Data Mining Assignment II

Fang Zhou

1. Introduction

The objective of this assignment is to leverage clustering algorithms to classify documents. As the important features of the articles have been extracted in the last assignment, I will focus on implementing the clustering algorithms with different parameters and distances/linkages, evaluating the performance, and analyzing the results.

2. Methods

2.1 Article Feature

In this assignment, I will classify documents based on the TF-IDF features. TF-IDF, short for term frequency–inverse document frequency, is a statistic used to reflect how important a word is to a document in a set. Although the word frequency is also a good feature, I choose TF-IDF because I expect to get the results with statistic meaning. I choose the TF-IDF features of 500 most important words in the corpus, which means each vector includes 500 features.

2.2 Data Pre-processing

However, the vectors collected in the last assignment don't fit well for this assignment. For example, some articles may miss the topic part so that it cannot be classified into any categories. I pre-process the data following the rules shown below:

- If an article does not have the body part or its body part is empty, then skip this article.
- If an article does not have the topic part or its topic part is empty, then skip this article.
- If an article has more than one topic, then random pick up one topic for this article.

After this processing, the number of articles decreases to 8654. And each article is bonded with only one topic. This processing makes the future processing easy and correct.

2.3 Clustering Algorithms

In this assignment, I choose k-means and hierarchical clustering to classify the document. For k-means, I use manhattan distance and cosine distance. For hierarchical clustering, I choose manhattan distance and cosine distance with single-linkage and complete linkage, respectively. The equations of these distances and the meanings of linkages are the same with the ones we discussed in the class so I don't list them here because of the limited space.

2.4 Evaluation Methods

I will evaluate a clustering from the perspective of scalability and quality. Scalability is measured by the running time. The less running time means the better scalability. The scalability becomes more and more important because the amount of data is increasing exponentially in the big data era. Topic

entropy, variance in number of elements in each clusters, and skewness are used to evaluate the quality of a clustering algorithm. Skewness is computed as the equation shown below. In this specific case, a good clustering algorithm should generate several clusters, which is similar to distribution of actual topics.

$$skew(X) = \frac{E\left[(X-\mu)^3\right]}{\sigma^3}$$

Where X is the number of each cluster, u and σ are average element number and standard derivation.

2.5 Preparations of Topics

For evaluating the clustering, I write a program to collect the topic of each article and the output is stored for future evaluation. I organize the topics and find the number of distinct topics is 60. This value is very helpful to determine the number of clusters configured in the clustering algorithms.

3. Experiments

3.1 Testbed

I test the program on a single machine. The machine has a 2.4 GHz 32-core Intel Xeon CPU E5-2630, 64 GB memory and one 500 GB Western Digital SATA hard drives.

3.2 Experiments Details

I use python to implement the whole assignment, including the pre-processing program, clustering algorithms, evaluation programs, and etc. I conduct each experiment for three times and calculate the average values. I try several distances and linkages on two clustering algorithms with different number of clusters: 5, 60, 120, 240, and 500. The reason is that 60 is the number of distinct topics of croups. So I pick up some typical number of clusters to test. It should be noted that I set the maximum loop number to 1000 in k-means clustering algorithm, which means if k-means cannot stop in 1000 times then the algorithm will stop and return results.

3.3 Optimization

After I finish the first version, I test the performance. Hierarchical clustering is so slow in my environment. The expected running time is longer than 3 days. This make me spend a lot of time on optimize my program written in Python. I tried multithreading in my program, but it doesn't work because the CPU utilization has exceed 100 percent. Finally, I optimize the union and update operations in the program and get an acceptable result. More details can be seen in my submitted code.

4. Results

In this section, I will show the scalability and quality results of each clustering algorithms.

4.1 Scalability

Time (Second)	|K|=5	|K|=60	|K|=120	|K|=240	|K|=500
K-means, Manhattan	2	16	28	46	142
K-means, Cosine	35	70	141	279	582
Hierarchical Clustering, Manhattan, Single	731	692	678	660	632
Hierarchical Clustering, Manhattan, Complete	1113	1113	1113	1112	1110
Hierarchical Clustering, Cosine, Single	798	798	796	786	763
Hierarchical Clustering, Cosine, Complete	1333	1333	1333	998	672

Table I: Scalability of Two Clustering Algorithms

Table I shows the scalability of two clustering algorithms with different parameters. As we can see, generally speaking, k-means clustering has a better performance than hierarchical clustering. For example, the average time of k-means with 5 clusters is 17.5 seconds. However, the hierarchical clustering in the same case needs 993.75 seconds on average.

For k-means clustering, distance model plays a very important role on the algorithm's scalability. Different distances bring different performance. Manhattan distance need less computation than Cosine distance. So the performance of Manhattan k-means is 4-17 times faster the one with Cosine distance.

But we can see the consuming time of hierarchical clustering is very stable. For example, the running time difference between hierarchical clustering with $|K|=500$ and the one $|K|=5$ is 3 seconds. This is because the hierarchical clustering is a bottom-up process. Before the process starts, hierarchical clustering spends a long time on initializing the distance between each two nodes in the feature space. After the initialization, the union and computation operations do not cost a lot of resource. So the finishing time of hierarchical clustering with 5 clusters is not changed much, compared to the one with 500 clusters.

To simply conclude, k-means clustering has the better scalability than hierarchical clustering, without considering the qualities of clustering algorithms.

4.2 Quality

First, I want to show the entropy results.

Entropy	|K|=5	|K|=60	|K|=120	|K|=240	|K|=500
K-means, Manhattan	2.115776	1.209066	1.001595	0.915537	0.732387
K-means, Cosine	2.094005	1.152526	1.771633	1.637217	2.24575
Hierarchical Clustering, Manhattan, Single	3.070338	3.03251	2.999185	2.935162	2.796019
Hierarchical Clustering, Manhattan, Complete	3.035178	2.709094	2.540623	2.31471	2.037487
Hierarchical Clustering, Cosine, Single	3.063842	3.037329	2.988697	2.901282	2.798079
Hierarchical Clustering, Cosine, Complete	2.812712	1.643551	1.21737	0.8991	0.651962

Table II: Entropy of Two Clustering Algorithms

From Table II, we can notice that except k-means with cosine distance, all other clustering algorithms have the less and less entropy with increasing numbers of clusters. Back to the definition of entropy, if the value is near to 0, then the classification is good; otherwise, it is not a good classification. Because we only have 60 distinct topics in the corpus, if the clustering algorithm works well, it should provide the very small entropy value in the third column (|K|=60) of Table II, compared to other numbers of clusters. The most important thing is the trend of this kind of clustering algorithm follows our expectations. Other algorithms have lasting decreasing entropy values, which is not reasonable. The reason is very complex and more analysis is provided in the discussion section.

Skewness and variance results are shown in Table III and IV.

Skewness	|K|=5	|K|=60	|K|=120	|K|=240	|K|=500
K-means, Manhattan	0.133251	5.241143	7.26562	8.188561	12.41439
K-means, Cosine	0.415799	1.582462	6.432503	6.843022	15.30137
Hierarchical Clustering, Manhattan, Single	1.5	7.550944	10.81702	15.3949	22.2934
Hierarchical Clustering, Manhattan, Complete	0.760886	2.229238	2.253763	2.244513	3.049088
Hierarchical Clustering, Cosine, Single	1.5	7.55091	10.81695	15.39383	22.29151
Hierarchical Clustering, Cosine, Complete	1.341917	6.466899	9.311368	12.9093	6.623629

Table III: Skewness of Two Clustering Algorithms

Variance	|K|=5	|K|=60	|K|=120	|K|=240	|K|=500
K-means, Manhattan	1406047	75073.21	28598.4	6643.963	1750.533
K-means, Cosine	999006.2	6204.812	44590.29	10436.87	22998.58
Hierarchical Clustering, Manhattan, Single	11968832	1207284	599579.2	291442.9	130598.4
Hierarchical Clustering, Manhattan, Complete	172268.2	1749.346	447.7531	112.3466	19.16114
Hierarchical Clustering, Cosine, Single	11748443	1161665	568894.5	259817.8	115231.9
Hierarchical Clustering, Cosine, Complete	1045715	62141.71	31042.55	12510.54	816.7891

Table IV: Variance of Clusters in Two Clustering Algorithms

In order to know the quality of skewness and variance, we first need to know these values of corpus topics. The corpus topics' values can be seen in Table V.

Skewness	5.869837
Variance	273478.6

Table V: Skewness and Variance of the whole topics

For skewness, both clustering algorithms with 5 clusters cannot provide similar value, compared to the corpus topics. The best entropy result, K-means with cosine distance, does not have the similar skewness and variance value. After observation, the most similar clustering algorithm is the hierarchical clustering with cosine distance and complete linkage, whose skewness value is 6.466899 and variance

value is 62141.71. The skewness is close to 5 when the number of clusters is 60 or 120. However, the variance changes a lot in different cases.

5. Discussions

It is very complicated and difficult to explain the phenomenon we have tested and observed. However, we can conclude some results we observed:

- The results are hard to explain for the unsupervised clustering algorithms. The data itself can tell people some stories, but it is hard to explain the stories.
- The number of clusters is the most important factor for clustering algorithms. As we have seen, the different clusters can make the totally different classifications.
- If the time is the limited, you should choose k-means clustering; if the time is enough, you'd better try both clustering.
- For TF-IDF features, to make the entropy least and get the most reasonable result, the best algorithm is k-means with cosine distance. To make the clusters more similarity with the actual data, the best way is to use hierarchical clustering with cosine distance and complete linkage. But both experience need a pre-condition, we know the number of distinct classifications in the corpus.
- These experiences only work for the documentation classification. It still need to test on other classification problems.