

# Ranking Techniques for Cluster Based Search Results in a Textual Knowledge-base

Shefali Sharma  
Fetch Technologies, Inc  
841 Apollo St, El Segundo, CA 90254  
+1 (310) 414-9849  
ssharma@fetch.com

Sofus A. Macskassy  
Fetch Technologies, Inc  
841 Apollo St, El Segundo, CA 90254  
+1 (310) 414-9849  
sofmac@fetch.com

## Abstract

This paper presents a framework and methodology to improve the search experience in digital library systems. The approach taken is to cluster a textual knowledgebase along multiple relations and return search results in the form of small, focused clusters. Specifically, we generate multiple relationship networks, one per relationship type, and then cluster these networks. At search time, we present a ranked set of clusters—one ranking per relationship type. The intuition for this approach is that returning clusters of contextually related information provides users with a situational and contextual awareness of the search results rather than returning a ranked list of only those documents that match the query. We address the use of both implicit (such as textual content) and explicit (such as citations, authors etc.) relations between documents. The primary question we focus on is how to rank the clusters, given a search query. We explore two approaches: a text-based rank (using the text's similarity to the user's query) and a social network-based rank (using information centrality). A comparison of these two ranking methods suggest that using information centrality for ranking is very useful for ranking clusters and its documents because the documents that characterize that cluster get the highest rank.

## 1. Introduction

By using only text to retrieve search results in a structured textual knowledge base, we lose the rich web of information hidden behind the pure textual information: the information of how the articles relate to each other. Users of digital libraries are interested not only on a single technical paper/research article but also the web of related content. Although text-based features are good for finding similar content, they are not as good at finding conceptually related work that uses different vocabulary or establishing the authority of documents [26]. The social networks formed within the corpus helps answer many questions for the user; questions like: what are the topics that the authors of this paper write about? Which are the articles that are similar to the article of interest? Which papers cite this paper? What are the articles that are talking about the same subject as this paper? What is the contextual importance of this document within the web of related documents? The work presented here describes a framework and methodology for using the network information to help answer these questions.

The intuition for this approach is that returning clusters of contextually related information provides users with a situational and contextual awareness of the search results rather than returning a ranked list of only those documents that match the query. The reason this is important is that people are good at searching and finding items if they know what they are looking for. However, often one is not entirely sure what one is looking for and this approach provides a novel search experience with contextual results that may not perfectly match the query but are nonetheless relevant through association with documents that do match the search query.

Existing online digital libraries like Citeseer [3], Rexa [23], Google Scholar [8] Academic.live.com [15] and many others [28] already use information other than pure text. For example, Citeseer uses header edit distance, citations and text similarity to find related documents. It has its own autonomous citation indexing system [7] and uses the citation information to identify types of papers such as survey papers. Google Scholar uses citation, author and publication information to rank articles in addition to the text. Rexa also provides cross-linked pages for papers, authors, topics and NSF grants and allows browsing by citations, authors, topics, shared-author, cited authors, and citing authors. Academic.live.com uses the authority of the paper along with the text to affect ranking. The ACM portal has recently added author profiles, citation counts, and other interactive features to its digital library.

This paper focuses on improving the search experience by providing search results that contain contextual information such as related documents, topical keywords, how the various documents relate to each other, etc. Specifically, the approach here provides search results in the form of clusters of documents. These clusters are created by first creating networks of documents from various types of information such as text, citations, shared-author, etc., and then by applying an iterative community finding algorithm on the networks to extract small, focused clusters.

The primary question this paper addresses is how to rank clusters that are relevant to a search query. One simple way is to use the average query score for each document in the cluster, or some other text-based ranking. Keeping with the network-oriented focus of the paper, the approach put

forth here is that of using the information centrality metric from social network analysis to rank clusters which have ‘central’ documents match the search query higher than clusters whose documents are on the periphery of the cluster. One of the key results in this paper is that using the information centrality metric to rank clusters as well as documents within a cluster makes it more transparent to a user what the topic of a cluster is and therefore easier for the user to judge whether that cluster is of interest.

One key idea put forth in this work is that giving a clustered view to the user will allow for easier browsing and faster analysis of the data. For example, the user can easily look at the clusters in the citation networks for a given query and browse cluster results, where each cluster will contain papers that cite each other extensively. Each document within the cluster would be ranked either by the text similarity of that document to a query or how central or relevant that document is within a particular cluster. The clusters will be ranked on some aggregate score of its constituent elements.

One subtle, but important, issue that needs to be addressed is how to cluster documents. Many textual knowledge-bases have rich structured information about its documents, making it impossible to create one coherent clustering of all the documents as each kind of relationship and content imposes a different kind of clustering. Rather than creating one global cluster view, the approach taken here is to create separate clusters for each kind of relationship and let the user chose which relations are important for a given query and information need. Specifically, each relationship (such as citations, text-similarity, shared-author, etc) imposes a different social network and hence a different view of the data. Each such view answers a different set of questions. For example, the citation relationship mentioned above gives the ‘cited by’ and ‘has a citation to’ information. The shared-author network answers questions like: What other topics the authors of a particular paper are writing about or who else are the authors of a paper collaborating with? Or if the user just wanted to find textually similar articles he can look in the text-similarity network.

To summarize, the main contributions of this paper is the framework and methodology for providing network-centric search results that are produced through the creation of multiple networks and clusters, the use of social network analysis metrics to rank the clusters and the way in which multiple relationships are returned as part of the search result. The intuition put forth for this methodology as that providing multiple views of the search results in small focused clusters will make it easier for the user to identify documents that are relevant to the current information need.

The remainder of the papers is organized as follows: Section 2 discusses relevant work, Section 3 describes our approach for social network based information retrieval, Section 4 describes our experimental results on the CoRA

data set, and we finish in Section 5 with a discussion and outline for future work.

## 2. Relevant Work

One important aspect of this paper is finding new ways of ranking the clusters that are created using the network information available to us. Researchers have in the past used cluster ranking for various uses such as improving the precision of the original search results. Kurkland and Lee have discussed various methods for calculating the ranking scores, using the HITS algorithm [12] and PageRank [13], as well as other centrality scores mentioned in [14]. The links are called generation links, which are based on the probability assigned by the language model induced from one document, from the term sequence comprising another. Their clustering is query dependent and uses the approach of overlapping nearest-neighbor clusters. The clustering method described in this paper is not dependent on the query. Instead, we identify communities within a network ahead of time, making our approach faster at query time. In addition, the relationships that we use are less noisy since we are dealing with the data in a textual knowledgebase that already contains clear relations between the data. Yaltaghian and Chignell improve upon the original ranking score of web documents by re-ranking [29]. They use co-citations network and a number of other network measures to find the relationships between the data and then use centrality measures to improve the original ranking of the documents using the extra network information.

Cluster ranking has also been applied to many other applications such as mining a mailbox network [1]. This paper proposes a new algorithm, C-Rank, that creates a list of overlapping clusters and ranks them by using their integrated cohesion. It uses fuzzy clustering in which each data point can belong to more than one cluster, leading to many more clusters. The clusters are then ordered by their strengths.

As discussed in the previous section, search within the domain of bibliographic content has used relationships like citations which add to the user experience. We concentrate on using known and exploring new relationships and using those relationships to provide a ranked clustered view.

## 3. Cluster Based Information Retrieval

The approach taken in this paper is to retrieve clusters of contextually related documents rather than documents themselves. Context is a broad and ambiguous concept. In this paper, we define context as how documents relate to each other with respect to a specific relationship such as text-similarity or shared authors. These relations each impose a different network of documents. Given such a network, it is possible to retrieve clusters of documents rather than a ranked list of documents. This section

describes our approach to retrieving document clusters, given an existing network of documents.

The information retrieval process involves three steps, two of which are done as part of a pre-computation analysis:

1. **Creating networks:** For each relationship, create a network of documents, with links weighted by the semantics of the relationship.
2. **Clustering:** For each relationship network, data is pre-clustered into cohesive groups by finding communities in the relationship network.
3. **Ranking:** For each relationship, rank both the clusters from the relationship network and the elements within each cluster.

The remainder of this section describes our approach to these steps in more detail.

### 3.1 Creating Relationship Networks

The cluster based methodology put forth in this paper uses graph-based techniques to cluster the documents. We create a relationship network for a particular relationship between documents, where a relationship has a strength which defines how strongly the two documents are connected through that relationship.

We consider two types of general relationships: explicit and implicit. Explicit relations are those where there is a clear relationship between documents such as one document citing another or sharing an author, etc. They are generally discrete values and often binary: does the relation exist or not?

On the other hand, example of an implicit relationship is the relationship between two documents if their text is similar or if they share a keyword, etc. Implicit relations are often continuous values between 0 and 1. These are somewhat analogous to the similarity functions that are used by standard clustering methods.

Each type of relationship induces a relationship network, where the strength of the relationship is dependent on the semantics of the relationship. We have identified a small set of concrete categories of relationships that are generally applicable and presently discuss how to use each of these within the cluster based information retrieval methodology.

#### 3.1.1 Explicit Relations

In many corpora documents have either an author or a creator attribute. These can be used to create an explicit relationship between documents, where the number of shared authors/creators indicates the strength of the relationship between documents. If two documents have no shared author, then they have no relationship using this definition. This kind of relationship is undirected.

One other explicit relationship that we use in this work is that of citations: if one document cites another then they share a citing relationship. This kind of relationship is

binary and directed: either a document cites another or it does not. For the purposes of this paper, we consider these relations to be undirected and only keep track of whether there is a relation between two documents or not. This also means that a citation-relation can actually have a value of 2 if the two documents cite each other—a rare occurrence.

#### 3.1.2 Implicit Relations

The most basic of implicit relationships when dealing with textual knowledge-bases is the *text-similarity* of the content of the documents (either abstract, description or full content). We here use a standard bag-of-words vector representation as used in information retrieval [24]: for each word in an instance, calculate the *tfidf* score for that word. The *tfidf* score is short for term frequency (*tf*) inverse document frequency (*idf*), where  $tf(w) = \log(1 + w_{doc})$  and  $idf(w) = \log(N / N_w)$ , where  $w_{doc}$  is the number of times word  $w$  appears in a given instance,  $N$  is the size of the corpus and  $N_w$  is the number of instances that word  $w$  appears in. We represent these scores in a vector the dimensionality of all words seen in the corpus such that the  $i$ -th element in the vector represents the  $i$ -th word in the corpus. Such a vector is likely to be sparse for any given instance as it will only contain a subset of all words ever seen. The weight of the relationship between two documents is then defined as the cosine of their respective *tfidf* vectors [24]. Since this leads to many relationships between the documents (nearly a fully connected graph), we decided to take just those relationships whose score is in the top 20—in other words, only choose the top 20 connections from each document, thereby ensuring that each document will have at least a degree of 20 (possibly more, as a relation in the top 20 from document A to document B may not be in the top 20 of the relations from document B).<sup>1</sup> This relationship helps the user look at other textually similar articles that the user might have been interested in, but perhaps, the search keywords that the user entered were not contained in those other papers, but they are actually using techniques or doing work related to the one the user is interested in and that information will be more meaningful to the user.

Some attributes are categorical in nature such as keywords or general terms from an agreed-upon vocabulary. In that case, a document is tagged with a few of these terms and using text-similarity as just described is not appropriate. For attributes such as these we make use of the Jaccard similarity coefficient [9], which is defined as the size of the intersection of two sample sets, divided by the size of their union of the sample sets:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

---

<sup>1</sup> We also tried using the top 5 and top 10 relations to construct the clusters but using top 20 empirically gave the best clusters.

The Jaccard coefficient is a value between 0 and 1, and is only non-zero if two sets share at least one term in common. For documents that have at least one term in common, we create a keyword or term relationship with a strength equal to that of the Jaccard coefficient.

### 3.2 Clustering

One precondition for returning clusters of documents for a search query is the ability to create document clusters. The key idea is that we want clusters to be able to include documents that are contextually relevant even though they do not match the search query. The clusters therefore need to be created outside the set of documents that match the query. The approach taken here is that we consider different views of the corpus, where each view imposes a clustering specific and unique to that particular view. These can therefore be pre-computed. The views that are considered in this paper are the large networks that can be instantiated based on a particular relationship such as text-similarity or shared authors. Clustering networks is related to unsupervised learning as well as finding communities in large social networks. Because many textual knowledge-based are quite large, we need to use algorithms that are efficient. We base our clustering on an existing algorithm that is near-linear for sparse graphs. This generates too large clusters, which is a general problem for any clustering algorithm, and we therefore need to modify the algorithm to generate small clusters. We presently describe the basic algorithm and our modification to it.

#### 3.2.1 Community Finding Algorithm

A community, or cluster, or cohesive subgroup is a subset of individuals among whom there are relatively strong, direct, intense ties [20]. ‘Relative’ meaning that there are more ties among the cluster nodes themselves than there are among the cluster nodes and outside nodes.

The research literature contains numerous graph-theoretic methods of finding communities—for example, the spectral bisection method [5,22] and the Kenigham-Linh algorithm [10], both of which suffered from drawbacks such as dividing the network into two groups and not an arbitrary number [20]. Division into more than two groups was achieved by repeated bisection with no guarantee that the best division into three can be arrived at by finding the best division into two and then dividing one of those again.

Many modern algorithms for finding community structure use hierarchical clustering. One such method is the one suggested by Girvan and Newman that uses “edge betweenness” [20]. The betweenness of an edge is defined to be the number of geodesic (i.e. shortest) paths between vertex pairs that run along the edge in question, summed over all vertices. The algorithm involves calculating the betweenness of all edges in the network and removing the one with the highest betweenness, repeating the process until no edges remain. This is a very computationally

expensive algorithm, and a new bottom-up agglomerative hierarchical clustering algorithm has been proposed [18]. This is a greedy algorithm that starts with each vertex in its own cluster, and then amalgamates communities in pairs, choosing at each step, the pair whose amalgamation will lead to the maximum increase of a quantity called modularity. The modularity measure identifies clusters that have more edges connecting vertices within a cluster than across clusters. Let  $e_{ij}$  be the fraction of edges that connect vertices in group  $i$  to those in group  $j$ . Then the modularity is given by:

$$Q = \sum_i (e_{ii} - a_i^2), \text{ where}$$

$$a_i = \sum_j e_{ij}.$$

For every pair of communities that have an edge joining them we calculate the change in  $Q$  upon joining those communities, which is given by:

$$\Delta Q = e_{ij} + e_{ji} - 2a_i a_j$$

The algorithm will keep joining communities until the modularity decreases, thereby having an information-theoretic stopping criterion. The algorithm is significantly faster than the original top-down algorithm proposed; it takes  $O((m+n)n)$  to construct the complete dendrogram, and  $O(n^2)$  on a sparse graph. This is still too slow on very large networks and we therefore use an efficient implementation as discussed in [4]. The implementation exploits short-cuts in the optimization problem and uses efficient data structures for representing sparse matrices leading to considerable saving of both memory and time. The running time of the optimized implementation is  $O(md \log n)$ , where  $d$  is the depth of the dendrogram and the total degree of the vertices is  $2m$ . We use a modified version of the algorithm which supports weights as discussed in [19].

#### 3.2.2 Iterative Clustering

The main reason for using the above method is that it is both scalable and efficient. In many real-world networks, that are sparse, it runs essentially in linear time. One drawback of the algorithm is that it tends to generate relatively few clusters. For very large networks, such as those from large knowledge-bases, the algorithm generates quite large clusters ( $\geq 100$ , some being  $\gg 100$ ). It is problematic presenting such large clusters as a result of the search query as they will lead to a significant cognitive load on the user. Instead, we need to generate smaller clusters that are much more focused.

The primary objective of this paper is to explore the ranking of clusters and we therefore use a heuristic based on cluster-size here. Our approach in this paper is to form relatively small clusters. We limit the cluster size to 50 and therefore recursively apply the community finding algorithm until all clusters are of size  $\leq 50$ .

### 3.3 Ranking

The primary focus of this paper is to identify and rank clusters that are relevant to a search query. This is actually a two-fold problem: the first is ranking the clusters and the second ranking the elements within the cluster such that the most relevant or important documents of a cluster show up at the top. This will help the user identify the clusters that are relevant to the current information need.

The ranking algorithm needs to favor clusters that are relevant, but it also needs to be pragmatic. Particularly, consider the case of returning a cluster of size 50 or a cluster of size 10. All else being equal, the smaller cluster is likely more focused and is likely to be of more interest to the user. The ranking function therefore needs to take not only the relevance of a cluster into account but also the size.

We first consider how to identify and rank clusters that are relevant to the user's search query. First, we define a *relevant cluster* as a cluster which contains at least one document that matches the search query. Second, we need to create a score for the cluster. This rank involves a combination of the relevancy score of the cluster as well as the size of the cluster as described above. We here focus on two types metrics to generate a cluster relevancy score—one is text-based and one is based on social network analysis. The rest of the section describes these two ranking methods.

#### 3.3.1 Text-based Cluster Ranking

There are various ways of using the text of documents within a cluster to generate a relevancy score. For example, one could use the average matching score, a matching score for the combined text of all the documents in the cluster or the ratio of documents in a cluster that matched the query. We here use the first approach: To calculate the 'cluster relevancy score' we take average query match score of all the documents within a cluster.<sup>2</sup> We also explored another text based metric where we just took the number of overlapping search results with the cluster elements to compute the score. It gave us almost the same results as the one above.

Nodes within a cluster are ranked based on their score.

As mentioned above, the size of the cluster is an input for rank calculation to bias towards cluster sizes that are more manageable for a user. Specifically, we here bias towards

clusters of size 10, which we consider to be the *ideal cluster size*.<sup>3</sup>

We use a kernel-based density function to bias towards the ideal cluster size to achieve a good tradeoff between the initial cluster relevancy score and the final ranking score. For example, we need to ensure that the bias is not too strong such that any cluster of size 10 will outweigh any cluster of size 9 or 11. On the other hand, we also need to ensure that a single perfectly matching document in a large cluster does not outweigh a somewhat good match in a smaller cluster (where the average score might lead one to choose the larger cluster). We note that because the ideal cluster size is not 25 (half of 50, which is the maximum cluster size), the density function need to be asymmetric—we need to shrink the bias for clusters smaller than the ideal size differently than clusters larger than the ideal size. The final ranking function was derived empirically by balancing these trade-offs.

The final rank of the text-based metric is as follows:

Let  $s$  be the size of the cluster and  $i$  be the ideal cluster size.

Then, if  $s \leq i$ :

$$R = score * e^{\frac{-(s-i)^2}{1000}}$$

Otherwise:

$$R = score * e^{\frac{-(s-i)^2}{100}}$$

#### 3.3.2 Centrality-based Cluster Ranking

The position of a web-document within the web of related web-documents and the links between them has been an important measure used to improve the ranking of web-based search engines. For example, the most significant ranking algorithms in the field of information retrieval such as PageRank [2] and HITS [11] use link-based ranking measures. PageRank uses hyperlink structure formed by the web pages on the World Wide Web to calculate a PageRank score. The HITS algorithm, an adaptation of PageRank, computes two types of scores—hubs and authorities—to help with the rank calculation.

These link-based ranking measures have also been applied to non web-based search [12, 13]. Yaltaghian and Chignell use various centrality based measures to re-rank documents returned by a search [29]. They use Google results to construct the co-citation network. The different centrality measures are then applied to the co-citation frequency matrix.

---

<sup>2</sup> We use the Lucene text search engine to compute this [16].

---

<sup>3</sup> Size 10 was chosen in an *ad hoc* fashion. Other sizes can easily be used, but we felt that 10 was a good manageable size for a user to quickly get a feeling for whether a cluster would be of interest.

The intuition behind using centrality is that the documents that are central are more representative of what a cluster covers than documents on the periphery (low centrality). Therefore, by finding how relevant the most central element(s) of the cluster is to a search query, we can judge the relevance of the cluster.

Centrality measures the contribution of document position to its importance, influence and prominence [29]. The literature coming out of social network analysis has defined many types of centrality metrics, but the metrics that are of particular interest in this paper are betweenness centrality [6] and information centrality [25]. Betweenness centrality is a measure of the number of times a node lies on a geodesic path that link pairs of nodes for all nodes in a graph [29]. Betweenness centrality ignores the fact that vertices with large degree are more likely to be used as shortest path than others [27]. Second, betweenness does not necessarily identify the vertices that can easily reach all other vertices in a graph such as the closeness degree. The information centrality measure defined by Stephenson and Zelen [25] takes these points into consideration. To calculate information centrality it considers the combined path from one actor to another, by taking all paths, including the geodesics, and assigning them weights, the inverse of the length of the paths that are combined. The weights assigned to the path are combined such that the information in the path being combined is maximized. Geodesics are usually given weights of unity, while paths with length longer than the geodesic length receive smaller weights, based on the information that they contain. Information of a node is then calculated as the harmonic average of the information for the combined path from the node to all other nodes.

To calculate information centrality, we first create a  $g \times g$  matrix  $A$ , which has diagonal elements

$$a_{ii} = 1 + \text{sum of values for all lines incident to } n_i$$

and off-diagonal elements

$$a_{ij} = \begin{cases} 1 & \text{if nodes } n_i \text{ and } n_j \text{ are not adjacent} \\ 1 - x_{ij} & \text{if nodes } n_i \text{ and } n_j \text{ are adjacent} \end{cases}$$

Where,  $x_{ij}$  is the value of the tie between actors  $i$  and  $j$ . We then take the inverse of  $A: C = A^{-1}$ , which has elements  $\{c_{ij}\}$ . We also need two intermediate quantities; the first is the trace or the sums of diagonals of elements of the matrix  $C: T = \sum_{i=1}^g c_{ii}$  and second is the row sum (which is same for all rows):  $R = \sum_{j=1}^g c_{ij}$ . With these two quantities, the information centrality index for actor  $i$ , is calculated as:

$$C_i(n_i) = \frac{1}{c_{ii} + (T - 2R)/g}$$

Finally, we calculate the information indices for each node:

$$C'_i(n_i) = \frac{C_i(n_i)}{\sum_i C_i(n_i)}$$

Calculating the information centrality indices of nodes in a graph is expensive. We note that although this has to be done for every cluster, the clusters are small ( $\leq 50$ ) and therefore the overall computation is relatively inexpensive.

We compute the ‘cluster relevancy score’ by finding the node in a cluster with the highest information centrality that also was a match to the search query.

Nodes in a cluster are ranked on the information centrality value of that node in the cluster.

For the information centrality method, the final rank is given by:

$$R_{ic} = \text{score}_{ic} * \frac{(s-d)}{s}$$

where  $s$  is the cluster size as before and  $d$  is the rank of the document with the highest information centrality that was also a match to the search query.

### 3.4 Returning query results

When the user enters a search query, the records that match the query are returned using Lucene [16]. Once we receive the matching documents we rank our clusters depending on the ranking algorithms (discussed in Section 3.3) selected by the user. The ranked clusters, with both the individual element score and a score for each cluster are then presented to the user.

## 4. Experimental Results

We here present an empirical and subjective evaluation of the cluster based information retrieval methodology introduced in this paper. The objective of this evaluation is to verify that returning clusters and using the centrality-based cluster rank does indeed provide greater insight into the results than using a text-based metric. The approach has been evaluated on three data sets within a prototype implementation of a search engine—screenshots of real queries are provided as part of the results.

### 4.1 Data

We have evaluated our approach on three datasets and present results from one of these data sets—the CoRA dataset—as it is more relevant to readers of this conference. The CoRA data set contains 4240 computer science research papers [17]. It includes the full citation graph as well as the name of authors. The attributes of the data that are relevant to this study is id, citations, author, title and abstract.

# Relevance Ranker

powered by  
Fetch Technologies

Search Query : **Gibbs sampler** Ranking Algorithm : **Information-Centrality** Data Set Used : **cora**

citation first >>		sharedauthor first >>		textlink-top-20 first >>	
<b>1</b>	<b>0.870175</b>	<b>1</b>	<b>0.962963</b>	<b>1</b>	<b>1</b>
0.106828	S.Sahu .. Bayesian Estimation and Model Choice in Item Response Models ..	0.048196	G.Roberts J.Rosenthal M.Cowles .. Possible biases induced by MCMC convergence diagnostics ..	0.091343	G.Roberts J.Rosenthal .. On convergence rates of Gibbs samplers for uniform distributions ..
0.106828	G.Roberts S.Sahu .. On Convergence of the EM Algorithm and the Gibbs Sampler SUMMARY ..	0.047752	G.Roberts J.Rosenthal .. On convergence rates of Gibbs samplers for uniform distributions ..	0.088505	G.Roberts S.Sahu .. Rate of Convergence of the Gibbs Sampler by Gaussian Approximation SUMMARY ..
0.079934	E.Alan .. Hierarchical Spatio-Temporal Mapping of Disease Rates ..	0.047752	G.Roberts J.Rosenthal .. Convergence of slice sampler Markov chains ..	0.088502	C.Geyer J.Hobert .. Geometric Ergodicity of Gibbs and Block Gibbs Samplers for a Hierarchical Random Effects Model ..
	<a href="#">See all 14 &gt;&gt;</a>		<a href="#">See all 27 &gt;&gt;</a>		<a href="#">See all 15 &gt;&gt;</a>
<b>2</b>	<b>0.209957</b>	<b>2</b>	<b>0.743044</b>	<b>2</b>	<b>0.109684</b>
0.102869	K.Mengersen R.Tweedie .. Rates of convergence of the Hastings and Metropolis algorithms ..	0.5	I.Mckeague P.Greenwood W.Wefe .. Outperforming the Gibbs sampler empirical estimator for nearest neighbor random fields ..	0.181548	C.Robert D.Titterington T.Ryd .. Convergence controls for MCMC algorithms with applications to hidden Markov chains ..
0.085768	L.Tierney R.Smith .. EXACT TRANSITION PROBABILITIES FOR THE INDEPENDENCE METROPOLIS SAMPLER ..	0.5	I.Mckeague P.Greenwood W.Wefe .. Von Mises type statistics for single site updated local interaction random fields ..	0.179802	A.Jouyenosas C.Robert M.Gruet .. MCMC control spreadsheets for exponential mixture estimation ..
0.081853	G.Roberts R.Tweedie .. Exponential Convergence of Langevin Diffusions and Their Discrete Approximations ..			0.160263	C.Robert K.Mengersen .. Reparameterisation Issues in Mixture Modelling and their bearing on MCMC algorithms ..
	<a href="#">See all 14 &gt;&gt;</a>		<a href="#">See all 4 &gt;&gt;</a>		<a href="#">See all 7 &gt;&gt;</a>
<b>3</b>	<b>0.086056</b>	<b>3</b>	<b>0.256218</b>	<b>3</b>	<b>0.087253</b>
0.104738	A.George C.Robert D.Robb F.Pa .. MCMC CONVERGENCE DIAGNOSTICS: A "REVIEW" ..	0.25	J.Lafferty .. GIBBS-MARKOV MODELS ..	0.139593	H.Cai .. EXACT BOUND FOR THE CONVERGENCE OF METROPOLIS CHAINS ..
0.092452	G.Roberts S.Brooks .. Assessing Convergence of Markov Chain Monte Carlo Algorithms ..	0.25	J.Lafferty .. GIBBS-MARKOV MODELS ..	0.134078	J.Rosenthal .. Theoretical rates of convergence for Markov chain Monte Carlo ..
0.093013	C.Robert .. MCMC Specificities of Latent Variable Models 1 ..	0.25	J.Lafferty .. GIBBS-MARKOV MODELS ..	0.119809	K.Mengersen R.Tweedie .. Rates of convergence of the Hastings and Metropolis algorithms ..
	<a href="#">See all 14 &gt;&gt;</a>				

Figure 1: Search result for 'Gibbs sampler'. Each column is a ranked set of clusters for a particular relationship

After we collected all the data from the dataset the next step is identifying a set of implicit and explicit links/relationships that will be useful for clustering the data. The explicit relationships used in this work are citations and shared-authors/creators. The implicit relation used is the text-similarity induced from abstracts. For each of these relationships, we create a network and use the community finding algorithm (Section 3.2) to create the clusters.

## 4.2 Evaluation Methodology

The evaluation of the effectiveness of the cluster based information retrieval methodology to improve the user's overall search experience is very challenging. The primary obstacle to this is that the methodology is not directly related to improving just one factor in the search like the ranking precision or quality of the clusters produced, or other concrete measures that are easily quantifiable.

Previous related work include web document clustering, cluster ranking methods, methods to improve the ranking or clustering of results in digital library search engines. The work presented here is sufficiently different from these, making any direct comparison difficult.

For example, research literature in the field of web document clustering have compared the various clustering algorithm on efficiency and quality of the clusters produced [30]. In addition, this work clusters results matching the search query and not all documents. Work in the area of the digital libraries has used ontology such as MeSH to improve clustering quality [6]. The main goal of this area

is to measure the quality of the clusters. Although it is important that we get clusters of good quality it was also important to us that we retrieve clusters that are contextually relevant in a broader sense and not only with respect to documents that match the search query. In addition, we also considered the "user friendliness" of a cluster and used heuristics to limit the cluster size.

Relationships other than pure text have already been recommended [26] and used in digital library search engines [3, 7, 15, and 23]. However, the result of these search engines is a ranked list of documents and not clusters such as what are returned in this work.

The main goal of this methodology is to not only help users find relevant information but also be aware of the contextually relevant information. Because context is subjective, our evaluation measure is more anecdotal than based on performance metrics. Specifically, we have evaluated the system on the CoRA data set using various queries with the goal of finding relevant information as well as finding relevant clusters. Because we have no other systems to compare against, we compare and contrast the two ranking metrics defined in Section 3.3.

## 4.3 Results

We found some very interesting results by comparing the two metrics for ranking the clusters that show us that information centrality can prove to be a very useful metric for characterizing a cluster. We present here the key result using the search query 'Gibbs sampler' and note that the differences in ranking metrics (within and across clusters)

Text-Sim /Centrality Score	Author/Creator/PI	Title	Abstract/Description
0.104738	A.George C.Robert D.Robb F.Paris K.Mengersen S.Knight	MCMC CONVERGENCE DIAGNOSTICS: A "REVIEWwww"	This review paper aims at presenting the various ..
0.093013	C.Robert	MCMC Specificities of Latent Variable Models 1	We derive from analyses of several specific laten..
0.092452	G.Roberts S.Brooks	Assessing Convergence of Markov Chain Monte Carlo Algorithms	We motivate the use of convergence diagnostic tec..
0.090075	C.Robert D.Titterington T.Ryd	Convergence controls for MCMC algorithms with applications to hidden Markov chains	In complex models like hidden Markov chains the ..
0.08467	D.Chauveau J.Diebolt	MCMC CONVERGENCE DIAGNOSTIC VIA THE CENTRAL LIMIT THEOREM	Markov Chain Monte Carlo (MCMC) methods as intro..
0.083241	A.Jouyenosas C.Robert M.Gruet	MCMC control spreadsheets for exponential mixture estimation	Bayesian inference for exponential mixtures is pr..
0.07901	P.Green S.Richardson	On Bayesian analysis of mixtures with an unknown number of components Summary	New methodology for fully Bayesian mixture analys..
0.065124	B.Yu P.Mykland	Looking at Markov Samplers through Cusum Path Plots: a simple diagnostic idea	In this paper we propose to monitor a Markov cha..
0.065088	G.Roberts P.Dellaportas S.Brooks	An Approach to Diagnosing Total Variation Convergence of MCMC Algorithms	We introduce a convergence diagnostic procedure f..
0.064748	A.Monfort C.Robert M.Billio	THE SIMULATED LIKELIHOOD RATIO (SLR) METHOD	Resume. Une methode de simulation fondee sur des ..
0.059501	C.Guihenneucjouyau C.Robert	Discretization of continuous Markov chains and MCMC convergence assessment	We show in this paper that continuous state space..
0.042811	C.Bielza D.Insua P.Muller	Decision Analysis by Augmented Probability Simulation	We provide a generic Monte Carlo method to find t..
0.039684	K.Roeder L.Wasserman R.Carroll	FLEXIBLE PARAMETRIC MEASUREMENT ERROR MODELS	Inferences in measurement error models can be sen..
0.035845	L.Bauwens P.Giot	A Gibbs Sampling Approach to Cointegration	This paper reviews the application of Gibbs sampl..

Copyright © 2008, Fetch Technologies, Inc. All rights reserved.

**Figure 2: Ranking within a cluster using the centrality metric for ranking based on ‘Gibbs sampler’ query. The scores on the left are information centrality scores for the documents.**

as presented using this query are prominent across a set of queries (but not all, as search results can be quite similar). In this particular query, the search engine returns a ranked set of clusters for each of the three relations we defined on the CoRA data set. Figure 1 shows the result of this query using the Information Centrality metric.

We here contrast the ranking methods used to rank documents within clusters. Focusing on the ranked clusters from the citation relationship graph (left column in Figure 1), we see that the top document (most central document) in the third cluster is titled ‘MCMC CONVERGENCE DIAGNOSTICS: A REVIEWwww’<sup>4</sup> and that the second most central paper is titled ‘MCMC Specificities of Latent Variable Models 1’, suggesting that the cluster is not as much about Gibbs sampling specifically as it is about MCMC. The fully ranked set of documents for that cluster is shown in Figure 2. When we use same query with the text-based ranking method, the cluster shows up as the fourth cluster with an entirely different ranking of documents as shown in Figure 3. As mentioned above, both the clusters had a very high rank in the cluster list(IC rank: 3, text-based rank: 4), confirming that the cluster itself is very relevant for the given query. As we look carefully at the contents of the cluster itself we find that the central topic of the cluster is the ‘Markov Chain Monte Carlo’ method (MCMC). This is exactly what the information centrality metric highlights by ranking of the documents containing MCMC higher whereas the text similarity ranks

documents containing “Gibbs sampler” higher because they retrieved a higher text similarity score to the search query. This ranking of the documents within the cluster may lead

the user to believe that the documents in the cluster primarily discuss Gibbs sampling when in fact they do not.

## 5. Discussion

We presented a methodology for cluster based information retrieval of documents in a textual knowledgebase. The intuition for this approach is that returning clusters of contextually related information provides users with a situational and contextual awareness of the search results rather than a ranked list of only those documents that match the query.

We described the process for representing a text corpus as a set of relationship networks (views of the data) and clustering these relationship graphs into small, focused clusters that contain contextually related documents. These clusters are then returned as a ranked set of clusters for each relationship such that a user can quickly find the cluster that is most relevant. Each cluster shows the top 3 most relevant documents within the cluster, making it easier and quicker to navigate these sets of clusters.

The key question we sought to answer in this paper was how to rank such clusters and we proposed two ranking methods: one is text-based similar to standard information retrieval processes and the second is based on social network analysis and uses the information centrality metric to identify central documents in a cluster and rank clusters based on whether the central documents of a cluster match the search query.

This work is a direction in information retrieval that has not received much attention, making it hard to compare against existing methods. We instead compared and contrasted these metrics on a specific query and showed that the information centrality ranking method put central documents at the top of a within-cluster rank and gives a

<sup>4</sup> This is the title as extracted as part of the CoRA research project.

Text-Sim /Centrality Score	Author/Creator/PI	Title	Abstract/Description
0.28423914	L.Bauwens P.Giot	A Gibbs Sampling Approach to Cointegration	This paper reviews the application of Gibbs sampl..
0.22037023	B.Yu P.Mykland	Looking at Markov Samplers through Cusum Path Plots: a simple diagnostic idea	In this paper we propose to monitor a Markov cha..
0.10968413	C.Robert D.Titterington T.Ryd	Convergence controls for MCMC algorithms with applications to hidden Markov chains	In complex models like hidden Markov chains the ..
0.09456637	G.Roberts P.Dellaportas S.Brooks	An Approach to Diagnosing Total Variation Convergence of MCMC Algorithms	We introduce a convergence diagnostic procedure f..
0.09267617	C.Robert	MCMC Specificities of Latent Variable Models 1	We derive from analyses of several specific laten..
0.0775584	A.Jouyenos C.Robert M.Gruet	MCMC control spreadsheets for exponential mixture estimation	Bayesian inference for exponential mixtures is pr..
0.047283184	D.Chaudeau J.Diebolt	MCMC CONVERGENCE DIAGNOSTIC VIA THE CENTRAL LIMIT THEOREM	Markov Chain Monte Carlo (MCMC) methods as intro..
0.016129777	A.Monfort C.Robert M.Billio	THE SIMULATED LIKELIHOOD RATIO (SLR) METHOD	Resume. Une methode de simulation fondee sur des ..
0.0	A.George C.Robert D.Robb F.Paris	MCMC CONVERGENCE DIAGNOSTICS: A "REVIEWwww"	This review paper aims at presenting the various ..
0.0	K.Mengersen S.Knight		
0.0	P.Green S.Richardson	On Bayesian analysis of mixtures with an unknown number of components	New methodology for fully Bayesian mixture analys..
0.0		Summary	
0.0	K.Roeder L.Wasserman R.Carroll	FLEXIBLE PARAMETRIC MEASUREMENT ERROR MODELS	Inferences in measurement error models can be sen..
0.0	G.Roberts S.Brooks	Assessing Convergence of Markov Chain Monte Carlo Algorithms	We motivate the use of convergence diagnostic tec..
0.0	C.Guihenneucjouy C.Robert	Discretization of continuous Markov chains and MCMC convergence assessment	We show in this paper that continuous state space..
0.0	C.Bielza D.Insua P.Muller	Decision Analysis by Augmented Probability Simulation	We provide a generic Monte Carlo method to find t..

Copyright © 2008, Fetch Technologies, Inc. All rights reserved.

**Figure 3: Ranking within a cluster using the text-based metric for ranking based on the ‘Gibbs sampler’ query. The scores on the left are the text similarity scores.**

better sense of what a cluster covers than when using the text-based metric.

We note that the work here leaves many open questions for improving the way relationship graphs are created as well as for many new ranking methods that are network-centric rather than text-based as is traditionally the case. In particular, we used the information centrality metric in a simple way here but believe that many improvements can still be done in a similar way to how text-based engines have tweaked and improved upon the tfidf calculations.

Another avenue worth further exploration is research into how to create more contextually rich relationships rather than the simpler relationships considered in this paper, including relationships that are defined only for a particular query. This also includes exploration of relationships that are already present in the literature, such as the common citation x inverse document frequency (CCIDF).

We noted early on that current clustering algorithms tend to generate very large clusters. The approach taken here was to iteratively cluster sub-graphs until the generated clusters were relatively small. Ideally, there should be an objective stopping criterion for this based on a cohesiveness metric or other metric for how ‘focused’ a cluster is.

Finally, the search experience based on this approach is not yet as friendly as one would like for a real system. There is still significant cognitive load on the user to understand what a cluster is really about, even with the help of information centrality. One clear question, which is already an active research area, is how to summarize or describe a cluster—especially with respect to how it differs from other clusters. Providing such a description may ease the cognitive load of the user.

## 6. Acknowledgments

This work was sponsored in part by the Department of Energy under award number DE-FG02-07ER84706. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Department of Energy or the U.S. Government.

## 7. References

- [1] Bar-Yossef, Z., I. Guy, R. Lempel, Y. S. Maarek, V. Soroka, Cluster Ranking with an Application to Mining Mailbox Networks. Sixth International Conference on Data Mining, 2006. ICDM '06.
- [2] Brin, S. and L. Page. The anatomy of a large-scale hypertextual Web search engine. Proceedings of the seventh International World Wide Web Conference 1998, pp.107-117.
- [3] Citeseer home page, <http://citeseer.ist.psu.edu/>
- [4] Clauset, A., M. E. J. Newman and C. Moore, "Finding Community Structure in Very Large Networks", cond-mat/0408187, 2004.
- [5] Fiedler, M. Algebraic connectivity of graphs. Czech.Math. J. 23, 298—305, 1973.
- [6] Freeman, L. C., Centrality in social networks: Conceptual clarification. Social Networks, 1:215–239, 1979.
- [7] Giles, C. L., K.D. Bollacker, and S. Lawrence, CiteSeer: An Automatic Citation Indexing System.
- [8] Google scholar homepage, <http://scholar.google.com/>
- [9] Jaccard, P. Étude comparative de la distribution florale dans une portion des Alpes et des Jura. Bulletin del la

- Société Vaudoise des Sciences Naturelles 37, 547-579, 1901.
- [10] Kernighan, B. W. and S. Lin, An efficient heuristic procedure for partitioning graphs. *Bell System Technical Journal* 49, 291—307, 1970.
- [11] Kleinberg, J. Authoritative sources in a hyperlinked environment. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, p. 668-677, ACM Press, New York, 1998.
- [12] Kurkland, O. and L. Lee, PageRank without hyperlinks: Structural re-ranking using links induced by language models. In *Proceedings of SIGIR*, pages 306-313, 2005.
- [13] Kurland, O. and L. Lee, Respect my authority! HITS without hyperlinks, utilizing cluster-based language models. In *Proceedings of SIGIR*, pp. 83--90, 2006.
- [14] Liu, X. and W. B. Croft. Cluster-based retrieval using language models. In *Proceedings of SIGIR*, pp. 186—193, 2004.
- [15] Live Search, <http://search.live.com/academic/>
- [16] The Apache Lucene homepage, <http://lucene.apache.org/java/docs/index.html>
- [17] McCallum, A., Nigam, K., Rennie, J., and Seymore, K. Automating the construction of internet portals with machine learning., *Information Retrieval*, 3(2):127—163, 2000.
- [18] Newman, M. E. J. Fast algorithm for detecting community structure in networks. Preprint cond-mat/0309508, 2003.
- [19] Newman, M. E. J. Analysis of weighted networks, submitted to *Phys. Rev. E*, 2003.
- [20] Newman, M. E. J. Detecting community structure in networks, *Eur. Phys. J. B* 38, 321–330, 2004.
- [21] Office of Scientific & Technical Information website home page, <http://www.osti.gov/>
- [22] Pothén, A., H. Simon, and K.-P. Liou, Partitioning sparse matrices with eigenvectors of graphs. *SIAM J. Matrix Anal. Appl.* 11, 430-452, 1990.
- [23] Rexa home page, <http://rexa.info/>
- [24] Salton, G., and M. J. McGill, *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., 1983.
- [25] Stephenson, K. and M. Zelen, Rethinking centrality: Methods and examples. *Social Networks*, 11:1–37, 1989.
- [26] Strohman, T., W. B. Croft, D. Jensen, Recommending Citations for Academic Papers, *Proceedings of the international ACM SIGIR conference on Research and development in information retrieval*, 2007.
- [27] Wasserman, S. and K. Faust, *Social Network Analysis : Methods and Applications*, pp. 192-197, 1994.
- [28] Wikipedia link for various academic search engines, [http://en.wikipedia.org/wiki/Academic\\_databases\\_and\\_search\\_engines](http://en.wikipedia.org/wiki/Academic_databases_and_search_engines)
- [29] Yaltaghian, B. and M. H. Chignell, Re-ranking search results using network analysis: A case study with Google., 2002.
- [30] Zamir, O., and O. Etzioni, Web document clustering: a feasibility demonstration, *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 46-54, 1998.