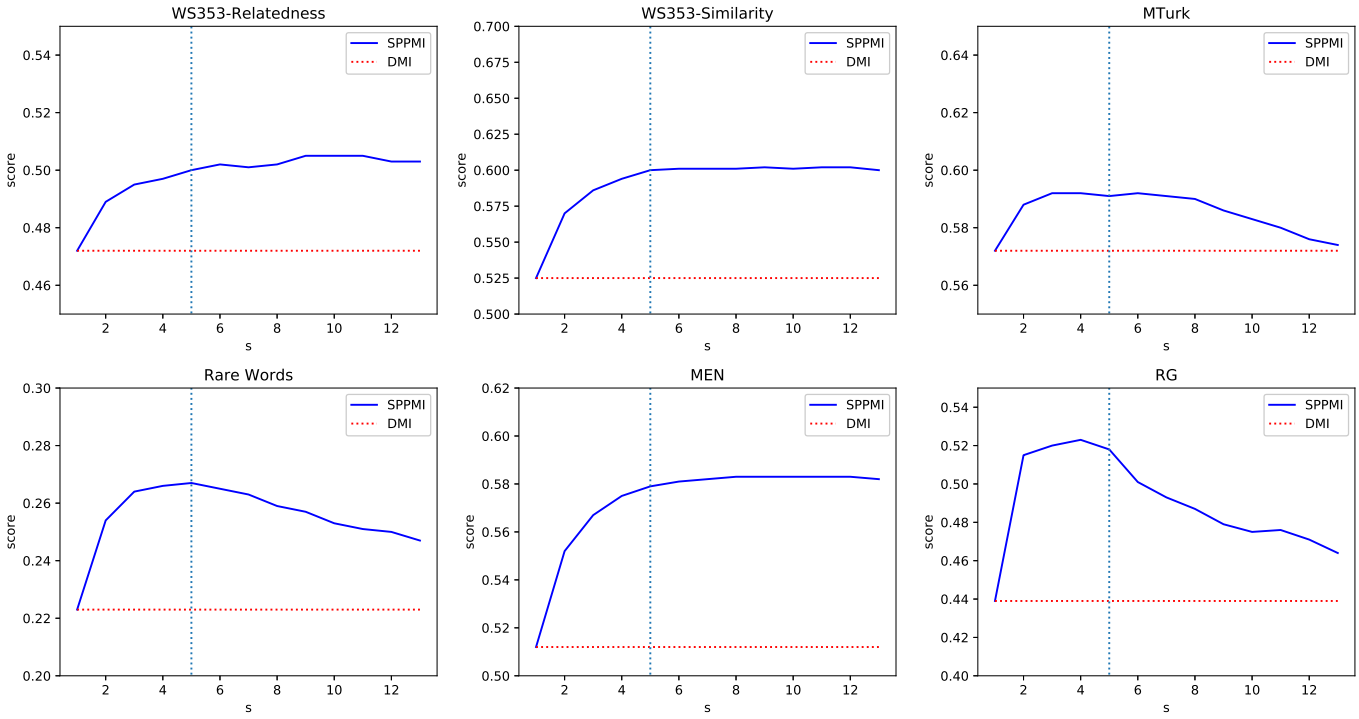


FIGURE 22: The influence of different shifting values in SPPMI on Reuters.

7.2.2 Word2vec Settings

There are a lot of hyper parameters in Word2vec Skip-Gram model, such as *learning rate*,*dimension of vectors*, *subsampling probabilities*, *window size*, *#negative samples*, *#iterations* and *min-count*. Some of the parameters are related to those in the distributional models, and in our experiment, we set the shared parameters the same to compare the performances, they are:

- ***window size***. The window size is set to $k = 5$ for every experiment.
- ***#negative samples***. The number of negative samples is the same as the parameter s in SPPMI. Since we set $s = 5$, the number of negative samples is also set to be 5.
- ***#iterations***. This parameter is related to the total number of word pairs we collected. To ensure all the models collect the same number of word pairs, the number of iterations is set to $k - 1 = 4$.
- ***min-count***. Min-count is used to remove the rare words. The words with the fre-

quency lower than min-count are removed from the vocabulary. In our experiment, we set min-count to 0 and 5 to see the influence of this parameter.

For the other parameters, we use default settings, and the values are: *learning rate=0.05*, *dimension of vectors=100*, *subsampling probabilities*=10^{-5}.

7.2.3 Results

The word similarity results of different test sets on different corpora are listed in Table 32, 33, and Figure 23, 24 demonstrates the results of different representations in a more straightforward way. Table 32 and Figure 23 show the performance without removing rare words, and Table 33 and Figure 24 demonstrate the results after removing words whose frequency is less than 5.

The performance of SPPMI is better than the original PPMI on all the test sets and different corpora. Our DMI outperforms SPPMI on WS353 Relatedness, WS353 Similarity, MTurk and MEN test sets, and does not have advantages over SPPMI on Rare Words test set. For RG test set, the improvement is not consistent, because the number of test pairs in RG is only 65.

Compared with Word2vec, DMI has advantages on most test sets except for rare words, because the vectors from distributional models are too sparse for rare words, and the low dimensional dense vectors are more suitable to represent rare words.

After removing the words whose frequency is less than 5, the performances on most data sets are improved, especially for Word2vec. Our DMI still outperforms SPPMI, but on some test set, the score is not as good as that in Word2vec. However, DMI still have a considerable advantage on WS-relatedness test set.

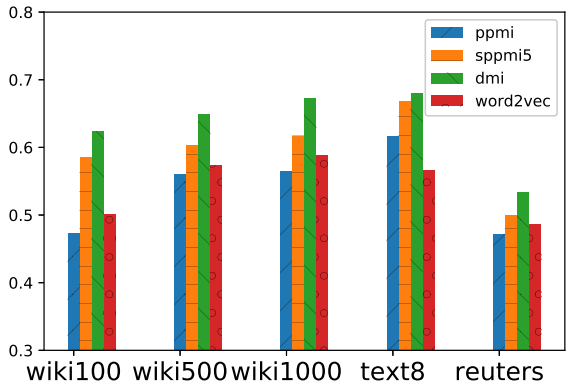
For the test set WS353 relatedness, a subset of WS353, our DMI has greater improvements than WS353 similarity set, which means our DMI is better at discovering the relatedness relationship between two words. These related words, such as *(computer, keyboard)* and *(OPEC, oil)*, tend to co-occur frequently in the same sentence, but their meanings are not the same, and the word cannot be replaced with each other in one sentence. For example, the words “computer” and “keyboard” are not referring to the same thing, but the

Data Set		s/a	WS353 Rel	WS353 Sim	MTurk	Rare Words	MEN	RG
Text8	PPMI	NA	0.616	0.612	0.622	0.121	0.613	0.650
	SPPMI	5	0.669	0.702	0.627	0.165	0.683	0.704
	DMI	6	**0.681**	**0.710**	**0.643**	0.149	**0.690**	**0.711**
	Word2vec	NA	0.567	0.658	0.564	**0.320**	0.532	0.547
Wiki100	PPMI	NA	0.473	0.564	0.561	0.075	0.553	0.460
	SPPMI	5	0.586	0.680	0.609	0.134	0.642	0.544
	DMI	11	**0.624**	**0.687**	**0.628**	0.120	**0.663**	0.555
	Word2vec	NA	0.501	0.675	0.509	**0.300**	0.558	**0.667**
Wiki500	PPMI	NA	0.560	0.635	0.583	0.201	0.660	0.653
	SPPMI	5	0.603	0.699	0.568	0.248	0.698	0.737
	DMI	11	**0.647**	**0.716**	0.607	0.240	**0.716**	**0.745**
	Word2vec	NA	0.573	0.710	**0.628**	**0.337**	0.672	0.740
Wiki1000	PPMI	NA	0.565	0.651	0.610	0.235	0.669	0.715
	SPPMI	5	0.617	0.699	0.632	0.278	0.707	0.778
	DMI	11	**0.673**	**0.731**	**0.664**	0.274	**0.727**	**0.787**
	Word2vec	NA	0.588	**0.731**	0.616	**0.346**	0.686	0.781
Reuters	PPMI	NA	0.472	0.525	0.579	0.223	0.516	0.439
	SPPMI	5	0.500	0.600	0.600	0.267	0.584	**0.518**
	DMI	19	**0.534**	**0.634**	**0.626**	0.242	**0.604**	0.502
	Word2vec	NA	0.486	0.586	0.619	**0.351**	0.494	0.400

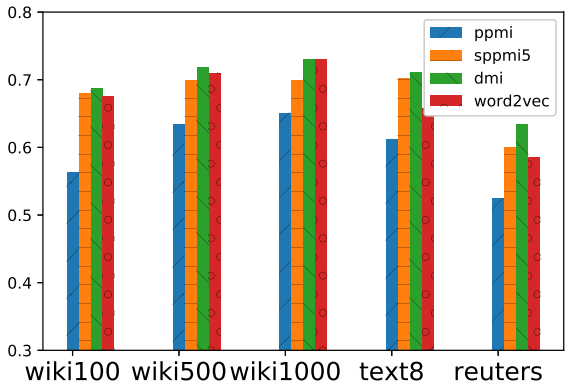
TABLE 32: The word similarity results on different corpus.*Min-count*=0

Data Set		s/a	WS353 Rel	WS353 Sim	MTurk	Rare Words	MEN	RG
Text8	PPMI	NA	0.650	0.687	0.640	0.182	0.641	0.668
	SPPMI	5	0.665	0.738	0.606	0.193	0.677	0.670
	DMI	6	**0.689**	**0.739**	**0.639**	0.196	**0.682**	**0.687**
	Word2vec	NA	0.619	0.705	0.616	**0.383**	0.566	0.512
Wiki100	PPMI	NA	0.549	0.656	0.624	0.180	0.623	0.527
	SPPMI	5	0.575	0.690	0.581	0.193	0.639	0.515
	DMI	11	**0.610**	0.690	**0.628**	0.210	**0.661**	0.535
	Word2vec	NA	0.554	**0.719**	0.580	**0.385**	0.603	**0.694**
Wiki500	PPMI	NA	0.537	0.637	0.609	0.211	0.651	0.606
	SPPMI	5	0.597	0.695	0.595	0.281	0.698	0.707
	DMI	11	**0.637**	0.716	0.615	0.285	**0.717**	**0.729**
	Word2vec	NA	0.616	**0.727**	**0.658**	**0.374**	0.693	0.716
Wiki1000	PPMI	NA	0.581	0.674	0.647	0.271	0.693	0.702
	SPPMI	5	0.593	0.689	0.608	0.290	0.712	0.730
	DMI	11	**0.656**	0.723	**0.658**	0.293	**0.729**	0.754
	Word2vec	NA	0.619	**0.755**	0.654	**0.371**	0.707	**0.775**
Reuters	PPMI	NA	0.475	0.575	0.609	0.279	0.558	0.551
	SPPMI	5	0.483	0.626	0.593	0.289	0.598	0.544
	DMI	19	**0.527**	**0.647**	**0.637**	0.303	**0.611**	**0.584**
	Word2vec	NA	0.455	0.584	0.613	**0.378**	0.494	0.373

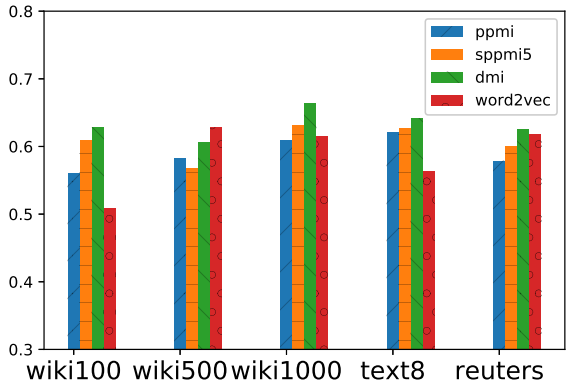
TABLE 33: The word similarity results on different corpus. *Min-count*=5



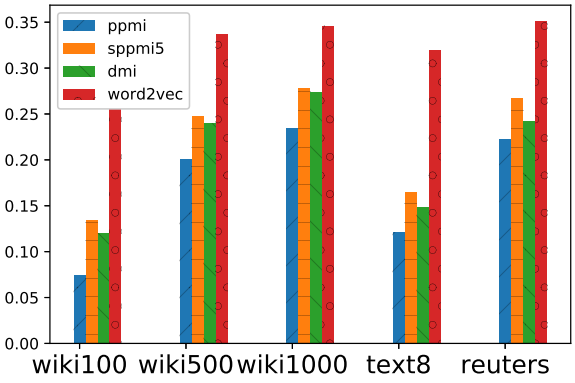
WS353 Related



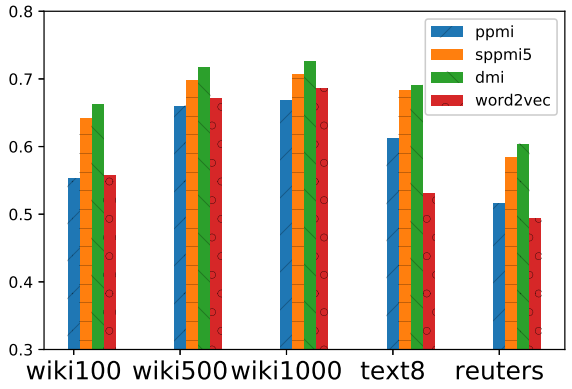
WS353 Simlarity



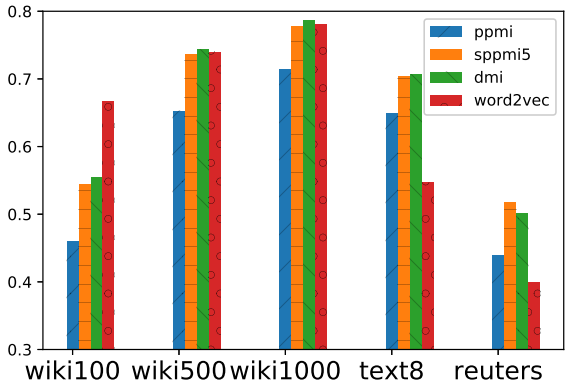
MTurk



Rare Word

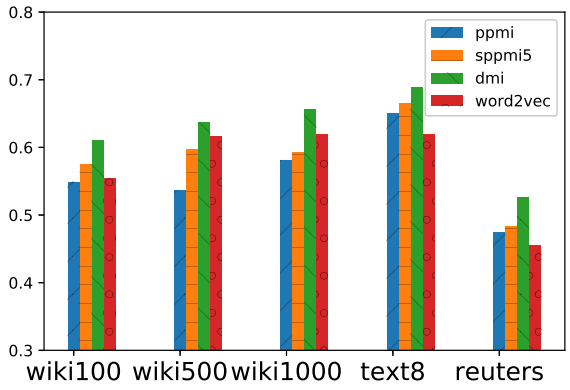


MEN

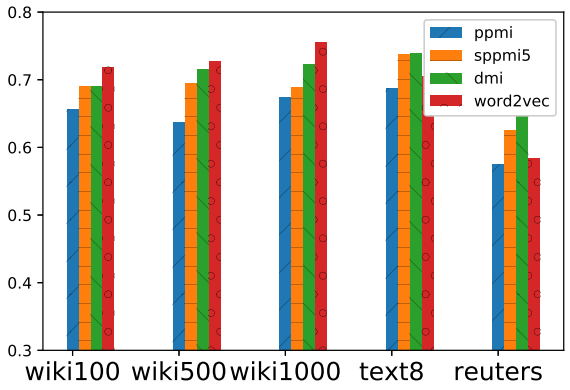


RG

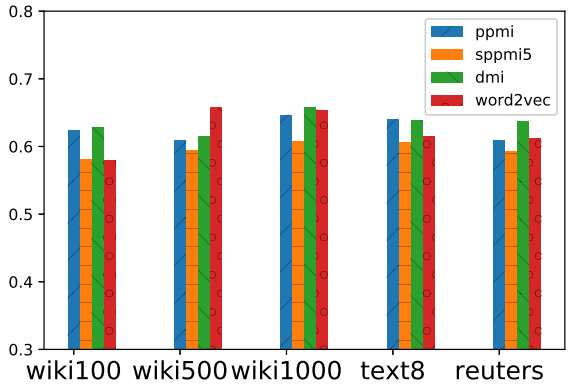
FIGURE 23: The word similarity results on different corpus.



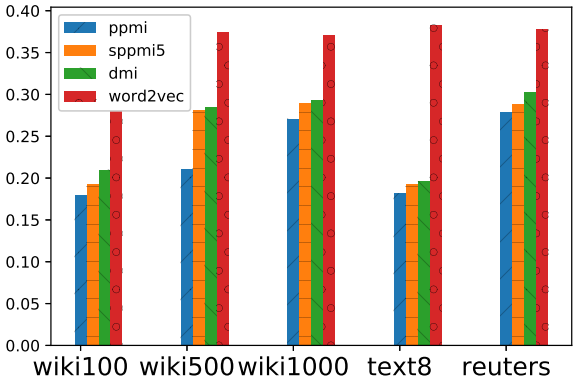
WS353 Related



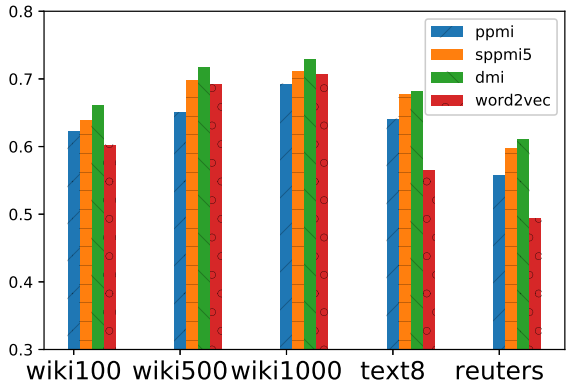
WS353 Simlarity



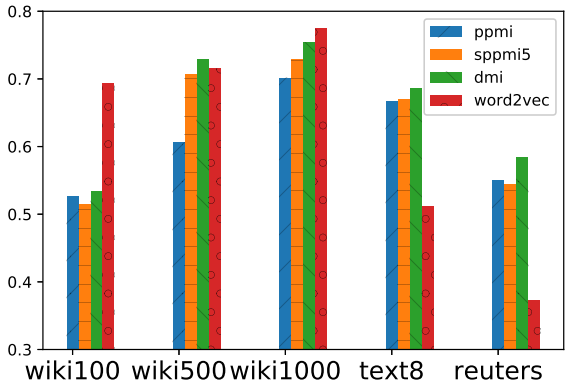
MTurk



Rare Word



MEN



RG

FIGURE 24: The word similarity results on different corpus. *Min-count=5*

"keyboard" is an important "computer" accessories. Another word association measure is the similarity, which is mainly tested in WS353 similarity test set, such as word pairs *(tiger, cat)* and *(car, automobile)*. These word pairs have similar meanings and sometimes we can replace one word with another. In our experiment, on Wiki-1000 corpus, Min-count=0, DMI outperforms SPPMI on WS353 relatedness by 9.1% , but only 4.6% on WS353 similarity test set. Also, DMI greatly outperforms Word2vec on discovering word relatedness.

Table 34 shows the word pairs evaluations. Let r_S, r_D ne the rank given by SPPMI and DMI respectively, r_0 be the ground truth rank, and r_1, r_2 be the distance between the ground truth and r_S, r_D respectively. Thus we have

$$r_1 = |r_S - r_0| \tag{31}$$

$$r_2 = |r_D - r_0| \tag{32}$$

If $r_2 < r_1$, the rank of the word pair given by DMI is closer to its ground truth than that of SPPMI. Otherwise, the DMI does not improve the evaluation.

The word associations for most of the improved pairs are relatedness. For example, the word pair *(maradona, football)* is related to each other but not similar, because maradona was a football star, and their given rank is 22. However, with the original PPMI, their rank is much lower than it should be, only 334. With SPPMI, the rank becomes better, but only 294, still underestimated their relatedness. When using DMI, the rank increased to 250, much closer to the ground truth. DMI improves the performance not only by increasing the rank of highly associated words, but also by decrease the rank of word pairs whose associations are overestimated. For the word pair *(lad, wizard)*, its labelled rank is only 345.5, which means they are irrelevant. But in PPMI and SPPMI, their similarity is over estimated, the rank is 231 and 232 respectively. DMI decreases the rank to 275, much better than PPMI and SPPMI, though not very close to the ground truth.

In MTurk and MEN test sets, when Min-count=0, the performances of DMI is better than that of SPPMI by over 2% on most corpora, and the pairs improved most and least are listed in Table 35 and 36 respectively. In both tables, most improved word pairs are related to each other, though some similar word pairs also have a better estimation. It also shows

$r_2 - r1$	w1	w2	Score	Rank Ground Truth	Rank PPMI	Rank SPPMI	Rank DMI
-60	announcement	effort	2.75	313	195	242	302
-52	monk	slave	0.92	345.5	276	257	309
-51	hundred	percent	7.38	106.5	224	224	173
-50	population	development	3.75	282	256	136	186
-45	grocery	money	5.94	202	235	278	233
-44	maradona	football	8.62	22	334	294	250
-43	lad	wizard	0.92	345.5	231	232	275
-42	video	archive	6.34	173.5	270	219	177
-39	precedent	cognition	2.81	312	145	169	208
-38	impartiality	interest	5.16	237	285	335	297
-38	dividend	calculation	6.48	163	57	86	124
-37	country	citizen	7.31	111	298	286	249
-34	street	children	4.94	246.5	254	211	245
-33	music	project	3.63	289.5	186	145	178
-32	movie	popcorn	6.19	188	293	299	267
-32	drink	mouth	5.96	200	175	164	196
-32	life	term	4.5	259	232	223	263
-31	money	property	7.57	87	132	178	147
-31	money	deposit	7.73	72.5	146	143	112
-30	governor	interview	3.25	299	245	244	274
...							
23	stock	jaguar	0.92	345.5	261	282	259
23	law	lawyer	8.38	33	100	73	96
23	monk	oracle	5	242.5	324	296	319
23	planet	space	7.92	62.5	101	104	127
23	peace	atmosphere	3.69	286	230	266	243
23	registration	arrangement	6	196	223	290	313
23	marathon	sprint	7.47	96.5	133	94	71
24	summer	nature	5.63	219.5	325	304	328
24	precedent	antecedent	6.04	193	207	259	283
25	forest	graveyard	1.85	333	302	276	251
25	planet	astronomer	7.94	60.5	148	78	103
25	game	round	5.97	198.5	126	150	125
26	arrival	hotel	6	196	250	273	299
26	morality	marriage	3.69	286	166	191	165
26	tiger	mammal	6.85	136.5	218	184	210
28	car	flight	4.94	246.5	190	235	207
29	psychology	mind	7.69	76 109	112	141	
34	chance	credibility	3.88	277.5	214	280	241
42	water	seepage	6.56	159.5	123	91	49
46	board	recommendation	4.47	261	225	225	179
49	consumer	energy	4.75	252	125	200	151
50	football	tennis	6.63	154.5	201	148	98
52	dollar	buck	9.22	5	257	251	303
96	asylum	madhouse	8.87	16	205	101	197

TABLE 34: Improvements in WS353. Dataset: Wiki-1000.

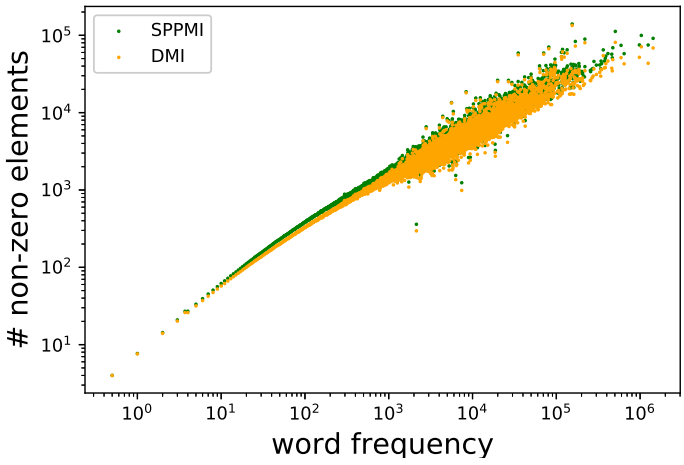


FIGURE 25: The average number of non-zero elements in the word vector as the word frequency increases

that our DMI is better in discovering word relatedness relation than similarity relation.

For the Rare Word test set, DMI does not get better results than SPPMI, because the vectors of rare words are very sparse, and DMI shifts more word pairs than SPPMI, thus, the DMI vectors for rare words will be too sparse to get satisfactory results. The average number of non-zero elements in the word vector is shown in Figure 25, and we can see that

- When the word frequency is small, the number of non-zero elements in the vector is under 100. It means the word vector is very sparse for rare words.
- The number of non-zero elements in the DMI vector is less than that in SPPMI vector, thus the rare words' DMI vectors are even sparser.

Therefore, DMI vectors for rare words contains too little information to represent the word well.

In RG test set, the improvements are limited, and when using the corpus Reuters, the performance of SPPMI is a little better than DMI. It is not surprising that, the

Besides, the performance keeps going up with the growth of the data set size. When using data sets Wiki-100 and Text8, the improvements are not very large, because the corpora size is very small, only 100MB. The correlation score is much larger in Wiki-500 and Wiki-1000 than that in Wiki-100 with the increase of the corpus size. For the data set

$r_2 - r1$	w1	w2	Score	Rank Ground Truth	PPMI	SPPMI	DMI
-83	drawing	music	2.583333333	141	192	234	151
-72	battleship	army	4.235294118	17.5	150	166	94
-64	college	scientist	2.8125	117.5	217	221	157
-60	brussels	sweden	3.176470588	83	225	237	177
-57	shariff	deputy	3.642857143	54	245	170	113
-49	artillery	sanctions	2.428571429	166	188	227	178
-48	plays	losses	3.2	81.5	231	216	168
-45	soccer	boxing	3.4	70	115	121	76
-44	pipe	convertible	2	239.5	148	165	209
-41	coin	awards	2.166666667	211.5	274	264	223
-39	horace	grief	1.764705882	265	166	134	173
-37	corruption	nervous	1.875	255	172	208	245
-37	angola	military	2.941176471	106.5	185	186	149
-36	pet	retiring	2	239.5	239	199	235
-36	acre	earthquake	2.125	217.5	203	180	216
-33	horse	wedding	2.266666667	191.5	147	154	187
-29	treaty	wheat	1.8125	260.5	240	226	266
-29	lincoln	division	2.4375	163.5	221	197	159
-29	money	quota	2.5	155.5	158	200	140
-29	politics	brokerage	2.5	155.5	262	279	250
...							
24	militia	weapon	3.785714286	42.5	92	103	127
25	slaves	insured	2.2	207.5	129	187	162
25	probability	hanging	2.058823529	232.5	209	245	270
26	radical	uniform	2.5	155.5	161	212	238
27	mystery	expedition	2.538461538	149	177	171	198
28	medium	organization	2.5625	146	233	219	247
28	heroin	shoot	2.692307692	130.5	104	143	171
29	france	bridges	2.235294118	200.5	252	189	160
29	germany	worst	1.4375	285	261	268	239
32	atomic	clash	2.785714286	120.5	198	204	236
32	feet	international	1.916666667	252.5	275	253	220
35	alcohol	fleeing	2.5625	146	181	182	217
38	bronze	suspicion	2	239.5	266	230	192
39	admiralty	intensity	2.647058824	134	199	220	259
42	sabbath	stevenson	2.214285714	205.5	184	190	148
44	body	improving	2.117647059	220.5	99	181	137
49	gossip	nuisance	3.0625	96.5	130	147	196
51	food	investment	2.25	195.5	128	168	117
61	libya	forged	2.461538462	160.5	142	169	230

TABLE 35: Improvements in Mturk. Dataset: Wiki-1000.

				Rank			
$r_2 - r1$	w1	w2	Score	Ground Truth	PPMI	SPPMI	DMI
-604	road	sidewalk	41	323.5	2514	2400	1796
-762	building	sidewalk	27	1396.5	2122	2405	1643
-649	origami	white	18	1983.5	2904	2846	2197
-535	house	staircase	30	1196.5	1863	2124	1589
-536	puddle	water	38	555.5	2443	2158	1622
-517	city	skyline	36	754.5	2484	2303	1786
-494	glittering	star	30	1196.5	2758	2695	2201
-459	daffodils	plant	42	248.5	2845	2261	1802
-445	cold	puddle	25	1521.5	2524	2630	2185
-443	baseball	hockey	38	555.5	1330	1375	932
-440	air	dew	23	1665.5	2531	2245	1805
-438	house	windmill	18	1983.5	2474	2424	1981
-431	man	sexy	30	1196.5	1816	2229	1798
-416	dripping	water	36	754.5	1942	1737	1321
-416	escalator	railway	20	1867	2924	2896	2480
-412	licking	rusty	5	2903	2581	2450	2944
-410	day	lunch	29	1263.5	2244	2268	1858
-409	art	collage	33	972	1709	1568	1159
-399	males	old	20	1867	2850	2704	2305
-391	husky	played	14	2259.5	2872	2822	2431
...							
339	rice	tickets	5	2903	2507	2472	2133
340	cottage	scenery	24	1589	1750	1830	2170
347	socks	white	28	1326.5	1082	876	529
351	pillow	stone	7	2773	2160	2248	1897
360	signed	stockings	7	2773	2871	2770	2410
368	flamingo	hummingbird	41	323.5	1998	1403	1771
377	puddle	red	7	2773	2729	2585	2208
393	feline	kittens	38	555.5	1286	859	1252
399	skirt	white	24	1589	1291	1158	759
404	female	makeup	30	1196.5	2178	1662	2066
405	downtown	hockey	14	2259.5	2292	2196	1791
421	haircut	hanging	15	2185	2149	2159	2632
429	city	downtown	29	1263.5	796	1107	678
434	black	skirt	24	1589	1346	1290	856
448	asphalt	water	10	2551	1870	1994	1546
457	dude	husky	16	2113	1906	2113	2570
460	flame	words	9	2634	2232	2465	2005
494	game	husky	6	2843	2804	2811	2317
636	canine	licking	25	1521.5	2226	1769	2405
681	black	paws	13	2329.5	2252	2309	1628

TABLE 36: Improvements in Men. Dataset: Wiki-1000

Reuters, the corpus size is about 1GB, but the score is still not high. It is mainly due to the small vocabulary size. Reuters has a smaller vocabulary than Wiki-100, though it is 10 times larger than Wiki100 in the length of the text.

7.3 Statistical Significance on Word Similarity Tasks

It has been pointed out that there has been an absence of statistical significance for measuring the difference in performance of word vectors on word similarity tasks [9]. Since the word similarity test sets are very small, and their average length is 973.5 word pairs as shown in Table 29, the smallest test set RG contains only 65 word pairs. Therefore, it is important to ensure the significance of the improvement.

Let S, D be the rankings produced by SPPMI, DMI respectively, and T be the ground truth ranking. Let r_{ST}, r_{DT} be the Spearman's correlation between ground truth and SPPMI, DMI respectively, and r_{SD} be the correlation between SPPMI and DMI. We want to test the null hypotheses of the forms $r_{ST} = r_{DT}$. Here r_{ST} and r_{DT} are two dependent correlations and they share one same index with each other. [27] suggested that William's T_2 test [30] is perhaps the best all-around choice for comparing two dependent correlations. The formula is

$$T_2 = (r_{ST} - r_{DT})\sqrt{\frac{(N-1)(1+r_{SD})}{2(\frac{N-1}{N-3})|R| + \overline{r}^2(1-r_{SD})^3}} \tag{33}$$

where

$$|R| = (1 - r_{ST}^2 - r_{DT}^2 - r_{SD}^2) + 2r_{ST}r_{DT}r_{SD} \tag{34}$$

$$\overline{r} = \frac{1}{2}(r_{ST} + r_{DT}) \tag{35}$$

N is the number of word pairs in the test set. T_2 has a t distribution with $df = N - 3$.

We use T_2 test to see whether our DMI outperforms SPPMI significantly. The improvements and p values of the T_2 test are listed in Table 37. If the p value is smaller than 0.05,

Data Sets	WS353 Rel		WS353 Sim		MTurk		Rare Words		MEN		RG	
	imp	p	imp	p	imp	p	imp	p	imp	p	imp	p
Text8	1.6	0.645	1.3	0.411	2.4	0.374	-9.6	0.496	1.2	**0.000**	0.4	0.944
Wiki100	6.5	**0.000**	1.0	0.440	3.1	0.099	-10.4	**0.009**	3.3	**0.000**	2.0	0.570
Wiki500	7.6	**0.000**	2.7	**0.018**	6.7	**0.002**	-3.2	0.106	2.7	**0.000**	0.9	0.577
Wiki1000	9.1	**0.000**	4.6	**0.002**	5.1	**0.002**	-1.4	0.499	2.8	**0.000**	1.2	0.648
Reuters	6.8	**0.017**	5.7	**0.008**	4.3	**0.050**	-9.4	**0.015**	3.4	**0.000**	-3.1	0.608

TABLE 37: Significance test on the improvements

the improvements are significant, otherwise, the performances of DMI are no better than SPPMI.

On rare word test set, SPPMI outperforms DMI by up to 10% on Wiki-100 corpus, that is due to the small value of their original spearman's correlation. Though SPPMI has better performances on rare words, most of the improvements(on Wiki-500, Wiki-1000 and Text8) are not significant, which means DMI and SPPMI have on par performances in measuring the similarity between rare words.

All the improvements on RG are not significant, because the T_2 test partly depends on the length of the test set, if the set is very small, it requires larger improvements to pass the significance test. However, the improvements on RG are not large enough to pass the test.

All the improvements are significant on MEN data set, and on WS353 relatedness, WS353 similarity and MTurk test set, the improvements are not significant when the data set is small, but as the corpora get larger, the improvements are all significant. Overall, our DMI outperforms SPPMI, and most improvements are significant on large data sets.

7.4 Word Analogy Tasks

The word analogy tasks present the questions in the form of "a is to $a*$ as b is to $b*$", where $b*$ is hidden, and we have to use the word representations to guess $b*$ from the whole vocabulary. In the end, the performance is evaluated by the accuracy of guessing $b*$.

In our experiment, the analogy questions are answered using two different methods, 3CosAdd[20] and 3CosMul [13]. The idea of 3CosAdd is that, since we know the word vector of a $a*$ and b, and the relations between $(a, a*)$ and $(b, b*)$ should be the same, thus

the vector should follow

$$w_a - w_{a*} = w_b - w_{b*} \tag{36}$$

Therefore, the vector $a * -a + b$ is the closest vector to the vector of $b*$, that is we need to find the most similar vector to $a*-a+b$ in the whole vocabulary. The formula of 3CosAdd is

$$\arg \max_{b* \in V \setminus \{a,a*,b\}} \cos(b*, a - a * +b) = \tag{37}$$

$$\arg \max_{b* \in V \setminus \{a,a*,b\}} (\cos(b*, a*) - \cos(b*, a) + \cos(b*, b)) \tag{38}$$

However, [13] improved the 3CosAdd method in recovering analogy relations, and they propose 3CosMul, the function is

$$\arg \max_{b* \in V \setminus \{a,a*,b\}} \frac{\cos(b*, a*) \cdot \cos(b*, b)}{\cos(b*, a) + \epsilon} \tag{39}$$

where $\epsilon = 0.001$ to prevent division by zero. They use multiplication and division between the word vectors instead of simple addition and subtraction. In our experiment, we will use both methods to discover the analogy relations.

The results of the analogy tasks are shown in Table 39. It can be seen that the performances on the Google test set are much better than these on MSR test set. Since MSR test set only contains the syntactic questions, it means neither PPMI, SPPMI nor DMI are good at answering syntactic analogy questions, they do better in discovering semantic analogy relations.

Another observation is that, 3CosMul is better than 3CosAdd in recovering analogy relations. For every model and every corpus, the accuracy using 3CosMul is higher than that using 3CosAdd, which is consistent with the claim that 3CosMul is superior to 3CosAdd in [13].

The third observation from the table is that our DMI does not have the advantage in

Data Set		s/a	Google		MSR	
			ADD	MUL	ADD	MUL
Text8	PPMI	NA	0.135	**0.225**	0.131	0.163
	SPPMI	5	0.061	0.161	0.020	0.031
	DMI	6	0.067	0.174	0.031	0.044
	Word2vec	NA	0.218	0.214	**0.275**	0.267
Wiki-100	PPMI	NA	0.176	**0.280**	0.070	0.098
	SPPMI	5	0.080	0.188	0.012	0.030
	DMI	11	0.071	0.171	0.013	0.027
	Word2vec	NA	0.234	0.231	**0.253**	0.247
Wiki-500	PPMI	NA	0.229	0.428	0.066	0.118
	SPPMI	5	0.126	0.249	0.010	0.025
	DMI	11	0.135	0.282	0.016	0.041
	Word2vec	NA	**0.471**	0.456	**0.356**	0.351
Wiki-1000	PPMI	NA	0.254	0.472	0.071	0.138
	SPPMI	5	0.155	0.306	0.011	0.036
	DMI	11	0.157	0.327	0.018	0.054
	Word2vec	NA	**0.506**	0.494	**0.412**	0.408
Reuters	PPMI	NA	0.244	**0.392**	0.114	0.180
	SPPMI	5	0.169	0.262	0.029	0.059
	DMI	19	0.157	0.194	0.024	0.031
	Word2vec	NA	**0.392**	0.372	**0.332**	0.320

TABLE 38: The word analogy results on different corpora.*Min-count*=0.

Data Set		s/a	Google		MSR	
			ADD	MUL	ADD	MUL
Text8	PPMI	NA	0.207	**0.315**	0.124	0.132
	SPPMI	5	0.139	0.211	0.051	0.045
	DMI	6	0.184	0.287	0.134	0.126
	Word2vec	NA	0.255	0.244	0.342	0.332
Wiki-100	PPMI	NA	0.286	**0.362**	0.086	0.106
	SPPMI	5	0.192	0.265	0.048	0.053
	DMI	11	0.195	0.279	0.099	0.107
	Word2vec	NA	0.263	0.254	0.309	0.291
Wiki-500	PPMI	NA	0.363	**0.500**	0.111	0.151
	SPPMI	5	0.271	0.413	0.055	0.088
	DMI	11	0.275	0.429	0.096	0.148
	Word2vec	NA	0.461	0.444	0.403	0.392
Wiki-1000	PPMI	NA	0.254	**0.472**	0.071	0.138
	SPPMI	5	0.267	0.364	0.044	0.074
	DMI	11	0.304	0.463	0.100	0.174
	Word2vec	NA	0.514	0.497	0.434	0.420
Reuters	PPMI	NA	0.255	**0.391**	0.102	0.161
	SPPMI	5	0.195	0.244	0.044	0.065
	DMI	19	0.238	0.335	0.103	0.156
	Word2vec	NA	0.391	0.372	0.343	0.327

TABLE 39: The word analogy results on different corpora. *Min-count*=5.

analogy tasks. On the other hand, PPMI has the best performance in the analogy tasks, and SPPMI is the second best. The method preserves most information has the best performance. In SPPMI and DMI, a lot of word pairs are removed, which means some information is lost at the same time.

Unlike word similarity tasks, it only requires the distance information from the word vectors, the analogy tasks require that for word pairs with similar relations, they must have the same relative position in the vector space. However, SPPMI and DMI only preserve the most important information in the vector, leading to the lack of the ability to present the word well in the vector space.

7.5 Implementation

The steps of getting the DMI vectors are listed below:

1. Collecting all the word pairs as the window moves and get the word co-occurrences for all unique word pairs. In this step, different window styles can be applied.

2. Calculate the DMI according to the equations for each unique word pairs. Moreover, other word association measures can be used, such as PPMI and SPPMI.

3. Build the DMI matrix using the unique word pairs and their DMI values. Each row and each column represents one word in the vocabulary. Each element in the matrix is the DMI value of the corresponding word pairs.

4. Evaluate the DMI word representations on different test sets. Each row of the matrix is the word vector representing the corresponding word.

Our DMI is a distributional model, and distributional models usually suffer from some efficiency issues. According to the descriptions of how to get the DMI vectors above, the issues are as follows:

- The size of the matrix is too big to load into the memory. The size of co-occurrence matrix is $|V| \times |V|$. If the vocabulary size is over 1e5, it takes up over 75GB memory.

Dataset	Text8	Wiki-100	Wiki-500	Wiki-1000	Reuters
Sparse(MB)	280	366	1,158	1,899	770

TABLE 40: Memory consumption of the co-occurrences in sparse matrix.

For the Wiki-1000 dataset, the vocabulary size is over 2e6, and it is difficult to fit in regular memory.

- The process of collecting word pairs and counting co-occurrences is rather slow, because the time complexity is $O(|V|^2)$. Moreover, the total number of word pairs increases quadratically with the length of the window.

In our experiment, we use different data structures to store the matrix, and use the inverted index to get the word co-occurrences if the corpus is very large.

7.5.1 Space Complexity

In natural languages, most words are irrelevant to each other, thus the co-occurrence matrix is extremely sparse. For instance, in Wiki-100 corpus, $|V| = 4.75 \times 10^5$, the total number of elements is over 10^{11}. However, there are only 37,882,362 non-zero elements, which means 99.99% elements are zero.

It is a good choice to store the co-occurrence matrix with sparse matrix, hence we use sparse matrix that is supported in Python $\mathrm{scipy.sparse.csr\_matrix}$ API. The memory consumption of the co-occurrences on different corpora in the sparse matrix are listed in Table 40 . The memory consumption is measured by a Python module called $\mathrm{memory\_profiler}$.

7.5.2 Word Pair Collection

According to Equation 3.2.2, there will be $nk(k-1)$ word pairs collected to form the co-occurrence matrix and we also have to count the word co-occurrence for each unique pairs. The process is described in Algorithm 1.

Algorithm 1 Collecting Word Pair Co-occurrences

Input: a sequence of text $C(w_1 w_2 \dots w_n)$, window size k $(k < n)$.
Output: a set of word pairs and the corresponding co-occurrence counts P.

```
1:  function CollectCooccurrences(C, k)
2:      P ← HashTable[]                          ▷ Initiate P as an empty HashTable
3:      for i ← 1 to n − k do                    ▷ Move the window by one token
4:          for j ← i to i + k do                ▷ Collect word pairs within the current window
5:              for h ← j + 1 to i + k do
6:                  if sorted((w_j, w_h)) in P.keys then   ▷ Count the co-occurrences
7:                      P[sorted((w_j, w_h))] +=1
8:                  else
9:                      P[sorted((w_j, w_h))] ← 1
10:     return P
```

7.5.3 Scalability

Even the sparse matrix structure is used to store the co-occurrences, the scalability is still a problem if the corpus keeps growing, especially for the corpus having a large vocabulary. In this case, it is impossible to build the whole co-occurrence matrix, and even extremely difficult to get the co-occurrence count for each word pair.

According to Equation 30, we can see that the co-occurrence count F_{ij} of a word pair is required to calculate the DMI value of the pair, and it can only be acquired through collecting pairs window by window. Other values like $\hat{F}_{ij}$ can be obtained directly from the corpus.

Therefore we can take advantage of using basic windows. Each window can be seen as a small document containing k words and there are n documents (windows) in total. Then we create an inverted index on these n windows, and it is very fast to search how many documents containing w_i and w_j. The number of documents is approximately the co-occurrence count for the word pair (w_i, w_j), because there might be multiple occurrences of one word in the same window, though we assume that there are k different words in one window. Thus, we can get the word co-occurrence counts using the inverted index.

Another question is whether we need to build the whole matrix to get the word representations. At least in our experiment, the answer is not. In our experiment, we only need the word vectors that are in the test sets, and as we stated before, most word vectors are

very sparse, it is pretty easy to get these vectors instead of the whole matrix.

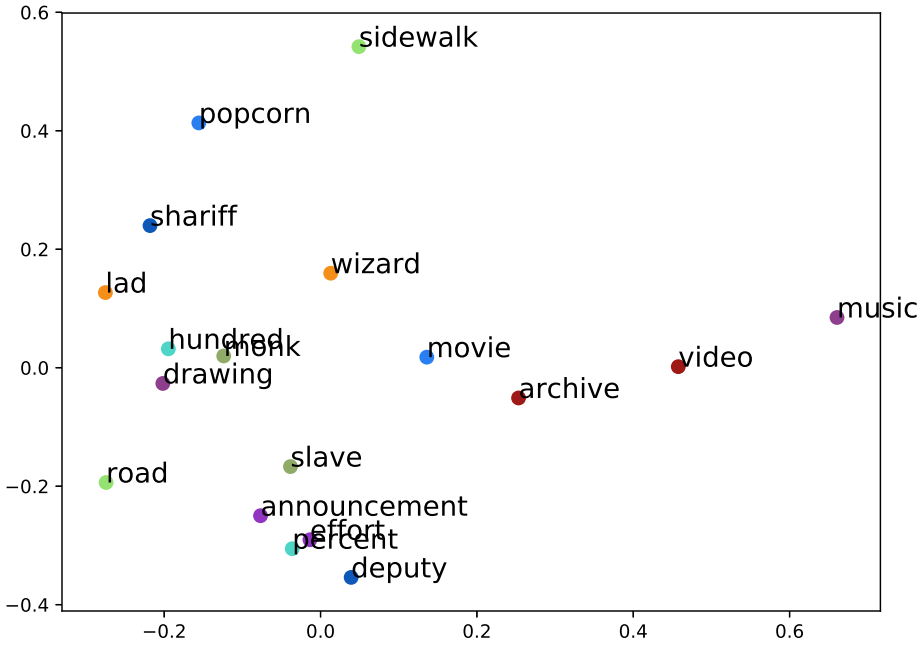
7.6 Examples of SPPMI and DMI

In this section, we will show how our DMI improves SPPMI on word similarity tasks by giving some examples on the test set.

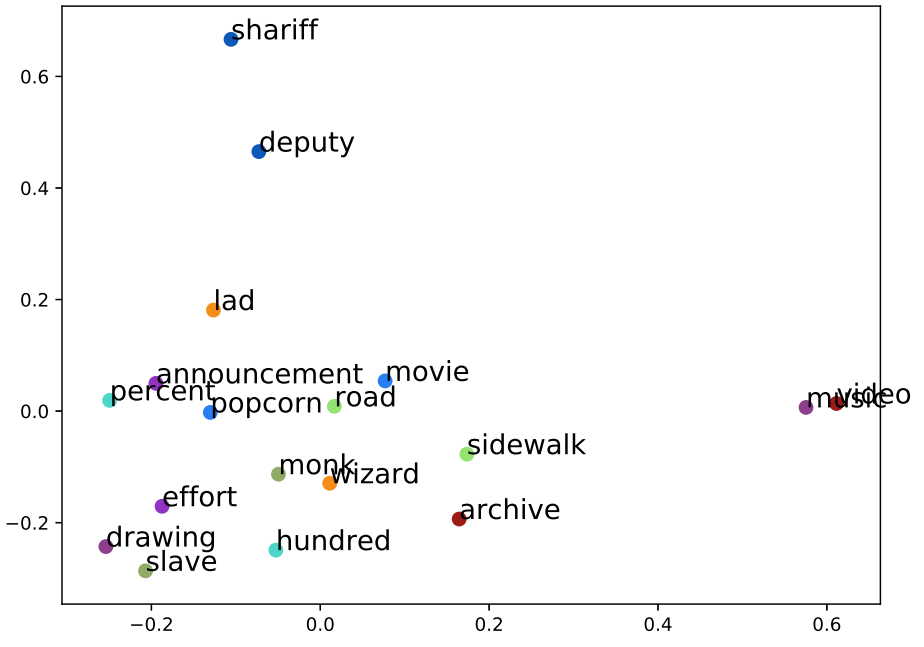
The 2D plot is the simplest way to reflect the similarity between two word vectors. If two words are highly associated with each other, their distance should be very close. Though the dimension of word vectors are pretty high, their distance can still be reflected on a 2D plot using dimension reduction methods. We use PCA to reduce the dimension of word vectors into 2, and each word is represented by a dot. The similarity between two words can be measured by their distance in the figure. Figure 26 shows some word pairs in the test data using 2 different word representations, SPPMI and DMI.

Two words in the same test pair are in the same color, and we can see that DMI improves SPPMI by moving similar pairs closer and dragging irrelevant pairs further away. For example, the irrelevant pair *(announcement, effort)* is given a score 2.75 out of 10 manually, which is pretty low, but in the SPPMI plot, they are very close to each other. However, in the DMI plot, the distance between them is larger. For these word pairs with higher human scores, such as *(popcorn, movie)* and *(deputy, shariff)*, their association is underestimated in SPPMI, but in DMI, they have higher similarity score and as reflected in the plot, they move closer to each other.

As we evaluate the word representations on the word similarity tasks, we use Spearman's correlation, which measures the correlation between the human labeled ranks and cosine similarity ranks given by the vector. With different word representations, the ranks of word pairs in the test set can be very different, better representation will lead to a ranking result closer to the human labelled ranks. Figure 27 shows 28 random test pairs' ranks in WS353 given by SPPMI and DMI. Each dot is a word pair in the test set, and the x-axis is the ground truth rank, y-axis is the rank given by DMI or SPPMI. If the word vectors are of great quality, the given rank of one word pair should be close to its ground truth, thus, the dots should be closer to the line $y = x$. Most word pairs like *(announcement, effort)*,

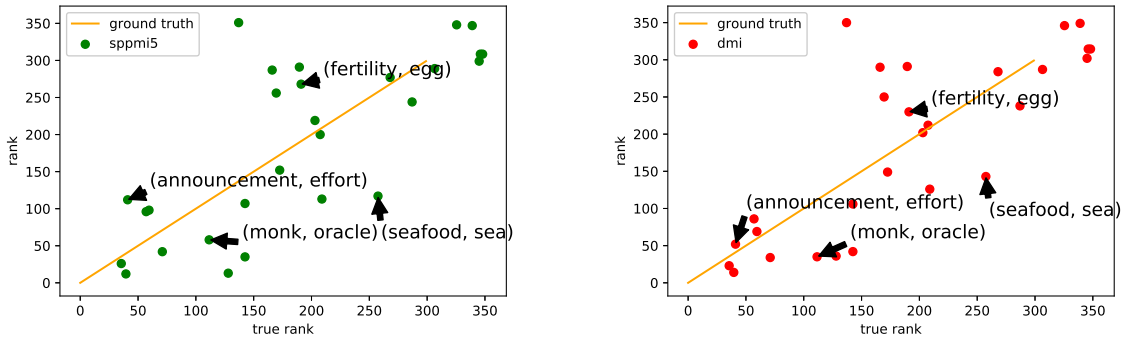


(A): SPMI



(B): DMI

FIGURE 26: Word pairs with SPPMI and DMI representations in 2D plot. The dimension is reduced by PCA, the dataset is Wiki-1000



(A): SPPMI (B): DMI

FIGURE 27: WS353 test pair rank changes in WS353.

(fertility, egg) and *(sea, seafood)* move closer to the ground truth line in DMI, and some of the word pairs do move further away, such as *(monk, oracle)*.

For those points above the line, the given rank (DMI or SPPMI rank) is larger than the labeled rank, thus their similarities are over estimated. Similarly, the similarities of the dots under the line are under estimated.

When calculating the cosine similarity between two words, we first normalize the vectors and then get the dot product of these two vector. Here we introduce a vector $\boldsymbol{m}_{(a,b)}$, the i th element of $\boldsymbol{m}_{(a,b)}$ is the product of the i th element of normalized word vector $\boldsymbol{a}$ and $\boldsymbol{b}$. That is $\boldsymbol{m}_{(a,b)}[i] = \boldsymbol{a}[i] \times \boldsymbol{b}[i]$. Thus, the cosine similarity would be the sum of all the element in $\boldsymbol{m}_{(a,b)}$. Our DMI reduces the similarity score above the line by making the vector $\boldsymbol{m}_{(a,b)}$ more sparser and increase the score under the line by keeping more values in the vector.

The similarity of word pair (fertility, egg) is over estimated in SPPMI, and the number of non zero values in $\boldsymbol{m}_{\text{(fertility,egg)}}$ is 139, while in DMI, the number of non zero values is reduced to 94, and the similarity drops. Same thing happens to the word pair (announcement, effort), the non zero element in $\boldsymbol{m}_{\text{(announcement, effort)}}$ decreases from 148 to 90.

However, the sparser vectors also diminish the performance on some test pairs. For example, The similarity of word pair (monk, oracle) is already under estimated and in SPPMI, $\boldsymbol{m}_{\text{(monk,oracle)}}$ is already sparse, there are only 73 non zero values. When applying DMI, the vector $\boldsymbol{m}_{\text{(monk,oracle)}}$ is even sparser, only 40 non zero values.

This is not always the case, for word pairs with higher similarity, such as (seafood, sea), our DMI preserves more non zero values in $\boldsymbol{m}_{(\text{seafood, sea})}$(106 in SPPMI and 123 in DMI), though the word vectors $\mathbf{w}_{\text{seafood}}$ and $\mathbf{w}_{\text{sea}}$ are sparser.

The number of non-zero values in the vector $\boldsymbol{m}_{(a,b)}$ represents the number of features two word vectors share. DMI reduces the number of shared features of word pairs above the ground truth line, and increases more shared features for the pairs under the line.

CHAPTER 8

Conclusions

In our work, we compared different window styles to collect word co-occurrence counts, word embedding models including PPMI, SPPMI and Word2vec, and proposed a new model called Dynamic Mutual Information(DMI) to improve SPPMI. DMI outperforms SPPMI by dynamically shifts the word pairs' Pointwise Mutual Information according to its variance instead of shifting a constant. Based on all the different word representations, we conduct several experiments to compare the difference between SPPMI and DMI, evaluate their performances on word similarity tasks and word analogy tasks. The works can be summarized as follows:

1. We compared different window styles including basic windows, Word2vec windows and weighted windows, and their co-occurrence matrices.

2. We talked about the relationship between Pointwise Mutual Information(PMI) and expected co-occurrence count $\hat{F}_{ij}$, and explained why shifted positive PMI(SPPMI) outperform positive PMI(PPMI).

3. Dynamic Mutual Information is proposed to improve SPPMI by dynamic changing the shifting values according to the variance.

4. We compared the different shifting schemes between SPPMI and DMI from different aspects.

5. Different corpora and test sets are used to evaluate our DMI.

6. We test the word embeddings on word similarity tasks and word analogy tasks, and the statistical significance on word similarity improvements are tested.

7. Some efficiency issues are pointed out and solved.

8. The improvements of DMI over SPPMI are analysed in different ways.

Since all the word embedding models are based on word co-occurrences, and few works have talked about the relationships between different windows and their influences on word representations, we analysed three mostly used windows: basic windows, weighted windows and Word2vec windows. We found that, for basic windows and weighted windows, their co-occurrence matrices and their co-occurrence distributions are the same. If the Word2vec window is only run once, the distribution of F_{ij} is so different from the other two window styles, but if we repeat the Word2vec window $k-1$ times, the co-occurrence matrix is roughly the same as the other two.

The PMI of a word pair can measure the association between these two words and it has been frequently talked about recently because it is related to the neural network based model Word2vec. We found that the PMI of a word pair is the (logarithm) ratio between its co-occurrence F_{ij} and its expected co-occurrence $\hat{F}_{ij}$, and SPPMI is simply abandon all the PMIs whose ratio is F_{ij} is not s times larger than its $\hat{F}_{ij}$, where s is usually set to 5. Same observation [6] have found that if PMI is less than 3 (the logarithm here is based on 2), their relation is not interesting. This is due to the inaccuracy of the estimation $\hat{F}_{ij}$, it tends to under estimate F_{ij}. Thus, the SPPMI can be explained that it shifts the noises in the PMI matrix by eliminating the unreliable estimations, and only preserves the values large enough.

However, the shifting scheme of SPPMI is too simple, all the values are shifted by a constant. We propose DMI to improve SPPMI by dynamically shifting PMIs according to the variance of the ratio. When the variance of the ratio is very large, it means the estimation is highly unreliable, thus we should give a large shifting value, otherwise, the shifting should be subtle. Moreover, we compared the shifting schemes between DMI and SPPMI in the following respect:

- Co-occurrence distribution. DMI gives a larger shifting value when F_{ij} is small and the preserves more frequent word pairs.

- Pair selection. DMI tends to select more associated word pairs than SPPMI.

- Values of Mutual Information. DMI values get larger with the growth of F_{ij}.
- Word vectors. The shape of word vectors in SPPMI is the same as that in PPMI, while DMI presents a different shape, as shown in Figure 12.

In the evaluation part, we use 5 corpora of different size to create the word representations using different models, and all the word representations are tested in two tasks: word similarity tasks and word analogy tasks.

For the word similarity task, we use 6 different test sets and our DMI outperforms the SPPMI on most of the test sets. In particular, our DMI is better at discovering relatedness relationships between words than similarity relationships, and most word pairs that receive a great improvement in the test set have relatedness relationship. In the Rare words test set, our DMI does not have a great performance because the vector is too sparse to work well. In addition, because the length of test sets is not large, and it is crucial to have a significance test on the improvements. We use T_2 test to see whether our DMI outperforms SPPMI significantly, and we find that when the corpus is large enough, the improvements of DMI is significant and SPPMI does not outperform DMI on Rare Word significantly.

In the Word Analogy task, we use two test sets, Google and MSR, and two methods to discover the analogy relations, 3CosAdd and 3CosMul. In the experiment, our DMI does not have advantages, and PPMI has the best performance with the 3CosMul. Also, we found that the distributional models are better at discovering semantic word analogy relations than syntactic analogy relations.

The distributional models suffer from some efficiency issues, such as that the matrix is too large to load into the memory and it is time-consuming to get the co-occurrence count. To solve the storage problem, we use sparse matrix to store all the matrices because the matrix is extremely sparse. Besides, we use the inverted index to calculate the word co-occurrence counts between each word pair.

In conclusion, we compare different word embedding models and propose a new distributional model called DMI. All the word representations are tested on word similarity tasks and word analogy tasks

REFERENCES

[1] Agirre, E., Alfonseca, E., Hall, K., Kravalova, J., Paşca, M., and Soroa, A. (2009). A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 19–27. Association for Computational Linguistics.

[2] Arora, S., Li, Y., Liang, Y., Ma, T., and Risteski, A. (2016). A latent variable model approach to pmi-based word embeddings. *Transactions of the Association for Computational Linguistics*, 4:385–399.

[3] Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

[4] Bollegala, D., Yoshida, Y., and Kawarabayashi, K.-i. (2017). Using k-way co-occurrences for learning word embeddings. *arXiv preprint arXiv:1709.01199*.

[5] Bruni, E., Tran, N.-K., and Baroni, M. (2014). Multimodal distributional semantics. *Journal of Artificial Intelligence Research*, 49:1–47.

[6] Church, K. W. and Hanks, P. (1990). Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.

[7] Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

[8] Dunning, T. (1993). Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19:61–74.

[9] Faruqui, M., Tsvetkov, Y., Rastogi, P., and Dyer, C. (2016). Problems with evaluation of word embeddings using word similarity tasks. *arXiv preprint arXiv:1605.02276*.

[10] Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., and Ruppin, E. (2001). Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pages 406–414. ACM.

[11] Gale, W. A. and Church, K. W. (1993). A program for aligning sentences in bilingual corpora. *Computational linguistics*, 19(1):75–102.

[12] Golub, G. H. and Reinsch, C. (1970). Singular value decomposition and least squares solutions. *Numerische mathematik*, 14(5):403–420.

[13] Levy, O. and Goldberg, Y. (2014a). Linguistic regularities in sparse and explicit word representations. In *Proceedings of the eighteenth conference on computational natural language learning*, pages 171–180.

[14] Levy, O. and Goldberg, Y. (2014b). Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pages 2177–2185.

[15] Levy, O., Goldberg, Y., and Dagan, I. (2015). Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

[16] Lund, K. and Burgess, C. (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior research methods, instruments, & computers*, 28(2):203–208.

[17] Luong, T., Socher, R., and Manning, C. D. (2013). Better word representations with recursive neural networks for morphology. In *CoNLL*, pages 104–113.

[18] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

[19] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

[20] Mikolov, T., Yih, W.-t., and Zweig, G. (2013c). Linguistic regularities in continuous space word representations. In *hlt-Naacl*, volume 13, pages 746–751.

[21] Palermo, D. S. and Jenkins, J. J. (1964). Word association norms: Grade school through college.

[22] Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

[23] Radinsky, K., Agichtein, E., Gabrilovich, E., and Markovitch, S. (2011). A word at a time: computing word relatedness using temporal semantic analysis. In *Proceedings of the 20th international conference on World wide web*, pages 337–346. ACM.

[24] Rastogi, P., Van Durme, B., and Arora, R. (2015). Multiview lsa: Representation learning via generalized cca. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 556–566.

[25] Rohde, D. L., Gonnerman, L. M., and Plaut, D. C. (2006). An improved model of semantic similarity based on lexical co-occurrence. *Communications of the ACM*, 8(627-633):116.

[26] Rubenstein, H. and Goodenough, J. B. (1965). Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.

[27] Steiger, J. H. (1980). Tests for comparing elements of a correlation matrix. *Psychological bulletin*, 87(2):245.

[28] Stratos, K., Collins, M., and Hsu, D. (2015). Model-based word embeddings from decompositions of count matrices. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1282–1291.

[29] Washio, K. and Kato, T. (2018). Neural latent relational analysis to capture lexical semantic relations in a vector space. *arXiv preprint arXiv:1809.03401*.

[30] Williams, E. J. (1959). The comparison of regression variables. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 396–399.

[31] Yang, W., Lu, W., and Zheng, V. (2017). A simple regularization-based algorithm for learning cross-domain word embeddings. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2898–2904.

[32] Zesch, T., Müller, C., and Gurevych, I. (2008). Using wiktionary for computing semantic relatedness. In *AAAI*, volume 8, pages 861–866.

[33] Zhao, Z., Liu, T., Li, S., Li, B., and Du, X. (2017). Ngram2vec: Learning improved word representations from ngram co-occurrence statistics. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 244–253.

VITA AUCTORIS

NAME:	Yaxin Li
PLACE OF BIRTH:	Yantai, Shandong province, China
YEAR OF BIRTH:	1992
EDUCATION:	Central University of Finance and Economics, B.Eng., Computer Science and Technology, Beijing, China, 2016
	University of Windsor, M.Sc in Computer Science, Windsor, Ontario, 2019