ANLP Assignment 3: Exploring Distributional Similarity in Twitter

s1802373 s1882930

1. Introduction

In this work, we compare three different methods on computing the context vectors and three different methods on similarity measures explored respectively. A benchmark list of word pairs is implemented to evaluate the performance of each method. With this data set, the various performance w.r.t the synonymy and non-synonymy will be discovered besides the discussion on general performance initially. Inspired by other previous work (Faruqui et al., 2016), tentative experiment on the limitation of statistical similarity is also performed.

2. The Question

The initial topic discussed in this work is how different methods rank the similarities between words. In this part of discussion, we would firstly focus on the lists of rank in different order computed by different methods and comparing them with a benchmark standard on synonymy and non-sysnonymy.

The second topic that would be discussed is about what limitations statistical similarity has due to different word sets.

3. The Words

The initial issue would be addressed w.r.t. the Rubenstein and Goodenough benchmark standard (Rubenstein & Goodenough, 1965). The ranked list of 65 noun word pairs from R&G (Rubenstein & Goodenough, 1965) would be implemented for the first task. Due to the limited data set, *serf* is deleted from the test word, as there are some word pairs with *serf* with zero count.

To explore what kind of similarity that is computed in the task, a list of test word is chosen by hand w.r.t. drinks and drinking appliance, which includes ['coffee', 'mug', 'cup', 'starbuck', 'costa', 'water', 'bottle'], adjectives, which includes ['like', 'admire', 'adore', 'love', 'hate', 'envy'], and Sherlock Holmes, which includes ['crime', 'murder', 'sherlock', 'holmes', 'detective', 'watson']. The words used in this part are chosen on purpose that there are some words with opposite meanings or strong correlationship.

4. The Methods

Computing similarities is involved with computing the context vectors and similarity measures.

4.1. Context Vectors

The traditional way to represent words vectors is to construct a high-dimensional sparse matrix M, where each row represents a word w in the vocabulary V and each column a potential context c in V. The value of each matrix cell M_{ij} represents the association between the word w_i and the context word c_j.

Besides positive pointwise mutral information(PPMI), there are other two methods implemented in this work to generate the context vectors, which are PPMI$_\alpha$ (Levy et al., 2015) and T-Test (Curran & Moens, 2002).

PPMI$_\alpha$ One way to reduce the bias toward low frequency events is to slightly change the computation for $P(c)$ by a different function $P_\alpha(c)$ that raises contexts to the power of α with subsection 4.1 and subsection 4.1.$PPMI_\alpha(w,c) = max(\frac{log_2 P(w,c)}{P(w)P_\alpha(c)}, 0)$ and $P_\alpha(c) = \frac{count(c)^\alpha}{\sum_c count(c)^\alpha}$

The value of α set in this part to compute the new word vectors is 0.75 as suggested (Levy et al., 2015).

T-TEST T test has been widely used for collocation discovery, which looks at the mean and variance of a sample of measurements (Manning et al., 1999). With the previous work (Curran, 2004), t test could be implemented into the computation for context vector as subsection 4.1. $A = \frac{P(w,f) - P(w)P(f)}{\sqrt{P(f)P(w)}}$

4.2. Similarity Measures

Apart from the cosine similarity, Euclidean distance and Jaccard index are both implemented and discussed in this work to compute the similarity among context vectors as well.

Euclidean distance The most simple way to get the simlilarity between two words is using the Euclidean distance(also called L2 norm). The formula is as follows:$sim_{L2}(\vec{v}, \vec{w}) = \sqrt{\sum_{i=1}^{N}(x_i - y_i)^2}$

When we want to comput the similiarity values, the word vectors($\vec{v}$ and $\vec{w}$) contain a list of unit values x_i and y_i, each representing the association between the context word and the target word. We use this formula to calculate the distance between two vectors as the similarity of two words.

Jaccard The more common method to get word similarity is Jaccard distance. From the formula, we can see that the numerator of the Jaccard uses the min function, computing the number of same features values in the vector. The denominator can be viewed as a normalizing factor, sumerising the the maximun between the two words $\vec{v}$ and $\vec{w}$ in the same vector position.$sim\_Jaccard(\vec{v}, \vec{w}) = \frac{\sum_{i=1}^{N} \min(v_i,w_i)}{\sum_{i=1}^{N} max(v_i,w_i)}$

5. Analysis

5.1. Q1

For the first question, we obtain the similarity values computed by various methods and compare the corresponding results to the top ten and the last ten word pairs from R&G (Rubenstein & Goodenough, 1965) due to the resource limitation. To compare the performance among different methods, we compute the Pearson correlation coefficient between the order from R&G (Rubenstein & Goodenough, 1965) and the order obtained by the methods implemented in the work.

In R&G (Rubenstein & Goodenough, 1965), the word pairs is ordered by the judgement on to what extent these two words are synonymy, which indicates that the top and last ten word pairs used in this part could test the performance among different methods on synonymy and non-synonymy respectively. The general performance of each methods would be approximately represented by the overall results on these 20 word pairs. The performance on synonymy, non-synonymy and the general performance is presented in Table 2, Table 3 and Table 4.

As is shown in Table 2, for the data set and word pairs chosen in this part of work, computing the context vectors by implementing PPMI_α presents the eminent performance, while the results of t test present overall the lowest Pearson correlation coefficient. Among the methods measuring similarity, cosine similarity and Jaccard perform similarly

The top ten word pairs chosed from R&G (Rubenstein & Goodenough, 1965) is the most similar word, i.e. the synonymy with the top ten highest ranking, which indicates that the Pearson correlation coefficient listed in Table 2 are tend to present the performance on synonymy with the methods we implemented. Corresponsively, the performance on non-synonymy is shown in the outcomes generated with the last ten word pairs in Table 3.

Measuring similarity with Euclidean distance perform the instable results on synonymy and non-synonymy, as the fluctuation among the similarity in Table 2 and Table 3. The results computed with 20 word pairs presents a more reasonable outcomes, as shown in Table 4, that the methods with Euclidean distance produce generally the worst evaluation, as computing similarity with Euclidean distance considers only the distance, only with which could not measure the similarities among the points with same ralatice position. For example, there are three vectors, $\mathbf{a} = [0, 1]$, $\mathbf{b} = [0, 3]$, $\mathbf{c} = [0, -1]$. The Euclidean distance between **a** and **b**, and **a** and **c** are both 2, but the mathematical interpretation on these two pairs are different.

5.2. Q2

Words which occur in a similar context tend to present similar meanings. Based on this hypothesis, we use vectors as representations in the context in the field of Lexical semantics. However, some underlying limitations of this method influence the accuracy of word similarity, which is opposite to human recognition. In order to explore the limitations of the semantic vectors, experiments with PPMI and cosine similarity are carried out, and results are presented in the appendix.

Experiment results show that the similarity(followed the pair) of ('coffee', 'starbuck')(0.36), ('love', 'hate')(0.17) and ('holm', 'sherlock')(0.41) take the first rank in each experiment. The statistical similarity in this task is computed with the context vectors which is computed w.r.t. the co-occurrence statistics of the words in the data set. Firstly, it strongly indicates that the similarity computed here is not strictly the similarity w.r.t. distribution semantics and could probably consider the Part of Speech (POS) tagging, e.g. *love* and *hate* could be assigned the same POS tag but these two words express opposite meanings, however, the similarity of ('love', 'hate') is eminent in the results. However, these two words could be classified into same category as they could be both interpreted as extreme sentiment. Secondly, ('holm', 'sherlock') forms the name of Sherlock Holmes. This method gives a high similarity to this word pair, while it belongs to different meaning and different word class from human recogniton. That is to say, the numerical description of similarity could not do a well-performed lexical semantic classification.

6. Conclusion

Overall, the combination of PPMI$_\alpha$ and cosine similarity produces the most accurate result w.r.t. the benchmark order from R&G (Rubenstein & Goodenough, 1965). To be more general, computing the context vector by PPMI$_\alpha$ or measuring similarity by cosine similarity would generate more accurate results w.r.t. the data set and evaluation standard used in this part of work.

Although word similarity could perform well in some semantic tasks under huge corpora (e.g. define words that have few synonyms), there are still several limitations among this method. Several conclusions are as follows. Firstly, the similarity method we use can classify the word according to the likeness of semantic content instead of the likeness of word meanings. Secondly, statistical similarity can capture word relationship under a certain context. Thirdly, there are still limitations (e.g. judging similarity by context correlation instead of human sense) among the methods in this work, which could lead to more deep research.

References

Curran, James R and Moens, Marc. Improvements in automatic thesaurus extraction. In *Proceedings of the ACL-02 workshop on Unsupervised lexical acquisition-Volume 9*, pp. 59–66. Association for Computational Linguistics, 2002.

Curran, James Richard. From distributional to semantic similarity. 2004.

Faruqui, Manaal, Tsvetkov, Yulia, Rastogi, Pushpendre, and Dyer, Chris. Problems with evaluation of word embeddings using word similarity tasks. *arXiv preprint arXiv:1605.02276*, 2016.

Levy, Omer, Goldberg, Yoav, and Dagan, Ido. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225, 2015.

Manning, Christopher D, Manning, Christopher D, and Schütze, Hinrich. *Foundations of statistical natural language processing*. MIT press, 1999.

Rubenstein, Herbert and Goodenough, John B. Contextual correlates of synonymy. *Communications of the ACM*, 8 (10):627–633, 1965.

A. Preliminary Task

The ranked list with correct similarity scores generated from the preliminary task is listed in Table 1.

SIMILARITY	WORD PAIR	COUNT 0	COUNT 1
0.17	('COMPUT', 'MOUS')	160828	22265
0.12	('CAT', 'MOUS')	169733	22265
0.09	('MOUS', 'DOG')	22265	287114
0.07	('CAT', 'COMPUT')	169733	160828
0.06	('COMPUT', 'DOG')	160828	287114
0.02	('@JUSTINBIEBER', 'DOG')	703307	287114
0.01	('CAT', '@JUSTINBIEBER')	169733	703307
0.01	('@JUSTINBIEBER', 'COMPUT')	703307	160828

Table 1. The ranked list with correct similarity scores

B. Experiment Results

	COS	EUC	JACCARD
PPMI	0.333	0.394	0.370
PPMIα	0.576	0.612	0.527
T-TEST	0.200	0.091	0.321

Table 2. The Pearson correlation coefficient computed by the results by using the top 10 words from R&G (Rubenstein & Goodenough, 1965)

	COS	EUC	JACCARD
PPMI	0.079	0.340	0.236
PPMIα	0.370	0.224	0.321
T-TEST	0.030	0.315	0.394

Table 3. The Pearson correlation coefficient computed by the results by using the last 10 words from R&G (Rubenstein & Goodenough, 1965)

	COS	EUC	JACCARD
PPMI	0.803	0.110	0.719
PPMIα	**0.854**	0.368	0.826
T-TEST	0.722	0.400	0.483

Table 4. The Pearson correlation coefficient computed by the results by using the 20 words from R&G (Rubenstein & Goodenough, 1965)

Constrained by the page limitation, we presents Table 5 only for the rank and Pearson Coefficient for all the 20 word pairs. The first 10 and last 10 is used in other two experiments for obtaining performance on synonymy and non-synonymy.

Table 5. Rank and Pearson Coefficient of 20 word pairs

	R&G	PPMI_cos	PPMI_min	PPMI_Jaccard	PPMI_a_cos	PPMI_a_min	PPMI_a_Jaccard	ttest_cos	ttest_min	ttest_Jaccard
gem, jewel	1	2	5	1	1	5	1	6	6	1
midday, noon	2	5	6	3	7	4	6	9	8	4
automobile, car	3	7	20	11	2	7	5	1	12	18
cemetery, graveyard	4	4	1	4	3	1	2	3	1	2
cushion, pillow	5	3	11	5	5	19	4	5	13	5
boy, lad	6	9	18	10	11	2	12	8	11	19
cock, rooster	7	10	7	14	8	12	9	10	5	14
implement, tool	8	1	16	2	4	15	3	2	7	3
forest, woodland	9	6	8	6	6	11	7	4	4	6
coast, shore	10	8	17	7	9	20	8	11	20	9
asylum, monk	11	15	4	9	14	9	13	19	14	7
asylum, fruit	12	17	14	18	13	14	14	17	15	16
grin, implement	13	18	9	17	19	6	19	16	3	13
mound, stove	14	16	2	15	16	3	17	13	2	8
automobile, wizard	15	13	12	13	15	18	15	14	17	11
autograph, shore	16	14	15	12	10	17	10	7	19	15
fruit, furnace	17	12	13	19	12	13	11	12	9	17
noon, string	18	11	10	8	18	10	18	15	16	12
rooster, voyage	19	19	3	16	17	8	16	18	10	10
cord, smile	20	20	19	20	20	16	20	20	18	20
Pearson	1	0.803008	0.109774	0.718797	0.854135	0.368421	0.825564	0.721805	0.4	0.482707