11-641 HW2 Report
Liruoyang Yu
liruoyay

1. Corpus Exploration

1.1 Dev Set

the total number of documents
942
the total number of words
254852
the total number of unique words
14063
the average number of unique words per document
174.592

1.2 Test Set

the total number of documents
942
the total number of words
249516
the total number of unique words
13924
the average number of unique words per document
173.372

1.3 First Doc in the Dev Set

the total number of unique words
161
all of the word ids that occurred exactly twice in the document
2, 5, 10, 18, 23, 27, 28, 30, 32, 42, 44, 45, 46, 50, 52, 60, 62, 69, 79, 87, 91, 99, 102, 107, 114, 141

2. Experiments and Analysis

2.1 Baseline Approach

Parameters set

Number of document clusters: 98

Number of word clusters: 84

Stopping criteria: all clusters (including all document clusters and word clusters) stay stable for five iterations. The criterion for being stable is in one iteration, no more than 0.5% of all the data points (both document and word) change clusters.

Statistics and results**

Run	Converging Time (s)	# of Iterations	Doc Sum of Cosine Sim	Word Sum of Cosine Sim	F1 score for documents
1	495.458756	23	601.294698	4493.594214	0.5745150393
2	512.584429	23	606.863076	4817.039731	0.5443961025
3	352.115393	17	608.148562	4721.612809	0.5745836431
4	466.235817	20	606.668092	4845.675347	0.5825719054
5	429.662234	20	604.879655	4670.721081	0.5489311011
6	314.720467	15	605.467539	4788.852596	0.6017513929
7	333.637781	16	**610.805292**	4779.966577	0.5532218958
8	427.617133	19	609.750063	4693.461724	0.5947097338
9	279.50532	13	609.374537	4865.515554	**0.613944732**
10	578.506742	26	608.080225	**4900.035561**	0.5741893298
Mean	419.004	19.200	607.133	4757.648	0.576
Variance	9395.422	16.400	7.724	14150.752	0.00053

** for raw data please see out_baseline in the zip file.

Document clustering statistics:

F1 mean: 0.576

F1 variance: 0.00053

F1 best: 0.614

F1 with the best SCS : 0.553

Word clustering statistics:

SCS mean: 4757.648

SCS variance: 14150.752

SCS best: 4900.036

Analysis

For documents, the internal evaluation results, i.e. the SCS, almost cannot reflect the external evaluation results, i.e. the F1 score, because the correlation coefficient of the two is merely 0.13825. They have very low level of correlation.

For words, the run with best SCS is the 10th run. The result that displays the clusters with the real words is shown in the word_cluster_dev_10 file. From the result, we can see some clusterings make sense, while some don't. For example, in cluster 80, we can find "minute" and "hour"; but in the same cluster, we also can find "kid" and "poison". And "breakfast" (cluster 76) is not in the cluster that "lunch" (cluster 79) is in, rather, it's in the same cluster with bleeding. So, the result is not quite I expected. I think maybe it's because the number of documents that are provided is not enough, which causes the words that should be closer in common senses don't appear more often than the words that are not as close.

2.2 Custom Algorithm

Motivation

One of the problems that the baseline approach has is that it just considers the word frequencies, which leads to the result that the more one word occurs, the more impact it causes when we cluster. Thus, if there are two words, and one occurs a lot times in almost every document, while the other one only occurs in two documents, then if we only consider the word frequencies, the many documents will be closer than the two documents, which doesn't make sense, because those two documents tend to be about a same area that's different from the many others.

So, to resolve this problem, I used tf-idf as the weights of the matrix, in hope that the algorithm clusters the documents and words more appropriately.

Statistics and results**

Run	Converging Time (s)	# of Iterations	Doc Sum of Cosine Sim	Word Sum of Cosine Sim	F1 score for documents
1	398.425154	17	538.510339	4878.971024	0.571447389
2	346.047201	15	529.485256	4895.549555	0.5504614103
3	450.661375	19	**540.202328**	**5190.371151**	0.5844035789
4	460.014841	20	533.172572	4925.993228	0.6169667077
5	363.97263	15	539.513592	5180.70503	0.5531304748
6	414.458873	18	531.83431	4967.367903	0.572011913
7	356.629903	15	530.344421	4794.444458	0.5429994308
8	327.304883	14	539.644146	5052.675239	**0.6255277931**
9	404.372286	17	537.971543	4863.618146	0.6112268478

10	344.670787	15	538.043978	4954.367701	0.5812501061
Mean	386.656	16.500	535.872	4970.406	0.581
Variance	2117.1196	4.0556	17.4737	17549.7201	0.00084

** for raw data please see out_custom in the zip file.

Document clustering statistics:

F1 mean: 0.581
F1 variance: 0.00084
F1 best: 0.626
F1 with the best SCS : 0.584

Word clustering statistics:

SCS mean: 4970.406
SCS variance: 17549.7201
SCS best: 5190.371

Comparison

With the customary, the mean SCS for word increased by 4%, and the mean F1 increased by 0.8%, while the mean SCS for documents dropped by 11%. So, this customary worked, because the F1 score increased, for both the best and the mean. But the improvement isn't big, maybe because the document frequencies of words in the dev set are not so different from each other. And the decrease of the SCS of the documents means that the documents are now less equally distributed among clusters, which means the features that separate the documents are more characterized now.
And the custom algorithm converges faster than the baseline approach.

3. Software Implementation and Data Processing

3.1 Software architecture

The software consists of 4 major parts, the bipartite clusterer, the data parser, the clustering user interface, and the analytics user interface. The clusterer is implemented as a python class which runs the clustering and calculates the SCS for the resulting clustering results. And the other 3 part are implemented as python modules.

The reasons for this implementations are as follows:

1. To maximize reuse and function separation. For example, all parts need parser, and interfaces shouldn't be mixed with underlying clusterer.
2. Since clusterer is stateful, so it's better to encapsulate it as a class, and let each instance manage their own states and behave accordingly.
3. Different functional needs result in different user interfaces.

3.2 Major data structures

Every matrix is represented by a list of lists. Each element of the outer list represents one row. The elements are also lists, and each of them contains a number of tuples. Each tuple is an inverse-indexed matrix cell.

Clusters are represented as lists of lists. Outer lists represent clusters, and each inner list contains a number of integers, which are the indexes of the data points (matrix rows) in the matrices.

3.3 Programming tools and libraries used

IDE: Jetbrains PyCharm 4.5.4
Libraries: python built-ins

3.4 Strengths, weaknesses, and problems

Strengths

The software can converge during a reasonable amount of time on the data using bipartite k-means clustering, and provide reasonable results.

The design achieved high granularity reusing and function separation, easy for understanding and maintenance.

Weaknesses

The number of clusters for word maybe a little bit low, resulting in big clusters. But for this number, it still has empty clusters.