Information Retrivial and Clustering (pp. – - –)
W. Wu, H. Xiong and S. Shekhar(Eds.)


# Finding Topics in Collections of Documents: A Shared Nearest Neighbor Approach [1]

Levent Ertöz
*Department of Computer Science and Engineering*
*University of Minnesota, Minneapolis, MN 55455*
E-mail: ertoz@cs.umn.edu

Michael Steinbach
*Department of Computer Science and Engineering*
*University of Minnesota, Minneapolis, MN 55455*
E-mail: steinbac@cs.umn.edu

Vipin Kumar
*Department of Computer Science and Engineering*
*University of Minnesota, Minneapolis, MN 55455*
E-mail: kumar@cs.umn.edu

## Contents



[1]This work was partially supported by NSF grant ACI-9982274, by LLNL/DOE grant #B347714, and by Army High Performance Computing Research Center contract number DAAH04-95-C-0008. The content of this work does not necessarily reflect the position or policy of the government and no official endorsement should be inferred. Access to computing facilities was provided by AHPCRC and the Minnesota Supercomputing Institute.

# 1 Introduction

Given a set of documents, clustering is often used to group the documents, in the hope that each group will represent documents with a common theme or topic. Initially, hierarchical clustering was used to cluster documents [5]. This approach has the advantage of producing a set of nested document clusters, which can be interpreted as a topic hierarchy or tree, from general to more specific topics. In practice, while the clusters at different levels of the hierarchy sometimes represent documents with consistent topics, it is common for many clusters to be a mixture of topics, even at lower, more refined levels of the hierarchy. More recently, as document collections have grown larger, K-means clustering has emerged as a more efficient approach for producing clusters of documents [4, 9, 16]. K-means clustering produces a set of un-nested clusters, and the top (most frequent or highest "weight") terms of the cluster are used to characterize the topic of the cluster. Once again, it is not unusual for some clusters to be mixtures of topics.

Our goal was to find an algorithm that would more consistently produce clusters of documents with strong, coherent themes, even if it were necessary to omit many of the documents in the process. After all, in an arbitrary collection of documents, e.g., a set of newspaper articles, there is no reason to expect that all documents belong to a group with a strong topic or theme. While this approach does not provide a complete organization of all

documents, it does identify the "nuggets" of information in a document collection and can profitably be applied to practical problems such as grouping the search results of a Web search engine.

To accomplish our goal of consistently finding coherent clusters, we developed a model of documents that allows us to clearly define what we mean by a coherent cluster. This model is based on the idea that a group of documents with a strong theme or topic is characterized by its use of words from a small number of specialized vocabularies. For example, documents about the Olympics would, in addition to words from a general vocabulary, tend to have many sports terms and international terms, i.e., country names. This model can be formalized as a generative probabilistic model, where each cluster of documents that represents a coherent concept is generated by selecting words from a few specialized vocabularies and one general vocabulary.

Our model indicates, quite realistically, that any individual document may be more similar to a document in another coherent cluster than to any document in its own cluster [16]. For example, a particular document about the Olympics may be more similar to another sports document, than to any other document about the Olympics. However, clustering methods, such as K-means and hierarchical clustering, tend to assume that an object belongs to a particular cluster only if it is closer to at least some object in that cluster then to some object in other clusters. To overcome this problem, we use a clustering approach based on how many nearest neighbors a document shares [8]. For documents, this somewhat indirect measure of similarity turns out to be more accurate than a direct similarity measure based, say, on the cosine measure. This approach also deals with another problem of document clusters, i.e., that there is a lot of variation in the "tightness" of different clusters. In such cases, many clustering techniques, e.g., K-means, will either combine two tight clusters into one cluster or split a loose cluster.

The basic outline of this paper is as follows. Section 2 provides a brief background in documents and clustering, and discusses some related work. Section 3 introduces our shared nearest neighbor clustering algorithm, while section 4 describes our document model. Section 5 presents some experimental results comparing our shared nearest neighbor clustering approach to K-means. Section 6 is a brief conclusion and an indication of areas for future work.

## 2 Document and Clustering Preliminaries

Documents are represented using the vector-space model [15, 11]. In particular, we remove stop words, perform stemming using Porter's suffix-stripping algorithm, and then weight each term based on its *inverse document frequency* (IDF). Finally, each document vector is normalized to unit length. To compute the similarity between documents, we used the cosine measure [11].

The two most common techniques used for clustering documents are hierarchical and partitional (K-means) clustering techniques [3, 12]. Hierarchical techniques produce a nested sequence of partitions, with a single, all-inclusive cluster at the top and singleton clusters of individual points at the bottom. Each intermediate level can be viewed as combining two clusters from the next lower level (or splitting a cluster from the next higher level). In contrast to hierarchical techniques, partitional clustering techniques create a one-level (un-nested) partitioning of the data points (documents). There are a number of partitional techniques, but we shall only consider the K-means algorithm, which is widely used in document clustering. K-means is based on the idea that a center point can represent a cluster. For K-means, we use the notion of a centroid, which is the mean of a group of points.

### 2.1 Related Work - General

Here we restrict our focus to the use of clustering for topic or theme related document tasks. Clustering has been proposed for use in browsing a collection of documents [2] or in organizing the results returned by a search engine in response to a user's query [18]. Document clustering has also been used to automatically generate hierarchical clusters of documents [13]. (The automatic generation of a taxonomy of Web documents like that provided by Yahoo! (www.yahoo.com) is often cited as a goal.) A somewhat different approach [1] finds the natural clusters in an already existing document taxonomy (Yahoo!), and then uses these clusters to produce an effective document classifier for new documents. Recent work to generate document hierarchies [14] uses some of the clustering techniques from [2]. Much recent work in document clustering has focused on K-means clustering [4, 9, 16]. Indeed, it has been suggested that variants of the K-means algorithm can produce both nested and un-nested sets of clusters that are as good or better than those produced by traditional hierarchal clustering techniques [16]. For that reason we only compare our algorithm to K-means. For recent developments comparing K-means and hierarchical clustering, we refer the

reader to [19].

Finally, note that our work is not closely related to Topic Detection and Tracking (TDT), which has a more temporal flavor. Also, our definition of a topic is not the same as that employed in TDT work. Instead, our work derives from work in document clustering, which has the goal of improved browsing or organization of documents.

### 2.2 Related Work – Specific

Our clustering algorithm is based on a shared nearest neighbor clustering algorithm described in [8]. A similar approach, but for hierarchical clustering, was developed in [6]. Recently, a couple of other clustering algorithms have used shared nearest neighbor ideas [7, 10]. The work in [7] also gives an example of a situation where a measure of similarity, which is based on the number of neighbors two points share, was used to overcome the problem that two data points might belong to separate classes, but still be most similar to each other.

We explain the approach of [8], which we call Jarvis-Patrick clustering, in more detail in preparation for Section 3. First note that while cluster analysis sometimes uses the original data matrix, many clustering algorithms use a similarity matrix, which is an $m$ by $m$ matrix ($m$ = number of objects) containing all the pairwise similarities between the objects being considered. If $x_i$ and $x_j$ are the $i^{th}$ and $j^{th}$ objects,respectively, then the entry at the $i^{th}$ row and $j^{th}$ column of the similarity matrix is the similarity, $s_{ij}$ between $x_i$ and $x_j$. A similarity matrix defines a weighted graph, where the nodes are the points being clustered, and the weighted edges represent the similarities between points, i.e., the entries of the similarity matrix.

Thus, from a graph point of view, clustering is equivalent to breaking the graph into connected components, one for each cluster. We will describe the shared nearest neighbor algorithm in [8] in these terms. First the $n$ nearest neighbors of all points are found. In graph terms this can be regarded as breaking all but the $n$ strongest links between a point and all other points in the proximity graph. This forms what we call a "nearest neighbor graph." Note that the nearest neighbor graph is just a sparsified version of the original similarity graph, which is derived by breaking the links to less similar points.

We then determine the number of nearest neighbors shared by any two points. In graph terminology, we form what we call the "shared nearest neighbor" graph. We do this by replacing the weight of each link between two points (in the nearest neighbor graph) by the number of neighbors that

the points share. In other words, this is the number of length 2 paths between any two points in the nearest neighbor graph [7].

After, this shared nearest neighbor graph is created, all pairs of points are compared and if any two points share more than $T$ neighbors, i.e., have a link in the shared nearest neighbor graph with a weight more than our threshold value, $T(T \leq n)$, then the two points and any cluster they are part of are merged. In other words, clusters are connected components in our shared nearest neighbor graph after we sparsify using a threshold.

This approach has a number of nice properties. It can handle clusters of different densities since the shared nearest neighbor approach is self-scaling. Also, this approach is transitive, i.e., if point, $p$, shares lots of nearest neighbors with point, $q$, which in turn shares lots of nearest neighbors with point, $r$, then points $p$, $q$ and $r$ all belong to the same cluster. The transitive property, in turn, allows this technique to handle clusters of different sizes and shapes. We have extended the Jarvis-Patrick approach as described in the next section.

# 3 Our Clustering Algorithm

## 3.1 Is cosine similarity a good measure by itself?

The cosine measure makes perfect sense for the K-means algorithm. K-means tries to maximize the average pairwise similarity between documents within clusters. The overall pairwise similarity of a cluster is equal to the square of the norm of the centroid vector of the cluster if the cosine measure is used for similarity [4]. Each document is assigned to the cluster whose centroid is most similar to the document, which means that the average similarity between pairs of documents in the cluster is maximized.

In the case of hierarchical clustering, cosine similarity turns out not to be very suitable. For example, for the LA1 document set [17], a document's closest neighbor actually belongs to a different class 20% of the time. In such a scenario, hierarchical methods make many mistakes initially, and these mistakes can never be corrected, at least with standard hierarchical techniques. Thus, while hierarchical clustering techniques are often thought to be the highest quality clustering approach, in general, variants of the K-means algorithms have been found to work as well or better for document clustering [16].

### 3.2 Our Clustering Algorithm

We begin by calculating the document similarity matrix, i.e., the matrix which gives the cosine similarity for each pair of documents. Once this similarity matrix is calculated, we find the first $n$ nearest neighbors for each document. (Every document is considered to be its own $0^{th}$ neighbor.) In the nearest neighbor graph, there is a link from document $i$ to document $j$, if $i$ and $j$ both have each other in their nearest neighbor list. In the shared nearest neighbor graph, there is a link from $i$ to $j$ if there is a link from $i$ to $j$ in the near neighbor graph. The strength of this link is equal to the number of shared near neighbors of $i$ and $j$.

At this point, we could just apply a threshold, and take all the connected components of the shared nearest neighbor graph as our final clusters [8]. However, this threshold would need to be set too high, since this is a single link approach, and would give poor results when patterns in the data set are not very significant. When a high threshold is applied, a natural cluster might be split into many small clusters due to variations in tightness in the similarity within the cluster. We address these problems with the clustering algorithm described below.

There are two types of parameters used in this algorithm: one type relates to the strength of the links in the shared near neighbor graph, the other type relates to the number of strong links for a document. If the strength of a link is greater than a threshold, that link is labelled as a strong link.

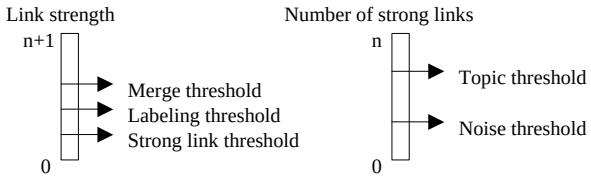


Figure 1: Types of Thresholds for SNN Clustering

The details of our shared nearest neighbor clustering algorithm are as follows:

1. For every point $i$ in the dataset, calculate the connectivity, conn[$i$], the number of strong links the point has.

2. For a point $i$ in the dataset, if conn[$i$] $<$ *noise threshold*, then that point is not considered in the clustering since it is similar to only a

few of its neighbors. Similarly, if conn[$i$] > *topic threshold*, then that point is similar to most of its neighbors and is chosen to represent its neighborhood.

3. For any pair of points ($i$, $j$) in the dataset, if $i$ and $j$ share significant numbers of their neighbors, i.e., the strength of the link between $i$ and $j$ is greater than the *merge threshold*, then they will appear together in the final clustering if either one of them (or both) is chosen to be a representative. Note that the algorithm will not suffer from the effects of transitivity since every other point on a chain of links has to be chosen to be a representative. In other words, two documents that are not directly related will be put in the same cluster only if there are many other documents between them that are connected with strong links, half of which must represent their own neighborhood.

4. Labeling step: Having defined the representative points and the points strongly related to them, we can bring back some of the points that did not survive the *merge threshold*. We do this by scanning the shared near neighbor list of all the points that are part of a cluster, and checking whether those points (a) have links to points that don't belong to any cluster and (b) have a link strength greater than the labeling threshold.

The method described above finds communities of documents, where a document in a community shares a certain fraction of its neighbors with at least some number of neighbors. While the probability of a document belonging to a class different from its nearest neighbor's class may be relatively high, this probability decreases as the two documents share more and more neighbors. This is the main idea behind the algorithm.

### 3.3 Impact of the Size of the Nearest Neighbor List

The size of the near neighbor list, $n$, is a measure of how focused the clusters will be. It is equal to the smallest number of documents within which we can find a coherent set of documents. If $n$ is equal to 1, then we will end up finding pairs of documents, which are each other's closest neighbors. If $n$ is equal to, say, 50, we could expect to find coherent sets of documents of size 1 to several hundreds. Note that transitivity comes into picture since the requirement for a document to belong to a community is that it must share several neighbors with many other documents in the community, although not necessarily with all of them. Singleton clusters should not be treated as

noise, since they were chosen to represent their neighborhood, i.e., there are themes associated with singleton clusters.

Changing the value of $n$ will change the clusters that are found. When $n$ is increased, two points that did not share many neighbors, might now share relatively more neighbors, due to the increased size of the neighbor list. On the other hand, two points that shared a lot of neighbors may now share relatively fewer neighbors.

The nearest neighbor list size should depend on the data set size. For example, if $n$ is fixed and there are infinitely many data points, then the resulting clusters will consist of points that are identical to each other.

### 3.4 Calculating Link Strengths – a Weighted Approach

Not all the shared nearest neighbors of two points are equally good. If two points share neighbors that are high in their nearest neighbor lists, then these neighbors should make a higher contribution to the similarity between the two points as compared to the case where two points share neighbors on the bottom of their neighbor lists [8]. One disadvantage of the non-weighted scheme is that, when the nearest neighbor list size is increased, all the points start looking the same. In the extreme case, when $n$ is equal to the number of documents less 1, every point has exactly $n$ shared nearest neighbors with every other point. The weighted scheme takes care of this problem to a certain extent and is also more intuitive.

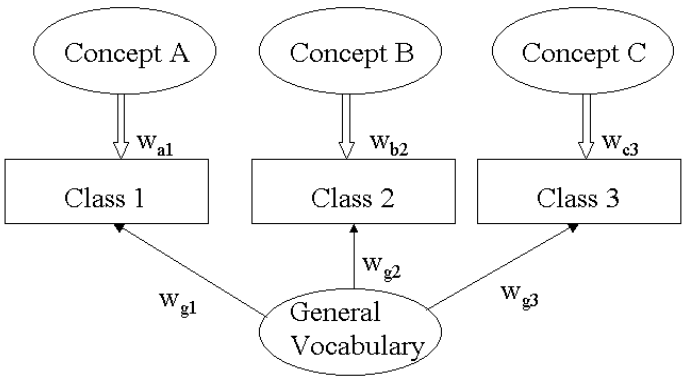


Figure 2: Abstract Data Model

# 4 Our Abstract Data Model

To enhance our understanding and to better test our algorithm (via synthetic data sets), we developed an abstract data model. In the abstract data model, there are concepts from which the documents "pick" their words. Each concept has a size (the number of words in the concept) and a frequency distribution for the words in the concept. A class has a size (the number of documents belonging to the class), and a length distribution for the documents that belong to the class. Furthermore, a class has links to the concepts from which its documents pick up certain fraction of its words (which is specified by the strength of the link). A word in a concept does not appear in another concept. Overlapping concepts can be modelled by, creating another concept from the intersection of the two concepts, and adjusting the weights accordingly. A simple model that consists of 3 classes and 4 concepts (general vocabulary can be treated as another concept) is shown in Figure 2.

There are several ways to change how "tight" the concepts will be and how "close" they will be to each other. One way is to change the weights of the links from the classes to the concepts. Another way is to change the sizes of the concepts and the general vocabulary.

Consider some scenarios. If the weights of the links from the classes to the general vocabulary are low, then the classes are well separated, since they won't share many words, given that the words in the concepts are distinct. However, even if the weights of the links to the general vocabulary are low, e.g., if the sizes of the concepts are very large compared to the general vocabulary (unlikely scenario), then the classes may not be well separated.

# 5 Experimental Results

## 5.1 A Synthetic Data Set Example

In this example there are 9 classes (0-8) and 10 concepts ($a-j$). All concepts except $e$ and $j$ have 30 words, and they appear with a frequency of at most 10 in any document. There are 200 words in concepts $e$ and $j$, and they appear in a document with a frequency of at most 20. We can think of the concepts $a-e$ as sports vocabularies, concept $e$ being a general sports vocabulary while concepts $a-d$ contain words specific to different sports. Class 0 picks 30% of its words from concept $a$, 50% of its words from concept $e$, and its remaining words from the union of all concepts. Similarly, class 1 picks its

words from $b$ and $e$, class 2 picks its words from $c$ and $e$, and so forth. The same structure exists between classes 4-7 and the concepts $f - j$. Class 8 picks all of its words from the union of the concepts, and is considered to be 'noise.' All the documents contain $20 - 100$ distinct words. Classes 0, 1, 2, and 3 have 100, 200, 300 and 400 documents, respectively. Similarly, classes 4, 5, 6, and 7 have 100, 200, 300 and 400 documents respectively. Class 8 has 1000 documents.

Table 1 below shows how K-means clusters our synthetic data. The first two columns show the cumulative size of the clusters versus the cumulative misclassification, while the next 9 columns are the confusion matrix. The clusters are sorted according to the norm of their centroids, which is given in the last column. (Recall again, that the norm of a cluster centroid represents the overall pairwise document similarity of the cluster.) In this example, 120 clusters were used, but only first 15 are shown.

Table 1: K-means Clustering of Synthetic Data.

| **cumulative** | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **size** | **mis.** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **norm** |
| 4 | 2 | - | - | - | 2 | - | - | - | - | 2 | 0.591 |
| 12 | 5 | - | 5 | - | 1 | - | - | - | - | 2 | 0.488 |
| 20 | 9 | - | - | - | - | - | - | - | 3 | 1 | 0.477 |
| 28 | 9 | - | - | - | - | - | - | - | - | 8 | 0.471 |
| 36 | 9 | - | - | - | - | - | - | - | - | 8 | 0.466 |
| 60 | 12 | - | 21 | 1 | - | - | - | - | - | 2 | 0.465 |
| 91 | 14 | - | - | - | - | - | 29 | - | - | 2 | 0.465 |
| 110 | 19 | - | - | - | - | 2 | 14 | 1 | - | 2 | 0.463 |
| 161 | 19 | - | - | - | - | - | 51 | - | - | - | 0.463 |
| 179 | 23 | - | 14 | - | 2 | - | - | - | - | 2 | 0.461 |
| 226 | 24 | - | - | - | 46 | - | - | - | - | 1 | 0.459 |
| 242 | 27 | - | - | - | 13 | - | - | - | - | 3 | 0.459 |
| 270 | 29 | - | - | - | 26 | - | - | - | - | 2 | 0.459 |
| 295 | 32 | 1 | - | 1 | 22 | - | - | - | - | 1 | 0.458 |
| 304 | 32 | - | - | - | - | - | - | - | - | 9 | 0.458 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **3000** | **480** | - | - | - | - | - | - | - | - | 15 | 0.365 |

Tables 2 and 3 show the clustering results at two different resolutions, using the method described in this paper. At a low resolution, Table 2,

Table 2: SNN Clustering of Synthetic Data - Low Resolution.

| **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **norm** |
|---|---|---|---|---|---|---|---|---|---|
| 83 | 183 | 261 | 351 | - | - | - | - | 1 | 0.354 |
| - | - | - | - | 89 | 168 | 266 | 339 | 1 | 0.354 |

Table 3: SNN Clustering of Synthetic Data - High Resolution.

| **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | |
|---|---|---|---|---|---|---|---|---|---|
| 78 | - | - | - | - | - | - | - | - | 0.459 |
| - | 169 | - | - | - | - | - | - | - | 0.438 |
| 1 | 1 | 228 | - | - | - | - | - | - | 0.424 |
| - | 1 | - | 280 | - | - | - | - | - | 0.423 |
| - | - | - | 4 | - | - | - | - | - | 0.674 |
| - | - | - | - | 81 | - | - | - | - | 0.449 |
| - | - | - | - | - | 160 | - | - | - | 0.443 |
| - | - | - | - | - | - | 223 | - | - | 0.426 |
| - | - | - | - | - | - | - | 4 | - | 0.672 |
| - | - | - | - | - | - | 1 | 279 | - | 0.420 |

our technique captures the way that the classes of documents are generated, putting documents from classes 0 – 3 in one cluster and documents from classes 4 – 7 in another. At a higher resolution, shown in Table 3, only the documents generated from single concepts are put together. Note that there are only 2 misclassified documents in the low-resolution case, and only 4 misclassified documents for the high-resolution results. Also, note that documents from the noise cluster are almost completely ignored. We also observed that K-means clusters tend to contain more of the 'general sports vocabulary' ($e$) in their most important word list, whereas SNN clusters contain the terms specific to their sport ($a$, $b$, $c$, $d$).

If we know the right number of clusters before running K-means, the results look better, but still not as good as for the SNN clusters. Besides, the right number of clusters is often difficult to determine for K-means.

### 5.2 Comparison with K-means on a Real Data Set

Data set LA1 is from the Los Angeles Times data of TREC-5 [17]. The words in the tables are the most important 6 words in each document. We see that

all the documents in the first cluster are related to NCAA, while all the documents in the second cluster are related to NBA. Even though both sets of documents are basketball related, our clustering algorithm found them as separate clusters. We ran the K-means algorithm on the same data set, and interestingly, all of the documents in these two clusters appeared in the same K-means cluster together with some unrelated documents, including a number of documents related to gymnastics and swimming. The reason that K-means put all these sports documents in the same cluster is that sports documents tend to share a lot of common words, such as *score*, *half*, *quarter*, *game*, *ball*, etc. This example shows that pair-wise similarity, by itself, isn't a good measure for clustering documents.

Table 4: The NCAA Cluster.

| | | | | | |
|---|---|---|---|---|---|
| wolfpack | towson | lead | tech | Scor | North |
| syracus | scor | georgia | dome | auburn | Louisvill |
| Scor | lead | throw | half | Free | Iowa |
| Scor | Fresno | unlv | lead | lockhart | jacksonvil |
| Panther | pittsburgh | sooner | brookin | Scor | Game |
| Iowa | minnesota | scor | illinoi | wisconsin | Burton |
| Scor | half | virginia | georgetown | lead | Kansa |
| Burson | louisvill | scor | ohio | game | Ellison |

Table 5: The NBA Cluster.

| | | | | | |
|---|---|---|---|---|---|
| Pacer | scor | piston | shot | game | hawkin |
| Cavali | mckei | charlott | scor | superson | cleveland |
| Scor | game | tripucka | basket | hornet | straight |
| levingston | hawk | jordan | malon | buck | quarter |
| daugherti | piston | warrior | cavali | shot | Eject |

The Tables 6 and 7 show the confusion matrices for the K-means and SNN techniques, respectively, on the LA1 data set, which has the class labels: *Financial*, *Foreign*, *National*, *Metro*, *Sports*, *Entertainment*.

We see, using the SNN approach, that we can get purer clusters, but that not all the documents are assigned to clusters. In order to make a fair comparison, we decided to remove, from K-means clusters, all documents that were relatively dissimilar to the centroid of their cluster. The misclassifica-

Table 6: Performance of K-means on LA1 Data Set.

| cumulative | | | | | | | |
|---|---|---|---|---|---|---|---|
| size | mis. | **Fin.** | **For.** | **Nat.** | **Met.** | **Sports** | **Ent.** |
| 11 | 3 | 1 | 8 | 1 | 1 | - | - |
| 94 | 3 | 83 | - | - | - | - | - |
| 149 | 3 | - | - | - | - | 55 | - |
| 233 | 3 | - | - | - | - | 84 | - |
| 283 | 11 | 2 | 42 | 2 | 4 | - | - |
| 325 | 18 | - | - | - | 5 | 35 | 2 |
| 363 | 23 | 1 | 1 | - | 1 | 33 | 2 |
| 389 | 40 | 9 | 7 | 8 | 2 | - | - |
| 475 | 68 | 3 | 58 | 6 | 8 | 7 | 4 |
| 517 | 84 | 2 | 1 | 1 | 3 | 26 | 9 |
| 626 | 95 | - | 1 | 4 | 3 | 98 | 3 |
| 755 | 96 | - | - | - | - | 128 | 1 |
| 828 | 135 | 11 | 34 | 16 | 11 | - | 1 |
| ... | ... | ... | ... | ... | ... | ... | ... |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 3204 | 756 | 3 | 2 | 9 | 10 | 1 | 5 |

Table 7: Performance of SNN on the LA1 Data Set.

| cumulative | | | | | | | |
|---|---|---|---|---|---|---|---|
| size | mis. | **Fin.** | **For.** | **Nat.** | **Met.** | **Sports.** | **Ent.** |
| 45 | 0 | 45 | - | - | - | - | - |
| 66 | 2 | 19 | - | 2 | - | - | - |
| 74 | 3 | 7 | - | 1 | - | - | - |
| 80 | 3 | 6 | - | - | - | - | - |
| 85 | 3 | 5 | - | - | - | - | - |
| 90 | 3 | 5 | - | - | - | - | - |
| 95 | 5 | 3 | 1 | - | 1 | - | - |
| 119 | 6 | - | 23 | - | 1 | - | - |
| 137 | 6 | - | 18 | - | - | - | - |
| 153 | 6 | - | 16 | - | - | - | - |
| 179 | 13 | 2 | - | 19 | 1 | - | 4 |
| 190 | 17 | - | 1 | 3 | 7 | - | - |
| 197 | 18 | - | - | 1 | 6 | - | - |
| 238 | 21 | - | - | 1 | 2 | 38 | - |
| 267 | 21 | - | - | - | - | 29 | - |
| 288 | 21 | - | - | - | - | 21 | - |
| 309 | 21 | - | - | - | - | 21 | - |
| 326 | 21 | - | - | - | - | 17 | - |
| 335 | 21 | - | - | - | - | 9 | - |
| 400 | 21 | - | - | - | - | - | 65 |

tion rate improved only slightly - it dropped from 23.6% to approximately 20% after throwing away half of the documents.

When we looked at the individual documents in a "supposedly not so good" SNN cluster, we found that the documents do form a coherent group even though they have different class labels. Table 8 below shows an SNN cluster of size 23, where 6 of the documents have a different label than 'metro'. If we look at the top 10 words in each of the documents, we can see that all the documents are about fire hazards. If we look at the second document in the list, we can see that there was a fire in a school in Chile. Since it was in Chile, the article appeared in the *Foreign* section of the newspaper. When we performed the same investigation on the K-means clusters, we found that there are actually several threads of stories in the 'not so good' K-means clusters. This artifact can be attributed to the self-similarity effect as discussed in the following.

Table 8: "Bad" SNN Cluster.

| **Bad Cluster Sample** | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | |
| **Document Label** | | **Top 10 words in the documents** | | | | | | | | | |
| Metro | | tree | christma | malfunct | koenig | fire | sullivan | blaze | firefight | glen | spark |
| Foreign | | chile | fire | eighti | eleven | school | confusion | fled | santiago | upi | eve |
| National | | remer | ren | fire | children | victim | couche | adult | stove | woke | anchorag |
| Metro | | fire | duct | jail | smoke | middleton | rubbish | burn | rise | waft | incid |
| Metro | | fire | alameda | inhal | plate | blaze | coron | oakland | smoke | occup | ident |
| National | | fire | smolder | charlott | cigarett | upi | swept | famili | carolina | injur | di |
| Metro | | palmer | fire | build | melodi | toni | fullerton | in | restaur | furnitur | reopen |
| National | | fire | rome | children | frame | blaze | kill | georgia | upi | swept | burn |
| Metro | | bucher | fire | collag | prado | mccarti | apparatu | casa | watercolor | celebr | depart |
| Metro | | sister | verduzco | fire | ornela | blaze | house | neighbor | angelica | bedroom | di |
| Metro | | apart | fire | granada | serrania | complex | casa | anaheim | karrie | blaze | awaken |
| Metro | | lonczak | lytle | fire | forest | acre | covina | creek | brush | cigarett | blaze |
| Metro | | fire | canyon | madlock | trabuco | wind | emori | firefight | joplin | keener | neal |
| Metro | | alarcon | lojacono | apart | fire | inhal | cate | damag | smoke | joseph | orang |
| Metro | | penelop | griffith | spaniel | blaze | springer | dog | peggi | fire | pet | burn |
| National | | outward | yonker | room | mclaughlin | door | fire | uncertain | blaze | nurs | badli |
| National | | apart | fire | richardson | blaze | kill | upi | tex | heavili | cause | damag |
| Metro | | tang | bird | fire | parrot | feenei | griebe | chilli | heater | anaheim | blaze |
| Metro | | weisenberg | griebe | fire | carport | divorc | car | anaheim | estrang | blood | wife |
| Metro | | fire | arson | duma | grove | garden | gasolin | damag | travel | destroi | busi |
| Metro | | church | williamson | firefight | arson | fire | blaze | incendiari | douse | etern | baptist |
| Metro | | apart | arson | fire | anaheim | complex | blaze | caus | carport | probabl | cross |
| Metro | | womble | fire | neighbor | house | demmer | spence | pendleton | clif | blaze | halloween |

Suppose that there is a cluster that contains two totally unrelated documents. If we calculate the similarity of each document to the centroid, we will get a value of 0.5 since the documents themselves constitute half of the centroid, i.e., all the similarity of each document to the centroid is similarity with that part of the centroid that represents the document itself, or self-similarity. If, however, the cluster contains two coherent threads of stories, instead of two single documents, then the situation will still be very similar to the two-document case. Due to the self-similarity effect, K-means

is not able to distinguish between a loose cluster and a cluster containing documents from several tight subclusters, representing several threads of stories. This is a problem because the average pairwise similarity, i.e., centroid norm, of coherent clusters varies a lot, typically from 0.25 to 0.60. This situation is exactly what we observed on the K-means clusters from the LA1 data set.

### 5.3 Word Clusters

Using the same dataset (LA1), we clustered the words instead of the documents. When we cluster the documents and look at the top terms in the centroid of the clusters, we get an idea of what the topic of the cluster is. When we cluster the words, which we do by transposing the document-term matrix and using the exact same algorithm as for clustering documents, we obtain coherent sets of words that form concepts. By contrast, the most important terms in document clusters (topics) may contain several concepts. The concepts found are specific to the dataset used. Here are some concepts that are found by the algorithm in LA1.

```
afghanistan embassi guerrilla kabul moscow rebel soviet troop ussr
withdraw

arab araf israe israel palestinian plo territory

chancellor chemic export german germani kadafi kohl libya libyan
plant poison weapon west

able ago associ believ bit bylin call com consid couldn dai datelin
didn do doesn don experi feel front get graphic gui happen haven
help hope includ isn life little ll look lot major maybe mind month
own people photo probabl re reason recent seen sit soon staf start
success tell time tough tri try ve wasn week wouldn writer

ahead ball basket beate brea chri coach coache consecut el final
finish foul fourth free game grab half halftim hill host jef lead
league led left los lost minut miss outscor overal plai player pointer
quarter ralli rank rebound remain roundup scor score scorer season
shot steal straight streak team third throw ti tim trail victori
win won
```

ab bengal bowl cincinnati craig dalla denver esiason field football francisco giant jerri joe miami minnesota montana nfl oppon pass pittsburgh quarterback rice rush super table taylor terri touchdown yard

When we look at the word clusters, they form a coherent set. The words in the $1^{st}$ cluster are all related to the USSR-Afghanistan conflict. The $2^{nd}$ cluster is about the Arabs, Israelis, and the Palestinians, while the $3^{rd}$ cluster is about the German-Libyan relationships. The $4^{th}$ cluster represents the general vocabulary in our abstract data model since it contains generic terms that could appear together in any document. The $5^{th}$ cluster represents general sports terms, while the $6^{th}$ cluster contain only the words related to football. As we can see, the word clusters correspond to the concepts in the dataset. Concepts are related to the top words in a document cluster, but there's no one-to-one correspondence. We can expect to have words from several concepts in the list of the top words of a document cluster, since a topic may have words from several concepts. Thus, while our abstract data model is a very simple model, it captures the nature of the text data in some important respects.

# 6 Conclusions and Future Work

Our research indicates that clustering based on shared nearest neighbors is a better approach than K-means clustering for finding groups of documents with a strong, coherent topic or theme. To explain and understand these results, we introduced a concept-based document model, where each document is generated by choosing its words from a small number of specialized vocabularies plus a general vocabulary, and where the probabilities with which words are chosen from each vocabulary depend on the class (topic) to which the document belongs. This model provides a solid foundation for the work in this paper (and future work) by providing a framework that explains **(a)** how it is possible for two documents to be most similar to each other, but yet be in different classes and **(b)** why a shared nearest neighbor clustering approach might work better than K-means or hierarchical clustering approaches, both of which are based on pairwise similarities.

Our future works relates to two areas: understanding and extending our document models and implementation. In the current implementation of the algorithm, topic and noise thresholds are picked as percentages of the total number of data points and the remaining thresholds are picked as

percentages of the number of links in the shared nearest neighbor graph. For example, if we have an idea about the amount of noise in the data, we could set the noise threshold accordingly. Using the same set of parameters, we obtain different link strength thresholds for different datasets since they depend on the structure of the data; they are not specified as pre-set values. While this method of selecting the parameters works a lot better than setting fixed thresholds, it is not fully automatic. Fully automating the selection of parameters requires better understanding of text data.

In terms of further developing our document models, we hope to extend our concept models to obtain a better implementation and to more thoroughly understand the behavior of a wide variety of clustering algorithms on a wide variety of data sets. For example, transaction data, e.g., customer purchasing data, is very similar to document data.

## References


[1] Charu C. Aggarwal, Stephen C. Gates and Philip S. Yu, "On the merits of building categorization systems by supervised clustering,"Proceedings of the fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Pages 352 – 356, 1999.

[2] Douglas R. Cutting, David R. Karger, Jan O. Pedersen, and John W. Tukey, "Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections", ACM SIGIR '92, Pages 318 – 329, 1992.

[3] Richard C. Dubes and Anil K. Jain, *Algorithms for Clustering Data*, Prentice Hall, 1988.

[4] Inderjit S. Dhillon and Dharmendra S. Modha, "Concept Decompositions for Large Sparse Text Data using Clustering," to appear in Machine Learning, 2000 (also appears as IBM Research Report RJ 10147 (95022), July 8, 1999).

[5] A. El-Hamdouchi and P. Willet, "Comparison of Hierarchic Agglomerative Clustering Methods for Document Retrieval," The Computer Journal, Vol. 32, No. 3, 1989.

[6] K. C. Gowda and G. Krishna, (1978), "Agglomerative Clustering Using the Concept of Mutual Nearest Neighborhood", Pattern Recognition, Vol. 10, pp. 105-112.

[7] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim, (1998), "ROCK: A Robust Clustering Algorithm for Categorical Attributes," In Proceedings of the 15th International Conference on Data Engineering, 1999.

[8] R. A. Jarvis and E. A. Patrick, "Clustering Using a Similarity Measure Based on Shared Nearest Neighbors," IEEE Transactions on Computers, Vol. C-22, No. 11, November, 1973.

[9] George Karypis and Eui-Hong (Sam) Han , "Concept Indexing: A Fast Dimensionality Reduction Algorithm with Applications to Document Retrieval & Categorization," CIKM 2000.

[10] George Karypis, Eui-Hong Han, and Vipin Kumar, (1999) "CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling," IEEE Computer, Vol. 32, No. 8, August, 1999. pp. 68-75.

[11] Gerald Kowalski, *Information Retrieval Systems – Theory and Implementation*, Kluwer Academic Publishers, 1997.

[12] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: an Introduction to Cluster Analysis*, John Wiley and Sons, 1990.

[13] Daphe Koller and Mehran Sahami, "Hierarchically classifying documents using very few words," Proceedings of the 14th International Conference on Machine Learning (ML), Nashville, Tennessee, July 1997, Pages 170-178.

[14] Bjorner Larsen and Chinatsu Aone, "Fast and Effective Text Mining Using Linear-time Document Clustering," KDD-99, San Diego, California, 1999.

[15] C. J. van Rijsbergen, *Information Retrieval*, Buttersworth, London, second edition,1989.

[16] Michael Steinbach, George Karypis, and Vipin Kumar, "A Comparison of Document Clustering Algorithms," KDD-2000 Text Mining Workshop, 2000.

[17] TREC: Text REtrieval Conference. http://trec.nist.gov

[18] Oren Zamir, Oren Etzioni, Omid Madani, Richard M. Karp, "Fast and Intuitive Clustering of Web Documents," KDD '97, Pages 287-290, 1997.

[19] "Evaluation of Hierarchical Clustering Algorithms for Document Datasets," Ying Zhao and George Karypis, CIKM 2002.