Evaluating Document Clustering for Interactive Information Retrieval

Anton Leuski
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, MA 01003 USA
email: leuski@cs.umass.edu

Abstract

We consider the problem of organizing and browsing the top ranked portion of the documents returned by an information retrieval system. We study the effectiveness of a document organization in helping a user to locate the relevant material among the retrieved documents as quickly as possible. In this context we examine a set of clustering algorithms and experimentally show that a clustering of the retrieved documents can be significantly more effective than traditional ranked list approach. We also show that the clustering approach can be as effective as the interactive relevance feedback based on query expansion while retaining an important advantage – it provides the user with a valuable sense of control over the feedback process.

1 Introduction

Locating interesting information is one of the most important tasks in Information Retrieval (IR). An IR system accepts a query from a user and responds with a set of documents. The system returns both relevant and non-relevant material, and a document organization approach is applied to assist the user in finding the relevant information in the retrieved set.

Generally a search engine presents the retrieved document set as a ranked list of document titles. The documents in the list are ordered by the probability of being relevant to the user's request. The highest ranked document is considered to be the most likely relevant document, the next one is slightly less likely and so on. This organizational approach can be found in almost any existing search engine [11, 14, 10]. It is assumed that the user will start at the top of the list and follow it down examining the documents one at a time.

A number of alternative document organization approaches have been developed over the recent years [2, 7, 9, 20]. These approaches are normally based on visualization and presentation of some relationships among the documents, terms, or the user's query. One of such approaches is document clustering.

Document clustering has been studied in the field of information retrieval for several decades. Willett [31] gives an excellent overview of the existing algorithms and applications. The use of clustering is based mostly on the Cluster Hypothesis: "closely associated documents tend to be relevant to the same requests" [29, p.45].

Croft [6] and more recently Hearst and Pedersen [13], showed that the Cluster Hypothesis also holds in a retrieved set of documents. However, they did not study how the clustering structure may help a user to find relevant information more quickly. In contrast to those studies Voorhees [30] could not find any conclusive support for the Cluster Hypothesis.

Numerous studies and anecdotal evidence hint that document clustering can be a better way of organizing the retrieval results [13, 25, 16]. However, we could not find any strong experimental results that support this assumption. In this paper we describe a set of experiments that show the clustering to be a much more effective way of directing a user towards relevant documents among the retrieved set than the ranked list.

1. K2 Skis 2001 Site	26. K2 Skis, Nordica Boots, Scott Ski Poles
2. K2 Skis	27. Epinions.com - Reviews of Alpine, K2 Skis
3. www.k2ski.com/	28. k2 skis
4. Welcome to K2 Sports	29. K2 SKIS FOR SALE
5. 411SKIING - K2 - k2 skis	30. K2 Skis
6. SkiNet.com | Gear | K2 Skis 2001-02	31. Devon Ski Centre, K2 skis
7. skinet | K2 Skis 2001-02	32. K2 Skis from Village Ski Loft
8. Aberdeen country and western line dance and coupl	33. K2 SKIS
9. Western Clipart	34. Berg's Alpine: K2 Skis
10. "K2 Skis For Sale" Backcountry World - [	35. K2 Skis
11. SKIING Magazine | Gear | K2 Skis 2001-02	36. K2 Skis Tutorial
12. K2 Skis at Footloose Sports in Mammoth Lakes Calif	37. K2 Skis Tutorial
13. GoSki Gear - K2 skis - The K2 ski line, K2 ski reviev	38. Waxed - Boarding, Blading and freestyle information
14. GoSki Gear - K2 skis - K2 ski line for 1998/99	39. K2 Telemark Skis - World Piste Piste Stinx Super St
15. AskAnOwner.com -- Get answers about K2 Skis	40. K2 Nordic Skis Directory
16. AskAnOwner.com -- Questions and answers about	41. Community/Forums
17. Department'K2 Skis 1999/2000'	42. Skiferie og skirejser pÂ Danmarks nye store skiguide
18. Colorado Firstrax, K2 Skis	43. K2 Mod Skis - www.ezboard.com
19. K2 Skis Review	44. Department'K2 Alpine Skis'
20. K2 skis 98/99	45. www.gregory1.com/cgi-bin/showlogo.pl?FN=K2,K2
21. Accessory K2 Skis	46. Hooger Booger 162 Snowboard/bindings - K2 170 S
22. GoSki Gear - K2 ski reviews - Other K2 skis	47. ANY OPINIONS ON TUA VS K2 TELE SKIS?
23. GoSki Gear - K2 skis - The 1997/98 K2 ski line	48. Re:ANY OPINIONS ON TUA VS K2 TELE SKIS?
24. SKI Magazine | K2 Skis 2001-02	49. K2 Four Skis
25. K2 Skis: 1997-98 Ski Review from The Mountain Zo	50. A pair of K2 Three Skis with bindings and poles

Figure 1: Top fifty documents retrieved by the Google search engine for the "K2 skis" query.

We also show that the clustering can be as effective as more traditional relevance feedback approaches based on query expansion.

Our study is task-oriented. We begin by specifying an exact IR task for which we apply the clustering approach. Next we describe our baseline and experimental systems. We then consider how the information provided by the clustering organization can help us dynamically adjust the order in which the retrieved set is examined. Using these observations we define an algorithmic browsing strategy and measure its performance on the standard data sets [12]. We conclude the paper with discussion of the experimental results and suggestions for future work.

2 Experimental Task

Imagine a user who is looking for a new pair of downhill skis. She is interested in the information about different K2 skis with a primary focus on what other people have to say about those skis. Our user turns to a web search engine and types in "K2 skis". Figure 1 shows the top 50 documents retrieved by the Google search engine (`www.google.com`) for that query. The ranked list is broken into two columns with 25 documents each. The list flows starting from the top left corner down and again from the top right corner to the bottom of the window. The pages are ranked by the search engine in the order they are presumed to be relevant to the query. The rank number precedes each title in the list.

Note that the documents in the retrieved set discuss many different aspects of the topic and not all of them are relevant. For example, the first document in the list is the homepage of the K2 Ski Corporation. That page does not have any ski reviews. There are also web-pages that sell skis (the document ranked 5

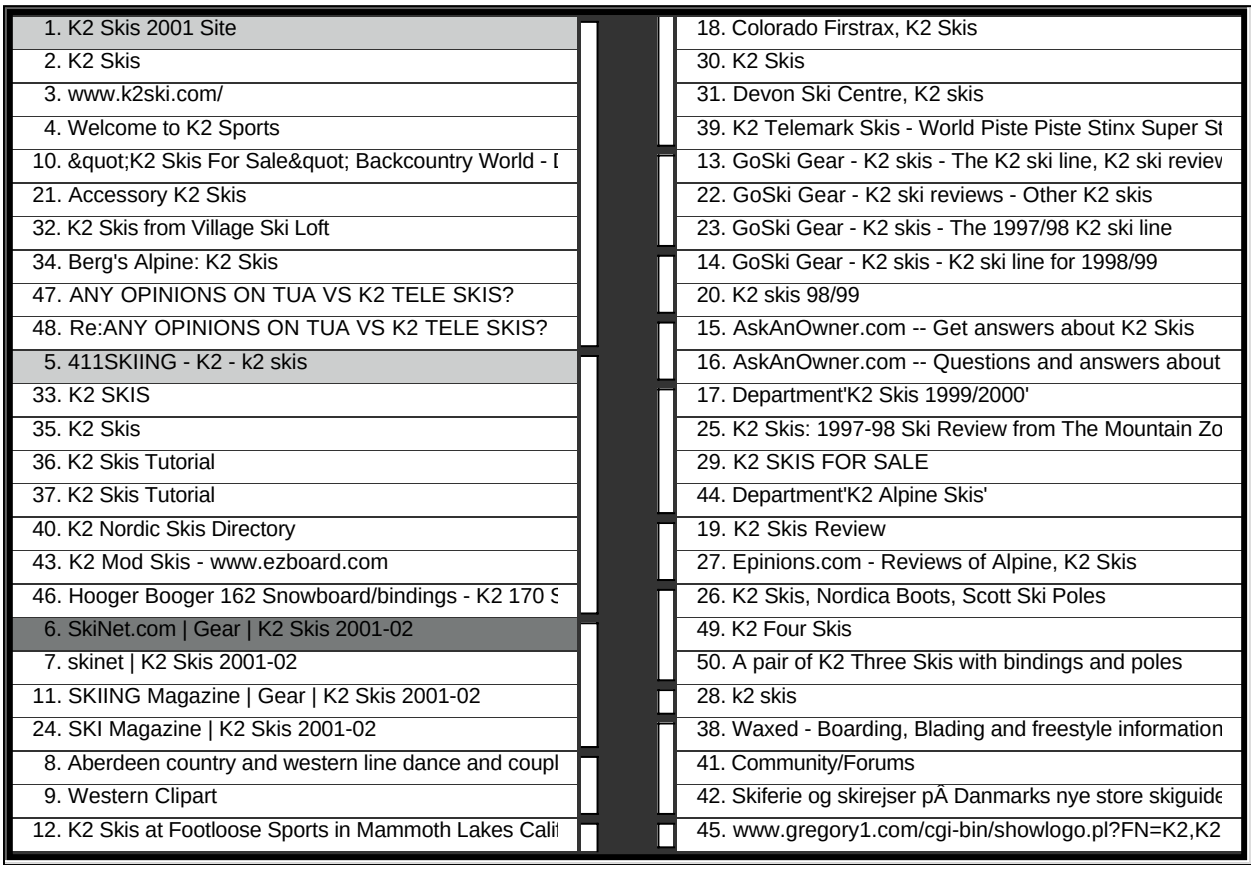

Figure 2: A set of clusters created for the top fifty documents retrieved by the Google search engine for the "K2 skis" query. A light-gray background indicates a non-relevant document. The relevant document has a dark-gray background.

in the list), discuss K2 Telemark skis (document 47), and completely unrelated pages such as document 9. The first relevant document is document 6.

The user's goal is to locate all relevant documents (the documents with reviews of K2 skis) in the ranked list as quickly as possible. The design of the ranked list organization assumes that the user will start at the top of the list and follow it down examining documents one at a time.

Figure 2 shows the same 50 documents partitioned into 14 clusters. Each cluster is represented by a rectangular bracket or "handle" that runs parallel to the cluster. We ordered the documents in the clusters using their Google rank and sorted the clusters using the rank of the highest ranked document in each cluster. The first document in the first cluster (document 1) is about K2 Ski Corporation. It is non-relevant and the figure shows it with a light-gray background. The rest of the documents in the cluster are similar to the first one and they are likely to be about the company as well. We go to the next cluster. The first document (document 5) is the page that sells skis and accessories. We mark it as non-relevant and skip to the next cluster. The first document (document 6) is a page that reviews skis and we marked it as relevant – it has a dark-gray background. The rest of the documents in the cluster are also relevant as they contain ski reviews from different sources. Thus we located the first relevant document after examining only 3 documents in the set. We had to examine 6 documents to find the same document with the ranked list.

This example illustrates both the retrieval problem we are interested in and our intuition why clustering can help us to solve it. Thus, we assume that the user is working with the document organization system to find the relevant material among documents retrieved by an information retrieval system. The experimental task is defined: *Given that no documents presented by the information organization system are marked as*

relevant or non-relevant, isolate the relevant material in the document set.

The first experimental question we study in this paper is whether we can locate the relevant documents in the set more quickly by using the clustering document organization than by following the ranked list.

3 System Design

We use the INQUERY information retrieval system as the retrieval engine for our experiments [4]. We run a query through the system and focus our analysis on the top 50 returned documents. There are two baseline document organization approaches in our study: the ranked list and interactive relevance feedback.

The ranked list is the order of the retrieved documents as determined by INQUERY. The system uses an inference network model and estimates probabilities of how much each document satisfies user's information need [28].

The interactive relevance feedback procedure is as follows: we start from the top of the ranked list. Each time a new relevant document is discovered, we submit all the examined documents to the INQUERY's relevance feedback subsystem to modify the weights in the original query. Additionally, the query is expanded by adding 100 highest ranked terms from the examined documents [1]. Note that this procedure takes into account both relevant and non-relevant documents. The unexamined documents in the set are reranked by INQUERY using the new query and we continue down the list. We have experimented with different amount of query expansion. Adding the top 100 terms gave us the best average performance.

3.1 Clustering Algorithm

For the clustering approach we have to measure the distances between documents. The INQUERY's retrieval model neither incorporates the notion of similarity between documents nor assumes the construction of document representations. To compute inter-document similarities we employ the vector-space model for document representation [27] – each document j is defined as vector V_j, where $v_{i,j}$ is the weight in this document of the i-th term in the vocabulary. The term weight is determined by an ad-hoc formula [4], which combines Okapi's *tf* score [26] and INQUERY's normalized *idf* score:

$$v_{i,j} = \frac{tf_{i,j}}{tf_{i,j} + 0.5 + 1.5\frac{doclen_j}{avgdoclen}} \cdot \frac{\log(\frac{colsize+0.5}{docf_i})}{\log(colsize+1)} \tag{1}$$

where $v_{i,j}$ is the weight of the ith term in the vocabulary in the jth document, $tf_{i,j}$ is the number of times the term occurs in the document, $docf_i$ is the number of documents the term occurs in, $doclen_j$ is the number of terms in the document, *avgdoclen* is the average number of terms per document in the collection, and *colsize* is the number of documents in the collection. The similarity between a pair of documents is computed as the cosine of the angle between the corresponding vectors ($\cos\theta$) [27]. In this study we use one over the cosine ($1/\cos\theta$) to define the distance between a pair of documents.

There exist many different clustering algorithms and a particular choice is often motivated by the algorithm efficiency. If we are to cluster a collection of hundreds of thousands of documents then using an algorithm that requires us to make $N\cdot(N-1)/2$ pair-wise comparison for N documents would be prohibitively expensive. A number of alternative solutions have been developed. For example, Scatter/Gather [8] interacts with a user – divides or merges clusters per user's request – in a constant time due to a clever near linear ($O(kn\log n)$) preprocessing phase.

Another popular approach is K-means clustering. The number K is a parameter of the algorithm and it determines the number of final clusters. The algorithm starts by defining K cluster centroid or "seeds" and then compares every objects with every centroid. The object is assigned to the cluster with the closest seed. The execution time is $O(K \cdot N)$.

These efficient algorithms sacrifice some accuracy in arranging the documents to receive a significant advantage in speed. This is achieved by ignoring some of the inter-object similarity information. If the number of documents is small, it is more cost-effective to employ $O(N^2)$ techniques that make use of all inter-object similarity information – the whole $N \cdot (N-1)/2$ set of pair-wise distances. This is the reason why we consider a system built around a hierarchical agglomerative clustering algorithm.

Table 1: Lance-Williams coefficients for most known agglomerative clustering methods. n_i denotes the size of the ith cluster.

Method	α_i	α_i	β	γ
Single linkage	0.5	0.5	0	-0.5
Complete linkage	0.5	0.5	0	0.5
Group average	$\frac{n_i}{n_i+n_j}$	$\frac{n_j}{n_i+n_j}$	0	0
Weighted group average	0.5	0.5	0	0
Centroid	$\frac{n_i}{n_i+n_j}$	$\frac{n_j}{n_i+n_j}$	$\frac{-n_i \cdot n_j}{(n_i+n_j)^2}$	0
Ward	$\frac{n_i+n_k}{(n_i+n_j+n_k)}$	$\frac{n_j+n_k}{(n_i+n_j+n_k)}$	$\frac{-n_k}{(n_i+n_j+n_k)}$	0

A hierarchical agglomerative clustering algorithm creates a hierarchy of clusters – it builds a tree where each node is a cluster of objects and the clusters corresponding to the node's immediate children form a complete partition of that cluster [24]. On input the algorithm receives a set of objects and a matrix of inter-object distances. It starts by assigning each object to its own unique cluster – the leaves of the future tree. The algorithm iterates through the cluster set by selecting the closest pair of clusters and merging them together forming a new cluster that replaces them in the cluster set. A node corresponding to this new cluster is created in the tree and the selected pair of clusters become its children. That procedure is executed until all objects are contained within a single cluster, which becomes the root of the tree.

This is a general algorithm that is instantiated by choosing a specific distance function for clusters. Indeed, the distance between a pair of singleton clusters is well-defined by the original distance matrix. If one of the clusters contains more than one object, the inter-cluster distance is determined by a specific heuristic. For example, we may define the inter-cluster distance as the smallest distance between two objects in both clusters. Other suggested alternatives include the average distance between two objects and the maximum distance. Lance and Williams [17] have shown that many different clustering methods can be derived from the following equation and computed quite efficiently:

$$d_{k,i\cup j} = \alpha_i \cdot d_{k,i} + \alpha_j \cdot d_{k,j} + \beta \cdot d_{i,j} + \gamma \cdot |d_{k,i} - d_{k,j}| \tag{2}$$

here, $d_{k,i\cup j}$, the distance between the cluster created by merging ith and jth clusters and an arbitrary cluster k is defined as a nonlinear function of distances between the individual clusters. The coefficients for the most commonly used methods are presented in Table 1.

In this study we consider six different clustering techniques based on the generalized agglomerative algorithm (Table 1). The *single linkage* method defines the distance between two clusters as the smallest distance between two objects in both clusters. The *complete linkage* uses the largest distance instead. These two methods represent two extremes of the generally accepted requirement that the "natural" clusters must be cohesive and isolated from the other clusters [24]. Single linkage clusters are isolated but not cohesive, while complete linkage produces cohesive groups that may not be isolated at all. The other four methods represent some compromise between the two extremes.

Thus the second experimental question we examine in this paper is the comparison of six different clustering algorithms to determine which is best suited for the task of helping the user to quickly locate relevant documents.

3.2 Creating Partition

The hierarchical agglomerative clustering algorithm produces a hierarchy of clusters. We are interested in an approach which presents the user with a partition of the document set – a set of clusters that divides the retrieved material into the groups of similar documents.

To create a partition of the document set from a cluster hierarchy we "cut" the hierarchy at some level, i.e., stop the clustering algorithm before it reaches the root of the tree. The clusters present in the set at that moment form the required partition. The problem is to decide at what point to make the cut. For example, the Scatter/Gather research [13] fixes the number of clusters in the document set. We, on the other

hand, set a threshold on the similarity distance between clusters – while iterating through the cluster set the algorithm stops as soon as the distance between the closest pair of clusters exceeds the threshold. If the threshold is kept constant from session to session, the density of the clusters becomes the system's invariant. The user will always know what minimal degree of similarity to expect from the documents placed in the same cluster.

To select the threshold value we conduct our experiments following a basic two-way cross-validation scheme. We divide our experimental data set into three parts: training, testing, and evaluation data. We select the threshold using the data from the former two data sets and evaluate the performance on the rest of the data.

3.3 Presenting Clusters

A clustering algorithm brings together similar documents. We show the document set to the user as a list of clusters where each cluster in turn is arranged as a list of document titles. We call this representation the *clustered list* by analogy with the ranked list (see Figure 2). Several past studies adopted similar approaches [13, 22]. However, we do not show any textual descriptions for the clusters. Instead we select one representative document from each cluster and place it at the top of the cluster's list. This document is supposed to be the most helpful to the user in establishing the overall relevance value for the cluster. Ideally, by looking at the representative document the user should be able to decide whether to examine the cluster or skip it and go to the next one. The rest of the documents inside the cluster are kept in their original order – they are ordered by the probability of being relevant to the user's request. This should insure an effective ranking [29, p.88]. The clusters are ordered using the original rank of the highest ranked document in each cluster.

The third experimental question we study in this paper is the choice of the representative document or the first document in a cluster list. We consider four different alternatives. The first, a rather obvious choice is to use the document that is the best representation of the cluster – the cluster centroid, or the document that is the most similar to the actual centroid. The second alternative we consider is the highest ranked document in the cluster. Our intuition is that if this document is non-relevant than the rest of the cluster is very likely non-relevant. The third choice is to use the lowest ranked document: if that document is relevant than it is very likely that the rest of the cluster is also relevant.

The documents at the top of the ranked list most likely are relevant and the documents at the bottom of the list most likely are non-relevant. Lewis [23] speculated that the best way to find the boundary between the relevant and non-relevant material in the list is to examine the documents in the middle. The last candidate for the cluster representative is the medium ranked document – the document whose original rank is the median of the cluster.

4 Search Strategy

Bookstein [5] argues that information retrieval should be envisioned as a process, in which the user is examining the retrieved documents in sequence and the system can and should gather the feedback to adjust the retrieval. In our previous studies we have adopted a similar notion while looking at organizing the retrieval results [21, 19]. We introduced a notion of a *search strategy* as a dynamic ranking process that orders the documents solely on the information provided by the document organization and relevance data obtained from the user.

For example, to build a ranked list the documents are ordered by probability of being relevant. The expected search strategy is to start at the top of the ranked list and proceed down the list examining the documents one-by-one.

In our previous work we studied a document organization system that visualizes documents as objects in two- or three-dimensional space positioned in proportion to the inter-document similarity [21]. A search strategy for such a system ranks the documents based on spatial proximity to the known-relevant objects.

There are two different information clues that the clustering system supplies to a user. The first, two documents are similar to each other if and only if they are listed in the same cluster. The second, like in the original ranked list, both the documents and clusters are ordered by their importance. The user should

start at the top of the list and follow it down. The only difference from the ranked list is that the user can abandon examining a cluster without looking at every document in it and jump to the next cluster in the list.

A search strategy reorders the documents in the retrieved set at discrete time steps, i.e., when the system receives relevance feedback about examined documents. The reordering is performed given some representation of the current state of the document set $\mathcal{D}_t$, where t is the time step. A mapping is computed between each unexamined document d and a numeric value: $F(\mathcal{D}_t, d)$. The documents are ordered using these numeric values $F(\mathcal{D}_t, d)$. We call the mapping function the *search strategy function* [18].

The search strategy for the clustering document organization system targets the documents that are most likely to be relevant. Following the idea expressed in the Cluster Hypothesis it selects the document with highest similarity to the known relevant documents and at the same time it minimizes the similarity to the known non-relevant documents:

$$F(\mathcal{D}_t, d) = \theta_1 \cdot \sum_{x \in \mathcal{R}_t} sim(x, d) + \theta_2 \cdot \sum_{x \in \mathcal{N}_t} sim(x, d)$$

where $\mathcal{R}_t$ and $\mathcal{N}_t$ are the sets of all examined relevant and non-relevant documents at time step t and $sim(x, d)$ is the binary similarity between two documents:

$$sim(x, d) = \begin{cases} 1, & \text{if } x \text{ and } d \text{ are in the same cluster} \\ 0, & \text{otherwise} \end{cases}$$

At each time step the search strategy selects the document d with the highest value of $F(\mathcal{D}_t, d)$. If two documents from different clusters have the same score, the search strategy prefers the document from the higher ranked cluster. For two documents from the same cluster the ties are broken by selecting the document with the highest place in the cluster's list.

The fourth question that we consider in this paper is the importance of the relevant and non-relevant information while computing the search strategy function. We experiment with different values for θ_1 and θ_2 and compare the performance of the resulting search strategies.

5 Experimental Setup

For our experiments we use TREC ad-hoc queries with their corresponding collections and the relevance judgments supplied by NIST accessors [11]. Specifically, TREC topics 251-300 and 301-350 are converted into queries and run against the documents in TREC volumes 2 and 4 (2.1GB) and TREC volumes 4 and 5 (2.2GB) accordingly. For each TREC topic we consider four types of queries: (1) a query constructed by extensive analysis and expansion [3]; (2) the description field of the topic; (3) the title of the topic; and (4) a query constructed from the title by expanding it using Local Context Analysis (LCA) [32]. A query of the last type has size and complexity between the corresponding queries of the first and second types.

Our assumption is that during a typical retrieval session a user does not generally look beyond the first screen showing the retrieved material – that is approximately equivalent to ten retrieved documents. Thus, we are interested in analyzing just the top portion of the ranked list. For each query we select the 50 highest ranked documents.

Thus our data consists of documents from two different collections (TREC-5 and TREC-6) retrieved by queries of four different types – eight different data sets. Each data set serves as a separate training data set – we exhaustively search for the threshold value that produces the best average performance on that data set. The training phase produces four potentially different threshold values for the documents from one collection, i.e., one threshold for each query type. We select one threshold value out of these four that gives us the best performance on all data sets from the same collection: the training data set and the other three data sets of documents retrieved from the same collection combined is the testing set. Now that selected threshold value is used to organize the documents from the other collection. Thus the other four data sets form our evaluation group. In the section detailing our experimental results below we report only the numbers from the corresponding evaluation data sets.

Table 2: Performance of the clustering search strategy on document set partitions created by the group average algorithm. Average precision numbers for two initial conditions and percent improvement over the original ranked list ("RL") are shown. The table also includes the precision values for the ranked list and interactive relevance feedback search strategies. The latter approach used 100 terms for query expansion.

Data set		RL Original	RF	group average	
				no-info (v. RL%)	first-rel (v. RL%)
TREC-5	Full	38.49	41.87	45.48 (18.16%)	44.14 (14.67%)
	Description	36.06	44.19	41.40 (14.82%)	40.84 (13.26%)
	Title	33.19	36.29	36.81 (10.91%)	37.38 (12.64%)
	Expanded Title	29.91	32.64	36.63 (22.45%)	33.43 (11.75%)
TREC-6	Full	53.67	54.48	57.69 (7.47 %)	56.73 (5.69 %)
	Description	46.66	58.54	54.69 (17.21%)	53.86 (15.44%)
	Title	46.19	56.23	51.78 (12.12%)	52.56 (13.80%)
	Expanded Title	51.64	53.12	54.61 (5.76 %)	54.34 (5.23 %)
total average		41.98	47.17	47.39 (13.61%)	46.66 (11.56%)

The outcome of a search strategy is a new document ranking as opposed to the original ranked list. Two rankings can be compared using traditional information retrieval measure. In this study we use the average precision [12]. Unless otherwise noted we use the paired two-tailed t-test with the cutoff level set to 5% ($p < 0.05$) to measure the statistical significance.

6 Experiments

6.1 Initial Conditions

The first question is to compare the performance of the clustering search strategy $F(\mathcal{D}_t, d)$ with the performance of the ranked list and relevance feedback search strategies. The last four columns of Table 2 show the average precision values for the clustering search strategy and the percentage difference from the ranked list ("RL"). The search strategy worked with clustering structures built by the group average algorithm. The highest ranked document was used as the cluster representative. The search strategy function weights were set to $\theta_1 = 1$ and $\theta_2 = -1$.

In addition to the "pure" clustering search strategy (the "no-info" columns) we considered a mixed approach. The last two columns ("first-rel") show the average precision for the clustering search strategy that starts with the highest ranked relevant document and all preceding non-relevant documents marked in the retrieved set. This situation simulates a user who has located the first relevant document by following the ranked list and then switched to the clustering organization.

We observe a small difference in average precision due to available relevant information. The values in the last two columns are slightly smaller than in the two preceding ones. It shows that one can locate the first relevant document in the set slightly faster with the clustering organization than by following the ranked list. However the difference is small and not statistically significant.

We observe 13.61% and 11.56% improvement over the ranked list. These differences are statistically significant. The differences between the clustering search strategy and the search strategy for interactive relevance feedback ("RF ") are small and not statistical significant.

6.2 Algorithms

The second experimental question is the comparison of six different clustering algorithms. Table 3 shows the corresponding average precision values for single linkage, complete linkage, group average, weighted group average ("weighted average"), centroid, and Ward algorithms. The highest ranked document was chosen as the cluster representative for each algorithm. The weights for clustering search strategy were set to $\theta_1 = 1$ and $\theta_2 = -1$.

Table 3: Performance of F search strategy on document set partitions created by six different clustering algorithms.

Data set		single linkage	complete linkage	group average	weighted average	centroid	Ward
TREC-5	Full	43.00	45.42	45.48	44.51	43.42	45.28
	Description	39.59	41.15	41.40	40.70	40.05	41.91
	Title	36.12	36.60	36.81	36.80	36.24	37.14
	Expanded Title	35.15	36.59	36.63	35.82	34.77	36.97
TREC-6	Full	56.16	57.30	57.69	56.06	56.35	56.25
	Description	52.27	53.24	54.69	54.08	52.91	54.20
	Title	50.40	52.89	51.78	51.78	52.21	52.47
	Expanded Title	54.52	54.70	54.61	54.11	54.77	54.06
total average		45.90	47.24	47.39	46.73	46.34	47.28

Table 4: Performance of the clustering search strategy on document set partitions created by the group average algorithm using four different types of cluster representative documents.

Data set		centroid	highest ranked	lowest ranked	medium ranked
TREC-5	Full	44.51	45.48	38.92	43.62
	Description	39.72	41.40	38.51	39.81
	Title	36.64	36.81	36.21	37.11
	Expanded Title	35.73	36.63	35.63	35.36
TREC-6	Full	54.63	57.69	52.89	53.63
	Description	52.35	54.69	50.71	52.20
	Title	51.45	51.78	47.98	47.79
	Expanded Title	51.08	54.61	49.39	48.07
total average		45.76	47.39	43.78	44.70

We observed that the group average and Ward algorithms result in better performance consistently across all experimental variables considered in this study. The difference between these two algorithms is insignificant. Also the differences between the group average algorithm and both weighted average and complete linkage are not statistically significant. The single linkage method is a clear "loser" in this competition.

6.3 Cluster Representatives

The third experimental question is to evaluate the effect of the cluster representative on the quality of the document organization. Recall from Section 3.3 that the representative document is the one placed at the top of each cluster's list. It is supposed to be the most helpful in establishing the overall relevance value for the cluster. Ideally, by looking at the representative document the user should be able to decide whether to examine the cluster or skip it and go to the next one.

Table 4 shows the average precision values obtained for four different cluster representatives. We used the document organization structure created by the group average algorithm. The relevant and non-relevant weights for clustering search strategy were set to $\theta_1 = 1$ and $\theta_2 = -1$.

The choice of the highest ranked document in a cluster as the cluster's representative is the most effective for the task of locating the relevant material. We observe an almost 4% drop in average precision while selecting the document closest to the cluster's center. The difference is statistically significant. Our explanation is that the highest relevant document allows user to quickly discard non-relevant clusters – if even the highest ranked document in the cluster is non-relevant, then it is very likely that the rest of the cluster is also non-relevant.

Table 5: Performance of the clustering search strategy on document set partitions created by the group average algorithm. Average precision values are shown for different ratios of relevant and non-relevant weights (θ_1 and θ_2) in $F(\mathcal{D}_t, d)$.

Data set		Relevant to non-relevant weight ratio					
		1:0	1:-0.25	1:-0.5	1:-1	1:-2	1:-4
TREC-5	Full	38.24	45.14	45.40	45.48	44.71	44.35
	Description	38.19	40.55	40.99	41.40	40.74	40.57
	Title	34.55	36.70	37.02	36.81	36.67	36.64
	Expanded Title	30.82	35.83	36.30	36.63	36.59	36.20
TREC-6	Full	54.31	56.99	57.03	57.69	57.51	57.76
	Description	50.93	54.36	54.45	54.69	53.12	53.46
	Title	49.49	51.94	51.98	51.78	52.90	52.47
	Expanded Title	49.25	54.27	54.31	54.61	54.86	55.26
total average		43.22	46.97	47.18	47.39	47.14	47.09

6.4 Search Strategy

The fourth experimental question we consider in this study is the optimization of the search strategy function $F(\mathcal{D}_t, d)$. Table 5 shows the performance of the search strategy on the clustering structures created by the group average algorithm using the highest ranked document as the cluster representative.

The six columns in Table 5 correspond to six different value settings for the relevant and non-relevant weights (θ_1 and θ_2) in F. The column titles are presented in the form θ_1:θ_2. There is a clear maximum in performance of the total average for $\theta_1 = 1$ and $\theta_2 = -1$. Increasing or decreasing the value for the non-relevant weight θ_2 from -1 leads to lower values of average precision. All pairwise differences between the maximum ("1:-1") and the other parameter sets are statistically significant.

7 Conclusions

The Cluster Hypothesis of Information Retrieval has been shown true on multiple occasions. In this paper we have studied how clustering can be used to interactively direct the user's search for relevant information in the top ranked portion of the retrieved set. We compared six different hierarchical agglomerative clustering algorithms. We defined an effective method for transforming a clustering hierarchy into a partition of the document set. We showed that these document set partitions can be much more helpful in locating the relevant information than the traditional ranked list. We also showed that the clustering can be as effective as the relevance feedback methods based on query expansion.

These results confirm, in the same way that relevance feedback experiments do, that user feedback can dramatically improve the effectiveness of a ranked list. Unlike most past efforts ([1] being a recent exception) we also show that it is also true when feedback is incremental – and even if no new documents are retrieved. Further, we have confirmed this result in a setting where we believe the user will have an crucial sense of control over the feedback process [15].

8 Discussion and Future Work

We see that clustering can greatly improve the effectiveness of the ranked list. In fact it can be as effective as the interactive relevance feedback based on query expansion. Surprisingly this high performance can be achieved by following a very simple strategy. Given a list of clusters created by the group average algorithm, a user starts at the top of the list and follows it down examining the documents in each cluster. As soon as she sees that a cluster has more non-relevant documents than relevant ones, she discards that cluster and switches over to the next one.

Figure 3 shows the result of applying this search strategy to the document set introduced at the beginning

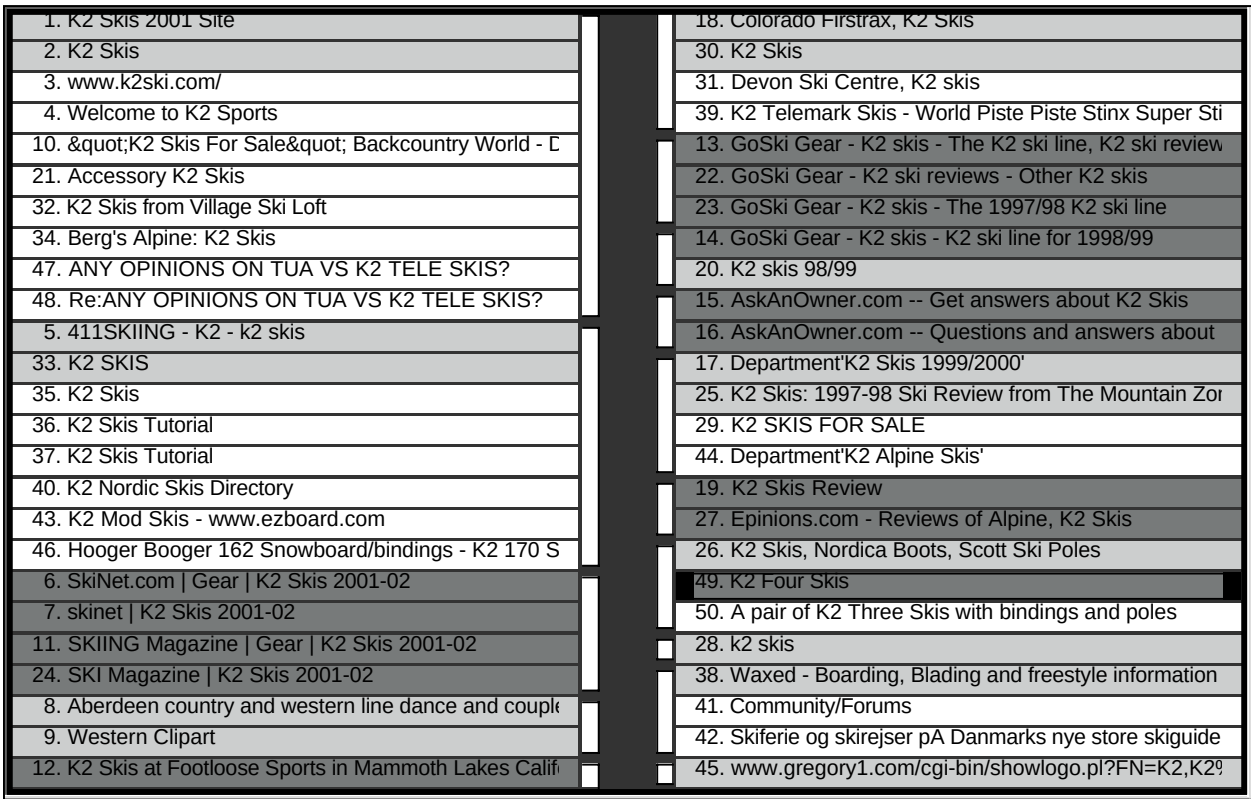

Figure 3: A set of clusters created for the top fifty documents retrieved by the Google search engine for the "K2 skis" query. Dark-gray and light-gray title backgrounds indicate relevant and non-relevant documents accordingly. The document organization is shown after the last relevant document is examined following the search strategy outlined in the paper.

of the paper. Here we marked all documents examined by the user that followed the strategy until the last relevant document was examined. There are 29 examined documents and 14 of those are relevant. The last relevant document is marked 49. If the user followed the ranked list, she would have found it after examining 48 other documents 35 of which are non-relevant.

Another nice quality of the clustering approach is the well-defined informational clues it uses. The preferred order in which documents should be examined is outlined by the clustered list. The inter-document similarity is also easily determined from the list. We believe these informational clues to be clear, obvious, and based on the skills possessed by anyone who has used a web search engine.

Note that our evaluation method results in a lower bound estimate of the system performance. It is possible that a user can find a more effective way of applying the informational clues supplied by the system. On the other hand, if the user adheres to the choices made by the algorithmic search strategy, she is guaranteed at least the level of performance predicted by the analysis in this paper.

We have observed that the threshold value can significantly affect the effectiveness of the system. In these experiments one threshold was used for all document sets. We believe the performance may be improved if the threshold value is adapted to individual user requests. We considered using the threshold based on the query complexity. Instead of selecting the best threshold on one collection overall we used the best threshold for the individual query types, i.e., after we find the best threshold for the TREC-5 Title queries we apply it to cluster the documents from the TREC-6 Title queries only. The average precision numbers were slightly worse than those that we have presented in this paper.

In our previous work we considered a spring-embedding visualization that presented the documents as spheres floating in two- or three-dimensional space positioned in proportion to inter-document similarity [21].

Since that presentation provides accurate and complete information about all pairwise distances between documents it can be used more effectively than the clustering approach. However, we have observed that unassisted searchers are unable to use that system to its full potential and tend to make mistakes selecting similar documents based on spatial distance. On the other hand, the clustering organization creates no confusion in identifying similar documents. We are interested in integrating both the clustering and spring-embedding approaches.

Acknowledgments

I am deeply grateful to James Allan for his support and contributions to the project.

This material is based on work supported in part by the Library of Congress and Department of Commerce under cooperative agreement number EEC-9209623,in part by the National Science Foundation under grant number IRI-9619117, TIDES and in part by SPAWARSYSCEN-SD grant number N66001-99-1-8912. Any opinions, findings and conclusions or recommendations expressed in this material are the author and do not necessarily reflect those of the sponsor.

References

[1] J. Allan. Incremental relevance feedback for information filtering. In *Proceedings of ACM SIGIR*, pages 270–278, 1996.

[2] J. Allan. Building hypertext using information retrieval. *Information Processing and Management*, 33(2):145–159, 1997.

[3] J. Allan, J. Callan, B. Croft, L. Ballesteros, J. Broglio, J. Xu, and H. Shu. Inquery at TREC-5. In *Fifth Text REtrieval Conference (TREC-5)*, pages 119–132, 1997.

[4] J. Allan, J. Callan, W. B. Croft, L. Ballesteros, D. Byrd, R. Swan, and J. Xu. Inquery does battle with TREC-6. In *Sixth Text REtrieval Conference (TREC-6)*, pages 169–206, 1998.

[5] A. Bookstein. Information retrieval: A sequential learning process. *Journal of the American Society for Information Science*, 34(5):331–342, 1983.

[6] W. B. Croft. *Organising and Searching Large Files of Documents*. PhD thesis, University of Cambridge, October 1978.

[7] W. B. Croft and R. H. Thompson. I^3R: A new approach to the design of document retrieval systems. *Journal of the American Society for Information Science*, 38:389–404, 1987.

[8] D. R. Cutting, D. R. Karger, and J. O. Pedersen. Constant interaction-time Scatter/Gather browsing of very large document collections. In *Proceedings of ACM SIGIR*, pages 126–134, 1993.

[9] D. Dubin. Document analysis for visualization. In *Proceedings of ACM SIGIR*, pages 199–204, July 1995.

[10] Google. `http://www.google.com/`.

[11] D. Harman and E. Voorhees, editors. *The Fifth Text REtrieval Conference (TREC-5)*. NIST, 1997.

[12] D. Harman and E. Voorhees, editors. *The Sixth Text REtrieval Conference (TREC-6)*. NIST, 1998.

[13] M. A. Hearst and J. O. Pedersen. Reexamining the cluster hypothesis: Scatter/Gather on retrieval results. In *Proceedings of ACM SIGIR*, pages 76–84, Aug. 1996.

[14] Infoseek. `http://www.infoseek.com/`.

[15] J. Koenemann and N. J. Belkin. A case for interaction: A study of interactive information retrieval behavior and effectivness. In *Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 205–212, 1996.

[16] E. Lagergren and P. Over. Comparing interactive information retrieval systems across sites: the TREC-6 interactive track matrix experiment. In *Proceedings of ACM SIGIR*, pages 164–172, 1998.

[17] G. N. Lance and W. T. Williams. A general theory of classificatory sorting strategies: 1. hierarchical systems. *Computer Journal*, 9:373–380, 1967.

[18] A. Leuski. Relevance and reinforcement in interactive browsing. In *Proceedings of Ninth International Conference on Information and Knowledge Management (CIKM'00)*, pages 119–126, November 2000.

[19] A. Leuski. *Interactive Information Organization: Techniques and Evaluation*. PhD thesis, University of Massachusetts at Amherst, May 2001.

[20] A. Leuski and J. Allan. Evaluating a visual navigation system for a digital library. *International Journal on Digital Libraries*, 3(2):170–184, 2000.

[21] A. Leuski and J. Allan. Improving interactive retrieval by combining ranked lists and clustering. In *Proceedings of RIAO'2000*, pages 665–681, April 2000.

[22] A. Leuski and W. B. Croft. An evaluation of techniques for clustering search results. Technical Report IR-76, Department of Computer Science, University of Massachusetts, Amherst, 1996.

[23] D. D. Lewis. An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of ACM SIGIR*, pages 37–50, 1992.

[24] B. Mirkin. *Mathematical Classification and Clustering*. Kluwer, 1996.

[25] Northern light. `http://www.northernlight.com/`.

[26] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In D. Harman and E. Voorhees, editors, *Third Text REtrieval Conference (TREC-3)*. NIST, 1995.

[27] G. Salton. *Automatic Text Processing*. Addison-Wesley, 1989.

[28] H. Turtle and W. B. Croft. Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems*, 9(3):187–222, 1991.

[29] C. J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 1979. Second edition.

[30] E. M. Voorhees. The cluster hypothesis revisited. In *Proceedings of ACM SIGIR*, pages 188–196, June 1985.

[31] P. Willett. Recent trends in hierarchic document clustering: a critical review. *Information Processing and Management*, 24(5):577–597, 1988.

[32] J. Xu and W. B. Croft. Querying expansion using local and global document analysis. In *Proceedings of ACM SIGIR*, pages 4–11, 1996.