REPORT DOCUMENTATION PAGE

AFRL-SR-AR-TR-04-

0196

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED
24-March-2004	Final Technical Report	Sept 1, 2003 - Feb 29, 2004

4. TITLE AND SUBTITLE
(STTR PH 1) Centering Resonance Analysis: A Superior Data Mining Algorithm for Textual Data Streams

5a. CONTRACT NUMBER
F49620-03-C0082

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Kevin Dooley, PhD
Steven Corman, PhD
Dan Ballard

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Crawdad Technologies, LLC
5412 W. Harrison Ct.
Chandler, AZ 85226-1935

Arizona State University
PO Box 874706
Tempe, AZ 85287-4706

8. PERFORMING ORGANIZATION REPORT NUMBER
STTR 001-1

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
USAF, AFRL
Air Force Office of Scientific Research
4015 Wilson Blvd, Room 713
Arlington, VA 22203-1954

10. SPONSOR/MONITOR'S ACRONYM(S)
AFOSR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

20040423 034

13. SUPPLEMENTARY NOTES

14. ABSTRACT Report developed under STTR contract for topic AF03T011 of STTR Program Solicitation FY 2003. Current knowledge-based intelligence systems do not perform well with streaming media because of performance shortcomings and an inability to work in storage constrained environment. The purpose of this research was to demonstrate that Centering Resonance Analysis (CRA) provides a superior approach to performing text mining under storage constraints. CRA is a radically different approach to modeling text compared to traditional word frequency-based approach. The project demonstrated that a CRA-based approach is superior to a word frequency approaches: up to 15 times better in identifying relevant documents, and up to 5 times greater precision in topic tracking experiments. A CRA data structure requires one-third the space required for raw compressed text, and can execute on a typical desktop computer. Future R&D efforts will focus on commercializing a product with applications to government and commercial business processes.

15. SUBJECT TERMS
STTR Report, streaming media, text mining, text analysis, information retrieval, knowledge management, ad hoc retrieval, topic detection and tracking, communication analysis

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dan Ballard
Unclassified	Unclassified	Unclassified	None	82	19b. TELEPHONE NUMBER (include area code) (480) 615-8543

Centering Resonance Analysis: A Superior Data Mining Algorithm for Textual Data Streams

Prepared by:
Crawdad Technologies, LLC
5412 W. Harrison Ct.
Chandler Arizona

Report No. STTR 001-1

For US Air Force Office of Scientific Research
Arlington, VA 22203

CDRL 0001AF
Final Technical Report
STTR Phase I

Contract Number F49620-03-C-0082

March 24, 2004

THIS PAGE INTENIONALLY LEFT BLANK

Table of Contents

List of Figures

List of Tables

Foreword

This document is the final report for US Air Force Contract F49620-03-C-0082 titled *Centering Resonance Analysis: A Superior Data Mining Algorithm for Textual Data Streams (STTR Phase I)*. This work was performed by Crawdad Technologies, LLC, a Chandler Arizona small business. Arizona State University was the primary subcontractor for this effort. Work was performed for the US Air Force Office of Scientific Research (AFOSR), Arlington VA. The AFOSR sponsor for this research was Dr. Robert Herklotz. The period of work performance was from September 2003 through March of 2004.

Acknowledgements

The work described in this report was performed by Crawdad Technology, LLC personnel and personnel from the Arizona State University Software Factory. The research team included:

Crawdad Technologies, LLC

Mr. Kevin Dooley, PhD

Mr. Steven Corman, PhD

Mr. Dan Ballard

Mr. Ballard served as the principal investigator for the research effort.

Arizona State University Software Factory

Mr. Steve Gordon, PhD

We also wish to thanks Dr. Robert Herklotz for his sponsorhip of the project at AFOSR; Mr. Peter LaMonica and Dr. Barrey McKinney from AFRL for their valuable interaction during the project; Software Factory programmers Sid Shah, Tapan Sanghvi, Yin Ding, and Akhil Pai for their excellent technical work on the project; Mr. Ken Soucy for his accounting expertise; and Mr. Dan O'Neill and Mrs. Sharon Ballard for their support in developing our commercial relationships.

Section 1. Executive Summary

Information analysts face an ever increasing challenge in dealing with information overload or information glut. There is simply too much information to access, analyze and act on. A significant amount of this information comes in the form of streaming media. Current knowledge-based intelligence systems do not perform well in this domain, however. The underlying technology most used for data mining of textual data is overly simplistic. Additionally, because these systems store all incoming raw data, storage problems are created. Data analysis and mining algorithms are needed that minimize storage capacity requirements, facilitate fast and accurate information retrieval and knowledge discovery, and determine the relevancy of an item as it is acquired from a data stream. The purpose of this Phase I research was to demonstrate that Centering Resonance Analysis (CRA) provides a superior approach to performing text mining under storage constraints.

CRA represents the text of a document as a network, and is thus a radically different approach to modeling text compared to a traditional word frequency-based approach. In a CRA network, nodes represent words while edges indicate discursive connectivity between words. A CRA network is a data model that can be used to implement efficient and effective information retrieval and text mining systems. Within a CRA network, CRA generates a measure of word importance called *influence* which quantifies the degree to which a word creates *coherence* in a text. CRA uses resonance as a measure of discursive (structural) similarity between two CRA networks. Resonance can be used as a high-precision metric for determining the relevancy of an incoming text item relative to a user's interests. Highly relevant documents can be stored for later retrieval and analysis while irrelevant documents are discarded.

This project established two primary technical objectives to determine feasibility and measure performance of a CRA-based approach:

1. Compare how well CRA determines relevancy of incoming news articles, relative to the traditional frequency-based approach.
2. Determine a data design for a CRA network that minimizes storage requirements, and measure the computational requirements associated with CRA.

With respect to the first objective, we have shown that a CRA-based approach is superior to a word frequency approaches along several dimensions of performance. We examined numerous metrics within an experimental design that tested performance against two corpora that differed in scope of content and average length of texts. We experimented with both a single (tracking) and multiple (ad hoc retrieval) thresholds for determining relevancy across topic query statements. We found that a relevancy metric based on word and word-pair influence was three times more strongly correlated with a human judgment of relevancy than metrics based on word frequency. A binary decision rule (relevant, irrelevant) based on CRA had a miss rate 15 times smaller than one based on word frequency. In the tracking experiments, the word-pair influence metric had three to five times greater precision than the frequency metrics in the focused short-text corpus. Both types of metrics had similar performance in retrieval and tracking within the less focused, longer-text corpus, but CRA-based metrics had higher precision where it mattered most; i.e., in the first several dozen documents retrieved.

Regarding the second task, we designed and implemented a CRA data structure that requires one-third the space of that required by the compressed raw text. A CRA-based text mining system was shown to only require the memory and computation speed of typical desktop computers.

In conclusion, we have demonstrated that CRA is a superior method for mining textual data. This is a significant result because most existing commercial and research applications are based on techniques using word frequency. By implication, substituting CRA-based metrics for frequency metrics should improve the performance of existing text analysis systems. We have shown that CRA is scalable to meet the computational and memory requirements of real-world government and commercial applications, and can effectively operate in a storage-constrained environment.

Further research and development is needed to commercialize the application, and this will be the focus of our Phase II proposal. In particular, we will develop a system that is capable of taking streaming data from numerous sources and fusing them into a single "operational" picture of the events in question. This work will create enhancements to CRA that optimize its performance for tracking new texts relative to a topic statement of interest, and that create insightful metrics and representations that give an analyst a deep view of the dynamic content of these different text streams. Phase III commercialization will be aimed at "voice of the cus-

tomer" and intelligence applications in commercial and government markets.

Section 2. Introduction

2.1. Identification and Significance of the Opportunity

The Air Force has played an important role in developing technology and applications in information retrieval and knowledge management, most recently through the High Performance Knowledge Bases program and the Rapid Knowledge Formation program. The Air Force CIO goals include "providing all AF members with on-demand access to all information they need to do their jobs" [1]. An increasing amount of such information comes in the form of streaming media [2]. Current knowledge-based intelligence systems do not perform well in this domain, however. The underlying technology most used for data mining of textual data is overly simplistic. Additionally, because these data mining systems store all incoming raw data, storage problems are created. Data mining algorithms are needed that minimize storage capacity requirements, facilitate fast and accurate information retrieval and knowledge discovery, and determine the relevancy of an item as it is acquired from a data stream.

The founders of Crawdad Technologies have been involved over the last six years in the development of a novel text mining system using a technology called Centering Resonance Analysis (CRA). CRA models textual data and measures similarity between texts. CRA can be used as both a model for storing textual data, and a mechanism for determining relevance of a text relative to a query or topic statement. CRA is grounded in a theory of human discourse, and contrasts radically with existing approaches involving term frequency or semantic analysis. CRA creates a rich, high-precision model of text, while eliminating the need for complicated semantic rules, training sets, or corpora. As Figure 1 shows, CRA provides rich information content and is easy to implement.

CRA represents a text as a network. In the resulting *CRA network*, nodes represent words while edges indicate discursive connectivity between words. A CRA network is a data model that can be used to implement efficient and effective information retrieval and text mining systems. CRA uses *resonance* as a measure of discursive (structural) similarity between two CRA networks. Resonance can be used as a high-precision metric for determining the relevancy of an incoming text item relative to a user's interests. Highly relevant documents can be stored for later retrieval and analysis while irrelevant documents are discarded.

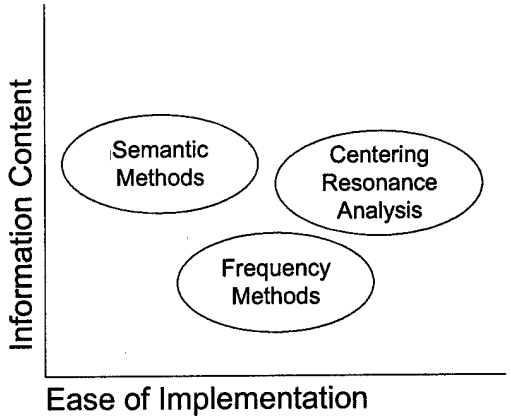

Figure 1. CRA Technology Map

Significant progress has been made in developing CRA and making it available for broader usage. As of the date of this report, Crawdad Technologies has released Crawdad 1.1, a desktop solution that performs knowledge processing, retrieval, and mining, for beta testing. Crawdad founders Corman and Dooley are authors of four pending patents concerning CRA. These patents are owned by Arizona State University, but Crawdad Technologies has negotiated exclusive rights for development and licensing of CRA. The theory underlying CRA has been described in *Human Communication Research* and *Management Communication Quarterly*, two of the top academic journals in the field of human communication [3, 4]. CRA has been shown to accurately model the "collective mind" of a group's interpretation of a common text [3].

Other researchers have recognized the potential of CRA. Crawdad's founders were invited to present their CRA research in the book Communication and Terrorism [5]. The research used CRA networks to analyze Reuters coverage of September 11, 2001 news events. CRA networks were also the focus of a recent workshop on network visualization, involving analysis of this same dataset by a team of data visualization experts [6]. CRA has been the basis of new applications in knowledge directories [7] and a new theory of organizational communication [8]. CRA has the ability to provide significant value in the information retrieval and knowledge management domains. Prior to this project, however, CRA had only been used in a relatively low volume, off-line environment, and had not been tested in an environment that requires processing of large volumes of streaming text. This project provided a timely

opportunity to compare CRA to other methods and determine its potential in text mining applications.

The primary purpose of this Phase I research was to prove that CRA provides both a feasible and superior approach to performing text mining under storage constraints.

2.2. Phase I Technical Objectives

The purpose of this research was to investigate the performance of the CRA data mining system when used for processing streaming news items. We wished to determine how streaming text data should be filtered and stored in a way that minimized storage needs, and maintained information integrity to the extent that retrieval and subsequent knowledge processing could proceed effectively. Our solution is the knowledge-based system depicted in Figure 2.

Streaming data is converted into an intermediate format, and a data model is produced for subsequent processing. The model must facilitate determination of relevancy for a particular text source. Relevant sources are compressed and stored in a database. Knowledge-processing algorithms are then used to extract concepts and rules from the database, and convert these to an appropriate knowledge representation and store in a knowledge base. Retrieval and mining methods are then used to find and present this data and knowledge to a user. Air Force applications for such a system include terrorist tracking, knowledge management (KM) of large-scale acquisition programs, and monitoring business process data.

The architecture of the streaming data system suggests that its performance is based on the performance criteria of several of its components. These criteria include:

a. The system must effectively handle large volumes of data under potentially *bursty* conditions.

b. The system must create a model of the text that is best for further information and knowledge processing purposes.

c. The system must use effective methods for determining the relevance of an incoming document relative to a query or topic statement.

d. The system must be able to store necessary data in a possibly storage-constrained environment.

e. The system must use effective methods for extracting knowledge-based rules.

f. The system must use effective methods for data retrieval and mining.

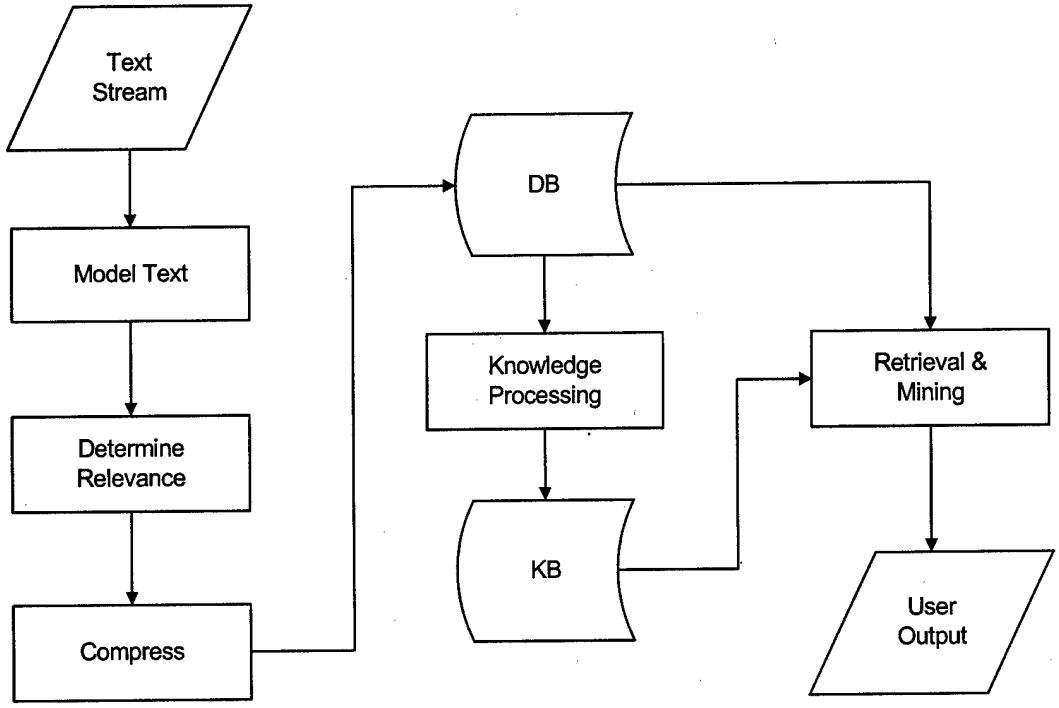

Figure 2. A Knowledge-Based Intelligence System for Streaming Data

g. The system must produce output in such a manner that the user can benefit from the system.

This Phase I research project focused primarily on criteria (b), (c), and (d). Other technology solutions exist today for handling criteria (a), and (e), whereas criteria (f), and (g) define activities that Crawdad Technologies is actively pursuing but are not within the scope of the work performed as part of the Phase I effort. Criteria (b), (c), and (d) are used to define our two main technical objectives:

Task 1--Performance Evaluation: Compare how well CRA determines relevancy of incoming news articles, relative to the traditional frequency-based approach.

Task 2--Scalability Analysis: Establish a data design and implementation for a CRA network that minimizes storage requirements and measure the computational requirements associated with CRA.

To the extent that CRA is superior in determining relevancy, it represents a significant finding in the field of information retrieval. It is important to note that while many technologies have been developed in the past decade to enhance information retrieval performance, they are all based on a model of the text based on word frequencies. If the CRA metric of *influence* is a more effective (or at least different and equally insightful) metric than *frequency*, then all of the methods based on frequency should theoretically be improved by using the metric *influence*.

This is depicted in Figure 3 which shows the computational flow of a text mining decision, starting from the modeling of text and ending with a decision outcome of some sort. A text is modeled by either counting word frequencies or calculating word influence values via the CRA network representation. This frequencies or influence values for each word in the text form a vector T, which is subsequently converted to a metric S. For example, it is typical to normalize T by the population word frequencies, in which case S would be a vector of the normalized word frequencies. Finally, an information processing method transforms S into output Z. It is Z that is used to make an information processing decision, e.g., determine if this text is relevant to the query or not? For example, Z might measure the similarity between two texts represented by vectors $T1$ and $T2$. In this case, relevance $Z = G(S1, S2) = G(f(T1), f(T2))$, thus demonstrating that (a) the quality of the text model T directly impacts the quality of information processing, and (b) the performance improvements observed via innovations made concerning the metric $S = f(T)$ or information processing output $Z = g(S)$ should be enhanced further by using a CRA-based model of the text if, in fact, such a representation is more powerful.

CRA must not only determine relevancy more accurately, it must be scalable enough for implementing a system for processing in a real-world streaming text environment. Thus, the purpose of Task 2 was to answer the questions: Can a data model for a CRA network be developed that minimizes storage requirements, and what are the computational requirements associated with implementing and operating a CRA-based knowledge management or information retrieval system?

In the end, why does better information processing matter? The two main benefits can be expected from the improved mining of textual data provided by CRA:

- Information retrieval tasks will be done more accurately — more "hits" and fewer "false alarms" — thus reducing analyst time spent on a text mining-task, e.g., document retrieval.
- Decisions based on subsequent operations such as question answering, hypothesis testing, causal reasoning, statistical modeling, and social network modeling will improve because better data will be being used as an input to such operations.

2.3. Detail of Technical Objectives

Task 1: Performance

Requirement: It is necessary to model a text in a meaningful way in order to do effective information retrieval and knowledge processing. Such a model may be stored locally in lieu of the actual text in a storage-constrained environment. As most retrieval tasks require a system to measure the similarity between two texts, it is reasonable to assume that a metric that performs well in measuring similarity between two texts will also do well in broader information retrieval and knowledge processing tasks. The basic requirement for a similarity measurement is that the proposed metric must distinguish between true matches and non-matches.

Solution: The CRA network provides a measure of word importance — influence — that is significantly different from word frequency. We thus posit that metrics based on word influence will outperform those based on word frequency for basic information retrieval tasks, specific ad hoc retrieval task and topic tracking tasks (these specific tasks will be described in detail in Section 3 of this report).

Background: Centering Resonance Analysis (CRA) is a form of text analysis, where *text* refers to a written document or transcribed conversation. Text analysis methods generally are of two different forms: semantic analysis (e.g. [9-11], or statistical- (i.e., frequency-) based approaches [12-14]. Frequency-based approaches

also include positioning methods that spatially represent populations of texts based on their similarity to one another [15-18].

Semantic approaches use natural language processing to draw inferences about a text's content. To accomplish this they apply rules or learned patterns to content that is directly given in the text, or distinguish important from unimportant material using similar sets of rules. Semantic approaches include Bayesian networks, automata-based production systems, and semantic analysis [19]. Semantic analysis provides valuable knowledge, but at a very high cost. Typically, significant effort is required for developing ontologies, and/or identifying and implementing training sets. Additionally, semantic analysis typically requires extensive processing. Both of these attributes make it unattractive as a generic method for handling large volumes of streaming textual data.

Frequency-based methods include keyword indices and other methods based on term frequency (TF) or inverse document frequency (IDF) [12]. These techniques produce representations of texts as a set of keywords with associated (weighted) frequencies. Frequency-based vector methods are often used in conjunction with positioning approaches [20]. Latent semantic analysis [17] and self-organizing maps [21] are notable examples. IDF methods represent the dominant paradigm for such tasks as text summarization in storage-constrained environments [22], sentence extraction [23], topic detection and tracking [24], summarization [25], document filtering [26], and information retrieval [12].

Statistical methods based on IDF produce data models of text that can be stored efficiently. Two problems exist, however. First, IDF measures for a text only have meaning within a given population of other texts; therefore training must occur before a system can be implemented. If a text is placed into a new population of texts, its IDF measures must be re-calculated. Second, the information retrieval performance of IDF schemes is limited. Most IDF and other statistical approaches ignore word order or discursive structure, and thus are constrained in the amount of information they capture. It is notable that after ten years of TRECs (Text Retrieval

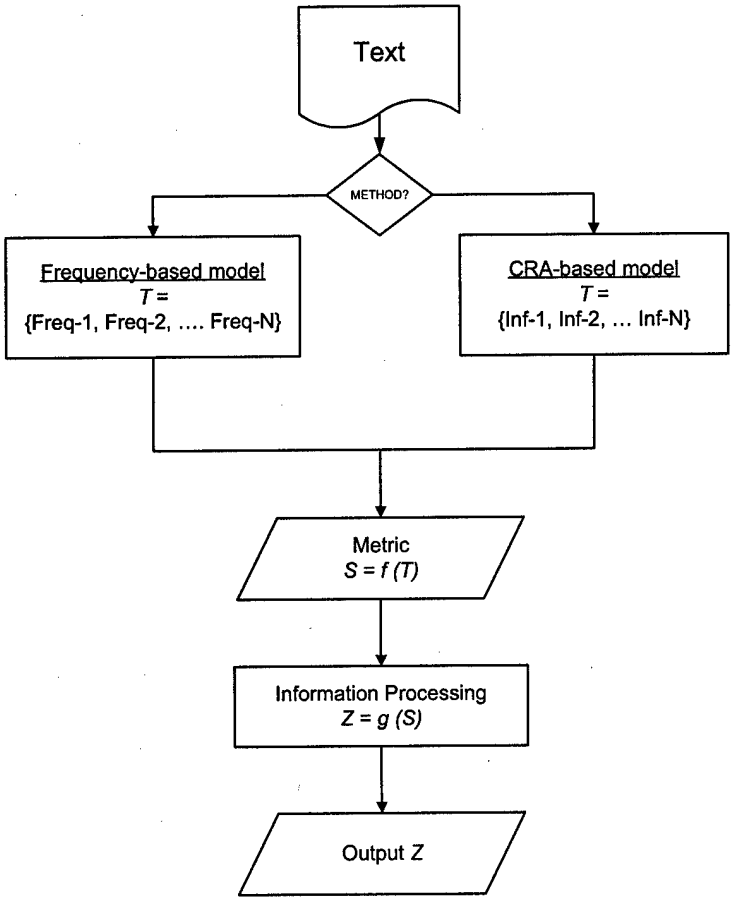

Figure 3. How Text Gets Converted Into a Decision Outcome

and Evaluation Conferences), IDF results on information retrieval tasks have not significantly improved. IDF schemes keep on getting increasingly complex, with little noticeable improvement in performance. *What is needed is a completely different approach.*

CRA is unique in that it is based on a third approach — discourse, which is concerned with how people create coherence in their communication through the use of backward and forward looking conversational centers [27-30]. *Centers* are noun phrases constituting the subjects and objects of utterances. In a written text, for example, each sentence has a *backward-looking* center that refers to a preferred *forward-looking* center expressed in the previous utterance. Though it does not code the centering process across sentences, CRA does code the ways authors deploy centers within sentences to create a structure of conceptual relationships in their texts.

The first step in CRA processing, *word selection*, uses grammatical parsing to unitize the centers by identifying noun phrases (NPs) and extracting them from text for further analysis. The second step is *word linking* where NP-component words (nouns and adjectives) are linked sequentially within sentences, and all possible co-occurrences of words within NPs are linked, yielding a CRA network. Third, CRA performs *word indexing* by analyzing the network of linked nodes generated in the previous step, and calculating the influence of each node, as made operational by its *centrality betweeness*. The final step in CRA is *word mapping*, whereby the CRA network is visualized. CRA thus combines elements of semantic analysis (via noun phrase parsing to identify centers) and statistical analysis (to examine the influence of words). *Visualization of CRA Networks can be used in lieu of sentence-based summaries for rapid assimilation of textual data in a streaming environment.*

A CRA map of this specific AF STTR area AF03T011 is shown in Figure 4; only the most influential words are shown. We can see that *data* is the most influential word (shown at top of graphic); *data set* and *data item* are the most influential word pairs (as denoted by the thicker edges), and other influential words include *algorithm*, *storage*, *application*, and *stream*.

Resonance is a latent property of the structure of a CRA network. To the extent that other texts or utterances deploy centers in the same way as in the network, they may be said to resonate with it. Assume that a criterion text and some query text have both been cast as CRA networks. Resonance is calculated based on the product of the influence values of co-occurring words, times the number of times they co-occur. A second statistic, *paired resonance* takes into account co-word pairs that co-occur across the two documents. Resonance tends to be a course-grained measure, whereas paired resonance is a fine-grained measure.

The more two texts and/or conversations frequently use the same influential word pairs, the more they resonate with one another. It follows that the more they reso-

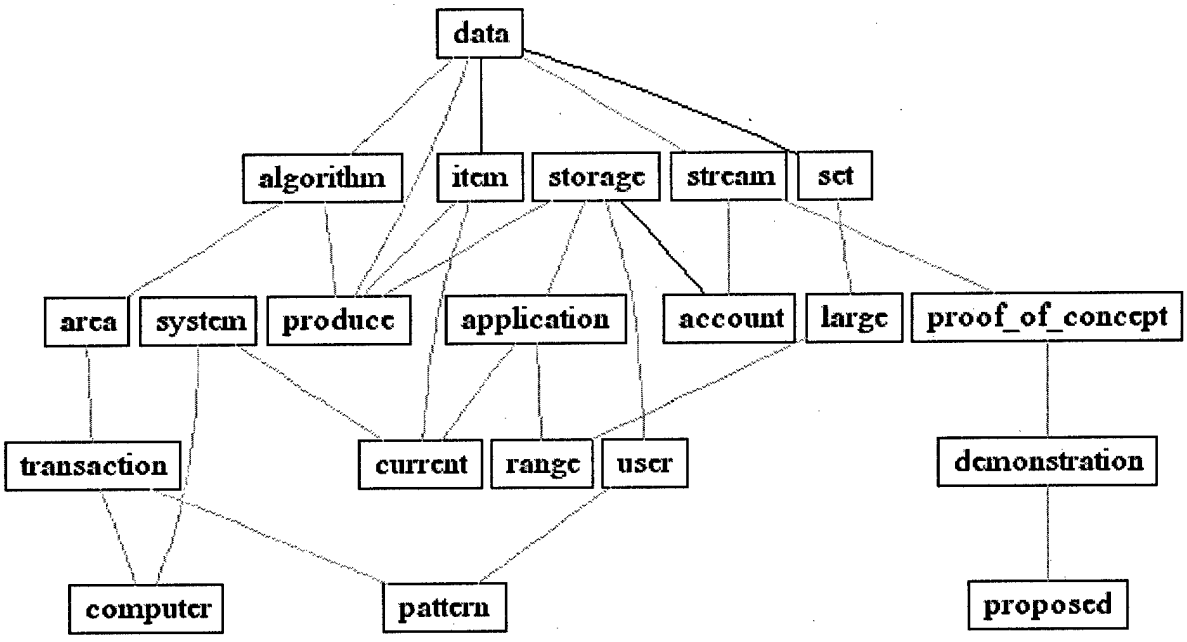

Figure 4. Visualization of CRA Network for RFP AF03T011

nate with one another then, the more their authors/ speakers assembled conversational centers in the same ways, in order to make their communication coherent. The resonance term can also be normalized to a (0,1) measure by taking its norm; this is relevant when the size of a text is not important, and yet the size of texts in the body of texts being examined differs greatly. From an algorithm perspective, resonance is similar to other vector-based methods using a cosine measure, and other co-word analysis methods [31-33], with the key difference being the vector values are word influence rather than word frequency (or IDF). As with other similar measures of similarity, resonance values associated with a population of texts can be assembled into an adjacency matrix and subjected to classification, thematic analysis, and spatial positioning. Crawdad 1.1 contains an automated classifier (that does not require a taxonomy *a priori*), a full text search engine, and a comparator that determines which elements of two texts are similar and different.

Figure 5 shows our solution for determining relevance of streaming texts. A full-text criterion is formed in one of two ways. First, the user/analyst can write a full text description of their topic of interest. Second, any document or set of documents can be pointed at and be used as a search criterion. A CRA network is then generated for the criterion. The incoming text has its CRA network generated, and the resonance is calculated between the next text and the criterion. Note this is equivalent to a full-text search capability using full-text queries. If the text resonates with the criterion greater than some chosen cut-off value, it can be retained. We use methods similar to those described by Motwani [34] to implement a dynamic threshold for resonance, based on current storage constraints.

In order to test the effectiveness of CRA in filtering incoming streams, we used standard methods and experimental procedures that measure *precision* and *recall* [12] as employed in the Text Retrieval and Evaluation Conferences (TREC). *Precision* measures how many false matches are retained, while *recall* measures how many true matches are discarded. We performed two different analyses. We first used the standard CACM corpus to perform a standard ad hoc retrieval test and then benchmark tested CRA's precision and recall performance against frequency-based methods.

Second, we performed a topic tracking task using a sample set of 83 topics form the TDT-3 corpus. The first relevant document in each topic was taken as the training document and scored against 31,276 English documents in the corpus using both CRA and term frequency-based methods. Performance was compared using CRA and frequency methods. We also collected CRA computational speed information by measuring the time required to perform these tasks and then used this information to evaluate scalability in a real-world streaming news environment.

Task 2: Scalability

Requirement: Due to storage constraints, it is desirable to store a representation of a text that is smaller than the original text. Such a representation should maximize the amount of information retained, but be obtained with reasonable computational effort in order

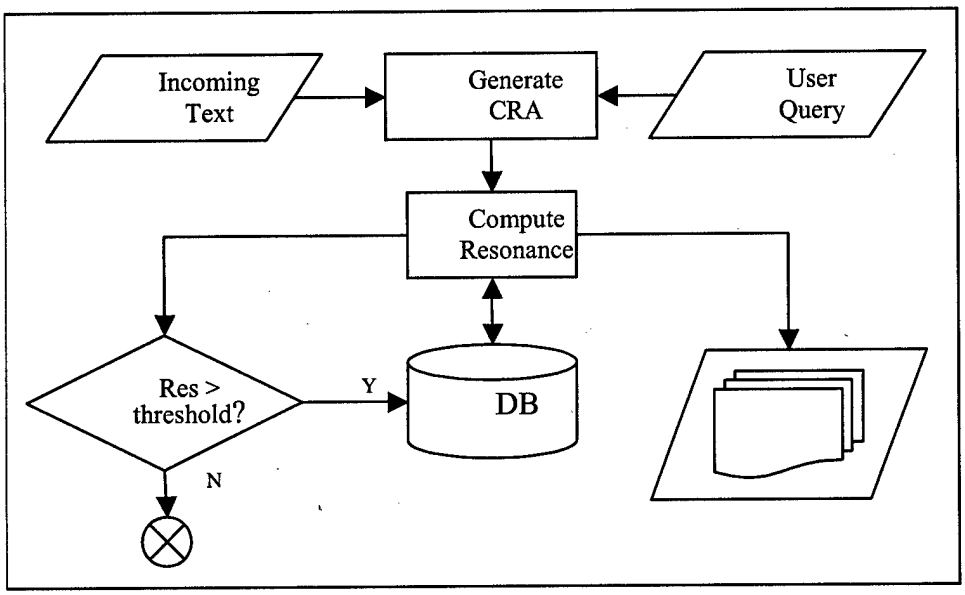

Figure 5. Process to Determine Relevance of Incoming Text

to be a scalable solution. Such a representation should facilitate efficient and effective knowledge processing.

Solution: CRA networks will require less storage capacity than the raw text, and maximize the amount of information retained. CRA will require reasonable computational effort and thereby represents a scalable solution.

Background: An appropriate solution must optimize several competing objectives: (a) minimize storage requirements, (b) maximize amount of information retained, (c) minimize computational requirements and other processing effort, and (d) facilitate subsequent processing. Table 1 compares four solutions for data storage of texts: CRA, the raw text (baseline), and semantic and frequency approaches. CRA compares very favorably to the other potential solutions. Storing the raw text requires too much space and does not facilitate subsequent processing. Semantic methods generate very large representations requiring much space, and processing effort is significant as methods much be trained, and/or ontologies developed and maintained. Frequency methods fail to retain much of the initial information from the original text, and usually require reference to a population of texts, thus requiring re-processing as contexts change. *Because CRA requires neither ontologies, training, nor corpora, and can create emergent taxonomies and perform classification "on the fly", it is best aligned as a solution to clustering issues involved with streaming data* [35, 36].

CRA network data files are smaller than the raw text but somewhat larger than required for a frequency-based representation. A frequency vector representing n words in a text requires $O(n)$ storage, one location for the frequency count of each word included. By contrast, CRA networks require $O(n + m)$ space, where m is the number of edges connecting the nodes in the CRA network. The data expansion is unavoidable because CRA networks simply carry more information about a text by encoding word connections intended by its author. The m component represents this additional data.

CRA first performs lexical analysis and noun phrase parsing, which reduces the size of n through (a) eliminating stop words such as articles, prepositions, and conjunctions, (b) stemming — the reduction of words to their base forms (for example, changing plural to singular forms), and (c) retention of only nouns and adjectives in the form of noun phrases. This, in turn, greatly reduces the required size of m. Semantic approaches either store all words (except perhaps stopwords), along with semantic tags, or use extensive processing effort in order to produce a more reasonably sized model of the text.

Three storage schemes are typically used to represent networks. An adjacency matrix is a two-dimensional array of size n^2, whose elements represent the presence/absence (or value) of an edge between the row and column nodes. When these array are used to represent undirected networks (like CRA networks) the size of the array can be reduced to $(n\ (n - 1))/2$. An edgelist represents a network by decomposing it into a set of linked edges. The space savings of this structure relative to the adjacency matrix is a function of the density of the network. For sparse networks such as CRA networks, space savings are achieved because non-existent links consume no storage. The *nodelist* or *adjacency list* is the most efficient means of representing a network. In this format, each node is listed along with pointers to its neighbors, so each node is listed only once, and there are only as many pointers as there are edges.

In order to facilitate subsequent processing, CRA networks also store information about the *influence* or importance of various nodes. These values are calculated from the network structure, and their storage in the CRA networks represents a trade-off between computational efficiency and storage efficiency. Once a text is converted into a CRA network and its influence values

Table 1. CRA Comparison with Other IR Techniques

Solution	Storage Requirements	Amount of Info Retained	Effort Required	Subsequent Processing	Total
CRA	4	4	4	5	17
Raw text	1	5	5	1	12
Semantic	2	5	1	5	13
Frequency	5	2	3	3	13

CRA Compares Favorably to Other Possible Solutions (1=poor, 5=excellent)

are calculated, it is directly accessible to Crawdad analysis tools and can be arbitrarily combined with other texts to form emergent collections; however, these influence values carry additional storage requirements.

All this said, as long as it is not necessary to reproduce the original text and there is a limited vocabulary of words, it is possible to specify a data structure for CRA networks that is space efficient relative to raw text. The minimal representation of a CRA network is in the adjacency list format described above. A given data line therefore includes a node identifier, and a list of adjacent nodes along with the edge values of connections to those nodes. It also includes the influence value calculated for the node.

Space savings on the adjacency list are achieved in two ways. First, assuming the documents analyzed never contain a vocabulary larger than a certain size, the node identifiers can be replaced with numerical codes. As words come into the system, they are assimilated (or found) in a tree data structure and associated with a fixed numerical code. There are about 200,000 English words in common use. A three-byte integer is sufficient to represent $2^{24} = 16{,}777{,}216$ words, which is more than enough to handle words in common use plus any slang or technical jargon that might be required. Second, because a CRA network only links words internal to itself, the adjacency lists can simply point to the ordinal position of the words linked. A two-byte integer can represent $2^{16} = 65{,}563$ in these "internal" positions, and it is unlikely that any individual text will exceed this number (it is over three times the size of the vocabulary of the average English speaker). The influence value requires a single precision real number. Further storage savings is achieved by eliminating any nodes from the network that have zero influence and only one edge connecting them to another node.

Word *influence* in a CRA network is operationalized as *betweenness* centrality [37], which requires computation of all-pairs shortest paths. The most efficient known computational algorithm for this problem was quite expensive in terms of computational power, requiring $O(n^3)$ time and $O(n^2)$ space [38], where n is the number of nodes in the network. These requirements would have made CRA impractical for texts containing more than a few hundred words. Fortunately, recent algorithmic advances [39] have reduced the computation time significantly. Therefore, calculation of CRA word influence now requires only $O(n(n + m))$ time and $O(n + m)$ space, where m is the number of edges in the network. Resonance is computed on collections of CRA networks and requires $O((n(n - 1)/2))$ computation time, where n is the number of CRA networks.

Section 3. Technology Approach

This section describes the use of Centering Resonance Analysis (CRA) to model and analyze texts. We begin by describing three basic approaches to text analysis, and describe the strengths and weaknesses of each approach. We then position CRA within that typology and describe how it differs from existing approaches. Finally we discuss various applications of CRA. Much of the material is derived from the foundational paper on CRA [3].

3.1. Computerized Analysis of Text

Computerized analysis can serve as a substitute, though not necessarily a replacement, for a complete reading of a text. This goal has been a priority in the well-established field of *text analysis* [40]. We distinguish three general approaches to text analysis: inference, positioning, and representation (see Figure 6). A text can be used to infer its deeper meaning within some specified context. This requires a pre-defined and constructed ontology, and/or training schemes. Second, a text can be compared to other texts through positioning methods. Third, the content of a text can be represented in a *raw* manifest form. Models based on inference or representation can be used for positioning. In general, inferential methods are useful for highly specific and static contexts; representational methods, such as CRA, are more robust to different contexts and require less effort to design and operate. Each of these approaches is now explained in more detail.

3.1.1. Inference

A first approach to textual analysis assigns meanings to linguistic inputs. However, these meanings are usually at a level of abstraction above the word content directly given in the text. For instance, if a text includes words such as *branches*, *leaves*, and *roots*, the method would conclude that the passage is likely about *trees*. To arrive at such a conclusion, these approaches typically apply rules, learned patterns, or ontologies to a text in ways that allow a computer program to distinguish important from unimportant material and infer meanings. A foundational assumption from this approach is that the meanings of words are discernible based on probabilistic causal relationships among other words in a text; this assumption underlies the common use of Bayesian networks [41] and leads this approach to be commonly linked with artificial intelligence techniques [42].

A primary methodological distinction in inference-based approaches is between syntax-driven lexical analysis and semantic grammars [19]. In *syntax-driven lexical analysis*, a textual input is parsed into its constituent linguistic units to arrive at a structural representation of the text. This is then passed through a semantic analyzer (software program) to arrive at a meaning representation. This semantic analyzer plays a crucial role in determining meaning, but the analyzer must be "trained" to look for particular word co-occurrences or grammatical forms.

The second approach, *semantic grammars*, develops rules based on entities and relations in a particular domain from which the text is drawn with the goal of resolving word ambiguities in particular contexts. The rules for making sense of ambiguous words are therefore specific to the domain, and are therefore often quite effective in automatically identifying the meanings of metaphors, pronouns, and the like. As in the case of the semantic analyzers mentioned above, these rules must be learned by the systems. Training is usually accomplished using a domain-specific dictionary or by analyzing a corpus of text from that domain to gain a familiarity with word usage.

An example of the inference approach is provided by Leacock, Miller, and Chodorow [10], who focus on identifying the *senses* of words. Their Topical/Local Classifier (TLC) is designed to statistically infer the

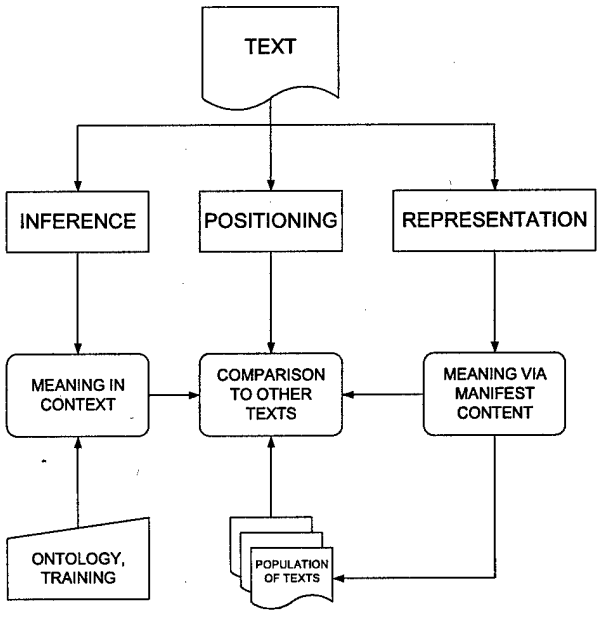

Figure 6. Approaches to Text Analysis

senses of verbs, nouns, and adjectives using words in the textual neighborhood. The TLC uses a Bayesian approach to find the word meaning that is most probable given a set of cues contained in a researcher-selected *window* of *N* words around a word with multiple senses. It then *tags* the text by separating grammatical units and reducing word variants to a base form. These tags and the meanings of the cue words are learned by the TLC using human coding and the WordNet lexical database [43]. WordNet includes a dictionary of words arranged conceptually that are used to teach the TLC about words including word synonyms, antonyms, hyponyms (super-ordinate-subordinate relationships), and the like. This training enables the classifier to recognize topics and meanings of words when analyzing subsequent texts.

Inferential approaches such as TLC are most useful when the goal involves making sense out of ambiguous words or finding related texts in a given domain. Increases in computing power allow these training approaches to be used relatively inexpensively. Thus, the accuracy of identifications can improve over time. Shortcomings of such approaches, however, relate to the need to teach a domain-specific grammar to the program. The rules it learns may not be germane in other contexts. In addition, relatively few approaches here employ a linguistic or semiotic theory to arrive at semantic representations. Rather, they assume that lexical co-occurrence provides the information necessary to make sense of word meanings.

3.1.2. Positioning

The second automated text analysis approach produces a characterization of a text in relation to a collection of other texts in a set or corpus; this approach has become increasingly common in knowledge management technologies for organizations [44]. Here, the key element is the creation of a *semantic space*: Lund and Burgess define this as:

> "a space, often with a large number of dimensions, in which words or concepts are represented by points; the position of each point along each axis is somehow related to the meaning of the word" (see page 203 of [45]).

This space can be constructed either by human raters' definitions of words or by computerized analyses of lexical co-occurrence which produce a vector for the focal unit that is placed within the semantic space [20]. For instance, CATPAC [46] is a positioning approach that scans a text to find the most frequently used words, then creates a word adjacency matrix indicating the frequency with which each word pair co-occurs in the text using a window that slides across the text, similar to that described in the inference approach. This matrix can then be analyzed using clustering techniques and multidimensional scaling to facilitate comprehension of the relationships [47]. Thus, methods such as this tend to assume that (a) authors of texts in the domain structure word co-occurrence in ways similar to other authors in the domain — that relationships between words are similar across a body of text, (b) that windows sliding across texts capture consistent patterns of co-occurrence, and (c) the semantic space into which word or text vectors are inserted is substantively important. In line with the example mentioned above, a text analyzed through a positioning approach creates a semantic space in which *leaves*, *branches*, and *roots* would be found close together, but would be distanced from *hoofs*, *tails*, and *manes*.

A prominent positioning approach is Latent Semantic Analysis (LSA), a theory and method designed by Landauer and colleagues [17, 48]. LSA uses a large corpus of machine-readable language to construct its semantic space, a matrix in which each row represents a unique word, and each column is a text passage or other context; the cell entries are the frequencies with which the row elements appear in the column elements. This matrix is analyzed by singular value decomposition (a type of factor analysis) so that the meanings of words can be represented as vectors in the resulting space. In contrast to the inference approaches discussed above, LSA:

> "uses no humanly constructed dictionaries, knowledge bases, semantic networks, grammars, syntactic parsers, morphologies, or the like, and takes as its input only raw text" (see [17], p. 263).

However, LSA requires that the semantic space be trained with a large corpus of written text [49]. The purpose of LSA differs from inference approaches in that it does not attempt to distinguish alternative senses of a word, but rather modifies vectors and semantic spaces with subsequent texts, producing, on average, overall semantic spaces of about 300 dimensions [49]. LSA, then, can represent both word and document meanings through vector techniques in multidimensional space based on semantic relatedness, and new entities can be represented in the same space.

Positioning approaches are based on the (often implicit) assumption that since humans learn the meanings of words through reading and hearing them used in particular combinations, word co-occurrence is a suitable basis for representing meanings [17, 49, 50]. Positioning approaches are particularly useful when researchers use textual sources to characterize the symbols used in a given social system, such as the International Communication Association [51]. It allows placement of words or documents in semantic space,

facilitating understanding of individual texts, comparisons between texts, and a consideration of the larger corpus. When, however, the methods require training to construct the semantic space (as in LSA), or the use of terms varies across social or temporal divisions, the positioning of vectors is open to questioning. Therefore, the validity of the placement of vectors in semantic space depends on the quality of the construction of the semantic space itself.

3.1.3. Representation

A final type of text analysis method attempts to produce stand-alone characterizations of a text by extracting or distilling meaningful content from its words without reference to a training set, corpus, semantic network, ontology, or collection of other texts. In other words, these representations are meaningful by themselves. Returning to the example employed above, a representational approach would identify *leaves*, *branches*, and *roots* as particularly prominent words important to the overall meaning of the text in question, and would produce a characterization of that text based on the information provided by these and other words. The selection of important words is based on a set of criteria that describe the functions of words in texts, such as the claim in keyword indexing that the importance of a word is inversely related to the frequency of its occurrence [12]. In contrast to the approaches described above, representational approaches do not attempt to make inferences about a words' common referent or to understand word meanings in a conceptual space, though they may be available to be incorporated in other analyses that involve inference or positioning.

Representational approaches tend to be similar to content analysis procedures that use manifest content (as opposed to latent) to arrive at descriptions of texts. Although such content analysis procedures often accept a wide range of symbolic material in addition to texts, their emphasis on objective, quantitative measurement of features of texts often detracts from their ability to produce stand-alone statements of texts' meaning, particularly through attempts to identify units of analysis and to generate coding categories.

An example of a representational approach that avoids the problems of manifest content analysis is provided by Danowski and colleagues [52-55]). Their method uses an automated content analysis that creates networks of words based on their co-occurrence in a researcher-selected *window* of *N* words. This window slides across a text, incrementing the frame one word each step, to locate clusters of words that frequently co-occur, as well as words that link clusters together. From this set of co-occurring words, a word network (as the representation of the text) is constructed. The content and structure of this network can then can then be linked to a variety of relevant information, such as organizational market share or financial performance [54], or can be used to manage organizational discussions through the introduction of optimal messages [52, 53]. This technique, although a considerable advance over traditional content analysis methods, raises two concerns. First, the size of the window (which was also used in several methods described above) is not determined with reference to a theory of linguistic structure, leaving open the possibility of researcher manipulation until an interesting network is produced. Second, since it does not filter words from the original text (except "stop" words like articles, conjunctions, and pronouns), word networks are potentially influenced by an author's grammatical style. Moreover, the networks often contain noise in the form of connections between words with little discursive importance [56].

Representational approaches such as this therefore avoid several of the shortcomings of inference and positioning methods, yet display some methodological limitations. A significant strength of Danowski and colleagues' method, like many other representational approaches, is that it produces a *network* of interconnected terms that is used in discerning the referential meanings of the text. A concept network is a well-recognized and respected format for accounting for meaning in the social sciences [57, 58]. In previous approaches, a network was used to either produce inference rules or position focal entities (including single words). In representational approaches, these networks are intrinsically meaningful. These networks can then be analyzed according to their global and local characteristics including qualities of the relationships between concepts and the differing conceptions of the same concept between networks [59, 60]. Additionally, representational approaches are more suitable when the goal is to produce a statement capturing the meaning of an entire text or discourse artifact rather than a smaller unit within a larger text. This can be particularly useful, for example, when comparing several similar texts over time or in searching for themes.

Representational approaches tend to be weak, however, in that they are rarely based on a linguistic theory about text production or interpretation, making their methodological choices (such as the *window* sliding across the text) and conclusions suspect. Further, some techniques in this category, such as TACT [61] require a researcher to manually *tag* the text before analysis by marking up particular elements for attention in later analysis. Although this aids in the retrieval of specific passages in a text, it is both labor-intensive and intro-

duces the possibility of error in the process, since the results of the program depend on the accuracy and comprehensiveness of the tagging.

In sum, the approaches to automated text analysis discussed here involve a variety of assumptions and methodological choices. We can conclude that attempts to deal with data glut must use computing power wisely to respond to the limitations of these methods. In particular, there is a need to develop methods that develop representations of texts that (a) create a network of associated terms, (b) employ a theoretically-justifiable unit of analysis rather than a window of arbitrary and varying size, and (c) are versatile because they are not dependent on dictionaries, corpora, or collections of other texts to produce representations. We now turn to a description of such a method.

3.2. Centering Resonance Analysis

CRA is a representational method, a form of network text analysis. CRA produces stand-alone representations of a text that do not depend on the sorts of dictionaries, semantic networks, or ontologies mentioned above; its representations may then be employed in analyses aimed at positioning or inference. Unlike other network text analysis methods, CRA is based on a theory of communicative coherence. Specifically, CRA draws on centering theory [30, 62] in assuming that competent authors/speakers generate utterances that are *locally coherent* by focusing their statements on conversational centers [27, 28]. *Centers* are noun phrases constituting the subjects and objects of utterances, and are generally entities such as objects, events or persons [63]. In a written text, for example, each sentence (except the first) has a *backward-looking* center that refers to a preferred *forward-looking center* expressed in the previous utterance. The author/speaker also establishes an ordered set of forward-looking centers to which the next utterance can coherently refer [29, 30]. A given utterance is made locally coherent by connecting the backward-looking center in a predictable way to previous forward-looking centers.

Under the assumptions of centering theory, then, communicators speak or write coherently by creating utterances that deploy a stream of centers — more specifically, noun phrases — in a strategic way, ultimately creating a semantic structure of centers.

Coherence, in turn, is a fundamental criterion for understandable, relevant communication [64, 65]. This notion provides the basis for an efficient automatic coding system for the content of communication, grounded in centering theory, replacing the arbitrary windows of the network text analysis approaches described above. Stock issues in coding are *unitizing* (breaking the stream into codable units) and *categorizing* (assigning units to theoretical categories of interest). In CRA we unitize communication in terms of intentional utterances strategically deployed by the communicator. These units are sentences or the conversational equivalent thereof [66] because sentences represent finite groups of centering objects constructed by communicators to fit into a coherent stream of other utterances.

Note that in CRA, we are not doing an analysis of the centering process itself. This would require looking at connections between utterances. In CRA, we are concerned with the deployment of a stream of centering objects within utterance units. CRA's attention to centering tokens within utterances is justified because our intent is to represent the content of messages and their impact on the coordination and control of activity, rather than to examine the ability of interactants to achieve coherence (though it is likely that the former goal subsumes the latter).

To explain the method in greater depth, we next describe the four steps involved in generating a CRA representation of a text: *selection*, *linking*, *indexing*, and *mapping*. These are illustrated in Figure 7. First a text may be pre-processed, in order to remove extraneous data. Optionally, content may be filtered through a predefined ontology. Noun objects embedded in noun phrases — i.e., nouns and adjectives — are next identified using Natural Language Processing (NLP) techniques. Specifically, we use a commercial, off-the shelf NLP parser. Each word is then linked to other words, depending on its semantic position within the text. The words and linkages are modeled as nodes and edges in a subsequent CRA network. Next, word influence is calculated. This influence value which measures the degree to which a word creates semantic coherence in a text. Finally, the CRA network is graphically rendered to allow ready visualization of the text.

3.2.1. Selection

CRA categorizes texts in terms of a pattern of connections between words that are crucial to the centering process. Compiling these connections in all utterances in a text yields a CRA network representing the text. The procedure is in the spirit of earlier network text analysis methods, but represents a more restricted form of linking that takes account of the discursive structure of the utterance. It begins with *selection*. Rather than linking all words that fall within an arbitrarily-sized window of text, CRA parses an utterance into its component noun phrases. A noun phrase is a noun (plus zero or more additional nouns and adjectives), which serves as the subject or object of a sentence. Since the center-

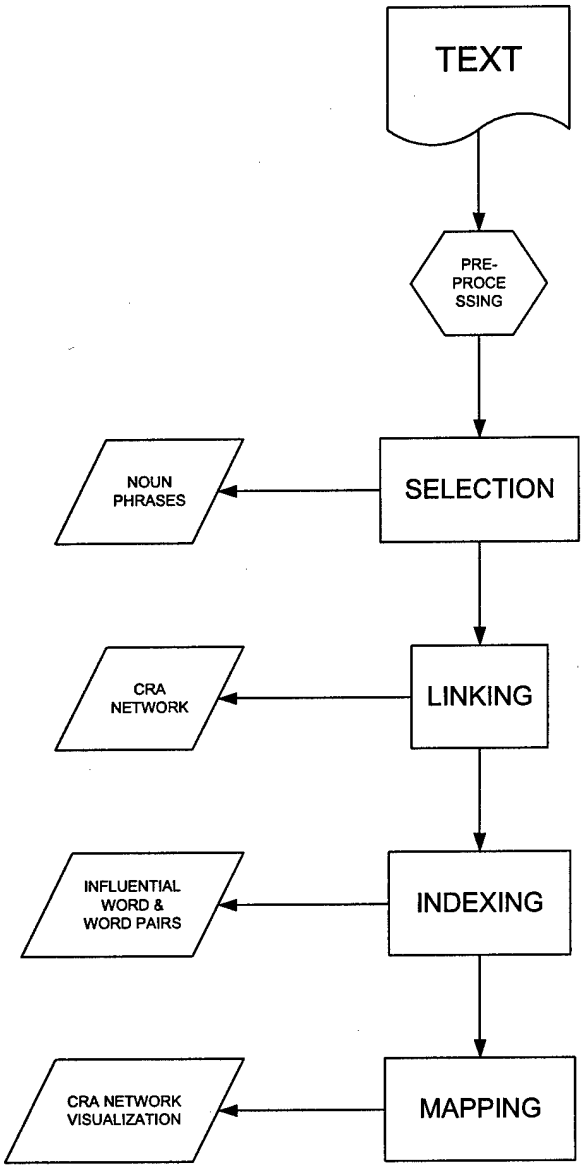

Figure 7. Steps and Outcomes Associated with Centering Resonance Analysis

ing process operates through noun phrases, this step acts as a filter, retaining only those words relevant to the centering process. We will refer to such words as *centering tokens*. A noun phrase can be composed of one or more centering tokens, and a sentence can consist of one or more noun phrases. Thus, the selection step of CRA turns sentences into sequences of centering tokens contained in noun phrases.

Before moving to the next step of the CRA method, we address a few important issues. First, CRA intentionally excludes the other main component of sentences, verb phrases. In the linguistic model directing CRA, verb phrases would be the action components linking different noun phrases in an utterance. As such, they are really a different kind of information, about the contexts of action in which the centering tokens were deployed. Given our concern for representing the manifest content of texts rather than providing inferences about the significance of particular utterances in ongoing interaction, the exclusion of verb phrases is logical. In line with centering theory, we need to focus on the entities that, when linked together, generate discourse coherence.

Noun phrases, according to linguistic semanticists (e.g., [67]), are the only elements that can be unambiguously classified as *entities* in discourse. Nouns denote conceptual categories that provide more salient discourse information than verbs and generally control the use and expression of verb phrases [68-70]. Moreover, nouns are less likely than verbs to be temporally situated and are thus more likely to be portrayed as entities (i.e., centers) in discourse [71]. In short, the parsing of texts into networks of noun phrases and the concomitant exclusion of verbs aligns with both our guiding model of discourse coherence and our desire to represent the manifest content of texts.

A second issue is whether to include pronouns in our analysis. Although a significant amount of research in linguistics is devoted to disambiguating the referents of pronouns in discourse [72-74] our approach makes the inclusion or exclusion of pronouns contingent on the purpose of the investigation and the quantity of texts involved. In most spoken and written texts, proper nouns or referents are introduced before pronouns [28] and topic shifts are introduced by specific nouns [75], meaning that little textual information is lost by dropping pronouns (as backward-looking centers) that appear later.

Third, and finally, there is the question of whether we should use *stemming* to convert words to more basic root forms before analyzing them. In general, we are wary of this technique. It is easy to think of cases where stemming might obscure important shades of meaning. The statements "the negotiators connected on the issues" and "there was a disconnect between the negotiators on the issues" would stem to the same set of objects, despite quite opposite meanings. Indeed, the effectiveness of stemming in general is an open question [76] and more sophisticated forms of stemming depend on sources of training data that are not always available or techniques that are not practical (e.g., Bazea-Yates & Ribeiro-Neto, [12], p. 168). Therefore we adopt only minimal affix stemming, going from plural to singular forms by removing "s" or "es" suffixes.

3.2.2. *Linking*

The second step, *linking*, converts the sequences into networks of relationships between centering tokens. According to the rationale above, the author or speaker of a text being analyzed with CRA intentionally groups the centering tokens into noun phrases and strings these phrases together (using verbs, pronouns, determiners, etc.) to form an utterance. CRA linking rules attempt to embody those choices. To begin with, all centering tokens in the utterance are linked sequentially. In the majority of cases, where noun phrases contain one or two words, the sequential connections capture all the linkage intended by the author; there are no higher-order connections possible without crossing the boundaries of the noun phrases. But there are cases where three or more tokens are contained in a single noun phrase. Here, the sequential links alone do not exhaust the connectedness possible in the set. Hence, we link all possible pairs of tokens within the noun phrase. For example, the phrase "complex discursive system" would generate the links: complex-discursive, discursive-system, and complex-system.

Accumulating these links over a set of utterances comprising a text (a paper, a collection of papers, a transcribed speaking turn or set of turns, etc.) yields a symmetric, valued, undirected network whose nodes are centering tokens. Link values represent the number of times the tokens were linked in the text according to the rules above. This network, when indexed as we describe in the next section, becomes a fundamental representation of the text and forms the basis for all applications of CRA.

3.2.3. *Indexing*

The third step in CRA is *indexing*. Here the network of centering token associations is analyzed to determine the relative influence of each node. This is a key step in differentiating the nodes and deserves some discussion here. Network metaphors are always based on some abstract notion of flow, and in the case of CRA networks, we would say there is a flow of meaning. To the extent that a CRA network is structured, some nodes are more influential than others in channeling flows of meaning. They are literally more *meaning-full* than other words in the network. Thus, identifying the *influence* of the nodes allows one to identify the most meaning-full words in the network and gain a measure of their meaning-fullness.

We operationalize this idea of influence as *centrality* of nodes in the CRA network. Although a variety of measures could be used, centering theory points us most clearly toward *betweenness centrality*. To our knowledge, the concept was first formalized in [77] where it is described as *rush* in a graph:

> "The rush in an element is the total flow through the element, resulting from a flow between each pair of vertices" (p.1 of [77]).

Freeman [37] contrasted betweenness centrality with other classic measures in a way that is instructive. Consider a minimal network of four peripheral nodes that are all connected to a single node in the middle (but not to each other). There are at least three senses in which the node in the middle is central. It is connected to a lot of nodes, relative to the others, which is the notion of *degree* centrality. It is also very directly connected to all of the other nodes, whereas the peripheral nodes are at least two steps away from each other. This reflects the notion of *closeness* centrality, usually measured as the average number of steps required to reach other nodes in the network from a focal node. The middle node is also central in the sense that any kind of resources flowing in the network (*meaning*, in the case of CRA networks) must flow through it. This is the idea of rush or betweenness centrality described above. Each of these measures can be computed for the network as a whole, as well as for the individual nodes [78].

Of the three kinds, betweenness centrality is clearly the most appropriate for estimating the influence of nodes in CRA. Degree centrality is the most often applied measure in earlier network text analysis efforts (e.g., [60]). Degree centrality takes only the local connections of each node into account. Closeness centrality is better in that it considers the entire network structure. However, it cannot be computed for disconnected graphs, which in CRA are not only possible but likely for low-coherence texts. More importantly, closeness undervalues the influence of nodes lying on paths connecting disparate parts of the network because nodes in the center of large, densely connected clusters will have higher closeness on average. From the standpoint of maintaining coherence in a structure of words, this *tying-together* function is crucial. Betweenness centrality therefore best represents the extent to which a particular centering token (represented by a network node) mediates chains of association in the CRA network. It tells us more than any other measure about how a given node channels the *rush* of meaning through a network of centering tokens. Therefore (adapting notation in [78]), the influence I of a node i in text T is operationalized as:

$$I_i^T = \frac{\sum_{j<k} g_{jk}(i)/g_{jk}}{[(N-1)\cdot(N-2)/2]}$$

where g_{jk} is the number of shortest paths connecting the j^{th} and k^{th} nodes, $g_{jk}(i)$ is the number of those paths

containing node x, and N is the number of nodes in the network.

Resonance is a latent property of the structure of a CRA network. While resonance is a property of a single network, it is only realized in the presence of an external signal (i.e. another network), just as a physical material only resonates when brought into contact with an external vibrating wave. To the extent that other texts or utterances deploy centers in the same way as a given network, they may be said to resonate with it. To understand how we make operational the resonance of one text with another, assume that texts A and B have been represented as CRA networks. The two texts may be of similar nature, or one may be considered a query and the other a text potentially relevant to the query. There are two ways of measuring resonance, one less specific and based on the words common in the two documents, the other more specific and based on word pairs common in the two documents.

Word resonance is calculated directly from the influence scores of the words in the two texts. Let the (unique) words (after parsing into phrases) for text A be represented by $\{w_1^A, w_2^A, \ldots w_{N(A)}^A\}$ with corresponding influence scores of $\{I_1^A, I_2^A, \ldots I_{N(A)}^A\}$, where $N(A)$ is the number of (unique) words in text A, and similarly, text B has words $\{w_1^B, w_2^B, \ldots w_{N(B)}^B\}$ with influence scores $\{I_1^B, I_2^B, \ldots I_{N(B)}^B\}$. In general $N(A) \neq N(B)$. Let the indicator function α^{AB}_{ij} be equal to 1 if w_i^A and w_j^B are the same words, and be equal to zero if w_i^A and w_j^B are not the same words. The word resonance between texts A and B, WR_{AB}, is defined by:

$$WR_{AB} = \sum_{i=1}^{N(A)} \sum_{j=1}^{N(B)} I_i^A \bullet I_j^B \bullet \alpha_{ij}^{AB}$$

The more two texts frequently use the same words in influential positions, the more word resonance they have. The more word resonance they have, the more the communicators used the same words, and the more those words were prominent in structuring the text's coherence. Word resonance is a more general measure of the mutual relevance of two texts, and has applications in the modeling of large corpora.

This measure is unstandardized in the sense that resonance will increase naturally as the two texts become longer in length and contain more words. There are cases, however, where a standardized measure is more appropriate. For example, in positioning documents relative to one another (as described below), one does not necessarily want to overemphasize differences in document length, number of nodes, and so on. In these cases the appropriate standardized measure of resonance is given by:

$$WR'_{AB} = \frac{WR_{AB}}{\sqrt{\sum_{i=1}^{N(A)} (I_i^A)^2 \bullet \sum_{j=1}^{N(B)} (I_j^B)^2}}$$

which is structurally equivalent to the manner in which the covariance between two random variables is standardized to a measure of correlation.

Pair resonance is estimated using co-occurring word-pairs, as opposed to co-occurring words. Let the *frequency weighted pair influence* of words i and j in text T be given by:

$$P_y^T = I_i^T \bullet I_j^T \bullet I_y^T$$

where I_i^T is the influence of w_i^T, I_j^T is the influence of w_j^T, and F_{ij}^T is the number of times that w_i^T and w_j^T co-occur (their corresponding nodes are connected directly by an edge) in text T. If text T has N (unique) terms, then there will be $(N \cdot (N - 1) / 2)$ pairs, but many of them will have a value of $F_{ij}^T = 0$ as they will not represent connected terms. Let the indicator function β^{AB}_{ijkl} be equal to 1 (a) if the two word sets (w_i^A, w_j^A) and (w_k^B, w_l^B) are equivalent (regardless of the manner in which the set elements are ordered), and (b) F_{ij}^A and F_{kl}^B both are equal to one (the sets represent connected nodes); otherwise the indicator is zero. In other words, the indicator function β^{AB}_{ijkl} is 1 when the corresponding pairs of co-occurring words also co-occur across the two texts. The pair resonance PR_{AB} is defined by Equation 1.

The more pair resonance two texts have, the more their producers assembled conversational centers in the same ways, in order to make their communication coherent. Pair resonance is a more sensitive measure of the mutual relevance of two texts than word resonance, because it takes account not only of the words and their position in the network, but how they were assembled in the utterances. As such pair resonance has applications in high-accuracy searching through a corpus of texts.

Both of the resonance measures just described yield unstandardized measures of resonance. There are cases,

$$PR_{AB} = \sum_{i=1}^{N(A)-1} \left(\sum_{j=i+1}^{N(A)} \left(\sum_{k=1}^{N(B)-1} \left[\sum_{i=k+1}^{N(B)} P_y^A \bullet P_{kl}^B \bullet P_{ijkl}^{AB} \right] \right) \right) \qquad \text{Equation 1}$$

however, where a standardized measure is more appropriate. For example, in positioning documents relative to one another (as described below), one does not necessarily want to overemphasize differences in document length, number of nodes, and so on. In these cases the appropriate standardized measure of resonance is the vector correlation between the influence scores or weighted pair influence scores for words or word pairs in the two texts.

In summary, CRA is a representational technique that describes the extent to which words are prominent in creating a structural pattern of coherence in a text. The description provided to this point shows that CRA possesses distinct advantages over other text analysis approaches. First, because its representations are independent of text corpora and training sets, CRA is ***highly scaleable***. It can be performed on single documents, parts of documents, or aggregations of documents, and does not depend on being trained to recognize particular patterns of words (e.g., the set trunk-root-leaf, as in inference approaches). Second, because it does not depend on training or rules sets, CRA can ***accommodate emergence*** of new terms or shifts in relationships among existing terms and concepts, as should be expected in knowledge development and other forms of innovation. Third, relative to other representational techniques, CRA is ***structurally sensitive*** in that it accounts for all likely chains of association between the centers that make texts and conversations coherent. This makes the technique more sensitive to complex associations in the text than statistical methods based on word frequency or local co-occurrence. Fourth, CRA is ***based on a theory of communicative coherence*** that avoids the imposition of an arbitrary *window* sliding over text to locate word co-occurrence.

3.2.4. Concept Mapping

The final step in CRA processing is *concept mapping* wherein the network or networks are appropriately visualized. In CRA network visualization, the object is to show as much as possible of the network of centering token associations representing a text. Because the networks of even modest-sized texts can contain 100 or more nodes, it is necessary to select a subgraph of the actual CRA network for display. Fortunately, the influence measure described above provides a criterion for selecting nodes. Ordering the nodes by their influence, we can generate a subgraph of nodes at or above some cutoff level. In doing so, we exclude tokens that are not very involved in chains of association which structure the network. Note that such choices are made for the purpose of visualization and enhanced comprehension; the less-influential nodes are not excluded from further analysis by this move. Once a subgraph has been extracted, it is displayed using graph drawing software.

3.3. Frequency versus Influence

If a key difference between CRA and all other statistical methods of text analysis is that it measures word influence instead of word frequency, a relevant question is: How much difference is there and why is word influence a potentially more sensitive measure relative to actual discursive intent of the author? In order to demonstrate the difference between word frequency, we examine an example concerning Reuters news coverage of the events on and following September 11, 2001.

Figure 8 shows a graph of the frequency of *terrorist* compared to the average for all words included in the sample. We looked at words making up noun phrases in the stories because this is where *terrorist* and similar words appear. For the first few days of coverage, *terrorist* is used a lot, an average 72 times per day. After that, there is a steep drop-off to an average of about 17 times per day because Reuters management insisted that writers not used the word *terrorist* directly in their stories concerning the event.

Figure 9 shows a graph of the influence of the word *terrorist* over the first 20 days of Reuters coverage compared to the average influence for all words. The differences from the previous frequency-based graph are striking. Except for one case on day 19 (September 29, 2001), *terrorist* has two (or more) times the influence of the average word. There's no large drop-off in influence after the first few days like there was for frequency. In fact, the influence of the word trends upward for that early period. For 16 out of the 20 days we analyzed, *terrorist* was at or above the 95th percentile in influence, and never fell below the 85th percentile. Thus, we see that while *terrorist* is not used as frequently, it is still just as influential in creating coherence in the discourse.

So Reuters's policy re-statement may have achieved a reduction in frequency of the word *terrorist*, but it really did nothing to decrease its influence as an idea tying together the story. Reporters and editors continued to rely on the word — apparently within the constraints of the policy — to make sense of the story for readers.

The semantic network represented by a CRA is particularly useful for demonstrating how particular words are defined by the words that are attached to them within the network. Figure 10 shows a map of the highest influence words for a CRA network formed from articles of September 26, 2001. In this network, the words are represented by the dots, and lines indicate that the words co-occurred inside a phrase (like "*blinding is a terrorist*"). This example demonstrates the importance

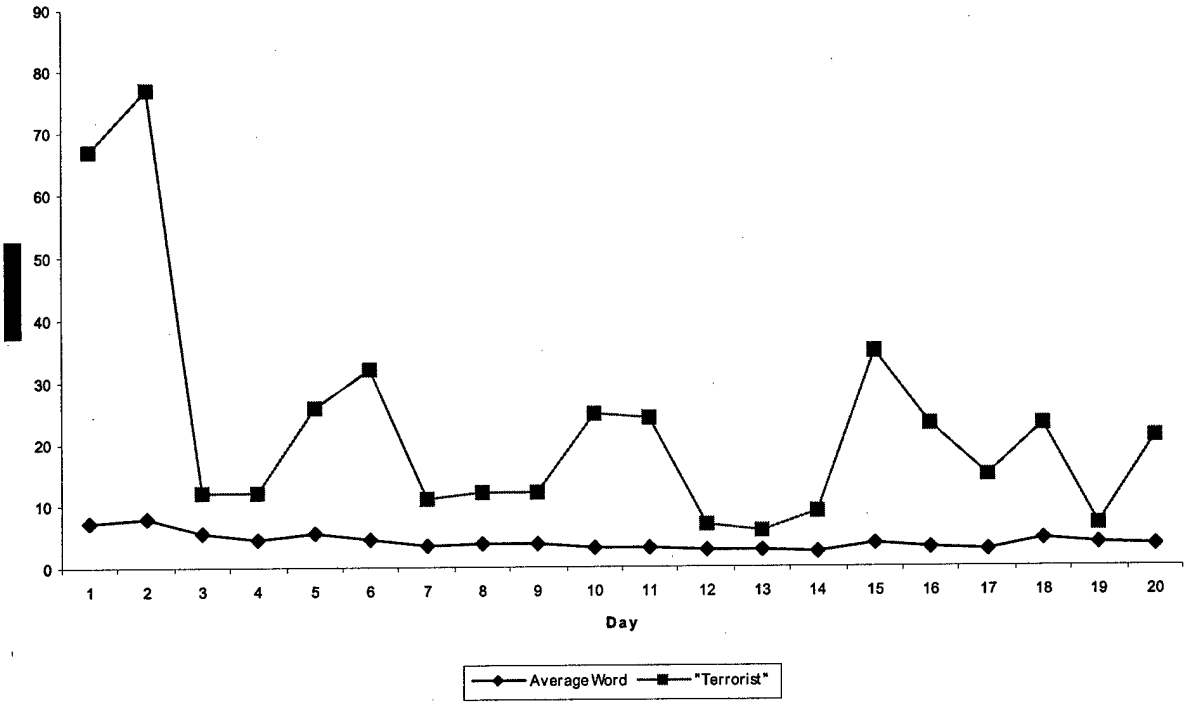

Figure 8. Frequency of *terrorist* in Reuters 9-11 Coverage

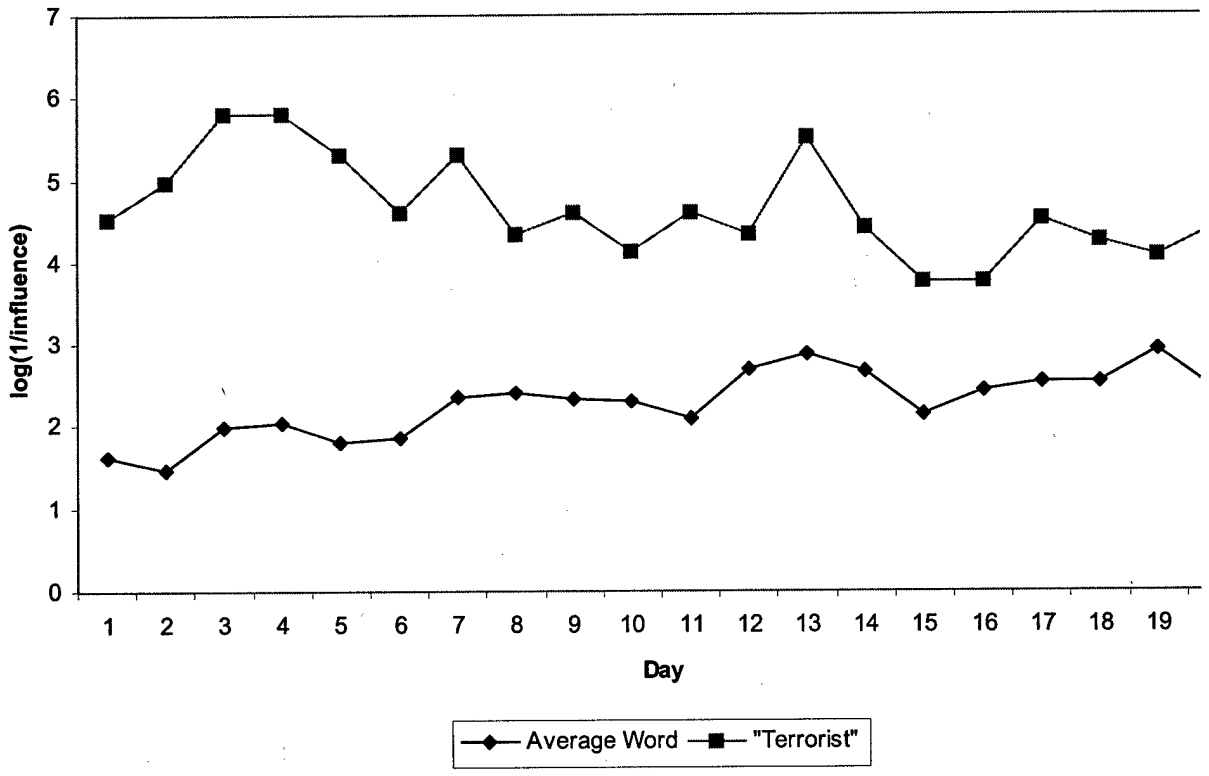

Figure 9. Influence of *terrorist* in Reuters 9-11 Coverage

of measuring word influence and the value of using a CRA network to understand embedded and contextual meaning.

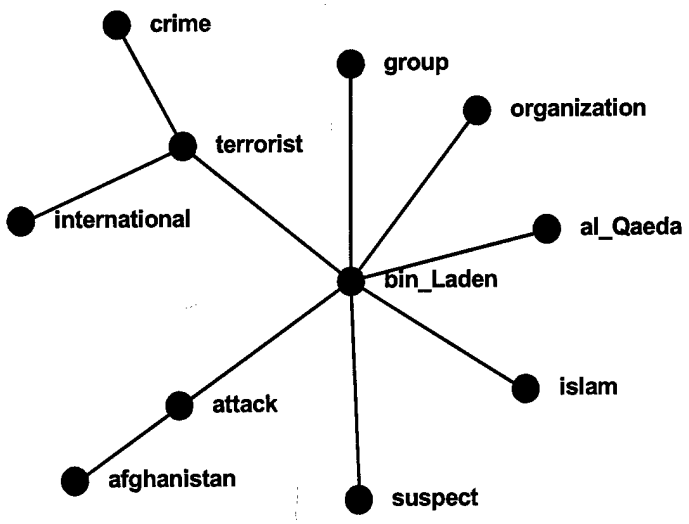

Figure 10. Highe Influence Words Associated with *bin_Laden*

3.4. Applications

The business value of using CRA in a text analysis project has been demonstrated across a variety of contexts:

- Strategic Planning. A university's research administration used CRA to determine how well its faculty's research strengths aligned with a particular federal research funding program. Over fifty faculty from 18 departments and five colleges were identified as having interests and skills in the environmental health area. This coincided with the university's push to obtain NIH research funding. Vitas, papers, and proposals were collected from each faculty member and a single "meta-text" was made for each faculty member, incorporating all their textual information. CRA was used to produce visual summaries of each faculty's expertise, and these were reviewed by administrators. Additionally, clustering was performed on the CRA Networks, and two major groupings (phsyical versus social sciences) and five minor groupings of faculty with similar research strengths were found. Recent RFPs from NIH were then collected and compared to the faculty descriptions. CRA found that the university's faculty did not "resonate" strongly with the RFPs, indicating a systemic weakness-the university was unlikely to succeed in getting large amounts of NIH funding with its current faculty base. This in part led to the university, redirecting its hiring efforts.
- Quality Control. An information technology firm used CRA to determine key issues, themes, trends, cause and effect logic embedded in the content of its problem reports. A global IT process was being managed from three different locations. Each location wrote 2-10 page problem reports whenever a system problem arose and was fixed. These reports went unread and unused, and there was no sharing of knowledge across the three sites. Over 500 problem reports were collected and CRA was performed. Key quality issues specific to particular sites, and cutting across each of the sites, were identified. CRA-based metrics were used to perform correlation analysis that linked particular root causes with observed symptoms. These metrics were also used to identify key themes (e.g. "slow user login"), and then the themes' influence values were analyzed on statistical process control charts to determine if the system was stable over time or not. Enterprise-level issues surfaced in the analysis were addressed in subsequent process improvements.
- Customer Service. A large service organization used

CRA to analyze its customer interview transcripts and determine systemic issues that cut across different industry sectors. The organization performed one-hour interviews with each of its key clients. Clients were asked to respond about strengths, weaknesses, and future plans for service. These interviews were then transcribed, and summarized by the interviewer. CRA was used on both the raw transcripts and the interviewer summaries. The two were compared to determine how accurately the interviewers wrote summaries. Content within the interview was used to cluster clients together, and these content-basec clusters were contrasted with client demographic information, such as size, industry sector, geographic location, size of account, etc. For example, results indicated that one cluster of clients were primarily concerned with quality and reliability of service, while another cluster of clients was concerned about innovation

- Group Dynamics. A consultant used CRA to determine the source of conflict when a project team split into two competing groups. The consultant was facilitating and observing a re-engineering effort within this organization. Interviews of key project team members were performed weekly. CRA was used to analyze the content of the interviews, and to identify how individuals were similar or different in the way they talked about the project. The consultant found that team members talked about the project using a shared language up until a specific date, and from there on talk about the project bifurcated along two tracks; one group focusing on customer issues and one focusing on technical issues. This change point was tracked back to an argument that occurred on that particular date during a group meeting, which led to a dissolution of group coherence.
- Public Perceptions. A researcher used CRA to determine how people were more fearful of Anthrax threat than airplane terrorism [5].A research team collected Reuters news articles pertaining to 9-11 events. Semantic themes such as "terror", "airport security", and "military response" were identified and their influence was tracked over time. Statistical analyses of CRA-based metrics were used to identify that the influence of the "Anthrax" theme was strongly correlated with influence of the word "fear"; in other words, public fear was associated more so than with other forms of terrorism.
- Organizational Change. A researcher used CRA to determine how an organization emerged from an entrepreneurial venture [79]. Interviews of the entrepreur were performed and transcribed every two weeks. Statistical analyses of CRA-based metrics demonstrated that the entrepreneur reformulated their conception of the organization at a specific moment in time, corresponding to a cognitive reframing of the product. The content of the interviews, via CRA-based metrics of word influence, was correlated with other quantitative data (dollars spent, time spent) to demonstrate how the change of "language" preceded the change in "activity".
- Knowledge Directory. An organization used CRA to develop a searchable knowledge base that would identify experts associated with particular issues [7]. Resumes and reports were collected from each organizational member and a single "meta-text" was made for each, incorporating all their textual information. CRA was used to produce visual summaries of each person's expertise, and these were reviewed by managers. A clustering of the workers demonstrated that expertise was organized more around how people performed their work, rather than the content of the work they performed. This led to a number of cross-functional initiatives to take advantage of these hidden commonalities.

Section 4. Task 1: Information Retrieval Performance

CRA is a new, innovative way to analyze documents and perform information retrieval (IR) and text mining tasks based on that analysis. A meaningful discussion of the performance of CRA can only be accomplished by comparing and contrasting CRA with traditional approaches such as term frequency-based approaches. This section provides a brief background discussion of these traditional approaches. The primary focus of this section is on methods of evaluating IR system performance. We discusses the accepted methodolgies which were primarily developed for frequency-based approaches. In particular, we discuss some of the problems with recall and precision measures and describe other methods of evaluating the performance of IR systems. The National Institute of Standards (NIST) Text Retrieval Conferences (TREC) and Topic Detection and Tracking (TDT) initiatives have provide significant insight into evaluation of these systems. Our discussion addresses how CRA can be applied/evaluated using TREC and TDT methodolgies. However, prior to our discussion of evaluation we first provide some basic background information about modern IR system implementations based on term frequency approaches.

4.1. Information Retrieval

IR typically requires examining a document or set of documents and determining if that document or set of document contains information of interest. Document retrieval and evaluation of that document is a multi-step process that typically includes a number of steps required to pre-process the document prior to analyzing its content. These steps include:

- Lexical analysis - removal of the document's punctuation, capitalization, etc. [80].
- Stop word removal - removing words like *a*, *the*, etc. that do not contribute to the overall meaning of a document [80].
- Word stemming - removal of suffixes such as *ed* and *ing* [81].
- Vector space computation - computing an information vector describing the information content of a document [82].
- Evaluation - comparing the computed information vector for a document with the information vectors of other documents that the user has rated as relevant and with the contents of the specified ontology.

Most traditional IR methods use metrics such as *term frequency* and *document frequency* in determining if a document likely contains desired information. It is instructive to examine some of these traditional systems in order to understand the unique advantages of the CRA approach.

The *vector-space* method of information retrieval/ evaluation provides a representation of a document based on its constituent words [82]. This model has been used extensively and has been shown to be competitive with alternative IR techniques [83, 84]. Using this technique, each document is represented as a vector in vector space so that documents with similar content have similar vectors. Each dimension of the vector space represents a word and its weight. The values of the vector element for a document are calculated as a combination of the term frequency $TF(w,d)$ — the number of times word w occurs in document d, and the document frequency — the number of documents the word w occurs in at least once. The inverse document frequency is given by:

$$IDF(w) = \log \frac{|D|}{DF(w)}$$

where $|D|$ is the total number of documents. The value $d^{(i)}$ of an element in the vector is then calculated as the product

$$d^{(i)} = TF(w_i,d) \times IDF(w_i)$$

After the document is lexically analyzed, the stop words removed and its words stemmed, the TF-IDF vector V_i is extracted. We assume an initial profile set V, |V|=0; the predefined number of TF-IDF vectors in the profile set is N and the preset number of elements of a vector is M. For a document of interest, we extract the TF-IDF vector V_i.

If $|V| < N$ (where $|V|$ is the number of vectors in the profile set V, then $V \Leftarrow V \cup V_i$; Otherwise, calculate the cosine similarity between every two vectors including the vectors in the set V and a new document vector V_i. If the profile vector set V is $\{V_1, V_2, \ldots, V_n\}$ where $n = N$. The similarity is given by

$$Sim(V_j,V_k) = \frac{V_j \bullet V_k}{|V_j| \times |V_k|}$$

where $j, k \in \{1, 2, \ldots, n, i\}$

Now combine the two vectors V_l and V_m with the greatest similarity:

$$V_l = V_l + V_m$$

$$Where\ (l,m) = \arg\max_{(x,y)} Sim(V_x, V_y)$$

The weights in the new vector V_k are sorted in decreasing order and we then keep the highest M elements.

The vector space method works well for certain classes of documents and provides good results in certain application such as intelligent search agents that analyze web pages. Basically information content is represented by a vector in information space. Two documents with similar vectors likely contain related relevant information of interest to an information seeker.

4.2. Evaluation of Information Retrieval Systems

4.2.1. Measuring Precision and Recall

IR is concerned with locating information that will satisfy a user's information needs. Traditionally, emphasis has been placed on text retrieval — providing access to natural language texts where the set of documents to be searched is large and topically diverse [85].

An *ad hoc retrieval task* occurs when a system knows the set of documents to be searched but cannot anticipate the particular topic to be investigated.

A *known-item search* is similar to an ad hoc search but the target of the search is a particular document (or a small set of documents) that is known to exist in the document collection.

In both the known-item and ad hoc retrieval task, the system's response is usually a ranked list of documents, and the system is evaluated by the rank at which the target document is retrieved.

In a *document routing* or *filtering* task, the topic of interest is known and stable, but the document collection is constantly changing. A typical document filtering task is an intelligence analyst examining a news feed. The filtering task generally requires a retrieval system to make a binary decision whether to retrieve each document in the document stream as the system sees it. The retrieval system's response in the filtering task is therefore an unordered set of documents (accumulated over time) rather than a ranked list.

Typically, retrieval tests are performed on a text collection and can be evaluated in a number of ways. For example, in TREC, all ad hoc retrievals are evaluated using the `trec_eval` package [84]. This package provides a number of metrics for retrieval system performance including recall and precision. Precision is the proportion of retrieved documents that are relevant, while recall is the proportion of relevant documents that are retrieved.

$Precision = N_{rr}/N_t$ and $Recall = N_{rr}/N_{tr}$

Where N_{rr} is the number of relevant documents retrieved, N_t is the total number of documents retrieved, and N_{tr} is the total number of relevant documents retrieved.

A cut-off level is a rank that defines the retrieved set. For example, establishing a cut-off level of ten mean that the retrieved set will consist of the top ten documents. The `trec_eval` program treats scores as averages over the set of topics where each topic is equally weighted.

Precision reaches a maximum value of 1.0 when only relevant documents are retrieved, and recall reaches it maximal value (also 1.0) when all the relevant documents are retrieved. These theoretical maximum values are not obtainable as an average over a set of topics at a single cut-off level because different topics have different numbers of relevant documents. For example, a topic with fewer than ten relevant documents will have a precision score less than one after ten documents are retrieved regardless of how the documents are ranked. Similarly, a topic with more than ten relevant documents must have a recall score less than one after ten documents are retrieved. At a single cutoff level, recall and precision reflect the same information, namely the number of relevant documents retrieved. At varying cut-off levels, recall and precision tend to be inversely related since retrieving more documents will usually increase recall while degrading precision and vice versa.

In most of the IR literature, precision/recall figures are presented as a continuous graph that shows changes of precision and recall that depends on how many documents are inspected. The result is a curve, going from high precision and low recall to low precision and high recall. Note that recall is a non-decreasing function of rank. Precision can be regarded as a function of recall rather than of rank.

Of all the numbers reported by `trec_eval`, the recall-precision curve and mean (non-interpolated) average precision are the most commonly used measures to describe TREC retrieval results. A recall precision curve plots precision as a function of recall. Since the actual recall values obtained for a topic depend on the number of relevant documents, the average recall-precision curve for a set of topics must be interpolated to a set of standard recall values. Recall-precision graphs show the

behavior of a retrieval run over the entire recall spectrum.

Mean average precision is the single-valued summary measure used when an entire graph is too cumbersome. The average precision for a single topic is the mean of the precision obtained after each relevant document is retrieved (using zero as the precision for relevant documents that are not retrieved). The mean average precision for a run consisting of multiple topics is the mean of the average precision scores of each of the individual topics in the run. The average precision measure has a recall component in that it reflects the performance of a retrieval run across all relevant documents, and a precision component in that it weights documents retrieved earlier more heavily than documents retrieved later. Geometrically, mean average precision is the area underneath a non-interpolated recall-precision curve.

4.2.2. Document Filtering

The purpose of document filtering is to retrieve just those documents in a document stream that match the user's interest as represented by the topic. There are three kinds of filtering tasks (defined for TREC 2002): an adaptive filtering task, a batch-filtering task, and a routing task.

For *adaptive filtering*, the system starts with a profile derived from the topic statement and a small number of examples of relevant documents and processes documents one at a time in date order. For each document in turn, the system must make a binary decision whether to retrieve it. If the system decides to retrieve the document, it obtains the relevance judgment for that document and can modify its profile based on the judgment if desired. The final output is the unranked set of retrieved documents for the topic.

The *batch-filtering* task is a simpler version of the adaptive filtering task. Here, the system is given a topic and a (relatively large) set of training documents such that each document in the training set is labeled as relevant or not relevant. From this data, the system creates a profile and a rule for when a document should be retrieved. The rule is applied to each document in the test set of documents without further modification. The final output is an unranked set of retrieved documents.

In the *routing* task, the system again builds a profile or query from a topic statement and a training set of documents, but then uses the query to rank the test portion of the collection. Ranking the collection by similarity to the query (routing) is an easier problem than making a binary decision as to whether a document should be retrieved (batch filtering) because the latter requires a threshold that is difficult to set appropriately. For TREC, the final output of the routing task is a list of 1000 documents ranked by decreasing similarity to the query.

Since neither batch filtering nor adaptive filtering produce a ranked list, they cannot be evaluated using the usual IR measures (e.g., `trec_eval`). Rather, these runs are usually evaluated using a utility function where a system is rewarded some number of points for retrieving a relevant document and penalized a different number of points for retrieving an irrelevant document. Raw utility scores do not average well so the scores for individual topics are normalized, scaled, and then averaged. Routing tasks can be evaluated using the usual IR measures.

4.2.3. Additional Metrics

The following metrics are used in the TREC Target Detection and Tracking (TDT) tasks. These provide an additional metric for performance evaluation [86].

This task is to associate incoming stories with topics that are known to the system. A topic is defined as known by its association with those stories that discuss that topic. Thus a target topic is defined by one or more stories that discuss that topic. TDT uses the following evaluation methodology. Detection performance is characterized in terms of the probability of miss and false alarm errors (P_{Miss} and P_{FA}). These error probabilities are then combined into a single detection cost, C_{Det}, by assigning costs to miss and false alarm errors:

$$C_{Det} = C_{Miss} \bullet P_{Miss} \bullet P_{target} + C_{FA} \bullet P_{FA} \bullet P_{non-target}$$

Where C_{Miss} and C_{FA} are the cost of a Miss and a False Alarm, respectively, P_{Miss} and P_{FA} are the conditional probabilities of a Miss and a False Alarm, respectively, and P_{target} and $P_{non-target}$ are the *a priori* target probabilities ($P_{non-target} = 1 - P_{target}$).

C_{Det} is the bottom-line representation of task performance and can be used to judge system performance. This cost measure represents a measure of application value, and consideration of the application can inform judgment as to the appropriate values for the relative costs of misses and false alarms and the target probability.

Since these values can vary with the application, C_{Det} is normalized so that $(C_{Det})_{Norm}$ can be no less than one (without extracting information from the source data). $(C_{Det})_{Norm}$ is defined as:

$$(C_{Det})_{Norm} = \frac{C_{Det}}{min(C_{Miss} \bullet P_{target}, C_{FA} \bullet P_{non-target})}$$

Using this formulation, the absolute value of $(C_{Det})_{Norm}$ represents the value (i.e., direct cost) of the solution.

There are two methods of estimating detection error probabilities. These are called *story-weighted* and *topic-weighted.* The story-weighted method assigns equal weight to each decision for each story and accumulates errors over all topics. The topic-weighted method accumulates errors separately for each topic and then averages the error probabilities over topics with equal weight assigned to each topic. The topic-weighted method provides a better estimate of performance. This method is also the method used in the TREC/TDT tasks.

4.2.4. Utility Measures

Park and Zhang propose performance classification based on a utility measure [87]. Linear utility is defined as

$$\text{Linear Utility} = a \bullet R_+ + b \bullet N_+ + c \bullet R_- + d \bullet N_-$$

Here R_+ represents the number of relevant and retrieved documents, N_+ represents the number of non-relevant but retrieved documents, R_- the number of relevant but non-retrieved documents, and N_- the number on non-relevant and non-retrieved documents and *a, b, c* and *d* are constant coefficients. For classifying large unstructured document collections, Park and Zhang defined linear utility measures:

$$LF_1 = 3 \bullet R_+ - 2 \bullet N_+$$

$$LF_2 = 3 \bullet R_+ - N_+$$

However, computing utility by averaging across all topics yields skewed results since a few topics can dominate the results. In order to show system performance, the concept of a scaled utility is introduced. Scaled utility is defined as:

$$ScaledUtility = \frac{max\{u(S,T), U(s)\} - U(s)}{MaxU(T) - U(s)}$$

The term $u(S,T)$ is the utility of system S for topic T, $Max\ U(T)$ is the maximum possible utility score for topic T, and $U(s)$ is the utility of retrieving s non-relevant documents. For a limited set of topics (say 10), the authors use LF_1. However, for a collection of 50 topics they use the scaled linear utility.

4.2.5. Efficacy of Precision and Recall Measures

Note that precision and recall are the traditional and very well-founded methods of evaluating document retrieval systems. However, as Greenwood [88] points out, these methods may be considered outdated. This is because precision and recall may no longer be adequate measures of effectiveness due to the way the information needs of society have changed over recent years (particularly with the use of the Internet).

Precision and recall were very good at measuring how good a system is at finding all the relevant documents within a collection (e.g., books in a library). However, this is no longer the predominant use of IR systems. IR now generally means Internet search for a document. Internet search engines work across vast document collections, ranking and returning relevant documents in much the same way that the older library retrieval systems did. However, users have fundamentally changed the way they use these kinds of systems. Users are not content to scroll through 100 relevant documents looking for information. Rather, they expect a relevant document to be in the top ten returned in response to their query so that a user can instantly see a document that appears relevant [88].

Because of this change in user expectations, Greenwood argues that what is needed is a new evaluation measure that allows systems to ranked according to how well they work in a search-engine-like application. He introduces two new measures called *position* and *answer. Position* is simply the position at which the first relevant document is ranked. Clearly the lower the position value the better the system. *Answer*, on the other hand, is the percentage of queries (for which there are relevant documents) where a relevant document is returned in the top ten, i.e., a document is returned which can answer the query. These measures are what we automatically associate with the performance of a good Internet search engine [88]. We note that CRA-based metrics did, in fact, perform better, in all circumstances, in *position* and *answer* performance.

4.2.6. Tools for IR System Performance Evaluation

One of the goals of our research is to be able to compare the performance of a CRA-based system with the performance of more traditional, frequency-based, approaches. To that end, evaluation tools are required that can be used to compare different approaches. In some cases, we have built our own simple term-frequency tools for comparing system performance. We have also utilized the Lemur toolkit for system evaluation. The Lemur toolkit [89] was developed at Carnegie-Mellon University and provides a TF-IDF implementation that is suitable for comparison testing with CRA. The following background information is provided to illustrate the Lemur toolkit capabilities.

Lemur provides a number of basic capabilities. These include the ability to construct an index for a document set and to retrieve documents using several baseline information retrieval algorithms such as TF-IDF

and Okapi. The toolkit has a number of capabilities that are not required for our present purposes (e.g., relevance feedback).

4.2.6.1. Indexing

The *indexer* indexes information about the terms in a collection of documents that can then later be accessed using either a term or a document as a reference. The indexer allows collection of term frequency, term position, and document length statistics.

The toolkit includes a number of *parsers*. However these parsers were designed to facilitate indexing many documents that are in the same file. The two most frequently used parsers are the TrecParser and the WebParser. The WebParser removes HTML tags. The parser assumes the use of NIST style format for documents.

4.2.6.2. Information Retrieval Evaluation

Lemur provides the RetEval application for information retrieval. RetEval takes a file of queries in the form:

```
<DOC #queryid>
    term1
    term2
    term3
</DOC>
```

and creates a result file containing the queryid, docid and a score for that document with respect to that query.

4.2.6.3. Evaluation of Retrieval Results

Lemur provides a Perl script called `ireval.pl` that can be used to calculate performance measurements in terms of precision and recall scores given he result file from running `RetEval` and a judgment file. A judgment file indicates which documents are relevant for each query.

The judgment file can be in a simple three-column format of `queryid`, `docid` and `relevance` (a value of 1 is assigned for a relevant document).

4.2.6.4. Precision/Recall Performance

The `ireval.pl` program is used to generate precision/recall statistics for both the CRA and TF-IDF methods.

4.2.7. Text Filtering

Text filtering is the process of classifying a stream of incoming documents arriving in an asynchronous manner to an information consumer from an information producer. The prototypical case is a news feed where the producer is a news agency (e.g., Reuters) and the consumer is a newspaper. The filtering process should block the delivery of documents that are not likely of interest. Filtering can be viewed as single-label text categorization — the classification of incoming documents into two disjoint categories, the relevant and the irrelevant. A filtering system may also further categorize incoming documents into thematic categories (e.g., a filtering system for a sports editor would categorize the articles a baseball, football, golf, etc.). SPAM E-mail removal is an example where the filter decides whether the mail is junk or not. The non-junk mail can be further classified into topical categories [90].

A text filtering system sifts through a stream of incoming information to find documents that are relevant to a set of user needs represented by profiles. Unlike a traditional search query, user profiles are persistent and represent a user's long-term information-seeking requirements. Using user feedback, a system can learn a better profile and thus improve its performance over time. For most filtering applications, the value of a document decays rapidly with time. This means that potentially relevant documents must be presented immediately to the user. There is no time available to accumulate and rank a set of documents. Evaluation must be based only on the quality of the retrieved set. Filtering differs from the search task in that documents arrive sequentially over a period of time [91].

A profile is initially specified by the user (and therefore resembles a standard query) and is updated by the system using feedback information from the user. This is termed *adaptive* filtering. The filtering task when no user profile is available is termed either *routing* or *batch* filtering. If documents need only be accepted or rejected then this constitutes batch filtering. If documents must be ranked in order of estimated relevance, this is a routing system. A routing system learns a static profile from training documents, and ranks all documents in the test set according to the profile. Filtering systems examine the test set one document at a time and must make a decision as each document is received whether to display that document to the user or not [92].

The filter can be installed at the producer end in which case it must route documents to only the interested consumer. The filter can also be installed at the consumer end to filter and classify documents. If the filter is installed at the producer end, then the system must build and maintain a profile for each user of the system. If the filter is installed at the consumer end, then only a single profile is required [90].

The term *information filtering* describes a variety of processes involving the delivery of information to peo-

ple who need it. The most common features of an information filtering system include [93].

- The system is designed for use with unstructured or semi-structured information.
- Information deals primarily with textual information.
- Filtering systems involve large amounts of data. Gigabytes of text are not uncommon.
- Filtering applications typically involve streams of incoming data, either being broadcast by remote sources (such as a newswire service) or sent directly by other sources (e.g., E-mail).
- Filtering is based on descriptions of individual or group information preferences — called profiles. These profiles typically represent long-term interests.
- Filtering is often meant to imply removal of data from an incoming stream, rather than finding data in that stream. This means that profiles may express what people want or what they do not want to see.

4.2.7.1. Comparing IR and Filtering Tasks

It is instructive to compare filtering tasks with the more traditional information retrieval processes. Belkin provides the following characterization [93, 94]:

- IR is typical concerned with single uses of the system, by a person with a one-time goal and one time query, information filtering is concerned with repeated uses of the system by a person or persons with long-term goals or interests.
- IR recognizes inherent problems in the adequacy of queries as representations of information needs. Filtering, however, assumes that profiles can be correct specifications of information interests.
- Where IR is concerned with the collection and organization of texts, filtering is concerned with the distribution of texts to groups of people or individuals.
- IR is typically concerned with the selection of texts from a relatively static database. However, filtering is mainly concerned with selection or elimination of texts from a dynamic data stream.
- IR is concerned with responding to the user's interaction with texts within a single information-seeking session. However, filtering is concerned with long-term changes over a series of information-seeking sessions.
- The timeliness of a text is usually an overriding consideration in filtering. This is usually not a concern in IR activities.
- IR has historically studied well-defined and very specific user domains (usually in science and technology). These users have generally been highly motivated in their information-seeking behavior. Filtering task are many times less well structured and across a diverse information domain (e.g., searching the web for information).

4.2.7.2. The TREC Evaluation Process

The TREC filtering track uses the RCV1 corpus provided by Reuters [95]. This corpus contains about 800,000 news stories, covering a time period in 1996 - 1997. Items in the corpus have unique identifiers and are dated but not timed. The assumption is made that the time order within one data is the same as the identifier order. Items from the first six weeks (20 August through 30 September) are taken as the training set. The remainder of the collection form the test set [91].

The adaptive filtering task is designed to model the text filtering process from the moment a profile is constructed. For the TREC evaluation, for each topic the last three relevant documents in the training set were used for this purpose. No other relevance judgments from the training set were allowed. Once a document is retrieved, the relevance assessment (when one exists) is immediately made available to the system. There is no interactive human judgment in the TREC assessment. Rather, the assessment is simulated by releasing the pre-existing relevance judgment for a document. Judgments for un-retrieved documents are never revealed to the system [91]. A complete description of the test collection and its development can be found in [92].

Systems are allowed to use the whole of the training set of documents (but no other relevance judgments than the three provided for each topic) to generate collection frequency statistics (such as IDF) or auxiliary data structures (such as automatically generated thesauri). Resources outside the Reuters collection could also be used. As documents were processed, the text are used to update term frequency statistics and auxiliary document structures even if the document was not matched to any profile [91].

In batch filtering, all the training set documents and all relevance judgments on that set are available in advance. Once the system is trained, the test set is processed in its entirety. For each topic, the system returns a single retrieved set. For routing, the training data is the same as for batch filtering, but in this case systems return a single retrieved set. For routing, the training data is the same as for batch filtering, but in this case, systems return a ranked list of the top 1000 retrieved documents from the test set. Batch filtering and routing are included to encourage participation of as many different groups as possible [91]. Because the text routing problem involves sending relevant incoming data to individuals or groups, it is essentially identical to filtering

4.2.7.3. TREC Filtering Evaluation Measures

Filtering systems are expected to make a binary decision to accept or reject a document for each profile. Therefore, the retrieved set consists of unranked list of documents. Standard measures (e.g., precision-recall curves) are not applicable. The choice of the primary measure of performance will impact the systems in a way that does not happen in ranked retrieval. While good ranking algorithms are generally independent of the evaluation measure used, good classification algorithms need to relate very strongly to the measure that it is desired to optimize. The traditional precision and recall measures of are limited utility in the text filter track since neither adaptive filtering or batch filtering return a ranked set of documents. The TREC filtering TREC uses the *F-beta* and *Linear Utility* as evaluation measures [91].

The F-beta measure is a function of recall and precision, together with a free parameter beta which determines the relative weighting of recall and precision. For any beta, the measure lies in the range zero (bad) to 1 (good). For TREC 11, a value of beta = 0.5 was chosen. This corresponds to an emphasis on precision (beta - 1 is neutral). The measure (with this choice of beta can be expressed as follows:

$$T11F = \frac{1.25 \times N_{RDR}}{N_{RTD} + 1.25 \times N_{RD}}$$

where N_{RDR} represents the number of relevant documents retrieved, N_{RTD} represents the number of retrieved documents, and N_{RD} represents the number of relevant documents. $T11F$ is defined as zero if the number of retrieved documents is zero.

Linear utility uses a credit of two for a relevant document retrieved and a debit of one for a non-relevant document retrieved and is given by

$$T11U = 2 \times N_{RDR} - N_{NRDR}$$

Here N_{NRDR} represents the number of non-relevant documents retrieved. This corresponds to the retrieval rule

Retrieve if $P(rel) > .33$

Filtering according to a linear utility function is equivalent to filtering by estimated probability of relevance.

When evaluation is based on a utility function, it is difficult to compare performance across topics. Simple averaging of the utility measure gives each retrieved document equal weight, which means that the average scores will be dominated by the topics with large retrieved sets. Furthermore, the utility scale is effectively unbounded below but bounded above. This means that a single very poor query might completely swamp any number of good queries.

For the purposes of averaging across topics, utilities are normalized by the maximum possible utility for the topic:

$$MaxU = 2 \times N_{RD}$$

Therefore:

$$T11NU = \frac{T11U}{MaxU}$$

The lower limit is some negative normalized utility, *MinNU* can be viewed as the minimum (maximum negative) utility that a user would tolerate over the lifetime of the profile. If the $T11NU$ value falls below this minimum, it will be assumed that the user stops looking at documents.

For each topic

$$T11SU = \frac{max(T11NU, MinNU) - MinNU}{1 - MinNU}$$

The *Mean*$T11SU$ is the mean of $T11SU$ over topics. Different values of *MinNU* can be chosen. The primary evaluation measure has *MinNU* = -0.5.

TREC provides official results tables that suggest a number of measures. Some of these include:

- Mean $T11SU$ over topics, over the whole period and broken down by time period for adaptive filtering. Mean T11F (F-beta, with beta - 0.5) over topics
- Mean set recall
- Mean set precision
- Zeros (number of topics for which no documents were retrieved over the period).

All means are averaged across topics. For the routing task, `trec_eval` is used for evaluating performance.

4.2.7.4. Filtering Summary

Much of the interesting research in the filtering track is concerned with development and improvement of adaptive filtering algorithms. These algorithms represent a class of machine-learning algorithms that change the filtering criteria based on user feedback. Competition in the adaptive filtering track will require developing a novel approach to providing adaptive feedback based on the CRA network. This has not been a focus of this research. However, the other two tracks (batch filtering and routing) can more readily utilize the CRA technology.

4.3. Topic Detection and Tracking (TDT)

4.3.1. Background

The purpose of Topic Detection and Tracking (TDT) is to "explore techniques for detecting the appearance of new, unexpected topics and for tracking the reappearance and evolution of them" [86]. This particular information retrieval task has significant value in intelligence and social system modeling activities. TDT defines two different objects:

- *Event*: A specific action involving specific people/ entities on a specific date (e.g. World Trade Center attack on 9/11/01)
- *Topic/Story*: An initial event and all subsequent related events (All events related to 9-11)

There are two basic tasks in TDT:

- *Tracking*: Train the system with N(t) documents that are "on topic". Evaluate each subsequent article as on or off topic. Only evaluate to one topic at a time.
- *Detection*
    a. Retrospective: Identify all events in a corpus of stories (texts), through clustering
    b. On-line new event detection: Identify the appearance of a new event that does not belong to any existing event (cluster).

Two other tasks, *First Topic Detection* and *Link Detection*, are special cases of tracking and detection. The third basic task, *Segmentation*, involves parsing a continuous text stream into distinct, discrete stories (texts). This function would be very important in applications involving transcription.

4.3.2. Performance Evaluation

Performance of TDT tasks is evaluated by examining misses and false alarms. Typically a cost equation is used that allows misses and false alarms to be weighted differentially into a single metric. The weights typically take into account the a priori probability of a text being on-target (and these probabilities are known to the method *a priori*). Additionally, the metric is further normalized so as to present costs scaled to a 0.1 to 1.0 scale. For details see pp. 143-144 of [24].

A corpus is identified and human judges identify a finite number of events within the corpus, and texts that are associated with those events. The corpus is then broken into two parts, for training and evaluation. A method is trained to identify what is on-target via the training data, and then the method must tag subsequent texts as on or off-target. The machine-generated results are then compared to the human judgments. Detection is currently run completely unsupervised (i.e. with no training).

The Linguistics Data Consortium, a research group affiliated with University of Pennsylvania, in collaboration with the natural language group at NIST, have sponsored a TDT conference since 1997. TDT has used a variety of corpora over the years:

- 1997 (1): TDT-1; 16,000 stories; 1994 Reuters and CNN (transcribed)
- 1998 (2): TDT-2; 60,000 stories; AP, NYT, CNN, AC, VOA, PRI (written and transcribed)
- 1999 (3): TDT-3; TDT-2 + 8 new English sources from 1998 and 3 Mandarin sources (in Mandarin and English-translation)
- 2000 (4): Used TDT-3 corpus
- 2001 (5): Expanded TDT-3 with additional stories, events, links
- 2002 (6): TDT-4; 45,000 stories from end of 2000; eight English, seven Chinese, and four Arabic sources
- 2003 (7): Used TDT-2, -3, and -4 for various tasks

There are a few key research teams in TDT (Linguistics Data Consortium/University of Pennsylvania, Carnegie-Mellon University, University of Massachusetts, University of Maryland, National Taiwan University, Chinese University of Hong, Kong, University of Iowa, RMIT (Australia), Texas A&M University, GE (Oasis product,) Dragon Systems, and IBM. In general, the field is small and has only recently expanded to Europe and Asia. The number of commercial entities is small.

All of the methods examined were based on vector scores, either based on straight frequency or an IDF/ Okapi-like score. Methods differed therefore in how they normalized their scores with respect to text length, how they trained, and how the clustered and scored for similarity. In general, researchers have found that:

- The nature of the *event* (e.g. scope) can have a large impact on relevancy scores
- Phrases tend to perform better over words; using *named entities* can help
- There are problems with clustering the translated documents
- A method good for one evaluation task may not perform well on another, and for any given method, its parameters need to be set differently for different evaluation tasks.

4.4. Experimental Design

The purpose of these experiments is to compare CRA to traditional frequency-based methods on perfor-

mance in tasks related to ad hoc retrieval and topic tracking. Our experiment contains three variables at two levels each, for a total of 2^3 or 8 experimental conditions:

- Method of Determining Relevance: The main variable in the experiment is the method used to determine whether or not a text is relative to a query (or equivalently, a topic statement). We examine metrics based on word influence (via CRA) versus metrics based on word frequency.
- Nature of Task: Determining relevancy of a document requires some judgment about whether a similarity score is large enough to warrant a proclamation of relevancy. These judgments are typically based on whether a score is above or below a particular threshold. A key question then becomes: Is the threshold constant across all queries, or is it allowed to change for each different query? The former condition is more general and is easier to implement; the latter may be viable in very specific situations where scope is narrow and definable and training can occur. In the language of TREC and TDT, the use of a single threshold across all queries measures performance "between topics", or tracking, while using a different threshold for each query measures performance "within topics", or retrieval. While the "within topics" metrics are common within TREC and TDT, we believe the tracking (i.e. single threshold) performance to be much more important from a practical standpoint.
- Corpus: We wished to experiment with two corpora of texts that had significantly different features, in order to enhance the generalizability of our results. The CACM corpus contains relatively short queries and texts and is narrow in scope of content. The TDT-3 corpus contains relatively long queries and texts and is broad in scope of content. Both corpora have been used extensively in information retrieval research; this allows comparison to other's results.

Performance is measured in a variety of ways:

- Recall-precision performance, as described earlier in Section 4.
- Correlation between relevancy judgment and metric.
- Determination of an optimal decision rule based on linear and non-linear (i.e. multi-layer perceptron) models.

4.5. CACM Corpus

4.5.1. Background

The CACM corpus consists of articles taken from the Communications of the ACM [96]. The corpus also contains a set of full text queries of varying length, as well as a set of relevancy judgments matching the queries to the texts. The document portion of the corpus contains 3204 total items. However only about half (1586) of these have an accompanying abstract. For these cases, abstracts have a mean of 93 words, with a standard deviation of 47.29. There are 64 queries. They have a mean length of 22 words, with a standard deviation of 16. In many cases, these queries will be too short to use with standard CRA. There are 796 judgments in the relations file. Abstracts are present for 650 (81%) of these cases.

CRA analyses of the CACM corpus were run to assess ad hoc retrieval and topic tracking performance of CRA on a standard dataset. After extraction of the CACM documents, it was determined that 1587 of the entries contained abstracts, and therefore provided enough text for meaningful analysis using CRA. These documents and the queries were extracted from the corpus and analyzed using standard procedures. Many of the queries were quite short.

Table 2 shows the various data that were collected for each experimental case of a query-text pair; this includes scores for various metrics based on either word frequency or word/word-pair influence. Note that we tested a number of metrics based on an inverse document frequency (IDF) weighting, as described earlier.

We found in pilot experiments that our initial influence-based metrics did not perform well because of computational problems associated with terms that were present in the text but had an influence value of 0.0. These zero influence terms can occur when queries only consist of a phrase or a single sentence. Therefore two different runs were made for each experimental case: One where each influence value has 1.0E-6 added to it as it was read into the computations ("plus epsilon"), and a second where 1.0 was added ("plus one"). Across all subsequent analysis we found that the *plus one* metrics were superior to the *plus epsilon* — therefore from hereon, results will only refer to the *plus one* metrics, and that label will be dropped.

Also, two different databases were constructed and analyzed in order to examine the effect of the disproportionate amount of irrelevant compared to relevant documents. While the results differ in magnitude, they do not in pattern, and so we will refer only to the results from analysis of the complete data set.

4.5.2. Basic Results

Table 2 shows the summary statistics from the experiment. We note the following

- Only a very small fraction of cases are judged relevant (0.4%).
- Average criterion size is only slightly larger (70%)

Table 2. Summary Statistics of CACM Corpus

	Valid N	Mean	Minimum	Maximum	Std.Dev.
relevance	205056	0.00388	0.000000	1.0000	0.06218
querysize	205056	10.09375	2.000000	33.0000	6.23930
querygraphinfluence	205056	0.30731	0.000000	1.0000	0.24476
criterionsize	205056	17.26998	1.000000	118.0000	15.95350
criteriongraphinfluence	205056	0.29917	0.000000	1.0000	0.25231
wordscore	205056	0.06724	0.000000	0.4254	0.04318
verbscore	205056	0.01048	0.000000	0.8165	0.04024
frequencyscore	205056	0.07090	0.000000	0.5937	0.05067
pairmatches	205056	0.02414	0.000000	12.0000	0.19761
pairfrequencyscore	205056	0.00094	0.000000	0.5341	0.01008
pairresonancescore	205056	0.00072	0.000000	0.4182	0.00702
idf_wordscore	205056	0.02657	0.000000	0.3666	0.02556
idf_verbscore	205056	0.00732	0.000000	0.7845	0.03031
idf_frequencyscore	205056	0.02664	0.000000	0.3677	0.02730
idf_pairresonancescore	205056	0.00051	0.000000	0.4757	0.00535

than query size, and both are relatively small. The largest criterion is about the size of a medium sized newspaper article.

- The largest number of pair matches is 12, again indicating a relatively small "opportunity space" for networks to resonate.

Next, a correlation analysis was performed. Table 3 shows the correlation between each of the data measures captured and the human judgment of relevancy. Higher correlation means better performance..

- Frequency and influence based metrics are equivalent in correlation level.
- The IDF normalization improves the performance of all metrics, slightly.
- Edge-based metrics are over 2*x* more correlated with relevance than nodes-based metrics (frequency, influence).
- Verbs are relatively unimportant.
- IDF normalized scores are about 80 percent correlated with their raw counterparts.
- Frequency and influence are correlated 88 percent. (Note: This is much larger than we have seen in other historical analyses — this is probably due to the small text size.)
- The size or graph influence of the query is not correlated with any other metric.

Node-level metrics (frequency, influence) are significantly (positively) correlated with criterion size and graph influence; edge-level statistics are only weakly correlated with criterion size. Thus, node-level metrics are more robust to short criterion sizes

Next, a t-test is performed, to see the difference in mean associated with relevant and irrelevant cases; see Table 4 for results. The *Mean-0* corresponds to the mean of the metric for cases judged irrelevant, and the *Mean-1* for cases judged relevant. A desireable result would be for *Mean-1* to be statistically significantly larger than *Mean-0* — this would mean it would be easier to assign a single threshold to discriminate relevant from irrelevant documents. All metrics except query size have averages that are statistically different. The t-values can be used to judge relative magnitude.

The edge-level metrics differ most greatly in mean between irrelevant and relevant cases, i.e., three to four times more than any of the node-level statistics.

Other results mimic the correlation analysis: verbs don't matter much, IDF helps slightly. Influence has a higher t-value than frequency, but frequency-IDF is slightly higher than influence-IDF, thus IDF helps frequency slightly more than it helps influence.

A variety of additional modeling was performed. Logit regression models were deemed to be inappropriate because of the high degree of co-linearity in the independent variables. Likewise, discriminant analysis was effected by the co-linearity. In order to isolate the difference between node and edge-level metrics, the analysis was simplified to two metrics: idf-frequency-score, and idf-pairresonancescore. Results are shown in

Table 7. The p-values indicate that both are significant in discriminating relevant from irrelevant documents, and they are only two percent redundant with one another.

Given two outcomes in the dependent variable, one classification function was extracted. The eigenvalues indicate that idf-pair resonance score is weighted about 3.5 times more than idf-frequency score within the classification function. Next, similar analyses were performed using only each of the metrics, alone/

Table 6 shows the percent correctly classified using a classification function using each of the metrics alone, and both together.

The results indicate that a classification function based on idf-frequency has slightly better performance for irrelevant documents; and a function based on pair

Table 3. Correlations Between Information Retrieval Metrics Using the CACM Corpus

	A	b	C	d	e	f	g	h	i	j	k	l	m	n	o
a/relevance	1.00	0.00	-0.01	0.04	0.02	0.05	0.02	0.05	0.17	0.15	0.16	0.07	0.02	0.07	0.18
b/querysize	0.00	1.00	0.18	0.00	0.00	-.01	0.01	-.01	0.05	-.01	-.01	0.05	0.01	0.05	0.00
c/ querygraphinfluence	-.01	0.18	1.00	0.00	0.00	0.00	0.00	0.00	0.00	-.01	-.01	-.01	0.00	0.00	-.01
d/ criterionsize	0.04	0.00	0.00	1.00	0.35	0.54	0.23	0.48	0.11	0.05	0.06	0.56	0.21	0.53	0.06
e/ criteriongraphinfluence	0.02	0.00	0.00	0.35	1.00	0.20	0.12	0.17	0.03	0.01	0.01	0.20	0.11	0.19	0.02
f/wordscore	0.05	-.01	0.00	0.54	0.20	1.00	0.16	0.88	0.15	0.12	0.14	0.82	0.15	0.78	0.13
g/verbscore	0.02	0.01	0.00	0.23	0.12	0.16	1.00	0.14	0.03	0.02	0.02	0.18	0.94	0.18	0.02
h/ frequencyscore	0.05	-.01	0.00	0.48	0.17	0.88	0.14	1.00	0.16	0.13	0.15	0.69	0.13	0.79	0.14
i/pairmatches	0.17	0.05	0.00	0.11	0.03	0.15	0.03	0.16	1.00	0.68	0.77	0.15	0.03	0.17	0.77
j/ pairfrequencyscore	0.15	-.01	-0.01	0.05	0.01	0.12	0.02	0.13	0.68	1.00	0.93	0.12	0.02	0.12	0.90
k/ pairresonancescore	0.16	-.01	-0.01	0.06	0.01	0.14	0.02	0.15	0.77	0.93	1.00	0.13	0.02	0.15	0.97
l/ idf_wordscore	0.07	0.05	-0.01	0.56	0.20	0.82	0.18	0.69	0.15	0.12	0.13	1.00	0.17	0.91	0.13
m/ idf_verbscore	0.02	0.01	0.00	0.21	0.11	0.15	0.94	0.13	0.03	0.02	0.02	0.17	1.00	0.17	0.02
n/ idf_frequencyscore	0.07	0.05	0.00	0.53	0.19	0.78	0.18	0.79	0.17	0.12	0.15	0.91	0.17	1.00	0.15

Table 4. T-Test of Means Between Relevant and Irrelevant Cases for Different Information Retrieval Metrics, CACM Corpus

	Mean-0	Mean-1	t-value	df	p
querysize	10.09411	10.00251	0.4134	205054	0.679337
querygraphinfluence	0.30749	0.26137	5.3063	205054	0.000000
criterionsize	17.23135	27.18090	-17.5746	205054	0.000000
criteriongraphinfluence	0.29891	0.36611	-7.5013	205054	0.000000
wordscore	0.06710	0.10278	-23.2999	205054	0.000000
verbscore	0.01044	0.02088	-7.3032	205054	0.000000
frequencyscore	0.07074	0.11052	-22.1307	205054	0.000000
pairmatches	0.02208	0.55276	-76.6959	205054	0.000000
pairfrequencyscore	0.00084	0.02577	-70.4570	205054	0.000000
pairresonancescore	0.00064	0.01922	-75.4824	205054	0.000000
idf_wordscore	0.02647	0.05377	-30.1496	205054	0.000000
idf_verbscore	0.00729	0.01526	-7.4032	205054	0.000000
idf_frequencyscore	0.02652	0.05839	-32.9512	205054	0.000000
idf_pairresonancescore	0.00045	0.01589	-82.5689	205054	0.000000

Table 5. Results from Discriminant Analysis, Comparison of Frequency and Pair Resonance Metrics, CACM Corpus

	Wilks'	Partial	F-remove	p-level	Toler.	1-Toler.
idf_frequencyscore	0.967822	0.997801	451.876	0.00	0.980412	0.019588
idf_pairresonancescore	0.994733	0.970807	6166.055	0.00	0.980412	0.019588

resonance has far superior performance to frequency. Combining both yields slightly better performance. Given the small amount of improvement in false alarms given by IDF-frequency, it is not likely that examining any kind of hybrid metric will improve results.

In summary:

- Edge-based metrics appear to be superior to node-based metrics. Their means differ more greatly between relevant and irrelevant cases, and they construct better performing classification functions.
- There is not much difference between the two types of node-based metrics, frequency and influence.
- IDF-normalization slightly improves performance of any metric.
- Verbs are not useful in determining relevance.
- A classification function based on pair-resonance (idf-normalized, fraction common edged weighted by influence) achieves a false alarm rate of 0.86 percent, and a hit rate of 24 percent.
- Results seem to indicate that a hybrid metric would be unlikely to outperform any of the individual metrics.

4.5.3. Retrieval

Next, recall-precision performance is measured by determining the average precision within a topic (*ad hoc retrieval*). Figure 11 shows recall-precision performance for the two *best competing* methods, as analyzed in the previous section:

- TF-IDF: Traditional term frequency — inverse document frequency metric
- IDF-Pairresonance: fraction of common edges in CRA networks, weighted by node influence value plus 1.0, IDF normalized.

TF-IDF slightly outperforms IDF-Pairresonance, especially in the recall range of 10-50 percent. Table 7 shows for each topic/query: number relevant (per judgments), average precision of IDF-Pairresonance, precision of IDF-Pairresonance at 10 retrieved documents, average precision of TF-IDF, precision of TF-IDF at 10 retrieved documents, the difference in the two precision values, the number of non-zero IDF-Pairresonance scores, and the log of the ratio of non-zero IDF-Pairres-

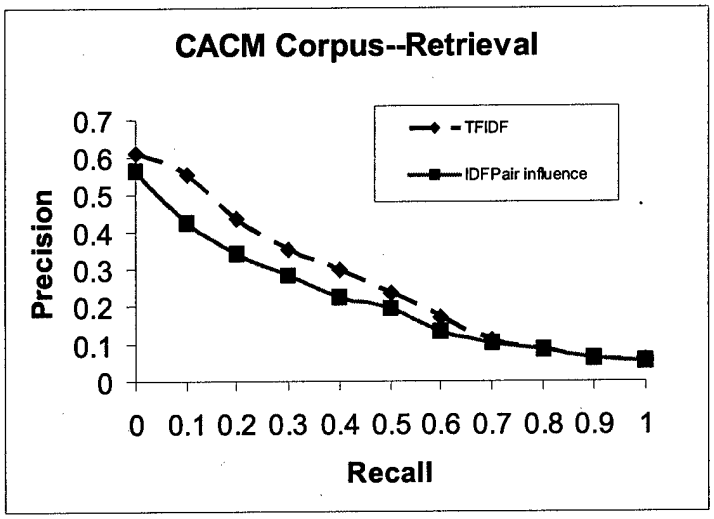

Figure 11. Recall-Precision Graph Comparing TF-IDF and IDF-Pair Resonance, CACM Corpus

onance scores to relevant documents. Only topics with one or more relevant documents are shown.

Average precision for IDF-Pairresonance is 0.205, and for TF-IDF is 0.250. Precision after 10 documents have been retrieved (the *web search* test) is similar. Average precision for IDF-Pairresonance can be improved to around 0.23 by using IDF-Wordresonance to break ties in rank. Regression models were used to see if there was any systematic pattern underlying TF-IDF's superior precision. No causal factors were identified.

Thus we conclude that from a conventional retrieval perspective, TF-IDF slightly outperforms IDF-Pairresonance.

4.5.4. Tracking

While TF-IDF is shown to be slightly superior from the conventional retrieval perspective, this is not of critical relevance to the objectives of this project because the retrieval perspective does not capture the elements of an actual news streaming environment as well as a filtering perspective does. The ad hoc retrieval task involves ranking every document in a corpus relative to its similarity to a topic query. The retrieval perspective and filtering perspective differ in two important regards:.

- In retrieval, every document is ranked according to similarity with the topic. However, there is no specific decision about whether a document is relevant or not. In filtering, a binary decision (relevant, irrelevant) is made; no distinction is made within each category (i.e. indicating degrees of relevancy). In a news streaming environment, it is important to be able to filter out irrelevant documents; if only a ranked list is given to the user, they must still sort out which documents are actually relevant or not.
- In retrieval, precision-recall performance is determined within each topic query. This means that each topic query is treated independently of one another — each has its own *threshold.* This is only a valid assumption in applications where a topic statement serves as a static filter over a long period of time. In a news streaming environment, it is unrealistic to assume that the decision criterion for determination of relevancy can be uniquely set and optimized for each different topic query. Instead, a single threshold value should be used across all query/document comparisons.

Table 6. Optimal Decision Rule, Comparing Rules Based on TF-IDF and IDF-Pair Resonance, CACM Corpus

idf_freq & idf_prres	% correct
irrelevant	99.13003
relevant	25.37688
idf_frequency only	% correct
irrelevant	99.93587
relevant	1.50754
idf_pairresonance only	% correct
irrelevant	99.13835
relevant	24.37186

In order to compare the tracking perspective to the retrieval perspective, recall-precision graphs were generated by examining the results across all the queries

simultaneously: (a) determine rank across all topics, (b) determine recall-precision performance across all topics. Figure 12 shows the tracking results for four different methods:

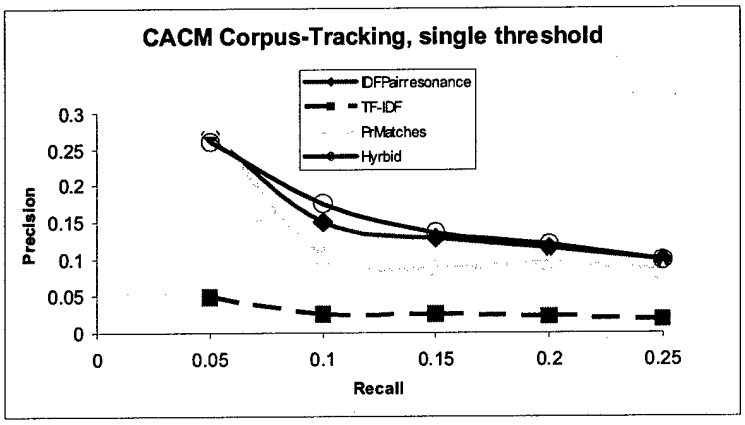

Figure 12. Recall Precision Graph Comparing CRA and Frequency-Based Metric, CACM Corpus, Single Threshold (Tracking)

- TF-IDF
- IDF-Pairresonance
- Number of matching pairs (edges). (This method has the advantage that it does not require computation of resonance, nor IDF normalization — it is only based on parsing and network construction and is therefore very computationally efficient compared to the other methods.)
- Hybrid (rank determined by number of matching pairs, then by IDF-Pairresonance, then by TF-IDF).

CRA-based methods significantly outperform TF-IDF. At 5 percent recall, precision for the three CRA-based methods is 5-6 times greater than for TF-IDF. The three CRA-based methods perform almost equivalently at 5 percent recall; the hybrid method does slightly better in precision in the 10-20 percent recall range. Considering these results with those found previously (see Section 4.5.2) demonstrates that CRA-based methods have superior discriminatory powers across a range of queries. We hypothesize two reasons:

- Metrics based on CRA network edges (i.e., *paired* methods) were shown to be robust to (i.e. have low variation in response to) the size of the topic query and criterion. Node based metrics (such as TF-IDF) are significantly more sensitive to topic query and criterion size. This means that edge-based metrics can function with a single threshold value for determining significance, whereas node-based metrics need different threshold values for different query and criterion sizes.
- Metrics based on CRA were shown to be statistically more *well-behaved* than metrics based on frequency. This means that CRA-based metrics are better able to function with using a single threshold value for determining relevance.

In summary, while TF-IDF slightly outperforms CRA-based methods in conventional ad hoc retrieval, CRA-based methods overwhelmingly outperform TF-IDF in terms of filtering capability. Remember that filtering capability is the relevant performance metric for the news streaming environment.

4.6. TDT-3 Corpus

4.6.1. Background

The TDT-3 corpus, which spans October through December 1998 and contains news data collected from 11 news sources in two languages (English, Mandarin) including newspaper and transcribed radio and TV contains approximately 30,000 articles. This corpus differs from the CACM corpus in that its topic statements and target texts are longer; the corpus is much more diverse in topical content; and there are many more texts in the corpus, and thus the fraction of relevant texts is significantly lower.

We adhered to the TDT2001 "Task Definition and Evaluation Plan" as published by NIST [86]. Specifically, we will follow the "basic required conditions":

- Train using only English documents
- Test all languages (use Mandarin translations)
- Number of on-topic training stories: one (We did experiments using 2 and 4 training stories, and the pattern of final results were consistent. Therefore, only the results corresponding to one on-topic training story will be discussed).
- Number of off-topic training stories: zero.
- Source data: Text sources and manual transcription of audio sources

Table 7. Recall-Precision Statistics for CACM Corpus, All Queries

Topic	# relevant	PrRes-precision	PrRes-10-prec	TF-precision	TF-10-prec	Dprecision	D-Prec10	#nozeroPrRes	nonzero/rel
1	5	0.058	0	0.07	0	-0.012	0	211	1.63
2	3	0.001	0	0.001	0	0	0	0	-2.00
3	6	0.089	0.1	0.05	0.1	0.039	0	6	0.00
4	12	0.118	0.1	0.18	0.2	-0.062	-0.1	11	-0.04
5	8	0.041	0	0.02	0	0.021	0	27	0.53
6	3	0.094	0.1	0.26	0.2	-0.166	-0.1	1	-0.48
7	28	0.233	0.2	0.33	0.6	-0.097	-0.4	20	-0.15
8	3	0.013	0	0.02	0	-0.007	0	73	1.39
9	9	0.069	0.1	0.31	0.2	-0.241	-0.1	69	0.88
10	35	0.314	0.6	0.52	1	-0.206	-0.4	5	-0.85
11	19	0.267	0.6	0.36	0.6	-0.093	0	42	0.34
12	5	0.253	0.1	0.41	0.2	-0.157	-0.1	62	1.09
13	11	0.132	0.2	0.18	0.2	-0.048	0	6	-0.26
14	44	0.063	0.2	0.08	0.1	-0.017	0.1	37	-0.08
15	10	0.154	0.1	0.18	0.2	-0.026	-0.1	8	-0.10
16	17	0.085	0.2	0.1	0.2	-0.015	0	71	0.62
17	16	0.092	0.1	0.1	0.3	-0.008	-0.2	11	-0.16
18	11	0.067	0.1	0.11	0.1	-0.043	0	38	0.54
19	11	0.489	0.5	0.3	0.3	0.189	0.2	7	-0.20
20	3	0.124	0.1	0.13	0.1	-0.006	0	80	1.43
21	11	0.02	0	0.04	0.1	-0.02	-0.1	14	0.10
22	17	0.324	0.5	0.33	0.5	-0.006	0	17	0.00
23	4	0.601	0.2	0.62	0.3	-0.019	-0.1	13	0.51
24	13	0.06	0.2	0.15	0.2	-0.09	0	2	-0.81
25	51	0.24	0.6	0.15	0.2	0.09	0.4	157	0.49
26	30	0.1	0.2	0.13	0.1	-0.03	0.1	83	0.44
27	29	0.1	0.2	0.22	0.4	-0.12	-0.2	110	0.58
28	5	0.27	0.4	0.66	0.3	-0.39	0.1	2	-0.40
29	19	0.47	0.8	0.63	0.9	-0.16	-0.1	1	-1.28
30	4	0.39	0.3	0.23	0.2	0.16	0.1	6	0.18
31	2	0.12	0.1	0.62	0.2	-0.5	-0.1	20	1.00
32	3	0.06	0	0.3	0.2	-0.24	-0.2	30	1.00
33	1	0.04	0	0.05	0	-0.01	0	33	1.52
36	20	0.22	0.3	0.1	0.2	0.12	0.1	13	-0.19
37	12	0.07	0.2	0.15	0.2	-0.08	0	30	0.40
38	16	0.4	0.6	0.39	0.6	0.01	0	23	0.16
39	12	0.17	0.4	0.19	0.3	-0.02	0.1	41	0.53
40	10	0.36	0.4	0.43	0.4	-0.07	0	26	0.41

Table 7. Recall-Precision Statistics for CACM Corpus, All Queries (Continued)

Topic	# relevant	PrRes-precision	PrRes-10-prec	TF-precision	TF-10-prec	Dprecision	D-Prec10	#nozeroPrRes	nonzero/rel
42	21	0.18	0.4	0.07	0.2	0.11	0.2	48	0.36
43	41	0.07	0.2	0.05	0.1	0.02	0.1	17	-0.38
44	17	0.06	0	0.14	0.2	-0.08	-0.2	11	-0.19
45	26	0.1	0.1	0.21	0.4	-0.11	-0.3	94	0.56
48	12	0.4	0.3	0.11	0.2	0.29	0.1	137	1.06
49	8	0.06	0	0.1	0	-0.04	0	83	1.02
57	1	1	0.1	1	0.1	0	0	3	0.48
58	30	0.07	0.2	0.2	0.4	-0.13	-0.2	67	0.35
59	43	0.16	0.3	0.21	0.5	-0.05	-0.2	65	0.18
60	27	0.16	0.2	0.3	0.6	-0.14	-0.4	84	0.49
61	31	0.23	0.2	0.22	0.4	0.01	-0.2	54	0.24
62	8	0.04	0	0.06	0	-0.02	0	2	-0.60
63	12	0.34	0.4	0.23	0.2	0.11	0.2	29	0.38
64	1	1	0.1	1	0.1	0	0	0	-2.00
		0.205	0.217	0.250	0.256				

- Story boundaries: As given by reference boundaries

4.6.2. Basic results

Table 8 shows the summary statistics for each of the metrics which we collected data. Note that the total experiment comprises approximately 2.5 million text-query pairs. We see that about one in every 500 documents is relevant. We also collected some data that was not collected during the CACM corpus experiment, specifically: number of matching and non-matching CRA network nodes and edges between the text and query, both unweighted and weighted by nodal influence values.

Table 9 shows the correlation between the metrics. The following is noted:

- Correlation between CRA-based word pair resonance and relevance (0.33) is slightly higher than between frequency and relevance (0.30). The next highest correlates with relevance are also the pair metrics (matching edges/pairs, weighted and unweighted). None of the other metrics collected relate very strongly to influence, so only frequency and word pair resonance will be used in further testing.
- Word influence and word frequency are strongly correlated (0.80) which is slightly less than in the CACM corpus. Word pair influence (resonance) and word frequency are less correlated (0.60) showing that they contain somewhat different information.

Optimal decision rules were found for word frequency and pair resonance, using data across all the queries, i.e. using a single threshold:

- Frequency:
  a. Determine resonance based on word frequency
  b. If resonance > 0.03, relevant
- Pair influence:
  a. Determine resonance based on pair word influence
  b. If resonance > 0.19, relevant

Table 10 shows relative performance. We can see that both have a trivially low false alarm rate; but pair influence does significantly better in terms of percent missed relevant documents (28.7%) compared the miss rate for frequency (41.4%). Thus, pair influence yields a 31% reduction in misses. This is a very significant difference from a practical standpoint.

In order to more systematically examine these optimal filters, a search was used to identify the best linear and nonlinear models for classifying texts as on-topic, across all topics. The set of texts was split into thirds for the three phases of analysis: training, validation, and testing. We found that while a nonlinear model (multilayer perceptron) had the best performance, with an error rate of 8.6%, the best linear model can achieve a 9.4% error rate, with approximately a 2-1 weight ratio

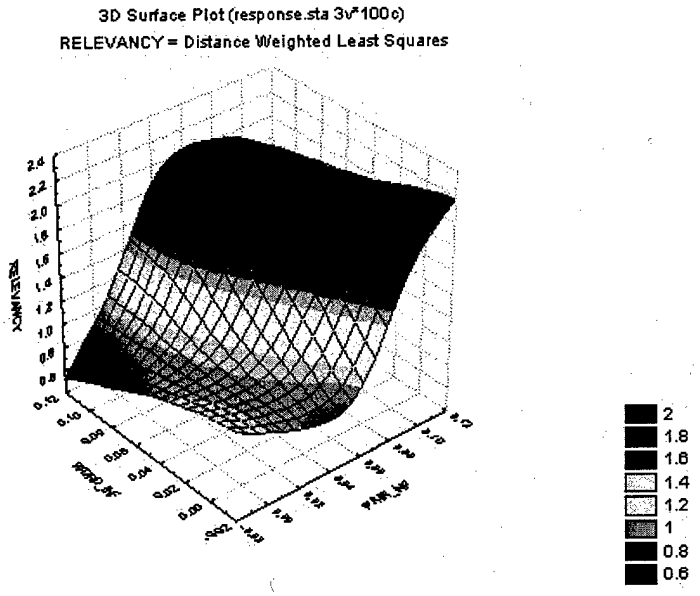

Figure 13. Response Surface of Optimal Decision Rule, TDT-3 Corpus

between pair influence and word influence. Figure 13 shows the response surface related to that optimal filter.

4.6.3. Retrieval

First, we examined recall-precision performance by calculating recall-precision graphs for each topic, and then averaging these values across all topics. Figure 14 shows average precision across all 84 topics.

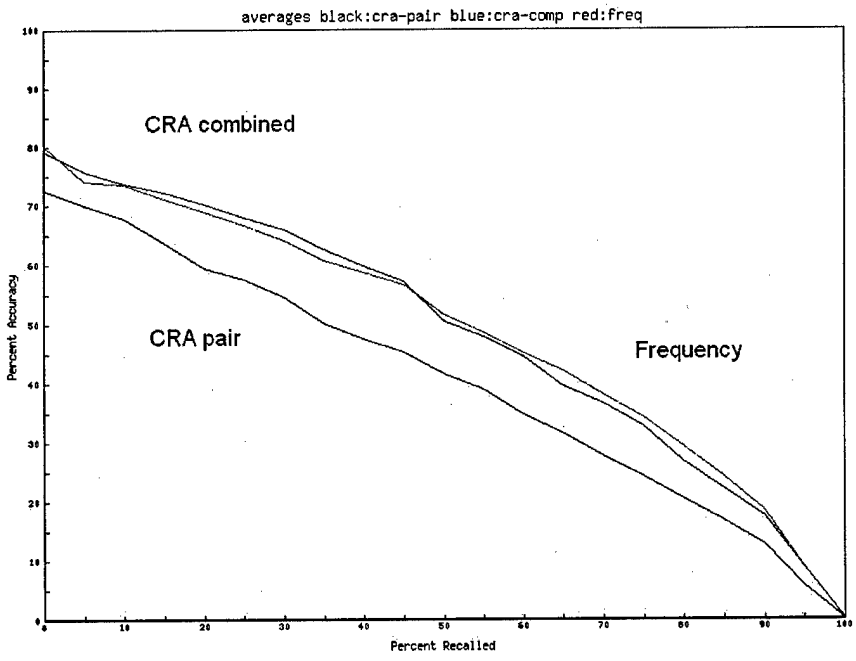

Figure 14. Recall-Precision Graphs Comparing CRA and Frequency-Based Metrics, TDT-3 Corpus

Table 8. Summary Statistics, Information Retrieval Metrics, TDT-3 Corpus

	Valid N	Mean	Minimum	Maximum	Std.Dev.
Relevance	2595825	0.0024	0.00000	1.000	0.0491
Word influence	2595825	0.0524	0.00000	1.000	0.0397
Squareroot word influence	2595825	0.2035	0.00000	1.000	0.1050
Pair influence	2595825	0.0019	0.00000	1.000	0.0074
Frequency	2595825	0.0419	0.00000	1.000	0.0464
Matching nodes	2595825	5.8578	0.00000	277.000	7.1923
Non-matching nodes	2595825	203.2291	0.00000	1559.000	122.3129
Total nodes	2595825	209.0869	17.00000	1685.000	127.3937
Matching nodes weighted by influence	2595825	12.1327	0.00000	561.057	14.7044
Nonmatching nodes weighted by influence	2595825	208.8448	0.00000	1564.999	122.6868
Total influence	2595825	220.9774	18.64836	1818.498	133.2408
Matching edges	2595825	0.2774	0.00000	486.000	1.2668
Non-matching edges	2595825	371.2800	0.00000	5124.000	288.3429
Total edges	2595825	371.5574	16.00000	5130.000	288.5503
Matching edges weighted by influence	2595825	0.8299	0.00000	1073.562	3.8449
Nonmatching edges weighted by influence	2595825	425.6015	0.00000	5766.873	346.3661
Total edge influence	2595825	426.4314	16.47314	5785.537	347.0329

We see results similar to what we found in the CACM corpus:

- Frequency and a combined metric of word-pair influence have the same average precision across the full range of recall (52.5%). Word-pair influence do better in the more important range of "early" recall.

A combined metric of word and pair influence does better than either metric by itself.

4.6.4. Tracking

Second, we examine recall-precision performance across all the topic queries by pooling all text-query pairs into one data set and using a single threshold to determine relevance. Results differ from what we found with the CACM corpus.

- The combined word-pair influence does about 7% better with *immediate recall* — the top-rank (or first) documents retrieved where precision is most important (i.e., 90 versus 85 percent precision).
- Overall though, frequency does slightly better, with a 36.2% average precision compared to 34.0% for word-pair influence. See Figure 15.

4.7. Summary

The purpose of this task was to determine whether metrics based on CRA had comparable or better performance when compared to traditional, frequency-based metrics in terms of the basic information retrieval tasks of ad hoc retrieval and filtering. In order to do so, we implemented a 2^3 factorial experimental design that had the following factors:

- *Method of Determining Relevance*: Metrics based on word and pair influence (via CRA) versus metrics based on word frequency.
- *Nature of Task*: Use a single threshold across all queries (tracking), versus using a different threshold for each query (ad hoc retrieval).
- *Corpus*: CACM corpus, which contains relatively

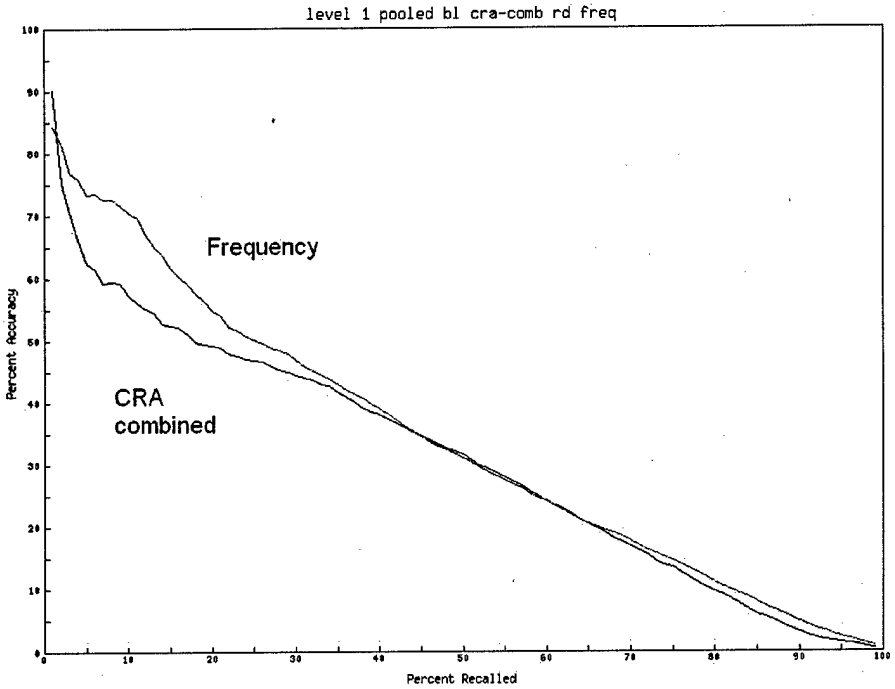

Figure 15. Recall-Precision Graph Comparing CRA and Frequency-Based Metrics, TDT-3 Corpus, Single Threshold (Tracking)

short queries and texts and is narrow in scope of content; and TDT-3 corpus, which contains relatively long queries and texts and is broad in scope of content.

Performance was measured by examining the manner in which recall and precision trade off with each another. This was done in three different ways:

(1) Finding the correlation between a human judgment of document relevancy and a metric score;

(2) Determining an optimal decision rule and measure its miss and false alarm rate;

(3) Using traditional TREC-based recall-precision curves.

Table 11 summarizes these results.

We can see in the eight different conditions where we performed a comparison, CRA was significantly better in one-half of these, and competitive in the other half. In the two most basic measures — correlation with relevance and an optimal decision rule — CRA-based metrics are significantly better. In retrieval and tracking, results are mixed. There is little difference in the retrieval task performance. In filtering, CRA is significantly better across the board in tests using the CACM corpus and in testing using the TDT-3 corpus, performed better than frequency-based approaches during the most important portion of the test, early recall.

In summary, we conclude that the experimental evidence suggests that metrics based on CRA perform better in information retrieval and text mining tasks than metrics based on word frequency. The best CRA-metrics are based either on word-pair influence or a combi-

Table 9. Correlation Between Information Retrieval Statistics, TDT-3 Corpus

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
A. relevance	1.00	0.19	0.11	0.33	0.30	0.10	0.00	0.00	0.11	0.00	0.01	0.28	0.01	0.01	0.29	0.01	0.01
B. wordres	0.19	1.00	0.93	0.48	0.80	0.77	0.37	0.40	0.78	0.37	0.42	0.45	0.38	0.38	0.44	0.36	0.37
C. sqrt wordres	0.11	0.93	1.00	0.34	0.70	0.68	0.40	0.43	0.69	0.40	0.45	0.31	0.39	0.39	0.31	0.38	0.38
D. pairres	0.33	0.48	0.34	1.00	0.60	0.27	0.02	0.03	0.28	0.01	0.04	0.76	0.04	0.04	0.75	0.03	0.04
E. freq	0.30	0.80	0.70	0.60	1.00	0.53	0.18	0.20	0.55	0.18	0.23	0.51	0.18	0.18	0.53	0.16	0.17
F. wordmatches	0.10	0.77	0.68	0.27	0.53	1.00	0.69	0.72	1.00	0.69	0.75	0.43	0.69	0.70	0.44	0.67	0.68
G. wordnonmatches	0.00	0.37	0.40	0.02	0.18	0.69	1.00	1.00	0.69	1.00	1.00	0.14	0.96	0.96	0.15	0.94	0.94
H. wordtotal	0.00	0.40	0.43	0.03	0.20	0.72	1.00	1.00	0.72	1.00	1.00	0.16	0.96	0.96	0.17	0.94	0.94
I. wordmatchinfluence	0.11	0.78	0.69	0.28	0.55	1.00	0.69	0.72	1.00	0.69	0.74	0.44	0.69	0.69	0.44	0.67	0.67
J. wordnonmatchinfluence	0.00	0.37	0.40	0.01	0.18	0.69	1.00	1.00	0.69	1.00	1.00	0.14	0.96	0.96	0.15	0.94	0.94
K. wordtotalinfluence	0.01	0.42	0.45	0.04	0.23	0.75	1.00	1.00	0.74	1.00	1.00	0.17	0.96	0.96	0.18	0.94	0.94
L. pairmatches	0.28	0.45	0.31	0.76	0.51	0.43	0.14	0.16	0.44	0.14	0.17	1.00	0.16	0.17	0.94	0.15	0.16
M. pairnonmatches	0.01	0.38	0.39	0.04	0.18	0.69	0.96	0.96	0.69	0.96	0.96	0.16	1.00	1.00	0.17	1.00	1.00
N. pairtotal	0.01	0.38	0.39	0.04	0.18	0.70	0.96	0.96	0.69	0.96	0.96	0.17	1.00	1.00	0.18	1.00	1.00
O. pairmatchinfluence	0.29	0.44	0.31	0.75	0.53	0.44	0.15	0.17	0.44	0.15	0.18	0.94	0.17	0.18	1.00	0.17	0.18
P. pairnonmatchinfluence	0.01	0.36	0.38	0.03	0.16	0.67	0.94	0.94	0.67	0.94	0.94	0.15	1.00	1.00	0.17	1.00	1.00
Q. pairtotalinfluence	0.01	0.37	0.38	0.04	0.17	0.68	0.94	0.94	0.67	0.94	0.94	0.16	1.00	1.00	0.18	1.00	1.00

Table 10. Optimal Decision Rule, Comparing Rules Based on TF and Pair Influence, CACM Corpus

	Miss Rate	False Alarm Rate
Frequency	0.41453	0.0067876
Pair influence	0.28731	0.0086327

Table 11. Task 1 Summary

	CACM	TDT-3
Corpus statistics		
Number of texts	1587	35275
Number of topic queries	64	84
Percent relevant	0.004	0.002
Performance results		
Correlation with relevance	CRA significantly better	CRA slightly better
Optimal decision filter	CRA significantly better	CRA significantly better
Retrieval (multiple thresholds)	Frequency slightly better	Tied; CRA better at early recall
Tracking (single threshold)	CRA significantly better	Frequency slightly better; CRA better at very early recall

nation of word and word-pair influence. CRA-based metrics appear to be more robust to the effects of widely varying text sizes.

Section 5. Task 2: Data Structure Design

The purpose of this section is to demonstrate the feasibility of using CRA in a real text-streaming environment. First, we will discuss our general notion for operating in a storage-constrained environment. Next, we discuss a data architecture for CRA that minimizes space requirements. Finally, we discuss the results of experiments we performed to determine the memory requirements and computational speed of a CRA-based text streaming system.

5.1. CRA Network as Metadata

A knowledge worker or information analyst operates in an information-rich environment. In dealing with voluminous amounts of data, we must pay particular attention to information storage requirements. The volume of information now available to assess is immense. Security analysts must deal with information flows that can reach terabytes of information per day. Google's news site actively tracks and abstracts some 4000 newspapers around the globe. In a typical information system application, such data is stored in a central location and accessed from remote users on an as-needed basis.

Consider a news streaming environment. It will be cost prohibitive if each file in a database must be indexed (i.e. modeled) in order to determine whether its contents are relevant each time a user seeks information. It is more cost effective to model the text once and then store that model as meta-data or a tag. In many current information retrieval systems, the tag associated with a text is a frequency vector — a word list and an associated tally of the number of times the word occurs in the text. As we have discussed though, a CRA network is an alternative model of the text, one that provides for a more effective information processing. Thus, the CRA network can be used as a document tag, i.e., a piece of metadata associated with a given document.

One of the problems with central storage of the file and associated tag is that local processing still depends on access from a remote user site. There are reliability issues with such a system, because a communication link breakage will result in the system not being able to perform its basic functions. Additionally, processing time increases when data access must move from the local to the global. Therefore, it is beneficial to store the tag associated with the text locally. This enables local information retrieval processing if (a) the tag (in this case, the CRA network) allows for effective information processing and (b) the tag can be made small in size. Requirement (a) was addressed in Section 4 of this report; the purpose of this chapter is to describe a solution for requirement (b).

Our notional system is shown in Figure 16. Each text is processed by CRA, which in turn produces a CRA network that serves as a tag for the text document. These tags can then be distributed to local databases. The user's information retrieval system can then directly access the local tag in order to locate relevant documents by requesting complete documents from the central repository only as needed.

5.2. CRA Network Implementation

CRA networks tend to have higher storage requirements because they are richer data structures than simple word counting that is normally used in frequency-based text analysis. Specifically, a frequency vector representing n words in a text requires $O(n)$ storage, one location for the frequency count of each word included. By contrast, CRA networks require $O(n + m)$ space, where n is once again the number of nodes (words) and m is the number of edges connecting the nodes in the CRA network. In a real sense the data expansion is unavoidable because CRA networks simply carry more information about a text by encoding word connections intended by its author. The m component represents this additional data.

The method of generating CRA networks mitigates this problem to some extent. First, certain word types such as articles, prepositions, conjunctions, etc., are excluded (frequency approaches do this to a lesser extent by eliminating *stop words*). Second, the grammatical parser used in the method performs *stemming* reducing *each* word to its base form (e.g., changing plural to singular forms). This can provide significant reduction of n and by reducing the available connection points for edges can also reduce the required size of m. On the other hand, unless a suitable lookup table is also

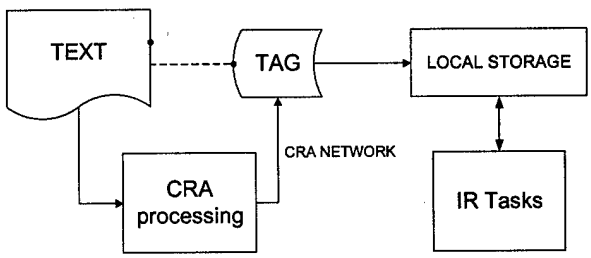

Figure 16. Using CRA to Create a Document Tag for Constrained Storage Environment

stored, stemming can complicate referral of words in a CRA network back to their positions in the original text.

Three storage schemes are typically used to represent networks. An *adjacency matrix* is a two-dimensional array of size n^2, whose elements represent the presence/absence (or value) of an edge between the row and column nodes. When they represent undirected networks (like CRA networks) the size of the array can be reduced to $n(n - 1)/2$. An *edgelist* represents a network by decomposing it into a set of linked edges. The space savings of this structure relative to the adjacency matrix is a function of the density of the network. For sparse networks space savings are achieved because non-existent links consume no storage. However for more densely connected networks edgelists can actually take more space than an adjacency matrix because node identifiers have to be repeatedly listed. The *nodelist* or *adjacency list* is the most efficient means of representing a network. In this format, each node is listed along with pointers to its neighbors; each node is listed only once, and there are only as many pointers as there are edges.

CRA networks also store information about the *influence* or importance of the various nodes. These values are calculated from the network structure, and their storage in the CRA networks represents a trade-off between computational efficiency and storage efficiency. Once a text is converted into a CRA network and its influence values are calculated, it is directly accessible to Crawdad analysis tools and can be arbitrarily combined with other texts to form emergent collections; however, these influence values carry additional storage requirements.

Given that it is not necessary to reproduce the original text and that there is a limited vocabulary of words, it is possible to specify a data structure for CRA networks that is space efficient relative to raw text. The minimal representation of a CRA network is an adjacency list format described above. A given data record would therefore include a node identifier and a list of adjacent nodes along with the edge values of connections to those nodes. It would also include the influence value calculated for the node.

Space savings on the adjacency list can be achieved in two ways. First, assuming the documents analyzed never contain a vocabulary larger than a certain size, the node identifiers can be replaced with numerical codes. As words come into the system, they are assimilated (or found) in a trie data structure and associated with a fixed numerical code. There are about 200,000 English words in common use. A three-byte integer would be sufficient to represent 2^{24}= 16,777,216 words, which would be more than enough to handle words in common use plus any slang or technical jargon that would be required. Additional research may allow reduction of this amount even further. Second, because a CRA network only links words internal to itself, the adjacency lists can simply point to the ordinal position of the words linked. Since a two-byte integer can represent 2^{16}= 65,563 of these *internal* positions, it is unlikely that any individual text will exceed this number (it is over three times the size of the vocabulary of the average English speaker).

Influence requires a single precision real number. Further storage savings can be achieved by eliminating any nodes from the network that have zero influence and only one edge connecting them to another node. Given this data, we can estimate the storage requirements of minimal CRA adjacency lists as follows:

- Node identifier code (reference to vocabulary list member): three integer bytes
- Each adjacent node: two integer bytes for internal pointer plus one integer byte for the edge value
- Influence value: two-byte single precision floating point number (depending on the operating system being used).

We tested this scheme on a sample of 1064 articles from the TIPSTER Wall Street Journal corpus. The minimal CRA files represented achieved a 54% reduction of file size compared to uncompressed ASCII text.

A final issue has to do with CRA network collections and how to best represent and process those. One problem has to do with cross-sectional analysis of collections of CRA nets using resonance measures. Resonance is an adjacency or distance measure, so collections of texts can themselves be viewed as undirected, valued graphs. Thus, the data architecture problems are similar to those already described for CRA networks of individual texts. The main differences are that these resonance networks must be amenable to use with a wider range of non-Crawdad statistical analysis tools, e.g., standard statistical packages. Also, meta information is significantly different than that in individual CRA networks because texts collections normally have no single author and may require information such as the reason or problem the collection was assembled to address. A second problem has to do with the analysis of changes in CRA networks over time. Here, interest is not only in resonance between texts in a sequence or the state of a present network, but in the evolution or history that led to the state of the network (see [97]). This requires the ability to generate connections between CRA networks over time.

Thus the data architecture for CRA networks to be developed in this project is constrained by the following requirements. The architecture must:

Table 12. Initial CRAZ File Layout

	Number of bytes	Data Type	Contents
Header			
	2	integer	number of nodes
	2	single real	density
	2	single real	graph influence
Node Data			repeated for each node
	3	integer	vocabulary token
	2	single real	influence
	2	integer	number of neighbors of the given node
Neighbor Data			repeated for each neighbor of the given node
	2	integer	neighbor index
	1	integer	weight of neighbor (count of co-occurrences)

- Be as space-efficient as possible by using a minimal representation of CRA networks as described above.
- Provide the means of linking stemmed words back to their position in the original text without creating large storage burdens for this capability.
- Include information about the influence of individual nodes to as to be compatible with existing Crawdad analysis tools.
- Allow the embedding of relevant meta-information, and support the filtering and/or analysis of the CRA networks based on these data.
- Support the inclusion of resonance-based networks-of-networks and their associated meta-information, and the analysis of these using both Crawdad tools and third-party packages.
- Encompass sequences of CRA networks and links between them allowing analysis of the history or evolutionary process that led to the present state of a network.

5.3. Initial CRAZ File Format Design

Our previous work developed an initial baseline CRAZ file format design. Table 12 shows the initial design for CRAZ files.

This design assumes that:

- The number of different vocabulary tokens in all files is no more than 2^{24}= 16,777,216.
- The number of different vocabulary tokens in any one file (i.e., nodes) is no more than 2^{16}= 65,536.
- The number of times two nodes are related in any one file is no more than 2^8 = 256.

The design is based on use of the *trie* data structure [98]. The trie can be used to find the index for any given token in a computationally efficient manner. Section 5.3.4 provides more information on use of the trie data structure.

The focus of our work in this project was refining that initial design to minimize storage requirements for each file. This was accomplished by:

- Eliminating unused fields in the CRA file format,
- Representing tokens by an index into a lexicon, and
- Writing the resulting file in binary rather than ASCII.

The initial CRAZ file format design was optimized during its implementation. The actual implementation differs as follows:

1. All real numbers are represented as a 2-byte binary number (i.e., short integer) by multiplying the value by 65,536 and rounding to the nearest integer. This takes advantage of the fact that all such values in a CRA file are numbers between 0 and +1 to produce the highest possible precision in 2 bytes.

2. The vocabulary token is shortened to a short (2-byte integer) using the following convention:
   a. If the string is found in a lexicon of 65,436 words, then the vocabulary token will be the index number of that word in the lexicon (counting from 100 instead of 0 or 1).
   b. If the string is an integer value, then the vocabulary token will be 0. The next 4 bytes in the file will be the binary representation of that integer.
   c. If the string is not found in a lexicon and is not an integer value, then the vocabulary token will be the length of that string, L. The next L bytes in the file will be the characters of the string.

Because we have only reserved vocabulary tokens 1 through 99 for this purpose, any strings not in the lexicon that are longer than 99 characters will use the vocabulary token 99 and only store the first 99 characters of that string in the file. This should be a very rare occurrence.

5.3.1. Final Prototype CRZ Design

A number of changes were made in the design of the file format during development. There were two rather minor changes and one major change. The minor changes were:

- The name of the file format was changed from CRAZ to CRZ, so that the file extension would be a more standard 3 characters.
- All 2-byte "single real" numbers from the original format are represented as a 2-byte binary integer by multiplying the value by 65,536 and rounding to the nearest integer. This takes advantage of the fact that all such values in a CRA file are positive numbers strictly less than +1 to produce greater precision within the 2 byte limitation.

5.3.2. Compressed Vocabulary Token Representation

We made one major change to our baseline design. The three-byte vocabulary token was replaced by a variable length representation that requires two-bytes in most cases. The following algorithm describes how we represent a token given a lexicon Lex:

Lex.lookup() returns a unique two-byte integer greater than 99 for each token in the lexicon (0 for tokens not in the lexicon). The lexicon can contain up to 65,436 distinct noun phrase tokens. The choice of 99 is arbitrary. Each character we remove from the longest string we can store outside of the lexicon allows us to store one more token in the lexicon.

These changes lead to the layout in Table 13.

5.3.3. Compression Trade-offs

Webster's Dictionary 3rd Edition has a vocabulary of around 54,000 word families [99.]. So, we can expect the vast majority of tokens encountered in a document to be among the 65,000+ most common noun phrase tokens in the language or sub-language of the document.

If the 65,000+ most common tokens constituted 95% of the tokens encountered in documents, then storing the

```
Write_Token(string T)
{
      index = Lex.lookup(T)
      IF (index > 0)
            WRITE index;
      ELSE IF (T is an integer)
      {
            WRITE 0;
            WRITE T as 32-bit integer;
      }
      ELSE
      {
            length = MAX(99, T.length());
            WRITE length;
            WRITE T.substring(0, length-1);
      }
}
```

Figure 17. Token Representation Algorithm

Table 13. Final CRZ File Layout

	Number of bytes	Data Type	Contents		
Header					
	2	integer	number of nodes		
	2	integer	density*65,536		
	2	integer	graph influence*65,536		
Node Data			repeated for each node		
	2	integer	0 → integer	1-99 → string length	index
	0-99	varies	4-byte integer	1-99 characters	0 bytes
	2	integer	influence*65,536		
	2	integer	number of neighbors of the given node		
Neighbor Data			repeated for each neighbor of the given node		
	2	integer	neighbor index		
	1	integer	weight of neighbor (number of co-occurrences)		

tokens themselves the other 5% of the time would still save space compared to always storing the token as three-bytes index, as long as the length of the average token not found in the lexicon was < 20. Table 14 illustrates the trade-offs between other percentages and average word lengths.

Table 14. CRZ Compression Trade-offs

% Tokens in lexicon	Bytes saved per token given average token length 5	10	15	20	Break-even average token length
99%	0.95	0.90	0.85	0.80	100.0
97%	0.85	0.70	0.55	0.40	33.3
95%	0.75	0.50	0.25	0.00	20.0
90%	0.50	0.00	-0.50	-1.00	10.0
80%	0.00	-1.00	-2.00	-3.00	5.0

In practice, the percentage of tokens found in this large lexicon should be well above 95% and the average token not found should be much less than 15 characters long. So, this approach should save more than 0.5 bytes per token in each document and usually much more.

Consider the example of the TDT-3 corpus. There were 113,699 distinct tokens. The 65,346 tokens selected for the lexicon were encountered in file a total of 2,876,072 times, compared to 52,748 times for the 48,263 tokens not selected for the lexicon. The percentage of tokens found in the lexicon was greater than 98%.

The average length of the tokens that were not selected for the lexicon was 7.2 characters long (including 54 numbers designated to be stored in 4 bytes). Storing all the tokens in the CRZ files required 2,876,072 2-byte indexes for the words in the lexicons and 52,748 2-byte lengths and 364,858 bytes of data for the tokens not in the lexicon, for a total of 6,204,253 bytes. If we had instead placed every token in the lexicon then each of the 2,928,820 occurrences of a token would have required three bytes of storage, for a total of 8,786,460 bytes. The savings is 2,582,207 bytes, or over 0.88 bytes per token.

5.3.4. Lexicon Lookup

The lexicon lookup algorithm must search the lexicon for a given token and return a unique two-byte

index for the token if it is found in the lexicon or 0 if it is not found. There are many ways to implement such an algorithm. Table 15 shows how many character operations will be required on average for selected algorithms (N = average length of tokens; L = number of tokens in the lexicon).

Table 15. Lookup Algorithm Complexities

Algorithm	Worst-Case Complexity
Linear Search	NL/2
Binary Search	Nlog(L)
Hash Table (not assuming perfect hash)	N+N(average size of collision set)
Hash Table (assuming perfect hash)	2N
Trie	N

Using a trie to implement the lookup is very time efficient. It is not particularly space efficient, but RAM limitations are not a critical factor.

A trie is n k-ary tree, where k is the number of characters in the character set for the lexicon. There is no reason to distinguish upper case from lower case characters. The digits and the characters '-', and '@' are important to preserve. The character '*' was substituted for any other character encountered. So, each node in our trie has an index value and up to 40 child nodes.

The trie is traversed by starting at the root node and moving to the child node that corresponds to the next character in a given string. In building the trie from strings in the lexicon, whenever a child node does not exist, we create it. The value 0 is stored in all new nodes, and is replaced by the index of a string in the lexicon after traversing the last letter in the string. When looking up a string, if a child node does not exist, then the lookup fails and returns 0. The efficiency of the trie is due to the fact that insertion or retrieval just processes each character in the string just once. Even a hash table has to process each character to compute the hash, and then may process them again when comparing the string to each string in the lexicon that happens to have the same hash.

5.3.5. Token Selection

For this project, we build an optimal trie for each given corpus of documents. Essentially, each such trie represents a lexicon for the sublanguage defined by the corpus.

The optimal trie for a corpus is that which results in the set of smallest CRZ files. This is not precisely the same as the 65,436 most common words in those documents. There are two factors at play:

1. Every token in a document appears exactly once in the document's CRZ file no matter how many times it appears in that document.
2. The longer the token, the more space is saved by having it in the lexicon.

Suppose we have the choice between putting the word *pig* or the word *horse* in our lexicon. If we put the word *pig* in, then we can save three bytes for every file that contains the word *pig*. If we put the word *horse* in, then we would save five bytes for every file that contains the word *horse*. So, if *pig* appears in p different files and *horse* appears in h different files, then we should pick *pig* over *horse* when $3p > 5h$.

To maximize compression, we sort the list of tokens the lexical analyzer finds in a corpus by the product of the length of each token by the number of files it was found in, and then select the 65,436 noun phrase tokens with the highest values for the lexicon.

In order to build the lexicon that would minimize the size of the CRZ files, we:

1. Calculate the product of the length of each noun phrase token by the number of files in which it was found;
2. Sort the collection of tokens in descending order by this product;
3. Create a list structure containing the 65,436 tokens with the highest values;
4. Insert each token from this list into the trie with the index of the token in this list as the value to be stored in the trie;
5. Write the list to a file (so it can be read into a list structure and each token can be quickly retrieved from this list structure given its index).

5.3.6. Lexical Representation

In order to quickly recognize if a token is in a given lexicon and what its index is, we represent the lexicon

using a trie data structure [98]. A trie is a tree data structure in which there is one node for every common prefix. Each node in our trie contains the unique index for the string associated with that node if that string is in the lexicon (otherwise, the node contains the value 0).

The trie structure is not a persistent object; i.e., it is not stored in a data file. We made this design decision because:

- The most efficient way to reproduce the tokens from indexes when working with CRZ files is via the list of tokens (stored in index order), so we needed to store the lexicon this way anyway.
- Creating the trie from a word list is fairly efficient.

For handling streaming news, the executable that produces CRZ files was made memory resident. This means the trie only has to be built from the word list once per session.

5.3.7. Implementation Issues

If no token in the lexicon had any common prefixes, then a naïve implementation of a trie would contain $N*L$ nodes, where N = average length of the tokens; L = number of tokens in the lexicon. Each node requires 162 bytes, so the worst-case size for the trie would be $40N$ MB.

In practice, the tokens in any lexicon have a significant number of common prefixes. For example, the trie for the TDT-3 corpus of 65,436 tokens with average word length 9.4 characters generated 281,421 nodes or about 43.5 MB, or about 11.5% of the worst case 376 MB.

We have not refined our trie implementation to minimize memory usage. Memory resource utilization is not a problems and most methods of reducing memory requirements result in increased execution time.

We also did not implement a way to write the trie for a lexicon out to a binary file and read it back in, because the program to produce the trie directly from the lexicon was quite fast. We did make the trie memory resident so that it would only have to be built once when constructing CRZ files for an entire collection of text files.

The compression of the trie in RAM, and the ability to write the trie to a compressed binary file and reproduce the trie by reading a compressed binary file are all feasible tasks for future phases of the project. The marginal utility of these tasks seem questionable right now, but the final determination of the cost-value trade-offs of these tasks should wait until a later phase of the project.

5.3.8. Operational Issues

Building corpus-specific lexicons has the advantage of minimizing file sizes, but presents some operational problems:

1. The need to pre-process the corpus to select the optimal set of tokens for the corpus;
2. The need to knowing which lexicon to use when decoding a CRZ file;
3. The need to calculate the resonance between CRZ files encoded using different lexicons which requires decoding them first.

No matter how the first issue is resolved, the second and third issues require that the CRZ file format have a field to indicate which lexicon it was built with.

Pre-processing the corpus is impractical in dealing with streaming data. There are a number of alternatives that should be explored in future phases of the project. These include:

1. Single Lexicon - Using a single lexicon for each language (e.g., English, French, Chinese, ...). This is operationally feasible, but may lead to poor file compression.
2. Pre-Built Domain Lexicons - Pre-build a lexicon for the sublanguage of each anticipated domain. This solution will likely uses document meta-data to determine the applicable domain. When a domain without a pre-built lexicon is encountered, the generic lexicon of alternative 1 can be applied.
3. Streaming Domain Lexicons - Build lexicons for domains by adding tokens to the trie for that domain as they are encountered. Domains would also have to be identified via document meta-data under this approach.

Tokens that occur in early documents but turn out to be uncommon in the domain will likely lead to poor file compression unless some dynamic mechanisms is added. For example, every token could be added to the trie for the domain with counters stored along with non-zero indexes for only the first 65,436 entries. Periodically (e.g., when there is a pause in the incoming stream or when the counters for non-indexed words become comparatively large), the trie could be optimized using the counters. Eventually, the trie should reach a reasonable equilibrium.

This approach might require rewriting all the CRZ files produced before each optimization. An alternative approach is to designate each optimized lexicon as a new version and have each file reference the version of the lexicon that was used to build it. This may be problematic and will likely adversely impact operational issue 3 (i.e., calculating resonance between files from the same domain/stream might still require decoding before calculation). It may be possible to use this approach until equilibrium is reached and then rewrite the files that used older versions.

5.4. CRZ Benchmark Performance

In order to benchmark the performance of the CRZ system, we processed the CACM corpus on the same machine under identical conditions using CRZ and Lemur's implementation of TF-IDF. In both cases, we were running on the same Windows 2000 system (a 2.00 GHz Pentium 4 with 256 MB RAM) with no other applications running. The benchmarking results are summarized in Table 16.

We did not measure RAM usage, since memory usage was observed to be inconsequential for both programs on a machine equipped with 256 MB of RAM.

In summary, CRZ produces files that are less than one third the size of the indexes for TF-IDF but currently requires slightly less than 40 times as long to do it.

5.5. Improving CRZ Performance

Our Phase I effort has focused on the effectiveness of CRZ for performing IR tasks and minimizing the space required to store the information that CRZ uses. We have briefly investigated how the execution time of the CRZ software can be reduced. Additional investigation will be conducted as part of the Phase II research effort.

5.5.1. Performance Bottlenecks

Preliminary analysis indicates that between 70 and 75% of the CRZ software's time is spent producing the sequence of tokens found in a given document, i.e., parsing the text. This requires:

1. Parsing the document (using a third-party parser) into an XML representation.
2. Processing the XML parse representation into a sequence of tokens.

Currently, the parser version for the Windows operating system processes a single file and then exits. The Linux release is has a "server mode" that allows the parser to remain in memory. We have not tested using the Linux version but our estimate is that this will save between 20 and 25% of the total execution time.

The parser can return its parse results in a tab-delimited format rather than XML. The tab-delimited files should be faster to produce and process because the tab-delimited files are significantly smaller and we can avoid the overhead of the XML parser. Our estimate is that it would also save between 20 and 25% of the total execution time.

If our estimates are correct, applying these improvements will approximately double the speed of CRZ. This means it will run between 20 and 24 times slower than Lemur's implementation of TF-IDF. At this point, the parsing and the CRZ calculations would be taking roughly equal amounts of time. It may be possible to gain further reduce processing time by optimizing the existing code base. However, since most processing time is spent in the parser, replacing the parser with a faster one (perhaps customized for finding only noun phrases) may make more sense.

5.5.2. Scalability

Our research to date has focused on algorithm and data structure development. Our prototype system was implemented on a desktop computer using the (rather inelegant) Microsoft Windows operating system. No attempt was made to implement a system suitable for use in a production environment (e.g., use by an intelligence analysts handling real-time or near real-time information).

Table 16. Benchmark Results

Metric	CRZ	Lemur (TF-IDF)	CRZ / Lemur
Clock Time	5:32:672	8:390	39.65
Input Size	980,739 bytes (Text files)	2,353,756 bytes (SGML)	n.a.
Output Size	515,128 bytes (CRZ files)	1,598,197 bytes (bindex files)	0.32

Our Phase II research will investigate methods of significantly speeding up processing by executing on a multiprocessor Unix platform. Platforms with four (or even eight) processors are relatively inexpensive and widely available.

This approach will allow us to use multiple instantiations of the parser and CRA software and allow us to increase speed by implementing parallel pipelines through these multiple processes and processors. Our initial research indicates that this approach can provide a cost effective platform for fielding a solution suitable for real-world problem solving.

Section 6. Conclusions

6.1. Summary

An intelligence analyst or knowledge worker is responsible for handling an ever increasing amount of information each day. Typically, this information is in the form of a streaming media. Current information retrieval and analysis systems do not provide adequate processing capability and performance in this environment. There are two major problems. First, existing systems use an inferior model of the text for computational processing and information retrieval tasks. Thus, the quality of the system is adversely affected by an inferior data model. Second, existing systems assume that all relevant data is stored centrally. In a streaming environment, it may be advantageous to store a compressed version of the text — a tag — that facilitates information processing. Text mining algorithms are needed that minimize storage capacity requirements, facilitate fast and accurate information retrieval and knowledge discovery, and determine the relevancy of an item as it is acquired from a data stream. The purpose of this Phase I research was to demonstrate that Centering Resonance Analysis (CRA) provides a superior approach to performing text mining under storage constraints.

CRA provides a superior method for modeling a text because it uses different knowledge about the content of a text. Our argument is that if CRA creates more salient knowledge about the content of a text, then information processing based on CRA will perform better than systems that do not have this knowledge. Specifically, CRA represents a text as a network; this stands in contrast to modeling text based on simple word frequency measurements.

In order to determine feasibility and measure performance of a CRA-based approach we established two technical objectives:

1. Compare how well CRA determines relevancy of incoming news articles, relative to the traditional frequency-based approach,
2. Determine a data design for a CRA network that minimizes storage requirements, and measure the computational requirements associated with CRA.

In order to meet our first objective, we conducted computer-based experiments using a prototype system implementation and performed specific information retrieval tasks using a realistic corpus of documents. Therefore, our results contain no bias in interpretation, and are reliable and valid. We followed well-accepted standard procedures for performing these experiments.

Our experimental design included the following variables:

- Metrics for determining relevance: We examined two families of metrics, one set based on word frequency and one based on word influence estimated using the CRA network. Raw and normalized scores were examined, and a number of hybrid combinations were created and analyzed.
- Corpus: We executed an experimental design that tested performance against two corpora, the CACM corpus, which contains relatively short queries and texts and is narrow in scope of content; and the TDT-3 corpus which contains relatively long queries and texts and is broad in scope of content.
- Method of determining relevance: We experimented with a system where a single, fixed threshold value was used to discriminate relevant from non-relevant documents across all queries (tracking), and a system where each query has its own threshold for relevance (ad hoc retrieval).
- Method of measuring performance: We used a variety of statistical comparisons in order to determine performance of each of the metrics:
  a. Correlation with the actual human judgment of relevancy.
  b. False alarm and miss rate performance based on an "optimal" decision rule.
  c. Recall-precision performance.

CRA-based metrics were shown to have superior performance. In particular, metrics based on word-pair influence had up to three times the correlation with document relevancy. An optimal filter based on CRA had a miss rate 15 times smaller than that based on word frequency. In the tracking experiment, the word-pair influence metric had 3-5 times greater precision than the frequency metrics in the CACM corpus. Both types of metrics had similar performance in retrieval and tracking within the TDT-3 corpus, but CRA-based metrics had higher precision where it mattered most — in the first several dozen documents retrieved.

For the second task which demonstrated practical feasibility of our approach, we designed a CRA data structure that requires one-third the space of that required by the compressed raw text. We used a novel data structure — the trie — which translates words into numerical indexes resulting in decreased storage requirements and faster look-up. Computer memory

requirements for processing were found to be compatible with a typical desktop computer, and processing speed was more than adequate to handle the volume of streaming text that one would expect in a normal text streaming application. For typical sized texts coming from a news stream, CRA networks can be generated at a rate of 600 pages per minute. We also briefly investigated how various elements of the existing system can be made much faster using multiple, parallel processors and faster noun phrase parsing.

6.2. Implications

In conclusion, we have demonstrated that CRA is superior approach for mining textual data. This is a very significant result because all existing commercial and research applications are based on metrics using word frequency. Existing frequency-based systems have been highly engineered to take advantage of nuances and context in grammar, language, and setting but are still based on word frequency statistics. By implication, substituting metrics based on CRA for word frequency metrics will improve the performance of existing systems. CRA computation and memory requirements have been shown to be compatible with real-world government and commercial applications and can effectively operate in a storage-constrained environment.

There are numerous ways in which better information retrieval performance translates to better operational performance:

- Documents that are relevant are not missed, leading to reduction in problem solving costs and time.
- Documents that are irrelevant are not retrieved, leading to reduction in problem solving costs and time, and reduction in local storage requirements.
- Second-order processing, such as link detection and extraction, question answering, taxonomy creation, document clustering, social network analysis, and trend and dynamic analysis all produce more valid results when the information content of the corresponding input data is improved.

6.3. Future Work

Further research and development is needed to commercialize the application, and this will be the focus of our Phase II proposal. In particular, we will develop a system that is capable of taking streaming data from numerous sources and fusing them into a single "operational" picture of the events in question. This work will create enhancements to CRA that optimize its performance for tracking new texts relative to a topic statement of interest, and that create insightful metrics and representations that give an analyst a deep view of the dynamic content of these different text streams. Phase III commercialization will be aimed at "voice of the customer" and intelligence applications in commercial and government markets.

Bibliography

[1] USAF, "Air Force information strategy guides information planning efforts," US Air Force Release * 101 5025, October 15 2002.

[2] M. Chesire, A. Wolman, G. Voelker, and H. Levy, "Measurement and Analysis of a Streaming Media Workload," presented at Proceedings of the 3rd USENIX Symposium on Internet Technologies and Systems, San Francisco, CA, 2001.

[3] S. Corman, T. Kuhn, R. McPhee, and K. Dooely, "Studying complex discursive systems: Centering resonance analysis or organizational communication," *Human Communication Research*, vol. 28, pp. 157 - 206, 2002.

[4] R. McPhee, S. Corman, and K. Dooley, "Organization knowledge expression and management: Centering resonance analysis of organizational discourse," *Management Communication Quarterly*, vol. 16, pp. 130 - 136, 2002.

[5] K. Dooley and S. Corman, "The dynamics of electronic media coverage," in *Communication and Terrorism: Public and Media Responses to 9-11*, B. Greenberg, Ed. Cresskill, NJ: Hampton Press, 2002.

[6] V. Batagelj, U. Brandes, S. R. Corman, J. C. Johnson, S. Koburov, L. Krempel, A. Mrvar, and D. Wagner, "Analysis and visualization of networks data," presented at Sunbelt XXII International Sunbelt Social Network Conference, New Orleans, LA, 2002.

[7] K. Dooley, S. Corman, and R. McPhee, "A knowledge directory for identifying experts and areas of expertise," *Human Systems Management*, vol. 21, pp. 217 - 228, 2002.

[8] K. Dooley, S. Corman, R. McPhee, and T. Kuhn, "Modeling high-resolution broadband discourse in complex adaptive systems," *Nonlinear Dynamics, Psychology & Life Sciences*, vol. 7, pp. 61 - 86, 2003.

[9] U. Hahn, M. Romacker, and S. Schulz, "Discourse structures in medical reports - watch out! The Generation of referentially coherent and valid text knowledge bases in the MEDSYNDIKATE system," *International Journal of Medical Informatics*, vol. 53, pp. 1 - 28, 1999.

[10] C. Leacock, M. Chodorow, and G. A. Miller, "Using corpus statistics and wordNet relations for sense identification," *Computational Linguistics*, vol. 24, pp. 147 - 165, 1998.

[11] J. Perez-Carballo and T. Strzalkowski, "Natural language information retrieval: Progress report," *Information Processing & Management*, vol. 36, pp. 155 - 178, 2000.

[12] R. Baeza-Yates, D. Ribeiro-Neto, and R. Swan, *Modern Information Retrieval*. New York, NY: ACM Press/Addison-Wesley, 1999.

[13] G. M. Galal, J. J. Cook, and L. B. Holder, "Exploiting parallelism in a structural scientific discovery system to improve scalability," *Journal of the American Society for Information Science*, vol. 50, pp. 65 - 73, 1999.

[14] S. M. Humphrey, "Automatic indexing of documents from journal descriptors. A preliminary investigation.," *Journal of the American Society for Information Science*, vol. 50, pp. 661 - 674, 1999.

[15] M. L. Doerfel and G. A. Barnett, "A semantic network analysis of the International Communication Association," *Human Communication Research*, vol. 25, pp. 589 - 603, 1999.

[16] A. S. Fraenkel and S. T. Klein, "Information retrieval from annotated texts," *Journal of the American Society for Information Science*, vol. 50, pp. 845 - 854, 1999.

[17] T. K. Landauer, P. W. Foltz, and D. Laham, "An introduction to latent semantic analysis," *Discourse Processes*, vol. 25, pp. 259 - 284, 1998.

[18] T. Nomoto and Y. Nitta, "Structuring raw discourse," in *New Methods in Language Processing*, D. B. Jones and H. L. Somers, Eds. London, UK: UCL Press, 1997, pp. 288 - 298.

[19] D. Jurafsky and J. H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Upper Saddle River, NJ: Prentice-Hall, 2000.

[20] G. Salton and C. Buckley, "Term weighting approaches in automatic text retrieval," *Information Processing and Management*, vol. 24, pp. 513 - 523, 1988.

[21] S. Kaski, T. Honkela, K. Lagus, and T.

Kohonen, "WEBSOM - Self-organizing maps of document collections," *Neurocomputing*, vol. 21, pp. 101 - 117, 1998.

[22] O. Buyukkokten, H. Garcia-Molina, and A. Paepcke, "Seeing the who in parts: Text summarization for web browsing on handheld devices," presented at Tenth International World Wide Web Conference, Hong Kong, 2001.

[23] D. Radev, H. Jing, and M. Budzikowska, "Centroid-based summarization of multiple documents: Sentence extraction, utility-based evaluation, and user studies," presented at TREC-11, 2002.

[24] J. Allan, "Detection of multi-topic tracking," *Information Retrieval*, vol. 5, pp. 139 - 157, 2002.

[25] I. Mani and M. Maybury, "TDT: Topic detection and tracking," in *Advances in Automatic Text Summarization*: MIT Press, 1999.

[26] N. Cancedda, N. Cesa-Bianchi, and A. Conconi, "Kernel methods for document filtering," presented at The Eleventh Text Retrieval Conference (TREC 2002), Gaithersburg, MD, 2002.

[27] G. McKoon and R. Ratcliff, "Memory based language processing: Psycho linguistic research in the 1990's," *Annual Review of Psychology*, vol. 49, pp. 25 -42, 1998.

[28] R. Lecoeuche, D. Robertson, C. Barry, and C. Mellish, "Evaluating focus theories for dialogue management," *Journal of Human-Computing Studies*, vol. 52, pp. 23 - 72, 2000.

[29] P. C. Gordon and R. Hendrick, "The representation and processing of Coreference in discourse," *Cognitive Science*, vol. 22, pp. 389 - 424, 1998.

[30] B. Grosz, A. Joshi, and S. Weinstein, "Centering: A framework for modeling the local coherence of discourse," *Computational Linguistics*, vol. 2, pp. 203 - 225, 1995.

[31] K. Collins-Thompson, P. Ogilvie, Y. Zhang, and J. Callan, "Information filtering, novelty detection, and named-page finding," presented at The Eleventh Text Retrieval Conference (TREC 2002), Gaithersburg, MD, 2002.

[32] P. W. Foltz, "Latent Semantic Analysis for text-based research," *Behavior Research Methods, Instruments and Computers*, vol. 28, pp. 197 - 202, 1996.

[33] K. Tolle and H. Chen, "Comparing noun phrasing techniques for use with medical digital library tools," *Journal of the American Society for Information Sciences*, vol. 51, pp. 352 - 370, 2000.

[34] R. Motwani, J. Widom, A. Arasu, B. Babcock, S. Babu, M. Datar, G. Manku, C. Olson, J. Rosenstein, and R. Varma, "Query processing resource management, and approximation in a data stream management system," presented at Proceedings of the First Biennial Conference on Innovative Data Systems Research (CIDR 2003), Asilomar, CA, 2003.

[35] S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams," presented at IEEE Symposium on Foundations of Computer Science, Redondo Beach, CA, 2000.

[36] L. O'Callaghan, N. Mishra, A. Meyerson, S. Guha, and R. Motwani, "High performance clustering of streams and large data sets," presented at Eighteen International Conference on Data Engineering (ICDE 2002), San Jose, CA, 2002.

[37] L. C. Freeman, "Centrality in social networks: Conceptual clarification," *Social Networks*, vol. 1, pp. 215 - 239, 1979.

[38] L. C. Freeman, "A set of measures of centrality based on betweeness," *Sociometry*, vol. 40, pp. 35 - 41, 1977.

[39] U. Brandes, "Faster evaluation of shortest-path centrality indices," *Journal of Mathematical Sociology*, vol. 25, pp. 163 - 177, 2001.

[40] C. W. Roberts, "Text Analysis for the Social Sciences: Methods for Drawing Statistical Inference from Texts and Transcripts." Mahwah, NJ: Lawrence Erlbaum Associates, 1997.

[41] L. M. R. Eizirik, V. C. Barbosa, and S. B. T. Mendes, "A Bayesian network approach to lexical disambiguation," *Cognitive Science*, vol. 17, pp. 257 - 283, 1993.

[42] A. Dong and A. M. Agoginao, "Text analysis for constructing design representations," *Artificial Intelligence in Engineering*, vol. 11, pp. 65 - 75, 1997.

[43] G. A. Miller, "WordNet: A lexical database for English," *Communications of the ACM*, vol. 38, pp. 39 - 41, 1995.

[44] K. A. Adams, "Word wranglers: Automatic classification tools transform enterprise documents from "bags of words" into knowledge resources," Intelligent KM available at http://www.intelligentkm.com, 2001.

[45] K. Lund and C. Burgess, "Producing high-dimensional semantic spaces from lexical co-occurrence," *Behavior Research Methods, Instruments, & Computers*, vol. 28, pp. 203 - 208, 1996.

[46] Terra, "The Galileo computer program," Terra Research and Computing, Birmingham, MI 1994.

[47] J. Woelfel and E. Fink, *The Measurement of Communication Processes: Galileo Theory and Method*. New York, NY: Academic Press, 1980.

[48] P. W. Foltz, W. Kintsch, and T. K. Landauer, "The measurement of textual coherence with latent semantic analysis," *Discourse Processes*, vol. 25, pp. 285 - 307, 1998.

[49] W. Kintsch, "Predication," *Cognitive Science*, vol. 25, pp. 173 - 202, 2001.

[50] D. P. Spence and K. C. Owens, "Lexical co-occurrence and association strength," *Journal of Psycholinguistic Research*, vol. 19, pp. 317 - 330, 1990.

[51] G. A. Barnett and J. A. Danowski, "The structure of communication: A network analysis of the International Communication Association," *Human Communication Research*, vol. 19, pp. 264 - 285, 1992.

[52] J. Danowski, "A network-based content analysis methodology for computer-mediated communication: An illustration with a computer bulletin board," *Communication Yearbook*, vol. 6, pp. 904 - 925, 1982.

[53] J. A. Danowski, "Organizational infographics and automated auditing: Using computers to unobtrusively gather as well as analyze communication.," in *Handbook of Organizational Communication*, G. Goldhaber and G. Barnett, Eds. Norwood, NJ: Ablex, 1988, pp. 385 - 433.

[54] J. A. Danowski, "Network analysis of message content," in *Progress in Communications Science*, vol. 12, W. D. Richards and G. A. Barnett, Eds. Norwood, NJ: Ablex, 1993, pp. 197 - 221.

[55] R. E. Rice and J. A. Danowski, "Is it really just a fancy answering machine? Comparing semantic networks of different types of voice mail users," *Journal of Business Communications*, vol. 30, 1993.

[56] K. M. Carley, "Extracting team mental models through textual analysis," *Journal of Organizational Behavior*, vol. 18, pp. 533 - 558, 1997.

[57] R. Axelrod, "Structure of Decision: The Cognitive Maps of Political Elites." Princeton, NJ: Princeton University Press, 1976.

[58] R. C. Schank and R. P. Abelson, *Scripts, Plans, Goals, and Understanding: An Inquiry into Human Knowledge Structures*. Hillsdale, NJ: Erlbaum, 1977.

[59] K. M. Carley, "Network text analysis: The network position of concepts," in *Text Analysis for the Social Sciences: Methods for Drawing Statistical Inferences from Texts and Transcripts*, C. W. Roberts, Ed. Mahwah, NJ: Lawrence Erlbaum Associates, 1997, pp. 79 - 100.

[60] K. M. Carley and D. M. Kaufer, "Semantic connectivity: An approach for analyzing symbols in Semantic Networks," *Communication Theory*, vol. 3, pp. 183 - 213, 1993.

[61] R. Siemens, "Practical content analysis techniques for text retrieval in large, untagged textbases," presented at Eleventh Annual International Conference on Systems Documentation, Kitchener, Canada, 1993.

[62] M. A. Walker, A. K. Joshi, and E. F. Prince, "Centering Theory in Discourse." Oxford, UK: Clarendon Press, 1998.

[63] P. C. Gordon, B. J. Grosz, and L. A. Gilliom, "Pronouns, names, and the centering of attention in discourse," *Cognitive Science*, vol. 17, pp. 311 - 347, 1993.

[64] K. Kellerman and C. Sleight, "Coherence: A meaningful adhesive for discourse," in *Communication Yearbook 12*, J. A. Anderson, Ed. Newbury Park, CA: Sage, 1989, pp. 95 - 129.

[65] D. Sperber and D. Wilson, *Relevance: Communication and Cognition*. Malden, MA: Blackwell Publishes, Ltd., 1995.

[66] F. Auld and A. M. White, "Rules for dividing interviews into sentences," *Journal of Psychology*, vol. 42, pp. 273 - 281, 1956.

[67] W. Frawley, *Linguistic Semantics*. Hillsdale, NJ: Lawrence Erlbaum, Associates, 1992.

[68] P. Hooper and S. Thompson, "The discourse basis for lexical categories in universal grammar," *Language*, vol. 60, pp. 703 - 752, 1984.

[69] P. Hooper and S. Thompson, "The iconicity of the universal categories of 'noun' and 'verb'," in *Iconicity in Syntax*, J. Haiman, Ed. Amsterdam: Benjamins, 1985, pp. 151 - 183.

[70] R. Langacker, "Nouns and verbs," *Language*,

vol. 63, pp. 53 - 94, 1987.

[71] T. Givón, *Syntax: A Functional-Typological Introduction*. Amsterdam: John Benjamins Publishing Company, 1984.

[72] M. Cloitre and T. G. Bever, "Linguistic anaphors, levels of representation, and discourse," *Language and Cognitive Processes*, vol. 3, pp. 293 - 322, 1988.

[73] D. Connolly, J. D. Burger, and D. Day, "A machine learning approach to anaphoric reference," in *New Methods in Language Processing*, D. B. Jones and H. L. Somers, Eds. London, UK: UCL Press, 1997.

[74] L. Gerken and T. Bever, "Linguistic intuitions are the result of interactions between perceptual processes and linguistic universals," *Cognitive Science*, vol. 10, pp. 457 - 476, 1986.

[75] R. J. Passonneau, "Interaction of discourse structure with explicitness of discourse anaphoric noun phrases," in *Centering Theory in Discourse*, M. A. Walker, A. K. Joshi, and E. F. Prince, Eds. Oxford, UK: Clarendon Press, 1998.

[76] W. B. Frakes and R. Bazea-Yates, "Information Retrieval: Data Structures and Algorithms." Englewood Cliffs, NJ: Prentice-Hall, 1992.

[77] J. M. Anthonisse, "The rush in a directed graph," Stichting Mathematisch Centrum, Amsterdam BN 9/71, October 1971.

[78] S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications*. New York, NY: Cambridge University Press, 1994.

[79] B. Lichtenstein, K. Dooley, and T. Lumpkin, "Dynamics of organizational emergence: A longitudinal study of new venture creation," *to be published in Journal of Business Venturing*, 2004.

[80] C. Fox, "Lexical Analysis and Stoplists," in *Information Retrieval: Data Structures & Algorithms*, W. B. Frakes and R. Baeza-Yates, Eds. Englewood Cliffs, NJ: Prentice-Hall, 1992.

[81] W. B. Frakes, "Stemming Algorithms," in *Information Retrieval: Data Structures & Algorithms*, W. B. Frakes and R. Baeza-Yates, Eds. Englewood Cliffs, NJ: Prentice-Hall, 1992, pp. 131 - 160.

[82] G. Salton and M. J. McGill, *An Introduction to Modern Information Retrieval*. New York, NY: McGraw -Hill, 1983.

[83] D. Harman, "Overview of the Third Text REtrieval Conference (TREC-3)," presented at 3rdText REtrieval Conference, Gaithersburg, MD, 1994.

[84] K. Sparck Jones, "Further reflections on TREC," *Information Processing and Management*, vol. 36, pp. 37 -85, 2000.

[85] E. M. Voorhees, "Overview of TREC 2002," presented at Eleventh Text Retrieval Conference (TREC 2002), Gaithersburg, MD, 2002.

[86] NIST, "The 2002 Topic Detection and Tracking (TDT2002) Task Definition and Evaluation Plan," National Institute of Standards and Technology, Gaithersburg, MD 2002.

[87] S.-B. Park and B.-T. Zhang, "Large Scale Unstructured Document Classification Using Unlabeled Data and Syntactic Information," presented at PAKDD 2003, 2003.

[88] M. A. Greenwood, "Implementing a Vector Space Document Retrieval System," University of Sheffield, Sheffield, UK December 2001.

[89] CIIR, "The Lemur Toolkit for Language Modeling and Information Retrieval," vol. 2004: Carnegie Mellon University Center for Intelligent Information Retrieval, 2004.

[90] F. Sebastiani, "Machine learning in automated text categorization," *ACM Computing Surveys*, vol. 34, pp. 1- 47, 2002.

[91] S. Robertson and I. Soboroff, "The TREC 2002 Filtering Track Report," presented at The Eleventh Text REtrieval Conference (TREC 2002), Gaithersburg, MD, 2002.

[92] I. Soboroff and S. Robertson, "Building a filtering test collection for TREC 2002," presented at 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Toronto, Canada, 2003.

[93] N. J. Belkin and W. B. Croft, "Information filtering and information retrieval: Two sides of the same coin," *Communications of the ACM*, vol. 35, pp. 29 - 38, 1992.

[95] T. Rose, M. Stevenson, and M. Whitehead, "The Reuters Corpus Volume 1 - from Yesterday's News to Tomorrow's Language Resource," Reuters Limited, London, UK 2002.

[96] E. A. Fox, "Characterization of Two Experimental Collections in Computer and Information Science," Cornell University, Ithaca, NY

TR 83 -- 561, 1983.

[97] Brandes and S. Corman, "Visual unrolling of network evolution and the analysis of dynamic discourse," *Information Visualization*, vol. 2, pp. 40 - 50, 2003.

[98] E. Fredkin, "Trie memory," *Communications of the ACM*, vol. 3, pp. 490 - 499, 1960.

[99] R. Goulden, P. Nation, and J. Read, "How large can a receptive vocabulary be?," *Applied Linguistics*, vol. 11, pp. 341 - 363, 1990.

Index

Q

R

S