SENTINEL: A Multiple Engine Information Retrieval and Visualization System[1]

Kevin L. Fox[2], Ophir Frieder[3], Margaret M. Knepper[2], and Eric J. Snowberg[2]

ABSTRACT

We describe a prototype Information Retrieval system, SENTINEL, under development at Harris Corporation's Information Systems Division. SENTINEL is a fusion of multiple information retrieval technologies, integrating n-grams, a vector space model, and a neural network training rule. One of the primary advantages of SENTINEL is its 3-dimenstional visualization capability that is based fully upon the mathematical representation of information within SENTINEL. This 3-dimensional visualization capability provides users with an intuitive understanding, with relevance feedback/query refinement techniques that can be better utilized, resulting in higher retrieval accuracy (precision).

1. INTRODUCTION

Information retrieval systems search and retrieve data from a collection of documents in response to user queries. Ever-increasing volumes of data are rendering traditional information retrieval systems ineffective in production environments. As data volumes continue to grow, it becomes increasingly difficult to develop engines that support search and retrieval with non-prohibitive search times. These larger data collections necessitate interactive paradigms to formulate accurate queries, as well as mechanisms that intuitively present the results to the user of the information retrieval system. We describe one such system, SENTINEL, a prototype information retrieval engine developed at Harris to support efficient and intuitive search, retrieval, and presentation of documents in response to user queries.

Two accuracy measures often used to evaluate information retrieval systems are recall and precision. Recall is the ratio of the number of the relevant documents retrieved from the total number of relevant documents available collection-wide. Precision is the ratio of the number of relevant documents retrieved from the total number of documents retrieved. In many interactive applications, however, users require only a few highly relevant documents to form a general assessment of the topic as opposed to detailed knowledge obtained by reading many related documents.

Time constraints and interest-level limit the user to reviewing the top documents before determining if the results of a query were accurate and satisfactory. In such cases, retrieval times and precision accuracy are at a premium, with recall potentially being compromised. A recent user study conducted by Excite Corporation demonstrated that less than five percent of the users looked beyond the first screen of documents returned in response to their queries [Wu - 97]. Other studies conducted on a wide range of operational environments have shown that the average number of terms provided by the user as input is often less than two and rarely greater than four [Fitzpatrick - 97]. The target users for the SENTINEL system also adhere to similar usage patterns in that high precision with efficient search times is vastly more critical than high recall. To meet user demands SENTINEL was designed to yield efficient, high precision retrieval.

Given the relatively few search words users often provide as query, an intutitive form of query refinement or enhancement is needed. SENTINEL's 3-D visualization provides users with such a capability. Using a user friendly visualization paradigm, users focus on the various key aspects of their search query and are graphically presented with the relationship of the documents based on these aspects to the query. By understanding these relationships, users can modify their search query to focus more or less on a given aspect.

SENTINEL is a fusion of multiple retrieval technologies, integrating n-grams, and a vector space model (VSM). High precision in SENTINEL is derived by providing users with multiple input interaction modes, fusing results obtained from multiple information retrieval search engines, each supporting a different retrieval strategy, and by supporting relevance feedback mechanisms. The accuracy improvement obtained by fusing results from multiple engines was demonstrated in [Gauch-96, Lee-97]. The need to fuse differing retrieval techniques as opposed to similar retrieval engines was demonstrated in [Alaoui-98]. In SENTINEL, the weight associated with each of the

[1] *This research was entirely funded under Harris Corporation's Internal Research and Development (IR&D) program.*

[2] *Harris Corporation - Information Systems Division * P.O. Box 98000 * Melbourne, FL 32902-9800*

[3] *Department of Computer Science and Applied Math, Illinois Institute of Technology, 10 W. 31st Street, Chicago, IL 60616*

individual retrieval engines can be modified to favor different engines based on the query types. For a comprehensive presentation of retrieval strategies and utilities, the reader is referred to a variety of recent references including [Grossman-98, Korfhage-97, Spark-Jones-97].

User understanding of the retrieved document set is enhanced by the use of an internally-developed, *n*-dimensional visualization system, called VisualEyes™. This tool supports multiple levels of data abstraction, clustered document presentation, data thresholding, and a variety of user interaction paradigms. The *n*-dimensional document visualization display enables the user to view different aspects of the document's topic. SENTINEL is able to reduce the display down to the most important aspects of the document. Displaying documents in a 3-dimensional space enables a user to see document clusters, the relationships of documents to each other, and also aids in new document identification. Documents near identified relevant documents (through SENTINEL queries) can be easily reviewed for topic relevance. The user is able to manipulate the dimensional view to gain new views of document relationships. Changing the document's dimensionality allows the information to be viewed for different topic aspects to aid in further identification of relevant documents.

2. SENTINEL OVERVIEW

SENTINEL is a C++ implementation of an Object-Oriented design and employs an Object-Oriented database (ObjectStore™ from Object Design Inc.). The basic structure of SENTINEL includes the following components:

- A web browser-based user interface that provides users with a mechanism to build queries for a topic of interest, execute the queries, examine retrieved documents, and build additional or refine existing queries.
- Multiple retrieval technologies utilizing n-gram and a Vector Space Model (VSM) to query the document corpus.
- A fusion component that combines and ranks the retrieved results of the various individual engines.
- A *3*-dimensional viewer that provides users with a mechanism to explore various aspects of retrieved documents, looking for additional relevant documents.

A user begins by defining a topic of interest, then proceeds to define one or more queries for that topic. User queries to SENTINEL can take the form of

- Keyword(s) or phrases
- An example document
- Document clusters

SENTINEL focuses on an interactive multi-pass approach. We do not assume that the information will be found immediately, and therefore the user needs to iteratively refine the query. SENTINEL allows the user to review the documents and select the documents most relevant to the topic. Relevant documents can be used as queries to further refine the topic. The user can then quickly query over the data with the additional queries.

3. SENTINEL'S RETRIEVAL ENGINES

Our goal in combining multiple differing retrieval technologies was to leverage the various strengths of each approach thus developing a more robust system, as shown in Table 1.

Table 1 Retrieval Engine Strengths and Weaknesses

	***n*-gram**	**Vector Space Model**
Strength	unique terms (e.g. proper nouns)	example documents used as input
	mis-spelled words	document meaning
	short documents (e.g. e-mail)	
Weakness	long documents	unique terms (e.g. proper nouns - terms that did not appear in the training corpus and hence do not appear in the VSM's dictionary)

3.1 ***n*-Grams**

SENTINEL employs an *n*-gram filter based on work with Least Frequent Tri-grams (LFT) [Yochum-85]. At present, SENTINEL uses a *3*-character sliding window (*3*-grams). Since typical early searches are keyword(s), the n-gram acts as a filter to quickly identify candidate documents to be reviewed early in the retrieval process. It is especially useful for phrases not in the dictionary of the Vector Space Model (VSM) retrieval engine.

3.2 Vector Space Model (VSM): Context Vectors

SENTINEL also uses a Vector Space Model to represent documents in an *n*-dimensional vector space. Words appearing in the document training corpus are represented as vectors, ω in the *n*-dimensional vector space, $\Re^n$.

The word vectors, $\omega \varepsilon \Re^n$, are normalized to unit length so that they all lie on the unit hyper-sphere and $\sum_{\forall_i} \omega_i^2 = 1$.

The similarity of two words is measured by computing the cosine similarity measure of the associated vectors, ω, $v \varepsilon \Re^n$:

$$\frac{(\omega, v)}{\|\omega\|_2 \|v\|_2} = \frac{(\omega \bullet v)}{|\omega||v|}$$

This measure is the cosine of the angle between the two vectors. Hence the higher the value (i.e., closer to +1), the smaller the angle between the two vectors. Since all of the word vectors are on the unit hyper-sphere, $\|\omega\|_2 = |\omega| = 1$ for all ω. Thus the cosine similarity measure reduces to

$$\frac{(\omega, v)}{\|\omega\|_2 \|v\|_2} = \frac{(\omega \bullet v)}{(1)(1)} = \omega \bullet v = \sum_{\forall_i} \omega_i v_i \ .$$

A vector for each document, $x \varepsilon \Re^n$, is constructed based on the terms in a document. Queries are treated like documents. Thus, documents and queries are compared by comparing their respective vectors in the vector space. Documents whose content, as measured by the terms in the document, correspond most closely to the content of the query are judged to be the most relevant. The documents are retrieved through keyword, word clusters (series of words), and example document queries mapped into the *n*-dimensional vector space. The documents whose vectors most closely coincide with the query's vector are retrieved.

Keywords, keyword phrases, single documents, and document clusters are provided as input to SENTINEL' s VSM. Queries constructed by SENTINEL' s VSM are broadly or narrowly focused, depending on the keywords, phrases, and example documents used in the queries. The document's score is obtained by computing the measure between the query and the document. Document scores range from 0 to +1. Negative scores indicate an irrelevant document. Testing has revealed that scores for relevant documents typically range from approximately .45 to 1. The closer to 1, the better the document matches the query.

Experiments have shown that the strongest performance of SENTINEL's VSM results from the use of example documents and document clusters. As passes are completed, top query results are reviewed and identified as relevant or irrelevant. Relevant documents from the query are input to the next pass of SENTINEL' s VSM.

3.2.1 Context Vector Training

A neural network (NN) training algorithm is used within SENTINEL to train the word vectors, $\omega \varepsilon \Re^n$, in our VSM. The NN training algorithm is based on the training rule for Kohonen's Self-Organizing Map [Kohonen-84, Haykin-94]. This unsupervised learning algorithm organizes a high-dimensional vector space based on features within the training data so that items with similar usage are clustered together. The words are trained on a representative sample document to be processed by the system. Heavier weights are placed on words in closer proximity. The NN also accounts for training that has already taken place by adjusting the less trained words. The training algorithm is shown in Figure 1.

Figure 1 Neural Network Training Rule

1. Initialize context vectors for each word in the dictionary. The values in the vector are chosen at random, and the vector is scaled to unit length, i.e., $\sum_{\forall i} \omega_i^2 = 1$

For N training epochs
For each training document
Set *word_index* to 0
Set *neighbor_word_index* to *word_index* +1
Adjust context vector for word(*word_index*) and word(*neighbor_word_index*). This accounts for proximity of words in the document, weighting more heavily words that are closer in proximity. This also accounts for training that has already taken place by adjusting highly trained words less. The algorithm is given as
$d = w_1 - w_2$
where w_1 = context vector for word(*word_index*)
and w_2 = context vector for word(*neighbor_word_index*)
$w_1(k+1) = w_1(k) - \mu_w k_1 d$
where μ_w = learning rate for word neighbor adjustments
and k_1 = (w1_num_updates * (*neighbor_word_index* - *word_index*)) $^{-1}$
$w_2(K+1) = w_2(k) + \mu_w k_2 d$
where k_2 = (w2_num_updates * (*neighbor_word_index* - *word_index*)) $^{-1}$
Renormalize w_1 and w_2
if (*neighbor_word_index* - *word_index*) < max_neighbor_words
Increment *neighbor_word_index*
Go to 3c
else if not done with all words in document
Increment *word_index*
Go to 3b
Calculate context vector for document
For every word in the document, adjust the word's context vector so that it is closer to the document's context vector. This steers words and the document that contains them towards a cluster of similar meaning in the vector space.
$d = w - v$
where *w* is the context vector for the word
and *v* is the context vector for the entire document
$w(k+1) = w(k) - \mu_d d$
where μ_d is the learning rate for word-to-document adjustment $(\mu_d \langle\langle \mu_w)$
Renormalize *w*
Note that early in the training, $\mu_w k_i$ should be much larger than μ_d, since the document's context vector is very random until some word training has been done. Eventually, μ_d may dominate $\mu_w k_i$ since k_i shrinks rapidly as word training continues.
Get next document and go to 3
Finish training epoch
Increment *epoch_count*
Reduce μ_w. This ensures that as training nears completion, updates are small even for words that have not been trained much.
If this is not the last epoch, go to 2, else done.

Application of the NN training rule causes the vectors for words with similar meaning (as defined by similar usage in the document corpus) to converge towards each other. Upon completion of training, words are assigned to the closest axis in the *n*-dimensional vector space. An axis represents the direction of each vector in the n-dimensional space. Words are assigned to the closest axis by taking the highest cosine similarity among the axes.

3.2.2 Document Reduction

Document's are reduced by removing stop-words, performing stemming, and inserting compound words. A document is further reduced by counting the number of words in each axis and retraining the axes with the highest count of words until over 70% of the document is represented. Figure 2 shows the percentage of words in the first 30 document axes before reduction and after reduction. Figure 3 shows a document that has been reduced from 58 to 26 axes.

Figure 2 Percentage of Words in the first 30 axes before and after Document Reduction

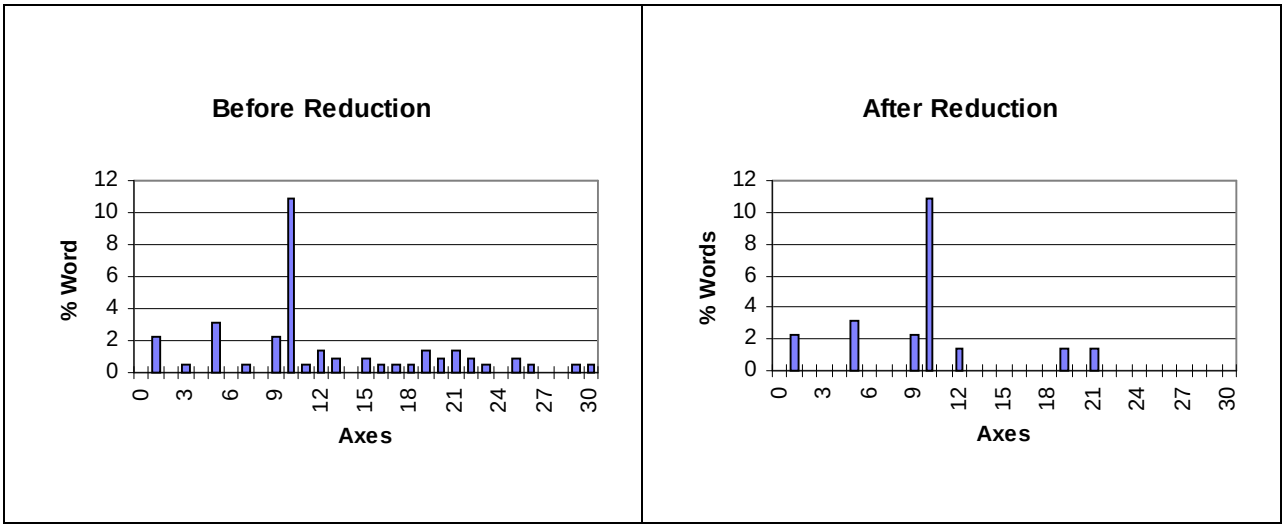


Figure 3 Text of a document being reduced

Rain, **snow** douse much of **Nevada wildfire**

Boys might not be **charged** with **sparking blaze**
June 25, 1996
GENOA, **Nevada** (CNN) -- **Firefighters gained** ground Tuesday on a **Nevada fire** that has charred about 4,000 **acres**, gutted four **homes** and forced thousands of **residents** to flee near **Lake Tahoe** since Sunday. With **fires** in five other states, federal **fire** officials are calling this one of the worst starts to the **annual wildfire season** in recent memory.

"This is definitely one of the worst, and the **season** is only starting," said a spokeswoman for the **National** Interagency **Fire** Center in Boise, Idaho.

The **Nevada** fire was 70 percent **contained** as of **midday** and **firefighters** expected full **containment** by Wednesday morning.

Cameraman Mike Conway **suffered burns** after being trapped in the **middle** of the **Nevada fire** while **filming**. He jumped in the back of a pickup truck to **escape** the flames.

rain(35) snow(93) nevada(53) forest_fire(19) charge(35) spark(9) blaze(183) nevada(53) firefight(37) gain(21) nevada(53) fire(10) acre(41) house(95) resident(52) lake_tahoe(5) fire(10) fire(10) annual(52) forest_fire(19) season(1) season(1) national(53) fire(10) nevada(53) fire(10) contain(10) midday(49) firefight(37) containment(10) suffer(54) burn(139) middle(53) nevada(53) fire(10) film(139) escape(139)

Reducing the number of words in a document reduces the procesing time to generate the internal document representation and, somewhat suprisingly also improves the query matching by eliminating some of the low content words. Via experimention, we determined that reducing beyond 70% resulted in removal of some high-content determining words in the document. Similar results were noted in [Grossman-97]. An example of test results obtained using a document collection comprising of over 2000 web news stories is shown below. The topic was "find awards/honors given to people and things (i.e., television shows)". The search started with the set of words: honor, award, mvp, noble prize, and hall of fame. The search was performed on documents which contain 100%, 90%, 80%, 70%, and 60% of the original document. We show the results of the keyword search using different reduction levels in Figure 4. As expected, the scores for recall drop as the document is reduced. The word retrieval precision starts to show an improvement for the 70% reduced document. The relevant documents found in the top 10 for each reduction were then used in the second pass to further define the query. Figure 5 shows the results of the keyword and document example search. For both the recall and precision the 70% document maintains the highest rate. The documents with minimal reduction start to have lower precision and recall scores.

Figure 4 Recall and Precision results from VSM component on SENTINEL'S different document axes reduction levels using just keyword retrieval

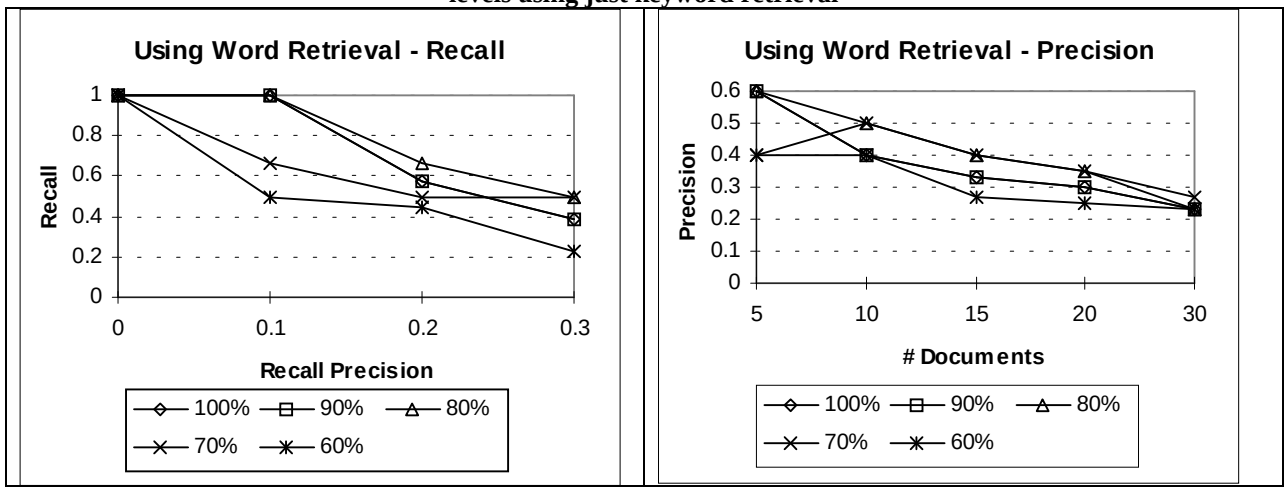
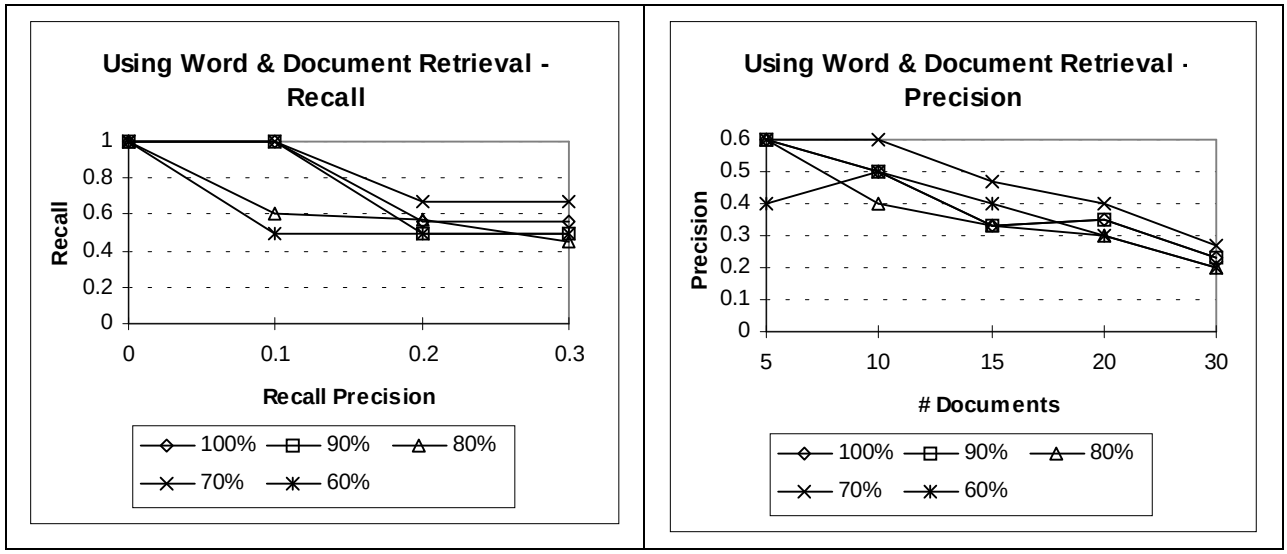


Figure 5 Recall and Precision results from VSM component on SENTINEL'S different document axes reduction levels using keyword and document example retrieval



3.2.3 Mathematical Representation of a Document

Information for a document is stored two ways: (1) the document context vector, and (2) the axis context vector. The document context vector, $x \varepsilon \Re^n$, is the sum of all the words that are in the document after reduction. It is used for building a single document query and to compare documents and queries. This mathematical representation of the document means the representation is built one time and doesn't need to be recalculated with every query. A document's axis context vector is the sum of the words in each axis vector after document clean up and reduction. The document context vector is used for building a query for a document cluster and the 3-D display.

Many implementations of the Vector Space Models count the frequency of occurrence of words and phrases to build the document queries. Frequency counts are done on the individual document files and for the document corpus. As new data are entered, the frequency counts must be updated (recomputed). Queries are built upon the highest frequency counts for documents, necessitating more computation time. SENTINEL creates an entirely mathematical representation of a document, and builds queries from that representation. The mathematical representation allows consistent grouping of the words so that they can be compared. Using our mathematical representation offers several advantages:

- Adding documents to the corpus does not require recalculation of word occurrence frequencies
- A large index is not required
- The vectors are small
- Documents are reduced
- Minimal calculations are required to build positive and negative queries – no document recalculation is required
- Document size independence
- Similarity equations are simplified

3.2.4 Query Building

Queries, like documents, are represented by n-dimensional context vectors, $y \varepsilon \Re^n$. Positive queries are combinations of word(s) and document(s). Single word and single document queries use the entire word/document as the query. When multiple documents are used to build the query, the document axes with the highest usage are used to build the query. Table 2 shows three documents with an example vector size 6, used to build a positive query. In this example, the query is built using axis 1, 3, and 5 since they contain the highest axis usage among the three relevant documents. The default is to use the axis used by all the documents and the next highest used axes. The user is allowed to lower or raise the number of axes used to build the query. When building a multiple document query, the documents should be similar. Otherwise, many irrelevant documents are retrieved.

Building a negative query is very similar to building a positive query. Instead of looking for the most frequently used axes, we look for the least frequently used axes in the specified relevant documents relative to the axis used by the bad document example. Table 3 shows a negative query being built. In this example, we use the least frequently used axes 2, 4, and 6 with respect to the good documents to build the negative query. The user can also raise or lower the number of axes used.

Table 2 Positive query example

Axis	Doc1	Doc2	Doc3	Query Axes
1	x	x	x	x
2		x		
3	x	x	x	x
4			x	
5	x		x	x
6	x			

Table 3 Negative query example

	Relevant Documents			Bad	Query
Axis	**Doc1**	**Doc2**	**Doc3**	**Doc4**	**Axes**
1	x	x	x	x	
2		x		x	x
3	x	x	x		
4			x	x	x
5	x		x		
6	x			x	x

4. RANKING

SENTINEL initiates each retrieval engine to calculate the document's score for each query. The retrieval engines maintain only the high level scores. The user can adjust the lowest acceptable score and retrieval engine weight to effect score results to favor/disfavor a particular retrieval engine. A ranking algorithm fuses the results of the retrieval engines and ranks the documents based on: the number of times the document was selected, highest score, lowest score, average score, location in the query list and number of retrieval engines locating the document. Irrelevant documents and queries can be removed.

The user specifies the lowest acceptable score for each of the retrieval engines. This helps eliminate the lower scoring documents. This also controls the number of documents shown to the user. The higher the number, the fewer the documents shown to the user. Additionally, the documents should reveal better matches to the query.

Each retrieval engine is assigned a specific percentage. The document scores for the retrieval engine are reduced by the specified percentage. Depending upon the query type, different retrieval engines can be emphasized.

- Potentially misspelled words may put more emphasis on the *n*-Gram retrieval
- Document example queries place more emphasis on the VSM retrieval engine

4.1 Scaling

SENTINEL standardizes the scores from each retrieval engine to range from 0 to 1. Each topic is composed of multiple queries from each of the retrieval components. The scores are scaled by query and for the entire topic (all the queries). Each set of scores are a separate entry into the ranking algorithm.

4.1.1 VSM Scaling

The query document for the VSM contains the entire vector and the axis vectors. Currently, scores are obtained by taking the cosine similarity measure of the query vector with the entire vector of each of the documents in the corpus. The scores range from -1 to 1. The closer to one the better the match - only high positive scores are kept. If none of the document scores equals one (1), then the documents are scaled based on the highest score for the query. This is a quick method of increasing the scores on potentially relevant VSM documents, thus allowing fusing of VSM highest results with the *n*-gram.

We have also started experimenting with applying the cosine similarity measure to the query vector against the corresponding axis of the documents in the corpus. Initial experiments have shown that it is only useful to do this for word queries since it limits the number of axes examined.

4.1.2 *n*-gram Scaling

The *n*-gram retrieval engine counts the number of occurrences of the least frequent term (*n*-gram). Since most early searches on the topic are keyword(s), the filter quickly identifies candidate documents to be reviewed early in the retrieval process. The *n*-gram engine is especially useful for keywords or phrases which did not appear in the document corpus used to train the VSM component. The identified relevant documents are used on the next pass through SENTINEL. A tri-gram, n=3, was selected due to the speed of processing and small amount of storage for the least frequency table.

The n-gram frequency count can vary widely; examples from TREC are shown in Table 4. The Text REtrieval Conference (TREC) is a yearly event sponsored by the National Institute for Standards and Technology focused on evaluating different information retrieval systems and approaches. TREC provides standardized data and queries which are used in the evaluation. The queries here are selected from the TREC collection. The *n*-grams need to be in the range from 0 to 1 to correspond with the standardized range. This presented some challenges because just taking the mean or dividing by the largest number does not provide a good representation of the *n*-gram documents.

Table 4 *n*-gram range of TREC data

	Query #301	Query #337	Query #350
# Files	79,777	550	3,271

# Files with 1 match	49,635	278	12
Largest # matches in a file	829	101	7,789

After examining the data, it appeared that we did not want to look at the documents that only had a few matching occurrences. We want to examine the documents that have a large number of the specified tri-Grams (*3*-grams) appearing in one document. The documents were divided into three groups: few matches, high matches, and scaled documents. The few matches were removed from the scaling calculation. Few matches consists of a large number of documents (greater than 50) with a couple of matches (approximately ranging from 1-20 matches) per document. High matches have a large number of matches in a single document. These are documents that need to be examined and are set to one. The remaining documents are scaled between zero and one. The scaling is to help identify documents that have the most *3*-gram matches and, most likely, should be reviewed.

4.2 Scoring Algorithm

An algorithm was developed to rank the document scores from different retrieval engines. The algorithm rates the following items:

- Number of times document identified
  - Per query
  - Per retrieval engine
- Maximum score
- Minimum score
- Average score
- Penalty points

For all the items except the penalty points each item is ranked, the higher the number the lower the score. The individual items are totaled, and the lowest final score indicates the best document. Penalty points are assigned to a document based on the number of retrieval engines locating the document and the document location in the individual query list.

A penalty is added to a document for each retrieval engine not identifying it as relevant. Multiple engines retrieving a document is a strong indication of a the document being relevant. This relevance correlation was also shown in [Lee-97]. The score must be above the minimum acceptable value after it is calculated for scaling and retrieval engine weight. During recent testing of the system the team placed a lot of emphasis on the number of times the file was identified by multiple queries and multiple retrieval engines locating the same file. Setting high penalties on these values brought relevant documents to the top of the list. More experimentation with different types of data will help identify the values that should be assigned to the parameters.

Originally, the scoring algorithm took all query scores and put them into one final list. No consideration was given to the document's list location in the individual queries. The algorithm was modified so that each document receives a penalty point for its location in each individual query list. This is intended to reward the documents that are located close to the top of the list in the individual queries, by assigning fewer penalty points.

5. SENTINEL'S USER INTERFACE AND *3*-D VISUALIZATION

Primary user interaction with SENTINEL is through a web-browser-based user interface. Users can build and tailor queries as the topic of interest is further defined, moving from a generic search to specific topic areas through query inputs. Queries may consist of a single keyword, multiple keywords (or phrases), keyword clusters [Korfhage-91], an example document, and document clusters. A set of documents retrieved for a particular topic may exhibit a variety of aspects. This can be seen, for example, in stories about the Oklahoma City bombing of 1996. There are relevant articles about the bomb, damage from the bomb blast, rescue work, the victims, suspects, the Timothy McVeigh trial, and the Terry Nichols trial, just to name a few. As an example, we apply SENTINEL to search for documents relevant to the trial of Timothy McVeigh for the Oklahoma City bombing. The document corpus consists of over 2000 news stories from the CNN web site on a variety of topics. In this case, a user begins by creating a new topic of interest: McVeigh Trial. Since this is a new topic, there are no queries associated with the topic. So our user creates a query by entering with a few keywords: "McVeigh", "trial", "bombing", and "Oklahoma City", as shown in Figure 8. The user has SENTINEL execute this query. As illustrated in Figure 9, a ranked list of documents, complete with

score, is returned to the user. Clicking on the document retrieves the text through HTML hyperlinks, enabling a user to determine the relevance, from his point of view, of a document to the topic. Top documents can be reviewed, and both relevant and irrelevant documents identified and marked as such. Irrelevant documents are filtered from subsequent queries. Removal of the higher-scoring irrelevant documents allowed lower scoring documents to be accepted on the final result list. Documents can also be marked for use as examples in additional queries for the topic.

Figure 8 Begin a topic for the McVeigh_trial with an initial keyword query consisting of the keywords "McVeigh", "trial", "bombing" and "Oklahoma City"

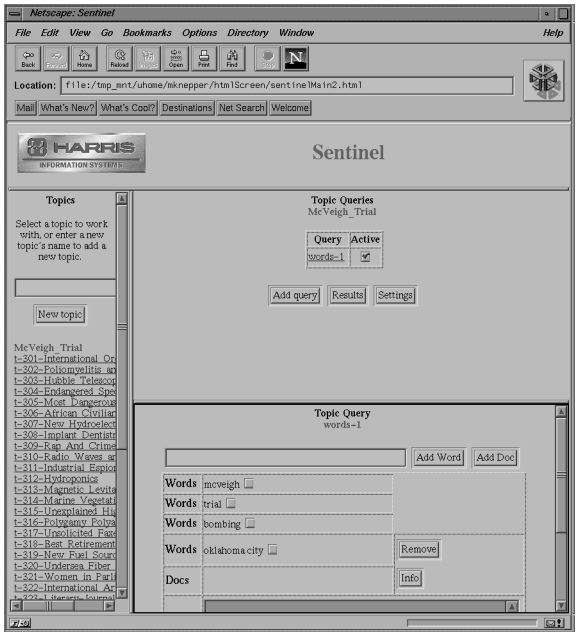
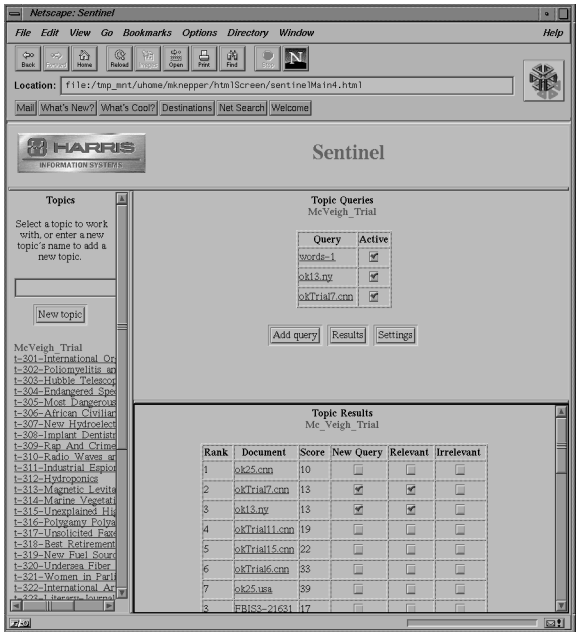


Figure 9 Results of the query are displayed in a ranked order list. A relevant document is added to the list of queries as an example document



In SENTINEL, we enhance user understanding of the retrieved document set through the use of VisualEyes™ , an internally-developed *n*-dimensional visualization tookit. The 3-D visualization tool supports multiple levels of data abstraction, clustered document presentation, data thresholding, and a variety of user interaction paradigms. The *n*-dimensional document visualization display enables the user to view different aspects of the document's topic. VisualEyes™ displays provides an intuitive display of document relationships and similarity. SENTINEL is able to reduce the display down to the most important aspects of the document.

As previously mentioned, the Oklahoma City bombing stories have a number of different aspects: the bomb, building damage, the victims, victim's families, the Timothy McVeigh trial, etc. Displaying documents in a *3*-dimensional space enables a user to see document clusters, the relationships of documents to each other, and also aids in the location of additional documents that may be relevant to a query. Documents near identified relevant documents (through SENTINEL queries) can be easily reviewed for topic relevance. The user is able to manipulate the dimensional view to gain new views of document relationships. Changing the document's axes allows the information to be viewed for different topic aspects to aid in further identification of relevant documents.

As illustrated in Figure 10, each document in the retrieved document corpus is represented mathematically in the *3*-D space by a colored cube. A red cube represents query request – in this case the words "trial", "bombing", "McVeigh" and "Oklahoma City". Yellow text is used to indicate the relevant documents found through text queries submitted to Sentinel. Other colors are used to indicated document clusters – documents related in some aspect. One can notice that relevant stories appear to separate from the other stories.

Figure 11, displays another perspective on the whole corpus of retrieved documents using different dimensions based on a story being the query. Again one can notice that the relevant stories appear to separate from the other stories. This separation is different from figure 10. This is focusing on different aspects of the story revealing a different configuration to the user. In both cases, the user can quickly identify areas to focus their search. Taking a look at some of the documents which have not been identified (green cubes), double click on any of the boxes to bring up the associated document, and make an inspection to determine whether or not it is relevant to the Timothy McVeigh trial.

Figure 10 Example Retrieval Visualization: 3-D viewing of keyword query to find stories about the Timothy McVeigh trial

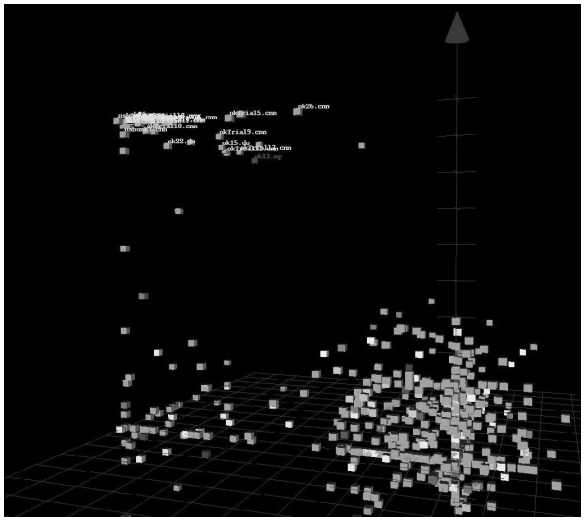
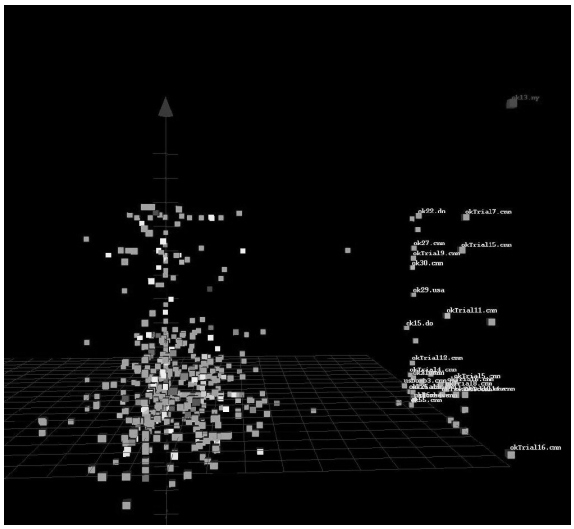


Figure 11 Screen shot using different dimensions - showing different aspects of the retrieved document set



6. CONCLUSION

The SENTINEL prototype has provided an efficient, high-level precision information retrieval and visualization system. It allows interactive formation of query refinement. It fuses results form multiple retrieval engines to leverage the strengths of the each. It has been designed for efficient maintenance, making it easy to add new documents. SENTINEL allows for multiple dictionaries and vocabularies – thus allowing a user to develop role-based dictionaries and/or vocabularies. Finally, SENTINEL provides a web-browser based interface for user interaction as well as a 3-D viewer for exploring the documents retrieved in response to a user's query.

As part of our evaluation of SENTINEL, we participated in both TREC-6 and TREC-7. In TREC-6, our 3-D visualization component was not yet available. In our second participation, we had the visualization capability. Accuracy results for TREC-7 are not yet available. However, from a personal standpoint, we significantly gained a greater understanding of our query results using our visualization component. We will continue to enhance SENTINEL to improve the system performance and capabilities.

REFERENCES

[Alaoui-98] S. Alaoui Mounir, N. Goharian, M. Mahoney, A. Salem, O. Frieder "Fusion of Information Retrieval Engines (FIRE)", 1998 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'98), Las Vegas, July 1998

[Croft-95] W. Bruce Croft, "What Do People Want from Information Retrieval?", D-Lib Magazine, November 1995.

[Damashek-95] Marc Damashek, "Gauging Similarity via *n*-Grams: Language Independent Categorization of Text", Science, 267(5199), Pp. 843-848.

[Fitzpatrick-97] Larry Fitzpatrick and Mei Dent, "Automatic Feedback Using Past Queries: Social Searching?", SIGIR-97 (Philadelphia, PA), Pp. 306-313, ACM, 1997.

[Gauch-96] Susan Gauch, Guijun Wang, "Information Fusion with ProFusion", in Webnet96 Proceedings, 1996

[Grossman-97] David Grossman, Ophir Frieder, D. O. Holmes, and D. C. Roberts, "Integrating Structred Data and Text: A Relational Approach," Journal of the American Society of Information Science, 48(2), February 1997.

[Grossman-98] David Grossman and Ophir Frieder, Ad Hoc Information Retrieval: Algorithms and Heuristics, Kluwer Academic Publishers, 1998.

[Haykin-94] Simon Haykin, Neural Networks: A Comprehensive Foundation, IEEE Press, Macmillan College Publishing Co., New York, 1994.

[Hearst-97] Marti A. Hearst and Chandu Karadi, "Cat-a-Cone: An Interactive Interface for Specifying Searches and Viewing Retrieval Results using Large Category Hierarchy", SIGIR-97 (Philadelphia, PA), Pp. 246-255, ACM, 1997.

[Korfhage-91] Korfhage, R. R., "To See or Not to See—Is That the Query?", In Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Infomration Retrieval, (Chicago, IL), Pp. 134-141, 1991.

[Korfhage-97] Robert R. Korfhage, Information Storage and Retrieval, John Wiley and Sons, 1997.

[Kohonen-84] Teuvo Kohonen, Self-Organization and Associative Memory, Springer-Verlag, New York, 1984.

[Lee-97] Joon Ho Lee, "Analysis of Mulitple Evidence Combination", in the proceeding of the 20th annual ACM SGIR Conference, (Philadelphia, PA), 1997.

[Schmitt-90] John C. Schmitt, Harris Corporation, U.S. Patent Number 5,062,143 (1990).

[Spark-Jones-97] Karen Spark-Jones and Peter Willet, Readings in Information Retrieval, Morgan Kaufmann Publishers, Inc. 1997

[Veerasamy-97] Aravindan Veerasamy and Russell Heikes, "Effectiveness of a Graphical Display of Retrieval Results", SIGIR-97 (Philadelphia, PA), Pp. 236-245, ACM, 1997.

[Wu-97] Jack Wu, Excite Corporation, personal communication with Ophir Frieder and David Grossman, November 21, 1997.

[Yochum-85] Julian A. Yochum, "High-Speed Text Scanning Algorithm utilizing Least-Frequent Trigraphs", IEEE Symposium on New Directions in Computing, 1985.