DMML Assignment 2

Arka Roy, Subhamon Supantha

March 2024

1 Introduction

This report aims to explain the various details of the models and methodology used for the tasks assigned. The task required us to cluster the documents in the "Bag of Words" data set from the UCL Machine Learning Repository that contains five text collections in the form of bags-of-words via K-means clustering for different values of K and determine an optimum value of K. As a similarity measure, we have to use the Jaccard index, that measures similarity between two documents based on the overlap of words present in both document. We have reported our results on the three smaller datasets (Enron emails, NIPS blog entries, KOS blog entries).

2 The Data Set

In the following passage, D is the number of documents in the collection, W is the number of words whose frequency is counted (i.e., W is the number of words in vocab.XYZ.txt) and NNZ is the number of non-zero frequency entries for this collection (i.e., NNZ is 3 less than the number of lines in docword.XYZ.txt). Information about the datasets in the repository:

```
Enron Emails:
D=39861
W=28102
N=3710420

NIPS full papers:
D=1500
W=12419
N=746316

KOS blog entries:
D=3430
W=6906
N=353160
```

As we can see the enron dataset is significantly larger than the other two.

3 Methodology

Each document is represented as a d dimensional vectors where the d is the number of words in the vocabulary and the i^{th} component is 1 if the document contains the i^{th} word, 0 otherwise.

It is not traditional to do KMeans clustering with Jaccard Index as a measure of similarity.

We assume the aim is to do the following:

Given a set of points $X = \{x_1, x_2, \ldots, x_n\}$, we need to find a partition $C = \{C_1, C_2, \ldots, C_k\}$ such that C is the argmin over the choice of $C_1, C_2, \ldots, C_k$ of the following function.

$$\sum_{j=1}^{j=k} \sum_{x_i \in C_j} J(x_i, c_j) \tag{1}$$

Here, c_j is the centroid of the j^{th} cluster, $J(x, y)$, is 1 - the Jaccard similarity index of x and y.

For the centroid of a cluster, we have used the mean of the vectors in that cluster. This is a probabilistic view of centroid. We deliberately haven't chosen the centroid in $\{0, 1\}^d$ as computing that is highly inefficient(exponential) in the exact case. For an approximation algorithm, where we take the mean of the vectors in the cluster and project that to $\{0, 1\}^d$, the result is meaningless for a sparse dataset like enron where all the centroids would just be $(0, 0, \ldots, 0)$.

To accommodate for the probabilistic view of the centroid, we have extended the definition of the Jaccard similarity index to

$$jaccard(x, y) = \frac{x^T y}{\|x\|_1 + \|y\|_1 - x^T y} \tag{2}$$

Note that when the vectors are in $\{0, 1\}^d$, this is the same as the traditional jaccard index. Also note that the above function is bounded in $[0, 1]$.

Now coming to the practical implementation part, we have loaded the data in a sparse matrix to reduce the space consumption and ease of indexing. We have also defined KMeans from scratch to incorporate the Jaccard index metric. Given the size of the dataset, we have tried to reduce space and time usage as much as possible. We have also tried to parallelize the computation of Jaccard distances from each point in a cluster to its centroid during the cluster update step of the algorithm. However, this did not show a significant improvement in performance. The next section states our findings.

4 Conclusion

We needed the optimal k values for each of the dataset. We have used the elbow method for finding the optimal k.

The results are tabulated as below.

Dataset	Optimal k	Time taken(in sec)
enron	6	¡22961.3
nips	4	532.4
kos	5	280.6

Table 1: Table to test captions and labels.

It must be noted that the time taken for enron is not an accurate representation of the actual time taken as during execution the device went to power saving sleep mode where performance is limited.