CIS530 HW5 Submission Example

2 SimLex-999 Dataset Revisited (10 points)

Note **5 Extra points** will be awarded for creativity and a more thorough qualitative analysis.)

2.1 What is the least similar 2 pairs of words based on human judgement scores and vector similarity? Do the pairs match? [3 points]

Results

- least similar 2 pairs based on human judgement scores
    1. (new, ancient)
    2. (shrink, grow)
- least similar 2 pairs based on vector similarity
    1. (house, key)
    2. (flower, endurance)
- Do the pairs match?
    - No

2.2 What is the most similar 2 pairs of words based on human judgement scores and vector similarity? Do the pairs match? [3 points]

Results

- most similar 2 pairs based on human judgement scores
    3. (vanish, disappear)
    4. (quick, rapid)
    5.
- most similar 2 pairs based on vector similarity
    3. (south, north)
    4. (north, west)
- Do the pairs match?
    - no

2.3 Provide correlation scores and p values for the following models:
How do those correlation value compare to each other? [4 points]

Results

- glove.6B.50d.magnitude
  - `Correlation = 0.18100126067449063,`
  - `P Value = 1.2242211264976856e-17`
- glove.6B.100d.magnitude
  - `Correlation = 0.20506409092608713,`
  - `P Value = 3.41228663395174e-22`
- glove.6B.200d.magnitude
  - `Correlation = 0.23670323199262908,`
  - `P Value = 4.9936324557833286e-29`
- glove.6B.300d.magnitude
  - `Correlation = 0.25894302181101986,`
  - `P Value = 2.080389068003349e-34`
- glove.840B.300d.magnitude
  - `Correlation = 0.2860664813618063,`
  - `P Value = 1.293335613361039e-41`
- How do those correlation value compare to each other?
  - When the dimensions of the magnitude increases, the p-value goes down significantly and the correlation increases. Which means the similarity is getting closer to human's judgement when the dimensions increases.

3.1 Cluster Randomly (1 points)

3.1 Run clustering on dev data, report the f_scores from the dev data [1 point]

- Dev data

```
+----------------+----+----------------+
|     Target     | k  | Paired F-Score |
+----------------+----+----------------+
|    smell.v     | 4  |     0.2857     |
|    image.n     | 9  |     0.1204     |
|   express.v    | 7  |     0.1562     |
|     talk.v     | 6  |     0.2428     |
|     play.v     | 34 |     0.0308     |
|     miss.v     | 8  |     0.1972     |
|   produce.v    | 7  |     0.1968     |
|    write.v     | 9  |     0.1507     |
|   provide.v    | 7  |     0.2204     |
|    party.n     | 5  |     0.2017     |
|     bank.n     | 9  |     0.0741     |
|     plan.n     | 3  |     0.3693     |
|   shelter.n    | 5  |     0.2500     |
|  difference.n  | 5  |     0.2564     |
|     eat.v      | 6  |     0.2186     |
|     mean.v     | 6  |     0.1772     |
|    treat.v     | 8  |     0.1610     |
|     use.v      | 6  |     0.2407     |
|   suspend.v    | 6  |     0.1304     |
|   judgment.n   | 7  |     0.1480     |
| organization.n | 7  |     0.1734     |
|   interest.n   | 5  |     0.1704     |
|    paper.n     | 7  |     0.2044     |
|   operate.v    | 7  |     0.1237     |
|   receive.v    | 13 |     0.0903     |
|    watch.v     | 5  |     0.2041     |
|     rule.v     | 7  |     0.1267     |
|    simple.a    | 5  |     0.2000     |
|  atmosphere.n  | 6  |     0.1812     |
|    expect.v    | 6  |     0.2105     |
|  different.a   | 1  |     1.0000     |
|    begin.v     | 8  |     0.0897     |
|     note.v     | 3  |     0.3684     |
|     win.v      | 4  |     0.2756     |
|    source.n    | 9  |     0.1260     |
| performance.n  | 5  |     0.2347     |
|     wash.v     | 13 |     0.0849     |
|     hear.v     | 5  |     0.1835     |
|    climb.v     | 6  |     0.1963     |
|    degree.n    | 7  |     0.1985     |
+----------------+----+----------------+

=> Average Paired F-Score:  0.1592
```

3.2 Cluster with Sparse Representations (6 points)

3.2.1 Run clustering on dev data, report the f_scores from the dev data [1 point]

- Dev data

```
+----------------+----+----------------+
|     Target     | k  | Paired F-Score |
+----------------+----+----------------+
|   judgment.n   | 7  |     0.1989     |
|   produce.v    | 7  |     0.3147     |
|     bank.n     | 9  |     0.3373     |
|    expect.v    | 6  |     0.3011     |
|   suspend.v    | 6  |     0.2222     |
|   receive.v    | 13 |     0.2028     |
|    paper.n     | 7  |     0.3892     |
| organization.n | 7  |     0.2775     |
|    degree.n     | 7  |      0.2947     |
|  atmosphere.n   | 6  |      0.2291     |
|     plan.n      | 3  |      0.5102     |
|      win.v      | 4  |      0.4304     |
|    simple.a     | 5  |      0.1818     |
| performance.n   | 5  |      0.2713     |
|     mean.v      | 6  |      0.3375     |
|  different.a    | 1  |      1.0000     |
|    begin.v      | 8  |      0.2694     |
|     hear.v      | 5  |      0.3368     |
|     wash.v      | 13 |      0.1026     |
|    smell.v      | 4  |      0.3301     |
|     miss.v      | 8  |      0.2182     |
|   express.v     | 7  |      0.2504     |
|     eat.v       | 6  |      0.2938     |
|    treat.v      | 8  |      0.2101     |
|     play.v      | 34 |      0.0884     |
|     talk.v      | 6  |      0.3462     |
|     note.v      | 3  |      0.5333     |
|   shelter.n     | 5  |      0.3636     |
|    write.v      | 9  |      0.1742     |
|    source.n     | 9  |      0.2374     |
|    watch.v      | 5  |      0.3717     |
|   interest.n    | 5  |      0.1792     |
|   provide.v     | 7  |      0.3820     |
|    image.n      | 9  |      0.2037     |
|     use.v       | 6  |      0.3366     |
|     rule.v      | 7  |      0.1811     |
|   operate.v     | 7  |      0.2400     |
|    climb.v      | 6  |      0.3655     |
|    party.n      | 5  |      0.2419     |
|  difference.n   | 5  |      0.4000     |
+-----------------+----+-----------------+
=> Average Paired F-Score:  0.2529
```

3.2.2 Provide a brief description of your method in the report, making sure to describe the vector space model you chose, the clustering algorithm you used, and the results of any preliminary experiments you might have run on the dev set. [5 points]

- Brief Description of your method
    - [Description / Background]

I have used a different clustering algorithm which is Agglomerative Clustering. The method is the same as Cluster with Sparse Representation. The cooccurrence matrix is the same, but instead of using K-Means algorithm, I switched to use Agglomerative Clustering. Agglomerative Clustering starts the cluster from each point and K-Means assign k random centroids at the beginning. That's the difference. Agglomerative clustering is slower than K-means but it's more stable.

- [the vector space model you chose]

The vector space I used is the same as Sparse Representation which is "coocvec-500mostfreq-window-3.filter.magnitude"

- ○ [the clustering algorithm you used]

I have used a different clustering algorithm which is Agglomerative Clustering, which is a type of hierarchical clustering algorithm. It supports 4 different linkage methods

Linkage	Explanation of distance between clusters
ward	Minimize total within-cluster variance
complete	Max of distances between points in each cluster
average	Average of distances between all pairs
single	Min of distances between points

- ○ [the results of any preliminary experiments]

Linkage	Overall F-score	
ward	0.2389	
complete	0.2832	
average	0.3430	
single	0.3599	Using “single” achieved highest f-score

Using “single” achieves best performance

The F-Score from Single in Agglomerative Clustering:

```
+----------------+----+----------------+
|     Target     | k  | Paired F-Score |
+----------------+----+----------------+
|    degree.n    | 7  |     0.4666     |
|  atmosphere.n  | 6  |     0.3595     |
|     play.v     | 34 |     0.1712     |
|    smell.v     | 4  |     0.4839     |
|     talk.v     | 6  |     0.6142     |
|     plan.n     | 3  |     0.6387     |
|    party.n     | 5  |     0.3572     |
|   express.v    | 7  |     0.4345     |
|     win.v      | 4  |     0.4813     |
|    paper.n     | 7  |     0.4570     |
|     rule.v     | 7  |     0.2880     |
|   produce.v    | 7  |     0.4385     |
|     note.v     | 3  |     0.6400     |
| performance.n  | 5  |     0.4427     |
|     bank.n     | 9  |     0.2429     |
|     miss.v     | 8  |     0.3511     |
|   operate.v    | 7  |     0.2844     |
|    write.v     | 9  |     0.3516     |
|    source.n    | 9  |     0.3378     |
|     mean.v     | 6  |     0.4361     |
|    expect.v    | 6  |     0.4205     |
|  difference.n  | 5  |     0.4778     |
|    image.n     | 9  |     0.2885     |
|   shelter.n    | 5  |     0.4244     |
|     use.v      | 6  |     0.6223     |
|     wash.v     | 13 |     0.1963     |
|   interest.n   | 5  |     0.3121     |
| organization.n | 7  |     0.3958     |
|    treat.v     | 8  |     0.4033     |
|    begin.v     | 8  |     0.3688     |
|    climb.v     | 6  |     0.3375     |
|  different.a   | 1  |     1.0000     |
|   provide.v    | 7  |     0.7063     |
|   suspend.v    | 6  |     0.2830     |
|    simple.a    | 5  |     0.2564     |
|     hear.v     | 5  |     0.3234     |
|     eat.v      | 6  |     0.4495     |
|   judgment.n   | 7  |     0.3309     |
|   receive.v    | 13 |     0.2289     |
|    watch.v     | 5  |     0.4400     |
+----------------+----+----------------+
=> Average Paired F-Score:  0.3599
```

3.3 Cluster with Dense Representations (8 points)

3.3.1 Run clustering on dev data, report the f_scores from the dev data [1 point]

- Dev data

```
+----------------+----+----------------+
|     Target     | k  | Paired F-Score |
+----------------+----+----------------+
|   judgment.n   | 7  |     0.2498     |
|   produce.v    | 7  |     0.3720     |
|     bank.n     | 9  |     0.5937     |
|    expect.v    | 6  |     0.4500     |
|   suspend.v    | 6  |     0.4151     |
|   receive.v    | 13 |     0.1486     |
|    paper.n     | 7  |     0.5574     |
| organization.n | 7  |     0.3622     |
|    degree.n    | 7  |     0.4295     |
|  atmosphere.n  | 6  |     0.3171     |
|     plan.n     | 3  |     0.6387     |
|     win.v      | 4  |     0.4936     |
|    simple.a    | 5  |     0.2857     |
| performance.n  | 5  |     0.4221     |
|     mean.v     | 6  |     0.2952     |
|  different.a   | 1  |     1.0000     |
|    begin.v     | 8  |     0.3933     |
|     hear.v     | 5  |     0.2595     |
|     wash.v     | 13 |     0.1672     |
|    smell.v     | 4  |     0.3117     |
|     miss.v     | 8  |     0.1795     |
|   express.v    | 7  |     0.4044     |
|     eat.v      | 6  |     0.4226     |
|    treat.v     | 8  |     0.3188     |
|     play.v     | 34 |     0.1568     |
|     talk.v     | 6  |     0.6120     |
|     note.v     | 3  |     0.8400     |
|   shelter.n    | 5  |     0.4574     |
|    write.v     | 9  |     0.3152     |
|    source.n    | 9  |      0.2659     |
|    watch.v     | 5  |      0.3140     |
|   interest.n   | 5  |      0.3440     |
|   provide.v    | 7  |      0.6249     |
|    image.n     | 9  |      0.3051     |
|     use.v      | 6  |      0.4022     |
|     rule.v     | 7  |      0.2888     |
|   operate.v    | 7  |      0.2835     |
|    climb.v     | 6  |      0.2811     |
|    party.n     | 5  |      0.3479     |
|  difference.n  | 5  |      0.4794     |
+----------------+----+----------------+

=> Average Paired F-Score:  0.3401
```

3.3.2 Provide a brief description of your method in the report that includes the vectors you used, and any experimental results you have from running your model on the dev set. [5 points]

- Brief Description of your method
  - [Description / Background]

I used the Agglomerative Clustering Method, for each word's paraphrases, I create k lists. K is the number of clusters. For each paraphrase, I get the embedding from the vector space model I use. If the paraphrase doesn't exist in the vector space, I randomly select a vector. I generated a matrix X by doing this. I created the model with K as the number of clusters. Then I fit the X with the model. After getting the labels through fitting, I reconstruct K paraphrase lists based on the model labels.

  - [the vector space model you chose]

Used GoogleNews-vectors-negative300.magnitude

  - [the clustering algorithm you used]

Used Agglomerative Clustering cluster algorithm with specified k
I have used a different clustering algorithm which is Agglomerative Clustering, which is a type of hierarchical clustering algorithm. It supports 4 different linkage methods

Linkage	Explanation of distance between clusters
ward	Minimize total within-cluster variance
complete	Max of distances between points in each cluster
average	Average of distances between all pairs
single	Min of distances between points

  - [the results of any preliminary experiments]

Linkage	Overall F-score	
ward	0.2915	
complete	0.3034	
average	0.3704	
single	0.3798	Using "single" achieved highest f-score

```
+----------------+----+----------------+
|     Target     | k  | Paired F-Score |
+----------------+----+----------------+
|   judgment.n   | 7  |     0.2521     |
|   produce.v    | 7  |     0.4331     |
|     bank.n     | 9  |     0.6154     |
|    expect.v    | 6  |     0.4675     |
|   suspend.v    | 6  |     0.3023     |
|   receive.v    | 13 |     0.2016     |
|    paper.n     | 7  |     0.6070     |
| organization.n | 7  |     0.3906     |
|    degree.n    | 7  |     0.4019     |
|  atmosphere.n  | 6  |     0.4491     |
|     plan.n     | 3  |     0.6432     |
|     win.v      | 4  |     0.4892     |
|    simple.a    | 5  |     0.2759     |
| performance.n  | 5  |     0.4370     |
|     mean.v     | 6  |     0.3646     |
|  different.a   | 1  |     1.0000     |
|    begin.v     | 8  |     0.3382     |
|     hear.v     | 5  |     0.3241     |
|     wash.v     | 13 |     0.2886     |
|    smell.v     | 4  |     0.4677     |
|     miss.v     | 8  |     0.3217     |
|   express.v    | 7  |     0.4253     |
|     eat.v      | 6  |     0.4584     |
|    treat.v     | 8  |     0.3906     |
|     play.v     | 34 |     0.1453     |
|     talk.v     | 6  |     0.6358     |
|     note.v     | 3  |     0.8400     |
|   shelter.n    | 5  |     0.5304     |
|    write.v     | 9  |     0.3484     |
|    source.n    | 9  |     0.3106     |
|    watch.v     | 5  |     0.4700     |
|   interest.n   | 5  |     0.3376     |
|   provide.v    | 7  |     0.7403     |
|    image.n     | 9  |     0.3347     |
|     use.v      | 6  |     0.6252     |
|     rule.v     | 7  |     0.3228     |
|   operate.v    | 7  |     0.2977     |
|    climb.v     | 6  |     0.3412     |
|    party.n     | 5  |     0.3807     |
|  difference.n  | 5  |     0.4788     |
+----------------+----+----------------+
=> Average Paired F-Score:  0.3798
```

The f-score is 0.3798. It’s more than the f-score from the sparse embedding experiment. Which indicate more density helps to improve the correlation to human’s results.

3.3.3 In addition, for Task 3.2 and 3.3, do an analysis of different errors made by each system – i.e. look at instances that the word-context matrix representation gets wrong and dense gets right, and vice versa, and see if there are any interesting patterns. There is no right for this. [2 points]

- To compare, make a dataframe **Target word | F score_ sparse | F score_dense | Difference**

K-Means on both sparse and dense embedings

Target Word	k	F Score Sparse	F Score Dense	Difference
write.v	9	0.2205	0.1633	0.0572
treat.v	8	0.2505	0.2658	-0.0153
note.v	3	0.5333	0.619	-0.0857
paper.n	7	0.3783	0.3784	-1E-04
talk.v	6	0.3507	0.2812	0.0695
wash.v	13	0.12	0.1789	-0.0589
source.n	9	0.1743	0.3111	-0.1368
degree.n	7	0.3279	0.3531	-0.0252
operate.v	7	0.231	0.2491	-0.0181
bank.n	9	0.3373	0.7333	-0.396
smell.v	4	0.2947	0.35	-0.0553
use.v	6	0.4184	0.2926	0.1258
provide.v	7	0.3312	0.2495	0.0817
mean.v	6	0.3804	0.3145	0.0659
hear.v	5	0.3368	0.2712	0.0656
suspend.v	6	0.1905	0.386	-0.1955
party.n	5	0.2762	0.4209	-0.1447
watch.v	5	0.3361	0.4964	-0.1603
climb.v	6	0.35	0.2798	0.0702
image.n	9	0.2062	0.3038	-0.0976

performance.n	5	0.2544	0.3798	-0.1254
organization.n	7	0.2523	0.2224	0.0299
eat.v	6	0.2574	0.3386	-0.0812
expect.v	6	0.3654	0.3512	0.0142
different.a	1	1	1	0
shelter.n	5	0.3129	0.3805	-0.0676
simple.a	5	0.1538	0.2857	-0.1319
play.v	34	0.0903	0.1218	-0.0315
receive.v	13	0.2018	0.1628	0.039
interest.n	5	0.2145	0.3395	-0.125
begin.v	8	0.2022	0.358	-0.1558
atmosphere.n	6	0.3489	0.3123	0.0366
difference.n	5	0.3088	0.4235	-0.1147
plan.n	3	0.572	0.4451	0.1269
rule.v	7	0.2077	0.2846	-0.0769
miss.v	8	0.25	0.2703	-0.0203
produce.v	7	0.2721	0.2705	0.0016
express.v	7	0.3106	0.2698	0.0408
win.v	4	0.4497	0.4737	-0.024
judgment.n	7	0.2245	0.3219	-0.0974

- Now sort based on Difference to identify words where one works better than other
    - [Target word | F score_ sparse | F score_dense | Difference table]
    - [Error analysis]

When the word has K as 1, there's no difference.

When the word has a very large K, for example "play" has k=34, it's f-score is very low.

Overall, the F-Score Dense performs better than F-Score Sparse.

Target Word	k	F Score Sparse	F Score Dense	Difference	Comments

				=F Score Dense – F Score Sparse	
bank.n	9	0.3373	0.7333	0.396	This work has highest f-score under densed embedding
suspend.v	6	0.1905	0.386	0.1955	
watch.v	5	0.3361	0.4964	0.1603	
begin.v	8	0.2022	0.358	0.1558	
party.n	5	0.2762	0.4209	0.1447	
source.n	9	0.1743	0.3111	0.1368	
simple.a	5	0.1538	0.2857	0.1319	
performance.n	5	0.2544	0.3798	0.1254	
interest.n	5	0.2145	0.3395	0.125	
difference.n	5	0.3088	0.4235	0.1147	
image.n	9	0.2062	0.3038	0.0976	
judgment.n	7	0.2245	0.3219	0.0974	
note.v	3	0.5333	0.619	0.0857	
eat.v	6	0.2574	0.3386	0.0812	
rule.v	7	0.2077	0.2846	0.0769	
shelter.n	5	0.3129	0.3805	0.0676	
wash.v	13	0.12	0.1789	0.0589	
smell.v	4	0.2947	0.35	0.0553	
play.v	34	0.0903	0.1218	0.0315	
degree.n	7	0.3279	0.3531	0.0252	
win.v	4	0.4497	0.4737	0.024	
miss.v	8	0.25	0.2703	0.0203	
operate.v	7	0.231	0.2491	0.0181	
treat.v	8	0.2505	0.2658	0.0153	
paper.n	7	0.3783	0.3784	1E-04	

different.a	1	1	1	0	
produce.v	7	0.2721	0.2705	-0.0016	
expect.v	6	0.3654	0.3512	-0.0142	
organization.n	7	0.2523	0.2224	-0.0299	
atmosphere.n	6	0.3489	0.3123	-0.0366	
receive.v	13	0.2018	0.1628	-0.039	
express.v	7	0.3106	0.2698	-0.0408	
write.v	9	0.2205	0.1633	-0.0572	
hear.v	5	0.3368	0.2712	-0.0656	
mean.v	6	0.3804	0.3145	-0.0659	
talk.v	6	0.3507	0.2812	-0.0695	
climb.v	6	0.35	0.2798	-0.0702	
provide.v	7	0.3312	0.2495	-0.0817	
use.v	6	0.4184	0.2926	-0.1258	
plan.n	3	0.572	0.4451	-0.1269	

3.4 Cluster without K (6 points)

3.4.1 Run clustering on `dev` data, report the `f_scores` from the `dev` data [1 point]

- `Dev` data

```
+----------------+----+----------------+
|     Target     | k  | Paired F-Score |
+----------------+----+----------------+
|    degree.n    | 7  |     0.4460     |
|  atmosphere.n  | 6  |     0.3825     |
|     play.v     | 34 |     0.1753     |
|    smell.v     | 4  |     0.5033     |
|     talk.v     | 6  |     0.6414     |
|     plan.n     | 3  |     0.6745     |
|    party.n     | 5  |     0.3739     |
|   express.v    | 7  |     0.4256     |
|     win.v      | 4  |     0.5445     |
|    paper.n     | 7  |     0.6022     |
|     rule.v     | 7  |     0.3317     |
|   produce.v    | 7  |     0.4968     |
|     note.v     | 3  |     0.7719     |
| performance.n  | 5  |     0.4761     |
|     bank.n     | 9  |     0.5688     |
|     miss.v     | 8  |     0.3058     |
|   operate.v    | 7  |     0.3060     |
|    write.v     | 9  |     0.3567     |
|    source.n    | 9  |     0.3109     |
|     mean.v     | 6  |     0.4011     |
|    expect.v    | 6  |     0.4642     |
|  difference.n  | 5  |     0.5078     |
|    image.n     | 9  |     0.3086     |
|   shelter.n    | 5  |     0.5194     |
|     use.v      | 6  |     0.6658     |
|     wash.v     | 13 |     0.2579     |
|   interest.n   | 5  |     0.3383     |
| organization.n | 7  |     0.3955     |
|    treat.v     | 8  |     0.4046     |
|    begin.v     | 8  |     0.3389     |
|    climb.v     | 6  |     0.3646     |
|  different.a   | 1  |     1.0000     |
|   provide.v    | 7  |     0.7470     |
|   suspend.v    | 6  |     0.4878     |
|    simple.a    | 5  |     0.3889     |
|     hear.v     | 5  |     0.3351     |
|     eat.v      | 6  |     0.4729     |
|   judgment.n   | 7  |     0.3298     |
|   receive.v    | 13 |     0.2256     |
|    watch.v     | 5  |     0.4646     |
+----------------+----+----------------+

=> Average Paired F-Score:  0.3949
```

3.4.2 Provide a brief description of your method in the report that includes the vectors you used, and any experimental results you have from running your model on the dev set. [5 points]

- Brief Description of your method
  - [Description / Background]

For each target word's paraphrases, I run the Agglomerative Clustering method with different k from 2 to the number of paraphrases. When the paraphrases contain less than 2 words, no need to do clustering. For each value for k, I record the silhouette score. I will pick K clusters which gives the highest silhouette score. Repeat this process for each word.

  - [the vector space model you chose]

The vector I used is GoogleNews-vectors-negative300.magnitude

  - [the clustering algorithm you used]

The clustering method used is Agglomerative Clustering. I used "single" in linkage because based on sparse and dense experiment, "single" always achieve the highest correlation.

  - [the results of any preliminary experiments]

```
+----------------+----+----------------+
|     Target     | k  | Paired F-Score |
+----------------+----+----------------+
|    degree.n    | 7  |     0.4460     |
|  atmosphere.n  | 6  |     0.3825     |
|     play.v     | 34 |     0.1753     |
|    smell.v     | 4  |     0.5033     |
|     talk.v     | 6  |     0.6414     |
|     plan.n     | 3  |     0.6745     |
|    party.n     | 5  |     0.3739     |
|   express.v    | 7  |     0.4256     |
|     win.v      | 4  |     0.5445     |
|    paper.n     | 7  |     0.6022     |
|     rule.v     | 7  |     0.3317     |
|   produce.v    | 7  |     0.4968     |
|     note.v     | 3  |     0.7719     |
| performance.n  | 5  |     0.4761     |
|     bank.n     | 9  |     0.5688     |
|     miss.v     | 8  |     0.3058     |
|   operate.v    | 7  |     0.3060     |
|    write.v     | 9  |     0.3567     |
|    source.n    | 9  |     0.3109     |
|     mean.v     | 6  |     0.4011     |
|    expect.v    | 6  |     0.4642     |
|  difference.n  | 5  |     0.5078     |
|    image.n     | 9  |     0.3086     |
|   shelter.n    | 5  |     0.5194     |
|     use.v      | 6  |     0.6658     |
|     wash.v     | 13 |     0.2579     |
|   interest.n   | 5  |     0.3383     |
| organization.n | 7  |     0.3955     |
|    treat.v     | 8  |     0.4046     |
|    begin.v     | 8  |     0.3389     |
|    climb.v     | 6  |     0.3646     |
|  different.a   | 1  |     1.0000     |
|   provide.v    | 7  |     0.7470     |
|   suspend.v    | 6  |     0.4878     |
|    simple.a    | 5  |     0.3889     |
|     hear.v     | 5  |     0.3351     |
|     eat.v      | 6  |     0.4729     |
|   judgment.n   | 7  |     0.3298     |
|   receive.v    | 13 |     0.2256     |
|    watch.v     | 5  |     0.4646     |
+----------------+----+----------------+

=> Average Paired F-Score:  0.3949
```

The f-score is 0.3949. It's highest comparing to sparse representation and dense representation. Which means that each word might have a best k which can device the paraphrases.

Leaderboards [2 points + 3 bonus]

- No writing is required here, but be sure to submit a valid result (not -1) to Gradescope. [2 points]