Lab 3 - Parallelizing k-means Stat 215A, Fall 2017

Yizhou Zhao 3032130362

October 24, 2017

1 Introduction

The paper of Ben-Hur gives a good guide on how to choose the number of clusters and the similarity metrics to measure the clusters. First, I will make a brief conclusion on the similiarity metrics which are introduce in the paper.

Definition: (Cluster Adjacency Matrix)

$$c_{i,j} = \begin{cases} 1, \ if \ i \ and \ j \ are \ in \ the \ same \ cluster \\ 0, \ otherwise \end{cases}$$

Let C^1, C^2 be two cluster adjacency matrix from data. Then the element-wise dot product $< C^1, C^2 >$ is defined as:

$$< C^1, C^2 >= \sum_{i,j} C^1_{i,j} C^2_{i,j}$$

And the cosine similarity-measure:

$$cor(C^1, C^2) = \frac{< C^1, C^2 >}{\sqrt{< C^1, C^1 >< C^2, C^2 >}}$$

Discussion: One possible correction or improvement here is to redefine $c_{i,i} = 1, i = 1, 2, 3....$ Intuitively, a sample is of the same cluster of itself. Also because when the number of samples are small, those ones can weight a lot the on the similarity measure.

And let $N_{i,j}$ for $i, j \in 0, 1$ be the number of entries on which C^1, C^2 have value i, j.

Definition:(matching coefficient)

$$M(C^1, C^2) = \frac{N_{0,0} + N_{1,1}}{N_{0,0} + N_{1,1} + N_{0,1} + N_{1,0}} = 1 - \frac{1}{n}||C^1 - C^2||^2$$

Definition:(Jaccard coefficient)

$$J(C^1, C^2) = \frac{N_{1,1}}{N_{0,1} + N_{1,0} + N_{1,1}} = \frac{< C^1, C^2 >}{< C^1, C^1 > + < C^2, C^2 > - < C^1, C^2 >}$$

2 Outline for coding

The steps to finish the tasks in labs include: reading the binarized data, subsampling data, k-means clustering and calculating the similaritiy metrics. I chose to parallelize the subsampling, clustering and the calculating similarity parts.

- subsampling: by *sample_frac* in **dplyr**
- clustering: by *kmeans* function in **R**
- calculating similarity: by memory-efficient version the functions written by **C++**.

And a shell script was written to run those R codes on our cluster.

3 Plot and results

3.1 R vs Cpp

Function name	Time(nanoseconds)	Language
dotProduct(M1, M2)	2973985.6	R
matchCoeff_R(M1, M2)	3064470.0	R
corSim_R(M1, M2)	8400983.2	R
jaccardCoeff_R(M1, M2)	11466100.4	R
sumProduct(A, B)	64560.8	C++
matchCoeff(A, B)	277244.8	C++
corSim(A, B)	147763.2	C++
jaccardCoeff(A, B)	187254.0	C++

The table above shows the running time comparison between R and Cpp for get the similarity metrics for two clusters of 100 objects. We can see from the table that Cpp runs a lots fast than R.

It is worth to mention that for the built-in functions, such as *kmeans,sampling_frac* and etc. , those functions run really fast. Because the base language to build R is C and Cpp. Therefore, it is unnecessary for us to rewrite functions to do random sampling and clustering.

3.2 Results

I chose to do the clustering as easy as possible: kmeans with L^2 penalty, 5000 as the maximum time of iterations and without projections to lower dimensional space. I set $k.max$ as 10 and for each number of clusters, I run ten times. Besides, I chose $r = 0.7$ as the sampling ratio.

Figure 1 shows the results after parallelized computation. Judging by the histograms and the CDF plot, there are transitions between 2 and 3 clusters and 3 to 4 cluster, and the histograms of three clusters and four clusters seem to be natural– gathered in a small range of interval. Therefore, it is probably that three clusters, or four clusters should be the best.

4 Comments

The algorithm provided by Ben-Hur, is an incredibly robust way to judge cluster results, even though that the results of cluster numbers tend to be favor of fewer clusters. I agree that the authors of the paper at least found a way to evaluate the distributions from the correlation similarities/matching coefficients/Jaccard coefficients. The transition between the histogram distributions from being concentrated is a good indicator.

And for the histograms, the similarity distributions gather on one or more intervals. Ideally, the distributions of them stay the same. However, one important caveat is that it is theoretically possible for the histograms to have more than one phase transition.

I trust that 3 is a reasonable cluster size, and informative for regional dialect relationships with geography. If there was more computational time, I would try run this algorithm for bigger k.max, different clustering methods, metrics and sample ratios.

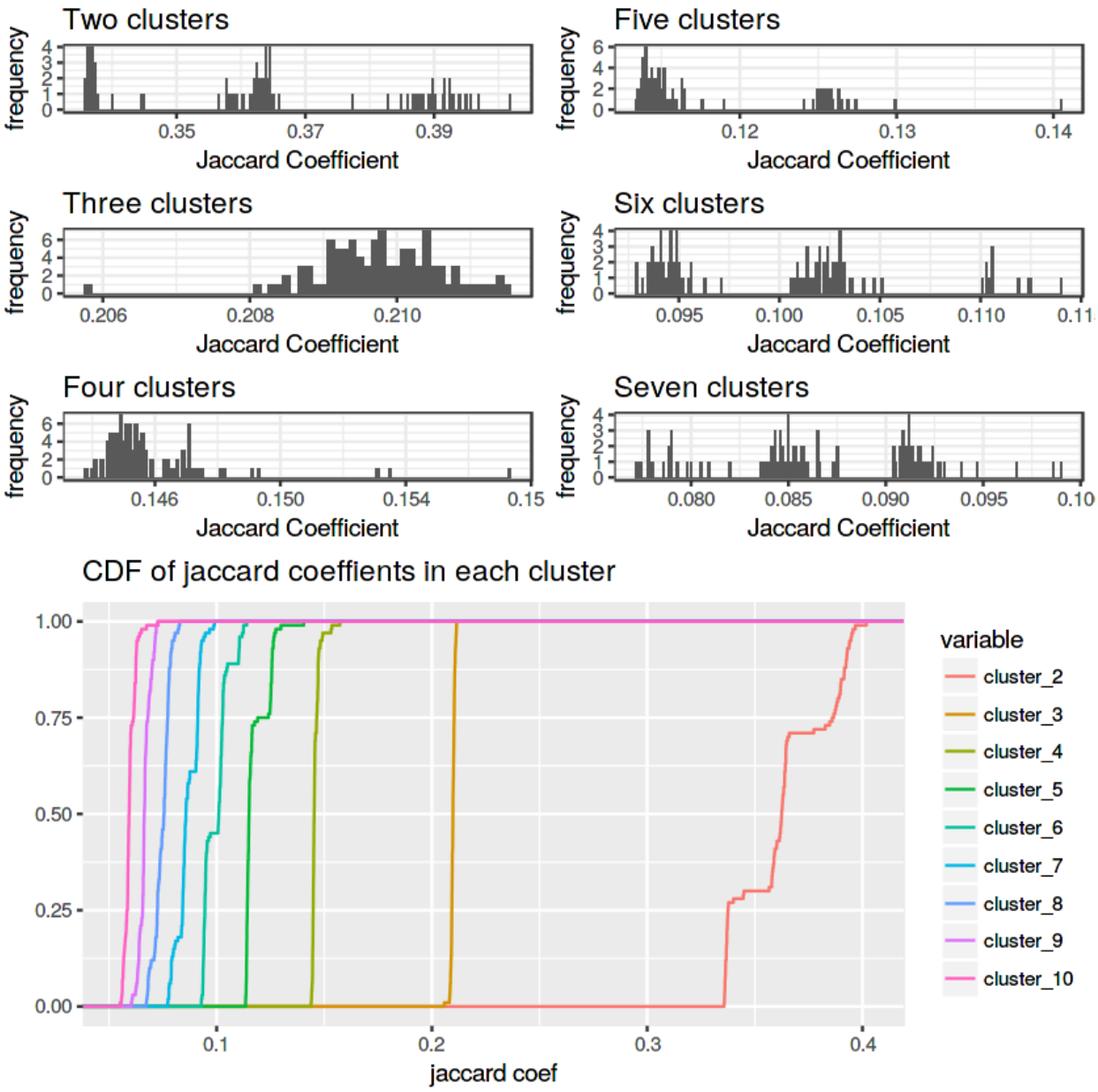

Figure 1: Left: histogram of the Jaccard Coefficient; right: overlay of the cumulative distributions for increasing values