Project C
CSE 494/598
Hemal Khatri

Overview

This goal of this project is to mine patterns in the search results by clustering the search results. The different methods used for clustering are: **1. K-Means 2. Buckshot 3. Bisecting K-Means**. For each of these methods, we show the algorithm used for computing the clusters as well as the time complexity for each of these algorithms. We also compare the quality of the clusters across these methods using cluster similarity measure such as intra-cluster similarity and inter-cluster similarity. We also experiment with different number of clusters as well as using varying number of documents for clustering. A GUI is developed as a web servlet which displays the clusters to a given user query in a web page.

K-Means

Offline Computation:

We compute and store the document vector corresponding to each document while computing the norm for each document. The document vector is then used for finding similarity between two documents based on cosine similarity model.

Online Computation

For a given query, we first extract all the results obtained using Vector Space Ranking as described in Part A. Then for the top N documents in the results, we apply k-Means algorithm as described below. The centroid for each cluster is computed using average of the term weights of documents in that cluster.

Algorithm K-Means

Input: K:number of cluster, D:Top N documents

Output: K clusters of documents

```
    Generate K centroids $C_1$, $C_2$, ..,$C_k$ by randomly choosing K documents from D
    Repeat until there is no change in cluster between two consecutive iterations
        for each document $d_i$ in D
            for j = 1 to K
                Sim($C_j$,$d_i$) = Find cosine similarity between $d_i$ and $C_j$
            end for
            Assign $d_i$ to cluster j for which Sim(Cj,$d_i$) is maximum
        end for
        Update centroid for each cluster
    end loop
end K-Means
```

Time Complexity

- Assume computing distance(cosine similarity) between two instances is O(*m*) where *m* is the dimensionality of the vectors which corresponds to the average number of terms that are common between two documents.
- Reassigning clusters requires O(*kn*) distance computations, or O(*knm*).

- Computing centroids: Each instance vector gets added once to some centroid: O(*nm*).
- Assume these two steps are each done once for *I* iterations: O(*Iknm*).
- Hence, overall time complexity is linear in all relevant factors, assuming a fixed number of iterations.

Space Complexity

K-Means is low cost method in terms of space as it only needs to store the document vector for Top N documents returned by Vector Space Model and the centroids for K – clusters.

Evaluation of Clusters

In order to evaluate the quality of clusters quantitatively, we use two measures:

(1) **Intra Cluster Similarity**: For each cluster j we measure the intra cluster similarity using:

IntraSim(j) = $\frac{1}{|S|}\sum_{d \in S} \cos ine(d, Cj)$

In order to compute the average intra cluster similarity, we compute an average over all the clusters.

AvgIntraSim = $\frac{1}{K}\sum_{i=1}^{K} IntraSim(i)$

(2) **Inter Cluster Similarity**: We compute the pairwise similarity between each pair of cluster by computing the similarity between their centroids. Average inter-cluster similarity is measured by computing an average over all the distinct pairwise similarity between centroids.

AvgInterSim = $\frac{2}{K(K-1)}\sum_{i=1}^{K-1}\sum_{j=i+1}^{K} Sim(Ci, Cj)$

Observations

- The time taken for doing clustering increases as the number of clusters increase.
- We also observe that K-Means is extremely sensitive to initial seeds and outliers. We tried multiple runs for K-Means and each time the clusters we got were significantly different. Sometimes due to selection of a document which had no similar documents as a seed, caused that cluster to have only one or two documents.
- Considering it is low cost linear time method, the results of K-means are reasonable. For e.g. for the query "**information retrieval"**, we get the three clusters which intuitively represent: 1. courses pages on information retrieval, 2. web pages of researchers 3. library and other services which give information. For the query "**computer science",** the three clusters intuitively represent: 1. Women in computer science 2. CS department pages for undergraduate program 3. Pages on research in computer science department.
- One approach to assign a name to a cluster could be to look for common terms appearing in all the documents in that clusters and assign the cluster name corresponding to the term that occurs maximum number of times.

- As the number of clusters increased, the intra cluster similarity increases and the inter cluster similarity decreases(see Figure 3). The intra-cluster similarity always remained greater than inter-cluster similarity.
- The following figure 1 shows the effect of increase in N on the quality of the clusters obtained by K-Means for the query "Computer Science" for K=3. We see that as N increases the quality of cluster degrade as the intra cluster similarity decreases and the inter cluster similarity increases. We speculate this is due to large number of documents being present in cluster which are barely similar to each other. However, if we increase the value of K with the increase in the value of N, the cluster quality will not degrade.

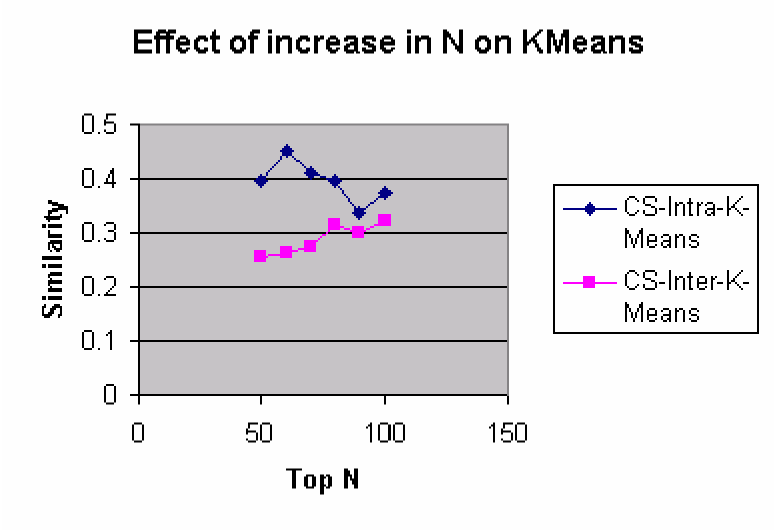


Figure 1: Effect of increase in N on K-Means

BuckShot

Buckshot algorithm combines K-Means and hierarchical agglomerative clustering(HAC). One of the problems with K-Means is that it is sensitive to selection of initial seeds. Hence we use HAC to generate the initial seeds for K-Means instead of picking random seeds or bootstrap k-means. But, HAC is costly as its running time complexity is $O(n^2)$, so we use only a sample of Top N documents to find the initial centroids. Typically, we chose $\sqrt{n}$ documents to do HAC and then chose initial k-centroids based on the clustering obtained using HAC. In order to compute similarity between two clusters we use **group-average**, which measures the average cosine distance between all pairs of neighbors in the two clusters.

Algorithm HAC

Input: List of Document Vectors D

Ouput: Hierarchy of clusters

```
Put every document in a cluster containing only itself
NumClusters = N
for i = 1 to N-1
    for ( j = 1 to NumClusters -1)
        for ( k = j+1 to NumClusters )
            compute similarity between cluster $C_j$ and $C_k$
        end for
```

```
                end for
                Pick two clusters that have highest similarity
                merge them into a single cluster
                NumClusters--;
                Update centroid for each cluster
        end for
end HAC
```

At each iteration, HAC stores the centroids of all the clusters at that level. Before the first iteration there are N centroids corresponding to the N documents. After the first iteration, there are N-1 centroids, and so on. Hence in order to get K centroid seeds for K-Means, HAC returns all the centroids corresponding to N-Kth iteration.

```
Algorithm BuckShot
Input: K:Number of clusters, D:Top N documents obtained by Vector Space Similarity
Output: K clusters
        Randomly select a sample S of √N documents from D
        Call HAC(S)
        Get K seeds from HAC
        Repeat until there is no change in cluster between two consecutive iterations
                for each document $d_i$ in D
                        for j = 1 to K
                                $Sim(C_j,d_i)$ = Find vector similarity between $d_i$ and $C_j$
                        end for
                        assign $d_i$ to cluster j for which $Sim(C_j,d_i)$ is maximum
                end for
                update centroid for each cluster
        end loop
end BuckShot
```

Time Complexity

The computational complexity of HAC is $O(N^2)$ where N is number of documents for doing clustering, however as we take only √N sample, hence the computation complexity of HAC is O(N). After selecting the centroids generated by HAC we do K-Means which is linear, hence the complexity of BuckShot algorithm is O(N).

Space Complexity

Buckshot is low cost method in terms of space as it only needs to store the document vector for Top N documents returned by Vector Space Model and the centroids for K – clusters. It also has to store an upper triangular matrix containing similarity between clusters which is of the size √N x √N.

Observations

- Buckshot tends to give clusters which have higher intra cluster similarity and lower inter cluster similarity than K-Means as is shown in Figure 2.
- It is not too sensitive to initial seeds like K-Means. Hence the cluster that we obtained by multiple runs of Buckshot are not too different. However, as the

sample used for getting the initial centroids using HAC is a random sample, sometimes the clusters obtained are dramatically different.

- The clusters formed are somewhat intuitive. For e.g. for the query: "**multimedia database"**, we get three clusters which are related to the following: 1. Course syllabus for database related courses 2. Prof. Candan's pages related to multimedia database research 3. Pages on multimedia writing. For the query "**parking decal"**, we could identify the following categories for clusters: 1. Pages related to Parking for east campus 2. Pages related to parking for main campus 3. Page related to parking for ASU Art museum. The last cluster in this particular run contained only one document because it possibly is not very similar to any other document in the top 50 results.
- One approach to assign a name to a cluster could be to look for common terms appearing in all the documents in that cluster and assign the cluster name corresponding to the term the occurs maximum number of times.
- The following figure 2 shows the effect of increasing the Top N documents on the query "Computer Science" for K=3. We see that intra-cluster similarity decreases as there are more dissimilar documents in a cluster for a fixed value of K.

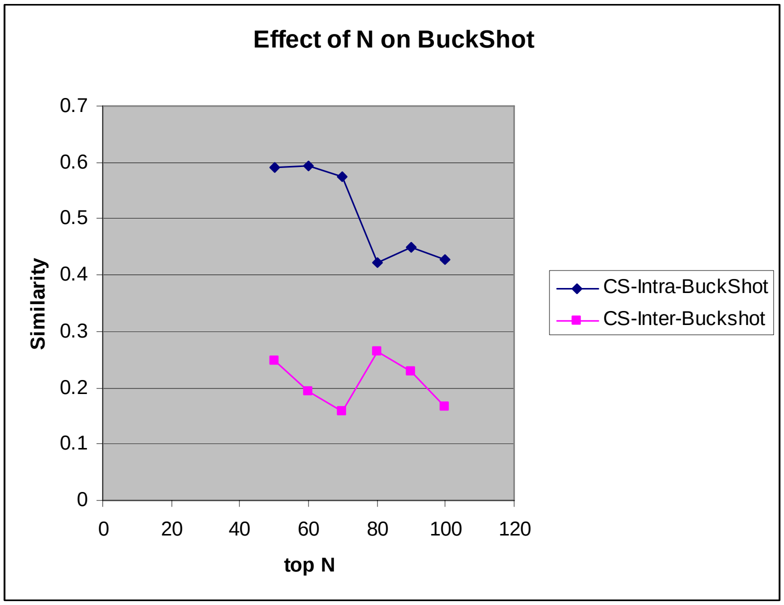


Figure 2: Effect of increase in N on BuckShot

Bisecting K-Means

This method is a type of divisive hierarchical clustering method using k-means. We start by putting all the documents in a single cluster. We partition the original cluster into two clusters by using K-Means i.e. K = 2. We make the cluster which has highest intra cluster similarity as permanent and recursively split the other cluster into two more clusters using K-means with K=2. We continue this until we get the desired number of clusters.

Algorithm Bisecting K-Means
Input: K: Number of clusters, D: Top N documents obtained by vector space similarity
Output: K clusters

```
    put all the N documents in a single cluster C
    for i=1 to K-1 do
        for j=1 to ITER do
            Use K-means to split C into two sub-clusters, $C_1$ and $C_2$
            if ( intra-cluster similarity($C_1$) > intra-cluster similarity($C_2$) )
                make cluster C1 as permanent
                $C = C_2$
            else
                make cluster $C_2$ as permanent
                $C = C_1$
            end if
        end for
    end for
end Bisecting K-Means
```

Time Complexity

Bisecting K-Means uses K-Means to compute two clusters with K=2. As K-Means is O(N), the run time complexity of the algorithm will be O((K-1)IN), where I is the number of iterations to converge. Hence Bisecting K-Means is also linear in the size of the documents.

Space Complexity

Bisecting K-Means is low cost method in terms of space as it only needs to store the document vector for Top N documents returned by Vector Space Model.

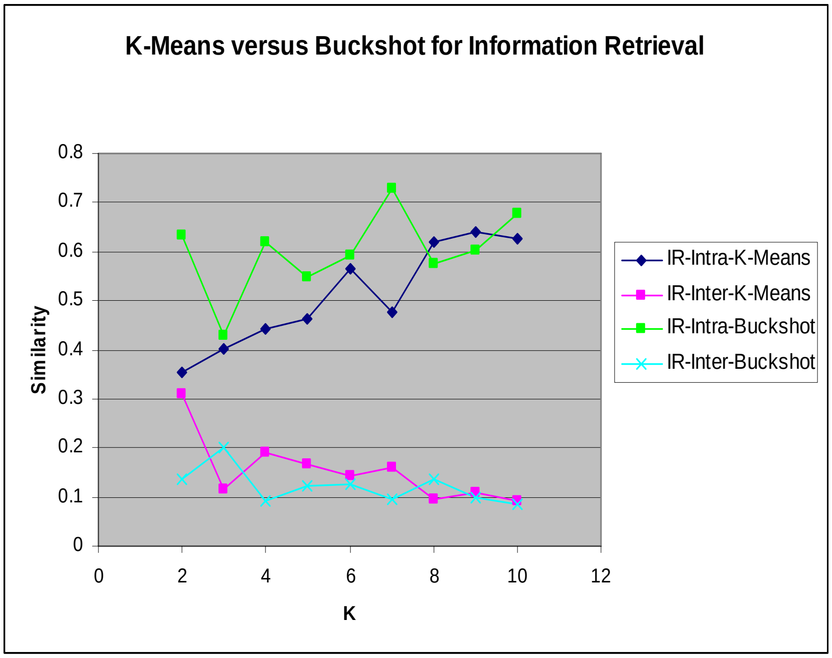
Observation

- Bisecting K-means is better than regular K-means in most cases. Even in cases where other schemes are better, bisecting K-means is only slightly worse.
- It is not sensitive to initial seeds as K-Means.
- When number of documents increases, the intra-cluster similarity decreases and the inter-cluster similarity increases as seen in K-Means and BuckShot.

Comparison of Approaches

K-Means vs Buckshot

The following figures show the comparison between K-Means and Buckshot method for different queries in terms of intra-cluster similarity and inter-cluster similarity.



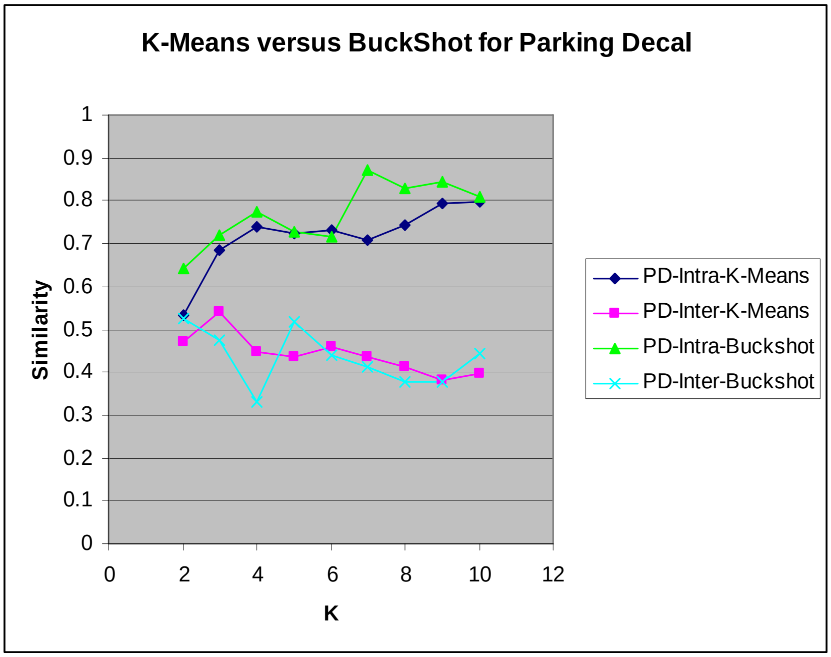


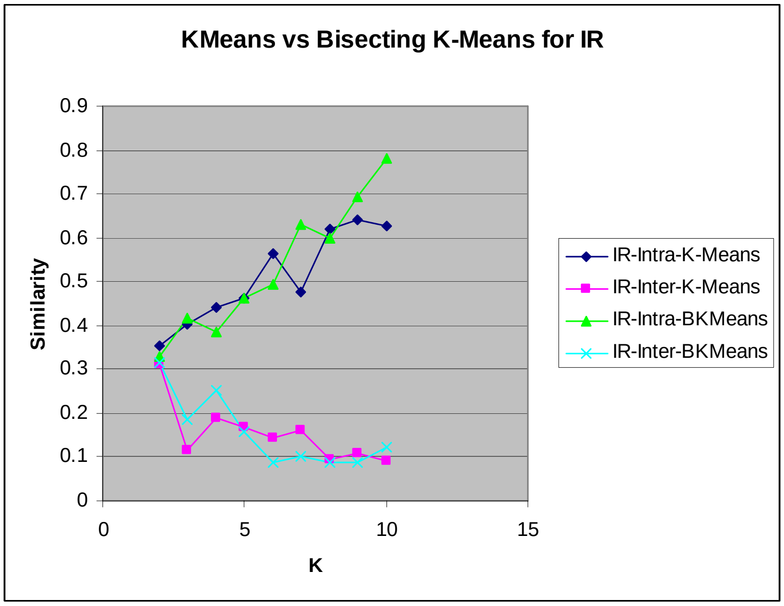
Figure 3: Comparison of K-Means and Buckshot

For the queries, “Information Retrieval” and “Parking Decal”, we see that Buckshot performs significantly better than K-Means as it has higher intra cluster similarity and lower inter cluster similarity for most of the values of K. This is due to the fact the Buckshot is not sensitive to initial seeds and outliers like K-means.

However, the time taken to compute K-Means is usually less than that of BuckShot algorithm.

K-Means vs Bisecting K-Means

The following figures show the comparison between K-Means and Bisecting K-Means method for different queries in terms of intra-cluster similarity and inter-cluster similarity.



For the query "Information Retrieval", both K-Means and Bisecting K-Means have more or less same intra-cluster similarity with bisecting K-Means slightly better for higher values of K. The inter-cluster similarity for Bisecting K-Means is slightly better than K-Means.

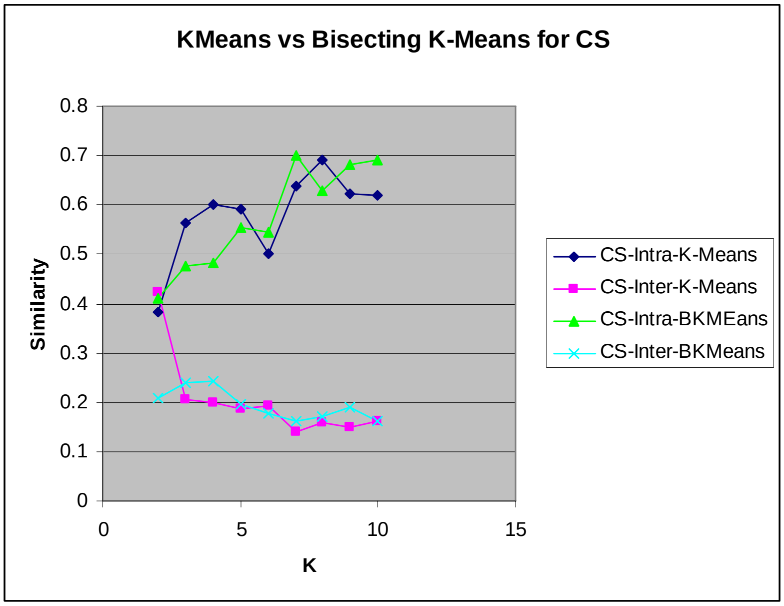


Figure 4: Comparison of K-Means and Bisecting K-Means

For the query "Computer Science", for lower values of K, K-Means seems to have higher intra cluster similarity than Bisecting K-Means, however for higher values of K, bisecting K-Means has higher intra-cluster similarity. The reason could be that higher values of K correspond to actual number of clusters for a given set of documents. Having less clusters will result in more dissimilar documents in one cluster. The inter-cluster similarity for both the methods is more or less the same.

The time taken by Bisecting K-Means is slightly more than that taken by simple K-Means.

BuckShot vs Bisecting K-Means

The following figure shows the comparison between BuckShot and Bisecting K-Means method for different queries in terms of intra-cluster similarity and inter-cluster similarity.

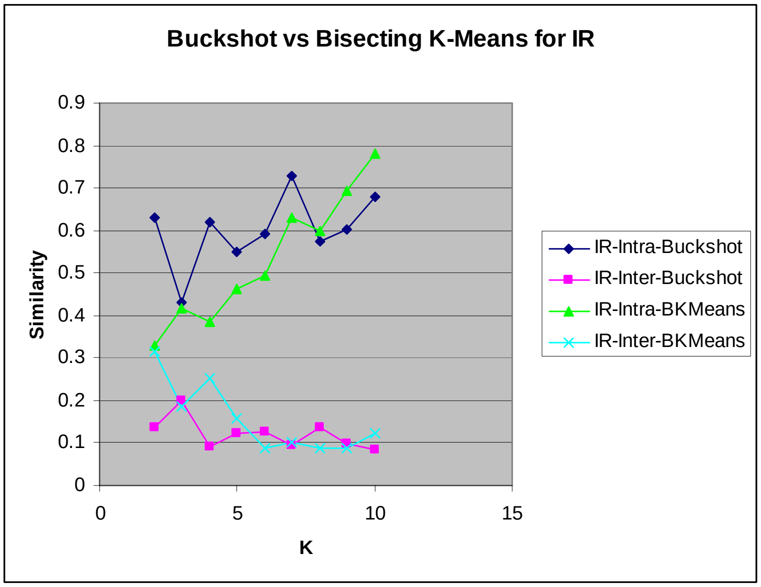


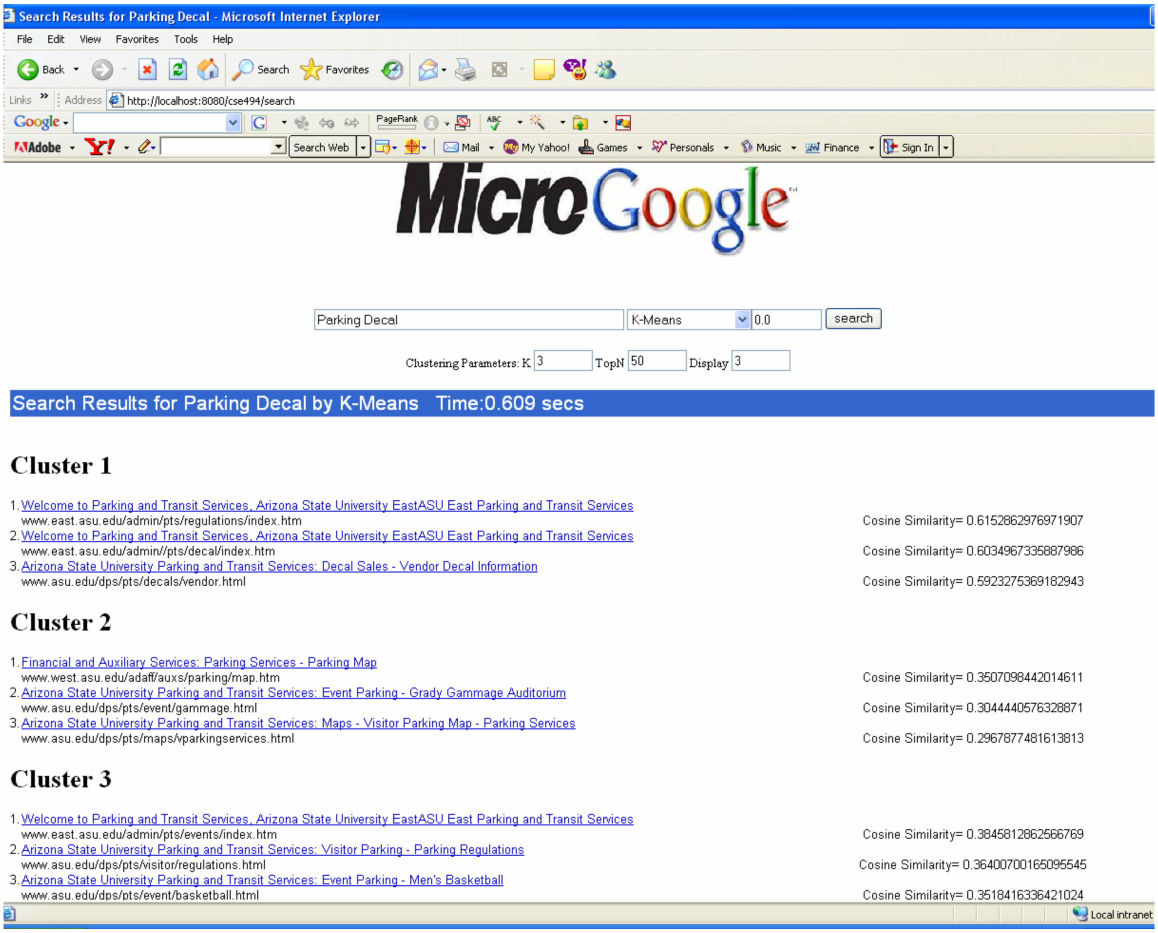
Figure 5: Comparison of Buckshot and Bisecting K-Means

For the query "Information Retrieval", Buckshot seems to have higher intra-cluster similarity than Bisecting K-Means for lower values of K from 1 to 7. However, for higher values of K, Bisecting K-Means seems to perform better than Buckshot.

OUTPUT OF GUI

The following shows the GUI implemented as a Java Servlet. It allows user to vary K-Number of clusters, TopN-number of documents to consider while clustering and how many documents to display for each cluster. It has following clustering methods:

1) K-Means
2) BuckShot
3) Bisecting K-Means



Search Results for Parking Decal - Microsoft Internet Explorer

Address http://localhost:8080/cse494/search

Parking Decal | K-Means | 0.0 | search

Clustering Parameters: K 3 TopN 50 Display 3

Search Results for Parking Decal by K-Means   Time:0.609 secs

Cluster 1

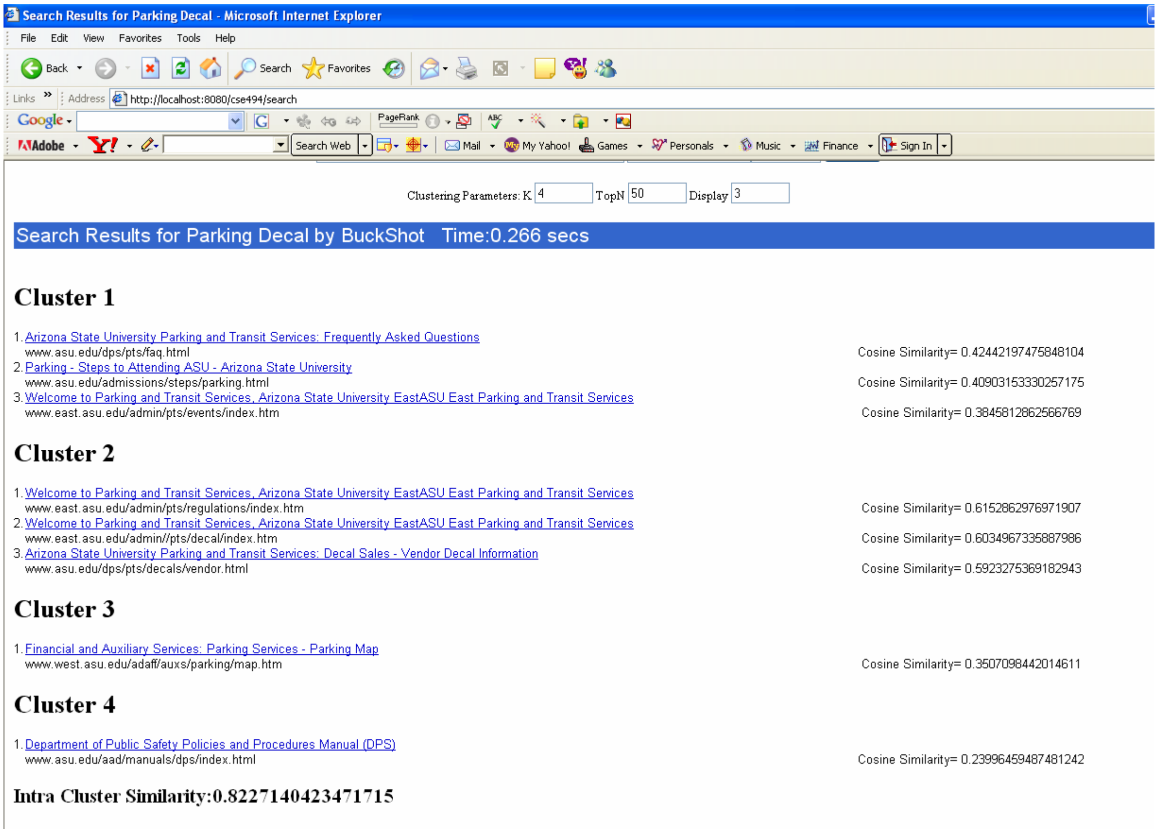
1. Welcome to Parking and Transit Services, Arizona State University EastASU East Parking and Transit Services
   www.east.asu.edu/admin/pts/regulations/index.htm   Cosine Similarity= 0.6152862976971907
2. Welcome to Parking and Transit Services, Arizona State University EastASU East Parking and Transit Services
   www.east.asu.edu/admin//pts/decal/index.htm   Cosine Similarity= 0.6034967335887986
3. Arizona State University Parking and Transit Services: Decal Sales - Vendor Decal Information
   www.asu.edu/dps/pts/decals/vendor.html   Cosine Similarity= 0.5923275369182943

Cluster 2

1. Financial and Auxiliary Services: Parking Services - Parking Map
   www.west.asu.edu/adaff/auxs/parking/map.htm   Cosine Similarity= 0.3507098442014611
2. Arizona State University Parking and Transit Services: Event Parking - Grady Gammage Auditorium
   www.asu.edu/dps/pts/event/gammage.html   Cosine Similarity= 0.3044440576328871
3. Arizona State University Parking and Transit Services: Maps - Visitor Parking Map - Parking Services
   www.asu.edu/dps/pts/maps/vparkingservices.html   Cosine Similarity= 0.2967877481613813

Cluster 3

1. Welcome to Parking and Transit Services, Arizona State University EastASU East Parking and Transit Services
   www.east.asu.edu/admin/pts/events/index.htm   Cosine Similarity= 0.3845812862566769
2. Arizona State University Parking and Transit Services: Visitor Parking - Parking Regulations
   www.asu.edu/dps/pts/visitor/regulations.html   Cosine Similarity= 0.36400700165095545
3. Arizona State University Parking and Transit Services: Event Parking - Men's Basketball
   www.asu.edu/dps/pts/event/basketball.html   Cosine Similarity= 0.3518416336421024



Search Results for Parking Decal - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Links
Address http://localhost:8080/cse494/search
Google
PageRank
Adobe
Search Web
Mail
My Yahoo!
Games
Personals
Music
Finance
Sign In

Clustering Parameters: K 4 TopN 50 Display 3

Search Results for Parking Decal by BuckShot Time:0.266 secs

Cluster 1

1. Arizona State University Parking and Transit Services: Frequently Asked Questions
www.asu.edu/dps/pts/faq.html Cosine Similarity= 0.42442197475848104
2. Parking - Steps to Attending ASU - Arizona State University
www.asu.edu/admissions/steps/parking.html Cosine Similarity= 0.40903153330257175
3. Welcome to Parking and Transit Services, Arizona State University EastASU East Parking and Transit Services
www.east.asu.edu/admin/pts/events/index.htm Cosine Similarity= 0.3845812862566769

Cluster 2

1. Welcome to Parking and Transit Services, Arizona State University EastASU East Parking and Transit Services
www.east.asu.edu/admin/pts/regulations/index.htm Cosine Similarity= 0.6152862976971907
2. Welcome to Parking and Transit Services, Arizona State University EastASU East Parking and Transit Services
www.east.asu.edu/admin//pts/decal/index.htm Cosine Similarity= 0.6034967335887986
3. Arizona State University Parking and Transit Services: Decal Sales - Vendor Decal Information
www.asu.edu/dps/pts/decals/vendor.html Cosine Similarity= 0.5923275369182943

Cluster 3

1. Financial and Auxiliary Services: Parking Services - Parking Map
www.west.asu.edu/adaff/auxs/parking/map.htm Cosine Similarity= 0.3507098442014611

Cluster 4

1. Department of Public Safety Policies and Procedures Manual (DPS)
www.asu.edu/aad/manuals/dps/index.html Cosine Similarity= 0.23996459487481242

Intra Cluster Similarity:0.8227140423471715