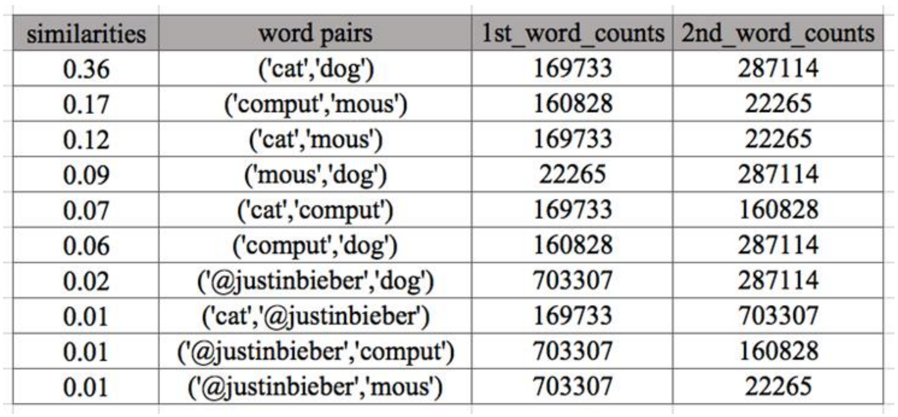
Student numbers: s1719048, s1755650

1. Preliminary task

The results of preliminary task are shown in Table 1.

Table 1 Sorted by cosine similarity



similarities	word pairs	1st_word_counts	2nd_word_counts
0.36	('cat','dog')	169733	287114
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
0.01	('@justinbieber','mous')	703307	22265

2. Choosing words and methods

Words

We wrote some codes to automatically choose test sets from a large range of frequency from 101 to 2490984. We used a large set (23 words with different frequencies and similarities) to explore how are different methods affected by word frequency, and a small set (5 words with different frequencies and similarities) to see specific examples about similarity ranking.

The large set is:

['gelena','ciley','#changmin','#yunho','today','work','tommorow','yesterday','morning','milk','beer', 'dog','cat','monitor','computer','apple','banana','samsung','sony', 'microsoft','tooth','brush','taylorswift']

The small set is:

['gelena', 'ciley', 'today', 'work', 'tommorow']

Methods

The context vectors computing methods we used here are co-counts, PMI, PPMI, Add-2 PMI and t-test. The similarities computing methods we used are Cosine, Jaccard and Dice. The definitions of these methods are shown in Table 2. Because co-counts just put the co-occurrences as the elements in context vector, we didn't show it in this table.

3. Experiments

3.1 Evaluation of different methods based on frequencies

We used the large words set here to find out how frequency can affect the methods chosen. We tried all five context vector computing methods with three similarity methods, the results of the relationship between frequency and similarity are shown in Figure 1.

We analysed these figures and got three conclusions:

a. For low-frequency words, in the cosine column, all the points on the left side of these five vector methods are overlapped together, which means when we look at a specific frequency, all similarities are the same, so cosine similarity perform badly on low-frequency words similarity estimation. While, both Jaccard and Dice perform better on all five vector methods (the point in the low frequency area are dispersive). And when looking at the Jaccard column, all vector methods give a relatively similar distribution, hence, Jaccard similarity is suitable for similarity estimation on low-frequency words.

b. For high-frequency words, cosine similarity has a better performance (the right side points are dispersive), while other similarity methods perform not so well. Hence, Cosine similarity is suitable for similarity estimation on high-frequency words.
c. From an overall view, the combination of Jaccard with PPMI or with t-Test is suitable for low-frequency words. And Cosine works well on high-frequency words when combined with any five context vector computing methods.

3.2 Evaluation of words ranking based on t-Test

We have tried all combination of context vector computing methods and similarity methods on the small words set, and found that only t-test ranked similarity differently when using different similarity methods. Hence, we focus on t-test here and tried to find out the differences among Cosine, Jaccard and Dice similarity measuring. And the results are shown in Table 3.

We can see from the Table 3 that the ranking given by Cosine meets our expects. Pair (ciley, gelena) co-occur 98 times in the given twitter file, which means they always occur in pairs and have a high similarity, and seldom occur with other words. Hence, both ('tommorow', 'gelena') and ('tommorow', 'ciley') should have a very low similarity. While in Table 3, Jaccard and Dice both rank ('tommorow', 'gelena') and ('tommorow', 'ciley') prior than ('today', 'work') and ('work', 'tommorow'), which is unreasonable based on our common sense. Therefore, we can get the conclusion that Jaccard and Dice are sensitive for low-frequency words, but when the test set is filled with words from a wide frequency range, they perform badly.

3.3 Some results surprised us

According to the definition of Spearman's rank correlation coefficient, this correlation coefficient should in [-1, +1], and when it is near 0, the frequency and similarity should have a low correlation, while when it is near -1 or +1, the frequency and similarity should have a high correlation. And as what we expected, the more correlated two things are, the points in the figure should be more like a line and when the correlation is low, the points should be more dispersive.

While, as shown in the figures of Figure 1, the coefficient of the Jaccard similarity based co-counts vectors is 0.10, which is near 0 and the points should be more dispersive, but it is not. And the coefficient of the Cosine similarity based on add-2 PMI vectors is 0.83, which is near 1 and the points should be more like a line, but it is not.

We think the reason why those happen is that co-counts vector computing itself is inaccurate, this inaccuracy might affect the evaluation of this Spearman's rank correlation coefficient.

Figures and Tables used in this report

Figure 1 The relationship between frequency and similarity on a large words set. The number below each figure is the Spearman's rank correlation coefficient.

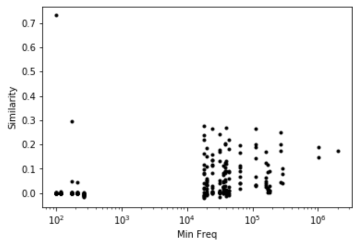
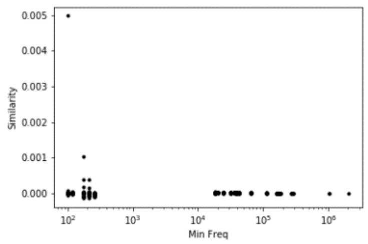
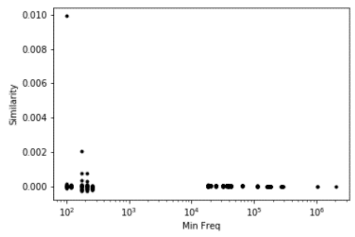
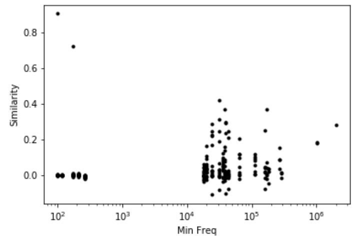
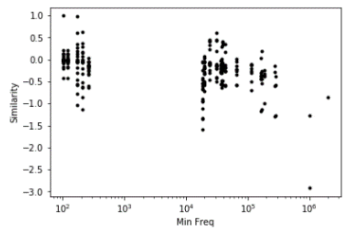
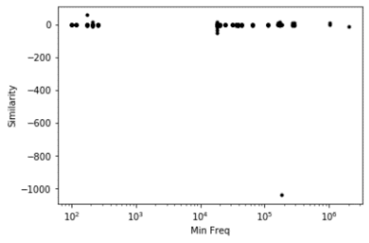
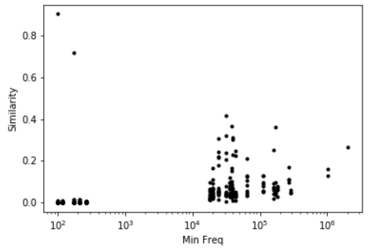
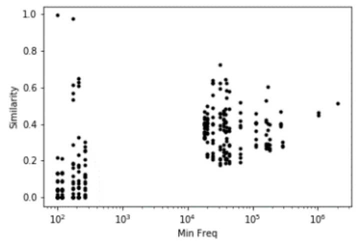
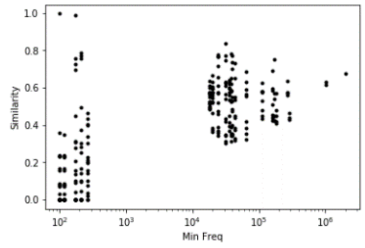
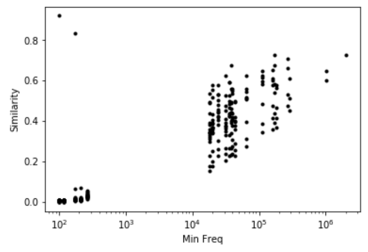
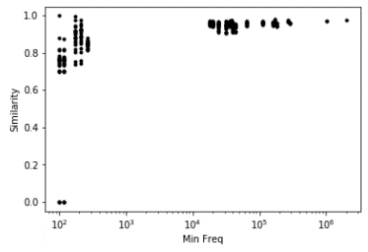
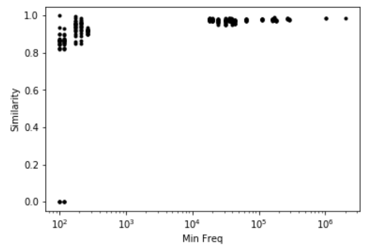
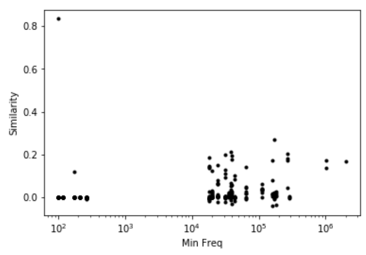
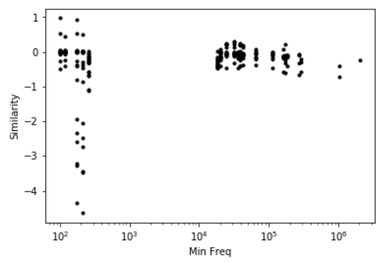
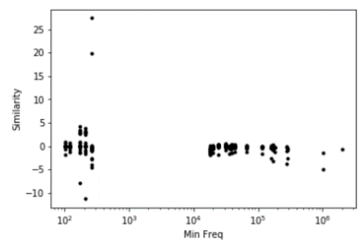
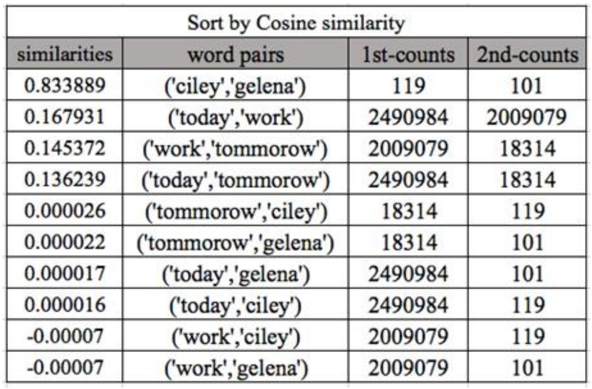
	Cosine	**Jaccard**	**Dice**
Co-occurrence frequency	 0.56	 0.10	 0.10
PMI	 0.39	 -0.39	 -0.30
PPMI	 0.76	 0.59	 0.59
Add-2 PMI	 0.83	 0.70	 0.70
T-test	 0.48	 -0.18	 -0.34

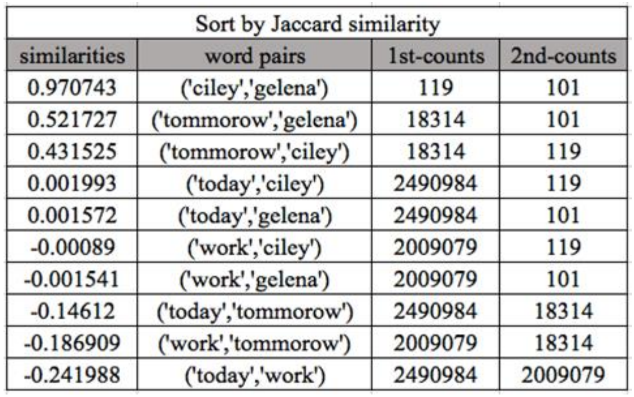
Table 2 Definitions of methods to compute context vectors and similarities.

Computing context vectors	Similarities computing methods
PMI: $\log_2 \frac{N*c\_xy}{c_x*c_y}$ PPMI: $\max(\log_2 \frac{N*c\_xy}{c_x*c_y},\ 0)$ Add-2 PMI: $\log_2 \frac{(N+N*2)*(c_{xy}+2)}{(c_x+2)*(c_y+2)}$ t-test: $\frac{(c_{xy}-c_x*c_y/N)}{\sqrt{c_x*c_y}}$	Cosine: $\frac{\sum_1^N v_i w_i}{\sqrt{\sum_1^N v_i^2}\sqrt{\sum_1^N w_i^2}}$ Jaccard: $\frac{\sum_1^N \min(v_i,w_i)}{\sum_1^N \max(v_i,w_i)}$ Dice: $\frac{2*\sum_1^N \min(v_i,w_i)}{\sum_1^N (v_i+w_i)}$

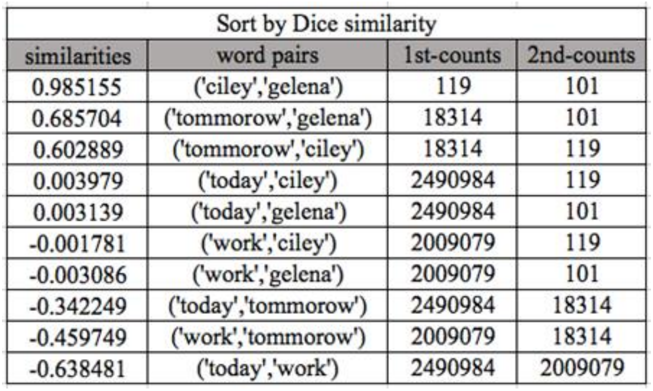
Table 3 The ranking of Cosine, Jaccard and Dice similarity on words set ['gelena', 'ciley', 'today', 'work', 'tommorow'] based on t-test



Sort by Cosine similarity			
similarities	word pairs	1st-counts	2nd-counts
0.833889	('ciley','gelena')	119	101
0.167931	('today','work')	2490984	2009079
0.145372	('work','tommorow')	2009079	18314
0.136239	('today','tommorow')	2490984	18314
0.000026	('tommorow','ciley')	18314	119
0.000022	('tommorow','gelena')	18314	101
0.000017	('today','gelena')	2490984	101
0.000016	('today','ciley')	2490984	119
-0.00007	('work','ciley')	2009079	119
-0.00007	('work','gelena')	2009079	101



Sort by Jaccard similarity			
similarities	word pairs	1st-counts	2nd-counts
0.970743	('ciley','gelena')	119	101
0.521727	('tommorow','gelena')	18314	101
0.431525	('tommorow','ciley')	18314	119
0.001993	('today','ciley')	2490984	119
0.001572	('today','gelena')	2490984	101
-0.00089	('work','ciley')	2009079	119
-0.001541	('work','gelena')	2009079	101
-0.14612	('today','tommorow')	2490984	18314
-0.186909	('work','tommorow')	2009079	18314
-0.241988	('today','work')	2490984	2009079



Sort by Dice similarity			
similarities	word pairs	1st-counts	2nd-counts
0.985155	('ciley','gelena')	119	101
0.685704	('tommorow','gelena')	18314	101
0.602889	('tommorow','ciley')	18314	119
0.003979	('today','ciley')	2490984	119
0.003139	('today','gelena')	2490984	101
-0.001781	('work','ciley')	2009079	119
-0.003086	('work','gelena')	2009079	101
-0.342249	('today','tommorow')	2490984	18314
-0.459749	('work','tommorow')	2009079	18314
-0.638481	('today','work')	2490984	2009079