Cross-lingual *C*ST*RD*: English Access to Hindi Information

ANTON LEUSKI, CHIN-YEW LIN, LIANG ZHOU, ULRICH GERMANN, FRANZ JOSEF OCH, and EDUARD HOVY
USC Information Sciences Institute

We present a cross-lingual information retrieval and information space navigation system that operates largely language-independent. Most of the cross-lingual tasks are executed by a statistical machine translation system that is trained on parallel text and requires only fairly shallow pre-processing of the input. Information retrieval and text summarization require only very little language-specific processing. In the 2003 Suprise Language (Hindi) experiment, we were able to show that our system is indeed easily adaptable to new languages.

Categories and Subject Descriptors: []:

General Terms:

Additional Key Words and Phrases: Cross-lingual Information Retrieval

1. INTRODUCTION

The ultimate goal of cross-lingual information processing is to provide humans access to information that is available only in languages of which they have no or no sufficient knowledge. Naturally, machine translation plays a pivotal role in this endeavor — the language barrier must be crossed at some point. While it is desirable in any case to shield the user from irrelevant information and minimize the amount of text he or she has to read in order to obtain the information needed, this is especially true for machine translation output. In an exercise on rapid devlopment of Tamil MT in 2001 [Germann 2001], evaluators were asked to extract information from ca. 10 pages of MT output. They experienced this task as extremely tedious, tiring and frustrating. Despite encouraging progress in MT quality over the past years, MT output is still, for the most part, ungrammatical and quite hard to read. Limiting the amount of text the user has to scan to obtain information is therefore crucial.

In this paper, we present a cross-lingual adaption of C*ST*RD,[1] an interactive information access system that provides information retrieval, document space exploration, multidocument summarization, and more. The system was built in less than a month during

[1] Clustering, Summarization, Translation, Reformatting and Display

This work was supported by the Darpa TIDES program under contracts Nos. N66001-00-1-8914 and N66001-00-1-8916.

the 2003 DARPA TIDES surprise language exercise by adapting and combining several existing technologies into one integrated information access interface.

C*ST*RD consists of two main components: *Lighthouse* for information retrieval and document space exploration, and *iNeATS* for interactive multidocument summarization. Other components, such as machine translation and multidocument headline generation operate in the background; the user does not interact with them directly.

2. LIGHTHOUSE

Lighthouse is an information organization and visualization system that supports full text search and focuses on presenting the retrieved documents to the user in such a way that facilitates effective browsing and exploration of the information. In contrast to the traditional search engines (e.g., such as Google) that arrange the retrieved documents in a linear list by their similarity to the query, Lighthouse exploits the other relationship among documents – the inter-document similarity. It uses the inter-document similarity in three different ways to organize the retrieved document set, presenting a visual summary of the set's content and helping the user to locate interesting information. In this section we briefly describe the main components of the system and discuss how it has been adapted to take into account a cross-lingual nature of the Surprise Language experiments. A full description of the Lighthouse system and its features can be found in Leuski [2001b] and Leuski and Allan [2003].

2.1 Cross-language retrieval

The version of Lighthouse used in the Surprise Language experiment is build on top of the Lucene search engine [Lucene]. Lucene is an open source search engine written in Java. It supports full text indexing and searching using techniques very similar to the best research retrieval systems. It has a powerful query language and it is easily expandable.

The default distribution of Lucene handles only European languages. We adapted the search engine to Hindi by implementing a word tokenizer for the language that breaks the input stream of text into individual terms or tokens. We then indexed the Hindi collection using the Lucene indexing functions.

Additionally, we implemented a query translation module for Lucene that takes a query in English and produces a translated query in Hindi. The translation algorithm is based on the English-Hindi dictionary provided by the University of Maryland College Park. The algorithm works by performing a greedy search on the English part of the lexicon using the query string. We select the Hindi parts of the lexicon corresponding to the matching English phrases and join them into the resulting query string. Matching long phrases in the query is preferred over matching the individual words. If the match is not found on the first pass, the search is repeated using the stemmed version of the words in the query. We used the Porter stemmer [Porter 1980] to stem both the query and the lexicon. If an English term had multiple Hindi translations, we added all the translation variants to the result. The English words that were not found in the lexicon were copied to the result unchanged.

Lighthouse shows the translated Hindi query to the user when the user submits the English query to the system. In the example presented in Figure 1, the user typed the word "bomb" and the corresponding translation is shown below the query string.

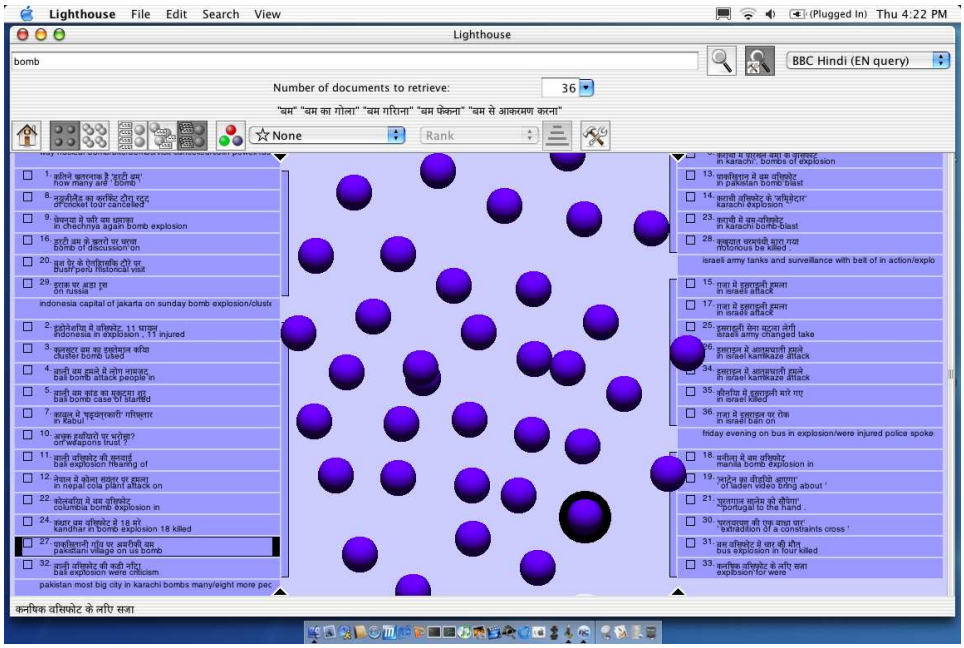

Fig. 1. The Lighthouse interface.

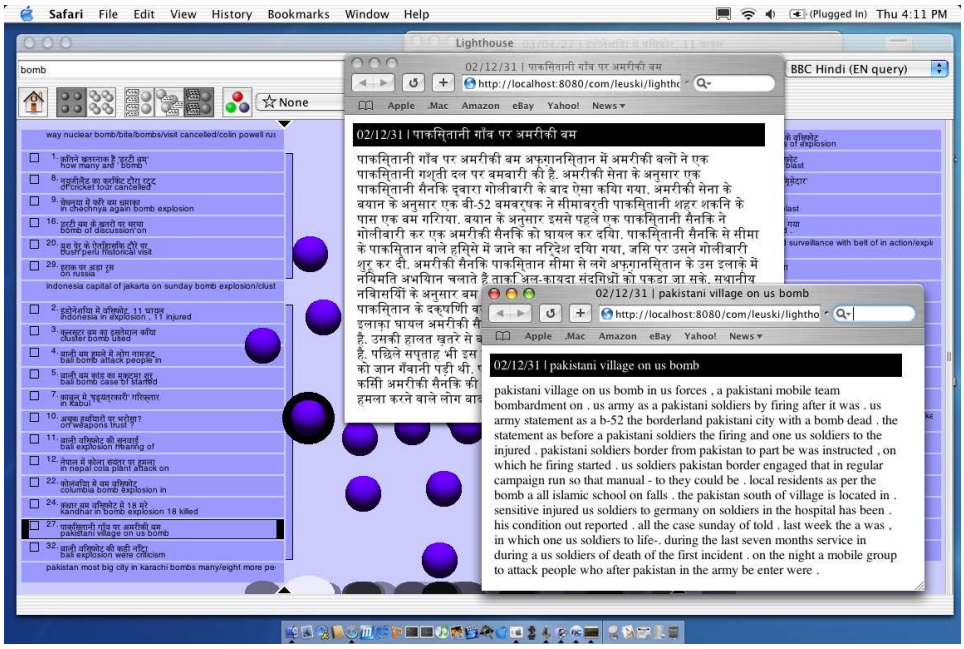

Fig. 2. The Lighthouse interface with open document windows.

2.2 Related work on clustering and visualization

Numerous studies suggest that document clustering (topic-based grouping of similar documents) is an effective way of organizing the text documents. The use of clustering is based on the Cluster Hypothesis of Information Retrieval: "closely associated documents tend to be relevant to the same requests" [van Rijsbergen 1979, p.45]. It has been studied in the context of improving the search and browsing performance by pre-clustering the entire collection [Willett 1988; Cutting et al. 1992; Cutting et al. 1993]. Croft [1978] and more recently Hearst and Pedersen [1996], showed that the Cluster Hypothesis holds in a retrieved set of documents. Their system breaks the retrieved results into a fixed number of document groups. Leuski and Croft [1996] considered a similar approach, but instead of fixing the number of clusters, they set a threshold on the inter-document similarity. While these systems operate with full-text documents, MetaCrawler-STC [Zamir and Etzioni 1998] at the University of Washington places the web search results into overlapping clusters based on the snippets returned by the engine. NorthernLight [NorthernLight] is an example of on-line commercial system that organize the search results into folders.

All these system generally leave open two questions: how to decide where to draw the border between clusters and how to express the relationship between objects in different clusters. So what we need is a system that does not require the hard decision, a system that visualizes the documents and leaves it to the user to isolate the clusters. Galaxies [Wise et al. 1995] computes word similarities and displays the documents as a universe of "docustars". In a space with a huge number of "docustars" it is not easy to select the object the user wants to explore.

The Vibe system [Dubin 1995] is a 2-D display that shows how documents relate to each other in terms of user-selected dimensions. The documents being browsed are placed in the center of a circle. The user can locate any number of terms inside the circle and along its edge, where they form "gravity wells" that attract documents depending on the significance of those terms in that document. The user can shift the location of terms and adjust their weights to better understand the relationships between the documents. The LyberWorld system [Hemmje et al. 1994] includes an implementation of Vibe, but presented in three dimensions.

The Bead system [Chalmers and Chitson 1992] uses a Multidimensional Scaling algorithm called spring-embedding for visualizing the document set. The system was designed to handle very small documents – bibliographic records represented by human-assigned keywords. The Bead research did not evaluate the system. The browsing system studied by Rodden et al. [1999] uses the same algorithm to organize small sets of images based on their similarity. They observed that such an organization is a more effective way to navigate the image collection when compared to a random arrangement of the same images.

Swan and Allan [1998] considered a system with the ranked list and the spring-embedding visualization. Their system presented complete, full-sized documents and it was studied in the context of searching for multiple topics across several query runs. There was no exploration of how the ranked list and the visualization could be effectively used together. Leuski and Allan [1998; 2000] adopted a similar approach and applied it to locating the relevant information among the retrieved documents. They have attempted an off-line analysis simulating a user browsing the system.

Other approaches exist that attempt to visualize the document space based on the concept similarities using some form of neural network such as Kohonen's self organizing

maps [Lin et al. 1991; Rushall and Ilgen 1996; Wise et al. 1995]. The Narcissus system [Hendley et al. 1995] applied the spring-embedding algorithm to visualizing structures of pages and links on the World Wide Web. Song experimented with visualization of clusters for a bibliographic database [Song 1998].

The prevalence of Web search engines points at the importance of search and the value of the ranked list. Clustering efforts illustrate the value of using inter-documents relationships to group the collection for understanding. Lighthouse combines both presentations – ranked list and clustering – but in a way that avoids the troublesome problems of hard decisions in clustering.

2.3 Ranked list

Lighthouse presents to the user the top portion of retrieved document set – usually it is the best 50 document found by the search engine. The size of the retrieved set can defined by the user. In Fig. 1, the limit is set to 36.

The retrieved documents are shown as the ranked list of document headlines and a set of spheres arranged in two- or three-dimensional space. Each sphere corresponds to a document in the retrieved set and the spheres are positioned in proportion to the inter-document similarity: a pair of similar document will be represented by two spheres that are close to each other and a pair of spheres that are far apart correspond to documents that are not related. We describe the details of the the latter presentation based on the spring-embedding visualization algorithm in Section 2.5.

The ranked list is broken into two columns of equal length each on the left and on the right side of the screen with the configuration of spheres positioned in the middle. The list flows starting from top left corner down and again from the top right corner to the bottom of the window. The pages are ranked by the search engine in the order they are presumed to be relevant to the query. The rank number precedes each document in the list.

For each document we show both the original Hindi headline and the English headline produced by the machine translation subsystem of C*ST*RD (see Section 5). The documents in the list can be ordered by their rank or alphabetically by either version of the headline. The retrieved documents can also be partitioned in a set of clusters and the corresponding headlines will be grouped together in the list (see Section 2.4).

Each sphere in the visualization is linked to the corresponding document title in the ranked list so clicking on the sphere will select the title and vice versa. The user can examine the clustering structure and place it in the best viewing angle by rotating, zooming, and sliding the whole structure while dragging the mouse pointer. (Only the spheres can be manipulated in this fashion – the ranked list remains in place.)

A click on the document title (or sphere) with the right mouse button brings up a pop-up menu that allows the user select either the original Hindi document text or the translated English text to be open in the web browser (see Figure 2).

2.4 Clustering

Figure 1 shows the 36 retrieved documents partitioned into 5 clusters. Each cluster is represented by a rectangular bracket or “handle” that runs parallel to the cluster. We order the documents in the clusters using their rank and sort the clusters using the rank of the highest ranked document in each cluster. Our mono-lingual English experiments show that such document organization can be much more effective in helping the user to locate the relevant information than the ranked list [Leuski 2001a].

Each cluster is preceded by its own headline. The cluster headline is produced by the GOPS multi-document headline generation system of C*ST*RD (see Section 4). The GOPS system is implemented as a Perl script which is called from the main Lighthouse Java code on demand. The cluster headlines are generated from the English translations of the Hindi documents produced by the machine translation subsystem of C*ST*RD (see Section 5).

Lighthouse uses the Ward hierarchical agglomerative clustering algorithm to generate the document set partition [Mirkin 1996]. On input the algorithm receives a set of objects and a matrix of inter-object distances. It starts by assigning each object to its own unique cluster. The algorithm iterates through the current cluster set by selecting the closest pair of clusters and merging them together forming a new cluster that replaces them in the cluster set. We terminate the clustering process as soon as the distance between the closest pair of clusters exceeds a predefined threshold. This threshold is set to the value which which generally produces good clusters [Leuski 2001a], but it can also be adjusted by the user.

To compute inter-document distances we employ the vector-space model for document representation [Salton 1989] – each document *j* is defined as vector V_j, where $v_{i,j}$ is the weight in this document of the i-th term in the vocabulary. The term weight is determined by an ad-hoc formula [Allan et al. 1998], which combines Okapi's *tf* score [Robertson et al. 1995] and INQUERY's normalized *idf* score:

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

where $v_{i,j}$ is the weight of the *i*th term in the vocabulary in the *j*th document, $tf_{i,j}$ is the number of times the term occurs in the document, $docf_i$ is the number of documents the term occurs in, $doclen_j$ is the number of terms in the document, *avgdoclen* is the average number of terms per document in the collection, and *colsize* is the number of documents in the collection. The similarity between a pair of documents is computed as the cosine of the angle between the corresponding vectors ($\cos\theta$) [Salton 1989]. In this study we use one over the cosine ($1/\cos\theta$) to define the distance between a pair of documents.

2.5 Spring-embedding

Partitioning the document set into non-overlapping clusters simplifies the inter-document relationships. It is very easy from a user's point of view to tell a pair of similar documents from a pair of dissimilar ones – similar documents are assigned to the same cluster. This simplification comes at a cost of losing the intricate details of the inter-document similarity, that might otherwise be useful for locating relevant information. Additionally, the clustering algorithm is a parametric approach. Determining the best value for the parameter that defines the final partition of the document set is a very hard question which one would like to avoid in a real-world system.

We described in Section 2.3 that Lighthouse visualizes each document as a sphere in two- or three-dimensional space and positions the spheres in proportion to inter-document similarity. For this presentation Lighthouse uses the same inter-document distance matrix described in the previous section. In contrast to the clustering presentation described there, the visualization discussed in this section does not require any clusters – if a user sees some spheres arranged in groups, that is just an artifact of the inter-document similarity. Simply put, it just draws the documents, illustrating any structure that is already present in

the data. Assigning any meaning to the structure is the user's task.

To generate a set of spheres that represent the multidimensional document vectors we use a Multidimensional Scaling (MDS) algorithm called the spring-embedding [Fruchterman and Reingold 1991]. The spring-embedding algorithm models each document vector as an object in 2- or 3-dimensional visualization space. It is assumed that the objects repel each other with a constant force. They are connected with springs and the strength of each spring is inversely proportional to the $1/cos$ dissimilarity between the corresponding document vectors. This "mechanical" model begins from a random arrangement of objects and due to existing tension forces in the springs, oscillates until it reaches a state with "minimum energy" – when the constraints imposed on the object placements by the springs are considered to be the most satisfied. The result of the algorithm is a set of points in space, where each point represents a document and the inter-point distances closely mimic the inter-document dissimilarity.

Our experiments with the spring-embedding visualization in monolingual settings showed that such presentation can be used effectively to interactively direct the user's search for relevant information in the top ranked portion of the retrieved set [Allan et al. 2000]. We have experimentally shown that this approach significantly exceeds the initial performance of the ranked list and rivals in its effectiveness the traditional relevance feedback methods.

2.6 Wizard

Lighthouse also partitions the documents based on the user's examples. The user selects one or several documents and assigns them to a particular category. Each category is associated with a color. The category assignments are indicated by painting the corresponding document titles and spheres with the category color. The user starts with two categories: "relevant" and "non-relevant". She can introduce new categories at will.

Lighthouse dynamically computes the likelihood of the other documents to be assigned to the category and presents this information to the user. The title and the sphere of the document that was assigned to the category by the user is filled with a bright shade of the category color. In contrast, the automatically assigned documents are indicated with a less intense shade of color and the intensity of the shading is proportional to the absolute value of the likelihood.

This category assignments are computed using a relevance feedback "wizard" based on a neural network [Leuski 2000]. The wizard takes into account the number of documents the user assigned to each category and the average distances between them and each unassigned document. If the user confirms the Lighthouse category assignments by marking the selected document, the system dynamically recomputes its estimates for other documents and directs the user to the most interesting information. Our experiments showed that wizard-directed browsing of the retrieved document set is significantly more effective then using the state-of-the-art relevance feedback method of information retrieval [Leuski 2000].

3. INEATS

The second main component of the C*ST*RD interface, iNeATS[2], allows the user to summarize and examine small groups of documents in more details then allowed by Light-

[2] **I**nteractive **NE**xt generation **A**utomatic **T**ext **S**ummarization

house. It can be invoked from within Lighthouse by selecting a group of documents on the Lighthouse screen and choosing "Summarize" from the pop-up menu.

An automatic multi-document summarization system generally works by extracting relevant sentences from the documents and arranging them in a coherent order [McKeown et al. 2001; Over 2001]. The system has to make decisions on the summary's size, redundancy, and focus. Any of these decisions may have a significant impact on the quality of the output. We believe a system that directly involves the user in the summary generation process and adapts to her input will produce better summaries. Additionally, it has been shown that users are more satisfied with systems that visualize their decisions and give the user a sense of control over the process [Koenemann and Belkin 1996].

We see three ways in which interactivity and visualization can be incorporated into the multi-document summarization process:

(1) give the user direct control over the summarization parameters such as size, redundancy, and focus of the summaries.
(2) support rapid browsing of the document set using the summary as the starting point and combining the multi-document summary with summaries for individual documents.
(3) incorporate alternative formats for organizing and displaying the summary, e.g., a set of news stories can be summarized by placing the stories on a world map based on the locations of the events described in the stories.

The iNeATS part of C*ST*RD addresses these three directions. It is built on top of the NeATS multi-document summarization system.

3.1 NeATS

NeATS [Lin and Hovy 2002] is an extraction-based multi-document summarization system. It is among the top two performers in DUC 2001 and 2002 [Over 2001]. It consists of three main components:

Content Selection. The goal of content selection is to identify important concepts mentioned in a document collection. NeATS computes the likelihood ratio [Dunning 1993] to identify key concepts in unigrams, bigrams, and trigrams and clusters these concepts in order to identify major subtopics within the main topic. Each sentence in the document set is then ranked, using the key concept structures. These n-gram key concepts are called topic signatures.

Content Filtering. NeATS uses three different filters: sentence position, stigma words, and redundancy filter. Sentence position has been used as a good important content filter since the late 60s [Edmundson 1969]. NeATS applies a simple sentence filter that only retains the N lead sentences. Some sentences start with conjunctions, quotation marks, pronouns, and the verb "say" and its derivatives. These stigma words usually cause discontinuities in summaries. The system reduces the scores of these sentences to demote their ranks and avoid including them in summaries of small sizes. To address the redundancy problem, NeATS uses a simplified version of CMU's MMR [Goldstein et al. 1999] algorithm. A sentence is added to the summary if and only if its content has less than X percent overlap with the summary.

Content Presentation. To ensure coherence of the summary, NeATS pairs each sentence with an introduction sentence. It then outputs the final sentences in their chronological order.

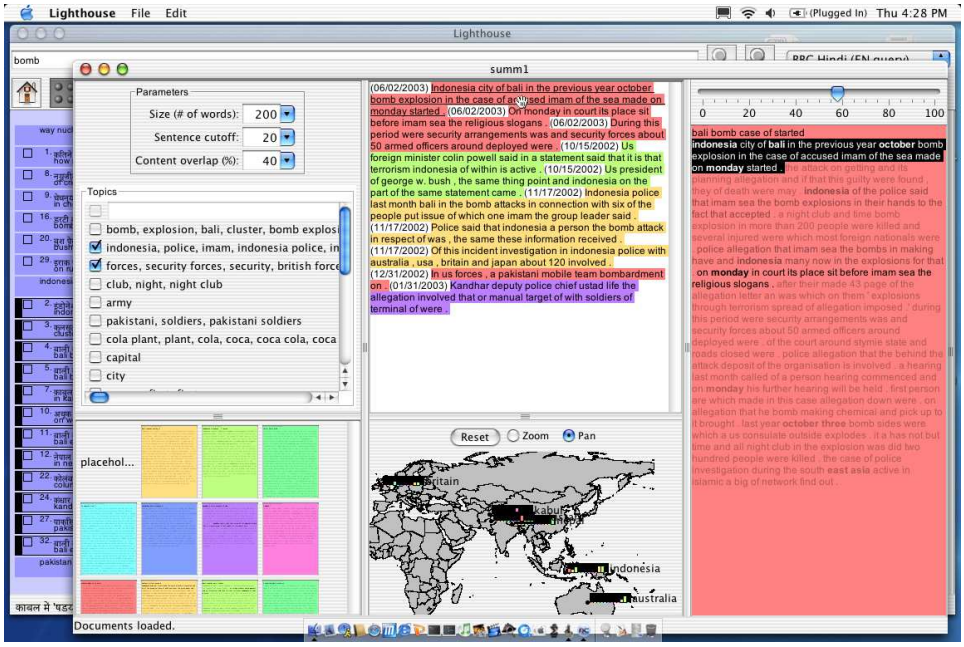

Fig. 3. The *iNeATS* interface.

3.2 Interactive Summarization

Figure 3 shows a screenshot of the iNeATS system. We divide the screen into three parts corresponding to the three directions outlined at the beginning of the section. The *control* panel displays the summarization parameters on the left side of the screen. The *document* panel shows the document text on the right side. The *summary* panel presents the summaries in the middle of the screen.

3.3 Controlling the Summarization Process

The top of the control panel provides the user with control over the summarization process. The first set of widgets contains controls for the summary size, sentence position, and redundancy filters. The second row of parameters displays the set of topic signatures identified by the iNeATS engine. The selected subset of the topic signatures defines the content focus for the summary. If the user enters a new value for one of the parameters or selects a different subset of the topic signatures, iNeATS immediately regenerates and redisplays the summary text in the top portion of the summary panel.

3.4 Browsing the Document Set

iNeATS facilitates browsing of the document set by providing (1) an overview of the documents, (2) linking the sentences in the summary to the original documents, and (3) using sentence zooming to highlight the most relevant sentences in the documents.

The bottom part of the control panel is occupied by the document thumbnails. The documents are arranged in chronological order and each document is assigned a unique color to paint the text background for the document. The same color is used to draw the document

thumbnail in the control panel, to fill up the text background in the document panel, and to paint the background of those sentences in the summary that were collected from the document. For example, the screenshot shows that a user selected the second document which was assigned the orange color. The document panel displays the document text on orange background. iNeATS selected the first two summary sentences from this document, so both sentences are shown in the summary panel with orange background.

The sentences in the summary are linked to the original documents in two ways. First, the document can be identified by the color of the sentence. Second, each sentence is a hyperlink to the document – if the user moves the mouse over a sentence, the sentence is underlined in the summary and highlighted in the document text. For example, the first sentence of the summary is the document sentence highlighted in the document panel. If the user clicks on the sentence, iNeATS brings the source document into the document panel and scrolls the window to make the sentence visible.

The relevant parts of the documents are illuminated using the technique that we call *sentence zooming*. We make the text color intensity of each sentence proportional to the relevance score computed by the iNeATS engine and a zooming parameter which can be controlled by the user with a slider widget at the top of the document panel. The higher the sentence score, the darker the text is. Conversely, sentences that blend into the background have a very low sentence score. The zooming parameter controls the proportion of the top ranked sentences visible on the screen at each moment. This zooming affects both the full-text and the thumbnail document presentations. Combining the sentence zooming with the document set overview, the user can quickly see which document contains most of the relevant material and where approximately in the document this material is placed.

The document panel in Figure 3 shows sentences that achieve 50% on the sentence score scale. We see that the first half of the document contains two black sentences: the first sentence that starts with “US Insurers...”, the other starts with “President George...”. Both sentences have a very high score and they were selected for the summary. Note, that the very first sentence in the document is the headline and it is not used for summarization. Note also that the sentence that starts with “However,...” scored much lower than the selected two – its color is approximately half diluted into the background.

There are quite a few sentences in the second part of the document that scored relatively high. However, these sentences are below the sentence position cutoff so they do not appear in the summary. We illustrate this by rendering such sentences in slanted style.

3.5 Alternative Summaries

The bottom part of the summary panel is occupied by the map-based visualization. We use BBN’s IdentiFinder [Bikel et al. 1997] to detect the names of geographic locations in the document set. We then select the most frequently used location names and place them on world map. Each location is identified by a black dot followed by a frequency chart and the location name. The frequency chart is a bar chart where each bar corresponds to a document. The bar is painted using the document color and the length of the bar is proportional to the number of times the location name is used in the document.

The document set we used in our example describes the progress of the hurricane Andrew and its effect on Florida, Louisiana, and Texas. Note that the source documents and therefore the bars in the chart are arranged in the chronological order. The name “Miami” appears first in the second document, “New Orleans” in the third document, and “Texas” is prominent in the last two documents. We can make some conclusions on the hurricane’s

path through the region – it traveled from south-east and made its landing somewhere in Louisiana and Texas.

4. MULTI-DOCUMENT HEADLINE GENERATION

Cluster headlines in the Lighthouse interface help the user to decide which document clusters are worth further examination. Since cluster headlines need to be short, sentence extraction is not an option.

Our multi-document headline generation module, a perl implementation of the *GOSP*[3] algorithm [Zhou and Hovy 2003], generates headlines for document clusters in two stages: First, it generates a headline for each document in the cluster. Then it selects among the individual document headlines those of the highest "informativeness".

4.1 Single-document headline generation

Single-document headline generation is performed in the following manner.

(1) **Select headline-worthy words from the document body**
Potential headline candidates are determined by statistical model trained on a collection of documents and their headlines. The scoring function combines two models of "headline worthiness":

$$Score(w) = P_{fo}(w) \times P_{tf}(w)$$

$P_{fo}(w)$ is the probability of a word w occurring in the headline given the position (measured by sentence number) of its first occurrence in the document body. It is estimated as follows.
Let $fo(w)$ be a function that returns the position (in terms of the sentence number) of the first occurrence of the word w in the document body of a given document, and let

$$Count\_Pos_i = \sum_{k=1}^{M} \sum_{j=1}^{N_k} \delta(fo_k(h_{k,j}) = i)$$

be the number of times a headline word has its first occurence in the document body in position i in a document collection, where M is the number of documents in the collection, N_k the number of words in the headline of document k, fo_k the "first occurrence" function with respect to document k, $h_{k,j}$ the j-th word in the headline of document k, and δ an evaluation function that returns 1 if the argument expression is true, 0 otherwise.
Then

$$P_{fo}(w) = \frac{Count\_Pos_{fo(w)}}{\sum_{k=1}^{Q} Count\_Pos_Q},$$

where Q is the highest sentence number in the training collection.
An evaluation of this measure in Zhou and Hovy [2003] showed that roughly 40% (310 out of 808) of the words in headlines also occur within the first 50 words of the document body. Similar observations can be found in Zajic et al. [2002] and Lin and Hovy [1997].

[3]Global Word Selection with Localized Phrase Clustering

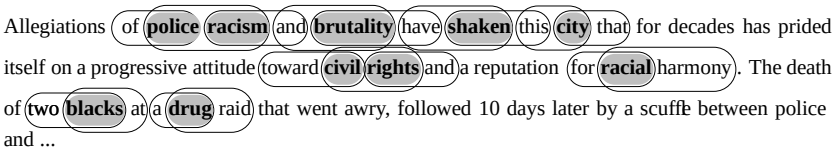

Fig. 4. GOSP forming bigram chains around headline-worthy words in the initial portion of the document body. The headline generated by the system is *police racism and brutality have shaken this city*.

The same evaluation in Zhou and Hovy [2003] also indicated that when the length of the headline is restricted, predictions are best if the sentence position model is combined with a lexicalized model based on the correlation of a word's occurrence in a document's body and it's occurrence its headline (cf. Jin and Hauptmann [2001]).

$$P_{tf}(w) = \frac{\sum_{j=1}^{M}(TF_{body}(w,j) \times TF_{headline}(w,j))}{\sum_{j=1}^{M} TF_{body}(w,j)},$$

where $TF_{body}(w,j)$ is the number of occurrences of the word w in the document body of the j-th document in the collection, and $TF_{headline}(w,j))$ the number of occurrences of w in the document's headline.

(2) **Extract phrases from the initial 50 words of the document body**
Next, the GOSP algorithm forms "bigrams chains" around each occurrence of the ten highest-scoring words within the first 50 words of the document body. This restriction is based on the aforementioned observation that the more important words in the document tend to have their first occurrence early in the text. These bigram chains form candidate headline phrases.

The headline candidate phrases are then sorted by their length in decreasing order. Starting with the longest phrase, candidate phrases are added to the set of final headline phrases until the length threshold is met.

(3) **Cleanup**
Finally, dangling verbs, particles, conjunctions at the beginning and the end of the final headline phrases are removed. In order to do so, a part-of-speech tagger is run on all input texts. Using a set of hand-written rules, dangling words and words in the stop list are removed.

4.2 Multi-document Headline Assembly

The procedure described so far generates sets of phrases for single-document headlines, resulting in a fairly large set of overlapping candidate phrases for the entire collection. (If there is any similarity between the document, there will also be similarity between the phrases selected for the headline.)

From this set of candidate phrases, we must now extract the ones with the highest "information value", as measured in the ratio of headline-worthy words to the total number of words in the phrase, and the least overlap between the phrases selected.

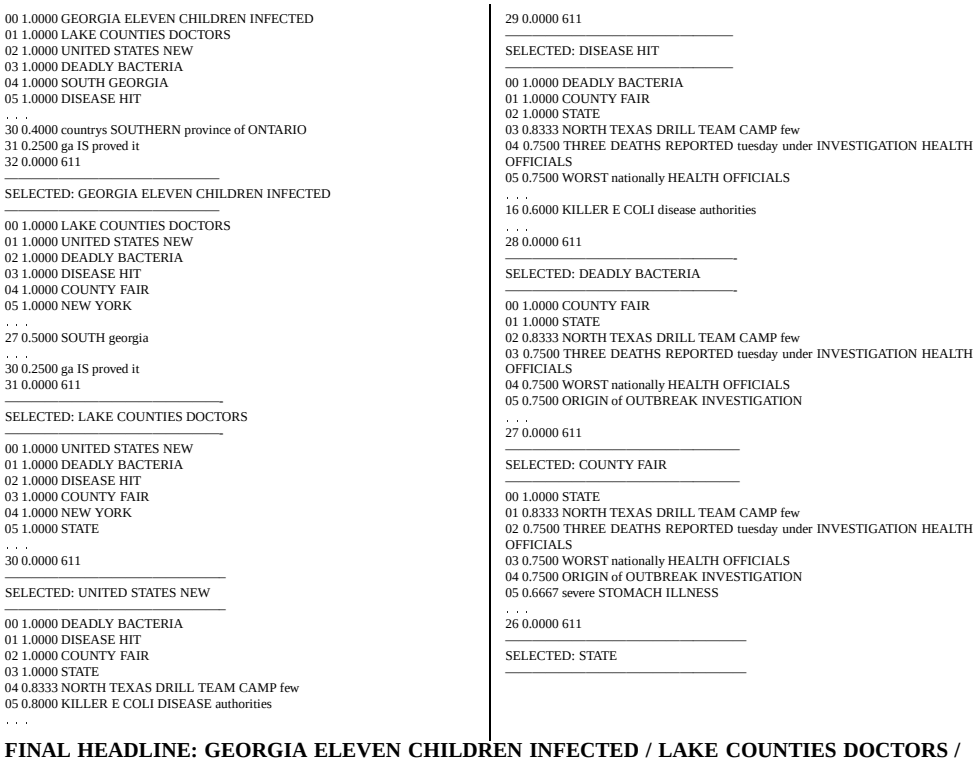

Fig. 5. Multi-document Headline Assembly. Candidate phrases are ranked (1st column) by the ratio (2nd column) of keywords (headline-worthy words; displayed in upper case) and total number of words in the phrase. After each phrase selection, the keywords in it become "downgraded" to non-keywords (displayed in lower case), and the remaining phrases are re-ranked. For example, SOUTH GEORGIA drops from rank 4 to rank 27 after GEORGIA ELEVEN CHILDREN INFECTED has been selected, because GEORGIA loses its value as a keyword. The process stops when the headline length limit has been reached.

The selection process works as follows. First, all phrases in the collection are ranked by the ratio of keywords (headline-worthy words) and the total number of words in the phrase. The highest ranking one is selected. (In the sample in Fig. 5, we prefer longer phrases over shorter ones if they have the same keyword ratio. If this is the best strategy has yet to be determined.) Once a phrase has been selected, all keywords in it lose their value as keywords, and the remaining phrases are re-ranked. Note, for example, that the phrase SOUTH GEORGIA SOUTH GEORGIA drops from rank 4 to rank 27 after GEORGIA ELEVEN CHILDREN INFECTED has been selected. This is because GEORGIA has lost its value as a keyword, so that the keyword ration drops from 100% to 50%. This procedure is repeated until the headline length threshold is met.

4.3 GOSP before MT, or MT before GOSP?

When generating headlines for document clusters in a cross-lingual application, an important decision must be made: Should the cluster headline be generated from the source

Table I. Comparative RED (n-gram overlap (recall)) score for Multidocument Headline Generation.

System	Unigrams	Bigrams	Trigrams	4-grams
HH	0.43 (±0.07)	0.16 (±0.06)	0.06 (±0.04)	0.02 (±0.02)
Trans	0.19 (±0.06)	0.02 (±0.02)	0.00 (±0.00)	0.00 (±0.00)
Gen	0.29 (±0.07)	0.07 (±0.04)	0.01 (±0.02)	0.01 (±0.01)
Gen10	0.27 (±0.08)	0.08 (±0.05)	0.03 (±0.03)	0.01 (±0.01)
Gen15	0.32 (±0.07)	0.08 (±0.04)	0.02 (±0.02)	0.01 (±0.01)
HH	overlap among reference translations			
Trans	headline generated from Hindi originals, then translated			
Gen	headline generated from MT output			
Gen10	same as Gen; optimized to achieve an average headlines length of 10			
Gen15	same as Gen; optimized to achieve an average headlines length of 15			

Notes:
- Only the first ten content words in the headlines were considered in the evaluation in order to favor short headlines.
- Confidence intervals (with $\alpha = .95$) were calculated by jackknifing (systematic resampling by selecting 3 out of 4 references for scoring).

language and then translated, or is it better to generate the headline from the document translations?

In order to answer this question, we compared the performance of both approaches with respect to the RED (Recall-based Evaluation for DUC) score. RED, introduced by Lin and Hovy [2003] (albeit not yet under the name RED), is a measure of n-gram recall between candidate summaries (or headlines) and a set of reference summaries / headlines. The data in Tab. I indicate that generating headlines from translations is significantly better than translating headlines generated from the original documents. Two factors may contribute to this phenomenon: First, translation of whole documents increases the sample size and therefore the chances of the translation engine "hitting the right words"; the impact of mistranslation of generated headlines is stronger than of occasional mistranslations in large amounts of text. Secondly, the translation engine was designed to translate whole sentences, not phrases.

5. MACHINE TRANSLATION

Obviously, machine translation is the key to the system's crosslingual capabilities. The Surprise Language experiment was, among other things, also a test of the promise of statistical machine translation to allow the rapid development of robust MT systems for new languages.

Statistical MT systems use statistical statistical models of translation relations to assess the likelyhood of a, say, English string being the translation of some foreign input. Three factors determine the quality of a statistical machine translation system: (1) the quality of the model; (2) the accuracy of parameter estimation (training); and (3) the quality of the search.

Our statistical translation model is based on the alignment template approach [Och et al. 1999] embedded in a log-linear translation model [Och and Ney 2002] that uses discriminative training with the BLEU score [Papineni et al. 2001] as objective function [Och

2003]. In the alignment template translation model, a sentence is translated by segmenting the input sentence into phrases, translating these phrases, and reordering the translations in the target language. A major difference of this approach to the often used single-word based translation models of Brown et al. [1993] is that local word context is explicitly taken into account in the translation model.

We use a dynamic programming beam-search algorithm to explore a subset of all possible translations [Och et al. 1999] and extract n-best candidate translations using A* search [Ueffing et al. 2002]. These n-best candidate translations are the basis for discriminative training of the model parameters with respect to translation quality.

More details on this system can be found in Oard and Och [2003].

During translation, word reorderings operations are the most time-consuming. At the same time, their payoff is often low [Germann 2003]. Since we needed to translate entire document collections for information retrieval, we performed these translations with *monotone* decoding, that is, while word reorderings were possible locally within the scope of the alignment templates, entire templates were not reordered. This decision was based on two considerations:

(1) Word order is not important for information retrieval.

(2) A more thorough search was impractical with respect to the computing resources required for high-quality, high-volume translations.

Even though we did not implement it, it would be conceivable to use TCP-socket based MT, which our system provides, to provide high(er)-quality translations of selected documents on demand from within the C*ST*RD interface. We have not evaluated to what degree human users would benefit from slightly better translations, and whether the inevitable delays are accepted by the human user in an interactive environment.

6. CONCLUSION

During the 2003 surprise language experiment, we build an integrated end-to-end system for advanced, state-of-the-art information access to information in Hindi for speakers of English. Anecdotal reports and a subjective, cursory evaluation of the tool indicates that it is indeed good enough to at least identify documents of high relevance, and to do so very efficiently.

We were able to accomlish this by relying on modules that employ linguisticly shallow techniques such as vector space models, term frequencies, etc. This shallow approach grants us a certain language-independence.

REFERENCES

ALLAN, J., CALLAN, J., CROFT, W. B., BALLESTEROS, L., BYRD, D., SWAN, R., AND XU, J. 1998. Inquery does battle with TREC-6. In *Sixth Text REtrieval Conference (TREC-6)*. Gaithersburg, Maryland, USA, 169–206.

ALLAN, J., LEUSKI, A., SWAN, R., AND BYRD, D. 2000. Evaluating combinations of ranked lists and visualizations of inter-document similarity. *Information Processing and Management (IPM) 37*, 435–458.

BIKEL, D. M., MILLER, S., SCHWARTZ, R., AND WEISCHEDEL, R. 1997. Nymble: a high-performance learning name-finder. In *Proceedings of ANLP-97*. 194–201.

BROWN, P. F., DELLA PIETRA, S. A., DELLA PIETRA, V. J., AND MERCER, R. L. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics 19*, 2, 263–311.

CHALMERS, M. AND CHITSON, P. 1992. Bead: Explorations in information visualization. In *Proceedings of ACM SIGIR*. Copenhagen, Denmark, 330–337.

CROFT, W. B. 1978. Organising and searching large files of documents. Ph.D. thesis, University of Cambridge.

CUTTING, D. R., KARGER, D. R., AND PEDERSEN, J. O. 1993. Constant interaction-time Scatter/Gather browsing of very large document collections. In *Proceedings of ACM SIGIR*. 126–134.

CUTTING, D. R., PEDERSEN, J. O., KARGER, D. R., AND TUKEY, J. W. 1992. Scatter/Gather: A cluster-based approach to browsing large document collections. In *Proceedings of ACM SIGIR*. Copenhagen, Denmark, 318–329.

DUBIN, D. 1995. Document analysis for visualization. In *Proceedings of ACM SIGIR*. Seattle, Washington, USA, 199–204.

DUNNING, T. E. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics 19*, 1, 61–74.

EDMUNDSON, H. P. 1969. New methods in automatic extraction. *Journal of the ACM 16*, 2, 264–285.

FRUCHTERMAN, T. M. J. AND REINGOLD, E. M. 1991. Graph drawing by force-directed placement. *Software–Practice and Experience 21*, 11, 1129–1164.

GERMANN, U. 2001. Building a statistical machine translation system from scratch: How much bang for the buck canwe expect? In *ACL 2001 Workshop on Data-Driven Machine Translation*. Toulouse.

GERMANN, U. 2003. Greedy decoding for statistical machine translation in almost linear time. In *HLT-NAACL 2003: Main Proceedings*, M. Hearst and M. Ostendorf, Eds. Association for Computational Linguistics, Edmonton, Alberta, Canada, 72–79.

GOLDSTEIN, J., KANTROWITZ, M., MITTAL, V. O., AND CARBONELL, J. G. 1999. Summarizing text documents: Sentence selection and evaluation metrics. In *Research and Development in Information Retrieval*. 121–128.

HEARST, M. A. AND PEDERSEN, J. O. 1996. Reexamining the cluster hypothesis: Scatter/Gather on retrieval results. In *Proceedings of ACM SIGIR*. Zurich, Switzerland, 76–84.

HEMMJE, M., KUNKEL, C., AND WILLET, A. 1994. LyberWorld - a visualization user interface supporting fulltext retrieval. In *Proceedings of ACM SIGIR*. 254–259.

HENDLEY, R. J., DREW, N. S., WOOD, A. M., AND BEALE, R. 1995. Narcissus: Visualising information. In *Proceedings of IEEE Information Visualization*. 90–96.

JIN, R. AND HAUPTMANN, A. 2001. Headline generation using a training corpus. In *Proceedings of the Second International Conference on Intelligent Text Processing and Computational Linguistics (CICLing01)*. Lecture Notes in Computer Science. Springer, Mexico City, Mexico, 208–215.

KOENEMANN, J. AND BELKIN, N. J. 1996. A case for interaction: A study of interactive information retrieval behavior and effectivness. In *Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems*. Vancouver, British Columbia, Canada, 205–212.

LEUSKI, A. 2000. Relevance and reinforcement in interactive browsing. In *Proceedings of Ninth International Conference on Information and Knowledge Management (CIKM'00)*, A. Agah, J. Callan, and E. Rundensteiner, Eds. ACM Press, McLean, Virginia, USA, 119–126.

LEUSKI, A. 2001a. Evaluating document clustering for interactive information retrieval. In *Proceedings of Tenth International Conference on Information and Knowledge Management (CIKM'01)*, H. Paques, L. Liu, and D. Grossman, Eds. ACM Press, Atlanta, Georgia, USA, 41–48.

LEUSKI, A. 2001b. Interactive information organization: Techniques and evaluation. Ph.D. thesis, University of Massachusetts at Amherst.

LEUSKI, A. AND ALLAN, J. 1998. Evaluating a visual navigation system for a digital library. In *Proceedings of the Second European Conference on Research and Advanced Technology for Digital Libraries (ECDL'98)*, C. Nikolaou and C. Stephanidis, Eds. Springer, Heraklion, Crete, Greece, 535–554.

LEUSKI, A. AND ALLAN, J. 2000. Strategy-based interactive cluster visualization for information retrieval. *International Journal on Digital Libraries (IJODL) 3*, 2, 170–184.

LEUSKI, A. AND ALLAN, J. 2003. Interactive information retrieval using clustering and spatial proximity. *User Modeling and User Adapted Interaction (UMUAI)*. In Press.

LEUSKI, A. AND CROFT, W. B. 1996. An evaluation of techniques for clustering search results. Tech. Rep. IR-76, Department of Computer Science, University of Massachusetts, Amherst.

LIN, C.-Y. AND HOVY, E. 1997. Identifying topics by positino. In *Proceedings of the 5th Conference on Applied Natural Language Processing*. Washington, D.C.

LIN, C.-Y. AND HOVY, E. 2002. From single to multi-document summarization: a prototype system and it evaluation. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL-02)*. Philadelphia, PA, USA.

LIN, C.-Y. AND HOVY, E. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *HLT-NAACL 2003: Main Proceedings*, M. Hearst and M. Ostendorf, Eds. Association for Computational Linguistics, Edmonton, Alberta, Canada, 150–157.

LIN, X., SOERGEL, D., AND MARCHIONINI, G. 1991. A self-organizing semantic map for information retrieval. In *Proceedings of ACM SIGIR*. Chicago, 262–269.

Lucene. Lucene search engine. `http://jakarta.apache.org/lucene/`.

MCKEOWN, K. R., BARZILAY, R., EVANS, D., HATZIVASSILOGLOU, V., SCHIFFMAN, B., AND TEUFEL, S. 2001. Columbia multi-document summarization: Approach and evaluation. In *Proceedings of the Workshop on Text Summarization, ACM SIGIR Conference 2001*. DARPA/NIST, Document Understanding Conference.

MIRKIN, B. 1996. *Mathematical Classification and Clustering*. Kluwer.

NorthernLight. Northern light search engine. `http://www.northernlight.com/`.

OARD, D. W. AND OCH, F. J. 2003. Rapid-response machine translation for unexpected languages. In *Proceedings of the MT Summit IX*. New Orleans, LA.

OCH, F. J. 2003. Minimum error rate training in statistical machine translation. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*. Sapporo, Japan.

OCH, F. J. AND NEY, H. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*. Philadelphia, PA.

OCH, F. J., TILLMANN, C., AND NEY, H. 1999. Improved alignment models for statistical machine translation. In *Proc. of the Joint SIGDAT Conf. on Empirical Methods in Natural Language Processing and Very Large Corpora*. University of Maryland, College Park, MD, 20–28.

OVER, P. 2001. Introduction to duc-2001: an intrinsic evaluation of generic news text summarization systems. In *Proceedings of the Workshop on Text Summarization, ACM SIGIR Conference 2001*. DARPA/NIST, Document Understanding Conference.

PAPINENI, K. A., ROUKOS, S., WARD, T., AND ZHU, W.-J. 2001. Bleu: a method for automatic evaluation of machine translation. Tech. Rep. RC22176 (W0109-022), IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY. Sept.

PORTER, M. F. 1980. An algorithm for suffix stripping. *Program 14*, 3, 130–137.

ROBERTSON, S. E., WALKER, S., JONES, S., HANCOCK-BEAULIEU, M. M., AND GATFORD, M. 1995. Okapi at TREC-3. In *Third Text REtrieval Conference (TREC-3)*, D. Harman and E. Voorhees, Eds. NIST, Gaithersburg, Maryland, USA.

RODDEN, K., BASALAJ, W., SINCLAIR, D., AND WOOD, K. 1999. Evaluating a visualisation of image similarity as a tool for image browsing. In *Proceedings of IEEE Information Visualization*. 36–43.

RUSHALL, D. AND ILGEN, M. D. 1996. DEPICT: Documents evaluated as PICTures: Visualizing information using context vectors and self organizing maps. In *Proceedings of IEEE Information Visualization*. 100–107.

SALTON, G. 1989. *Automatic Text Processing*. Addison-Wesley.

SONG, M. 1998. Bibliomapper: A cluster-based information visualization technique. In *Proceedings of IEEE Information Visualization*. 130–136.

SWAN, R. AND ALLAN, J. 1998. Aspect windows, 3-d visualizations, and indirect comparisons of information retrieval systems. In *Proceedings of ACM SIGIR*. Melbourne, Australia, 173–181.

UEFFING, N., OCH, F. J., AND NEY, H. 2002. Generation of word graphs in statistical machine translation. In *Proc. Conference on Empirical Methods for Natural Language Processing*. Philadelphia, PE, 156–163.

VAN RIJSBERGEN, C. J. 1979. *Information Retrieval*. Butterworths, London. Second edition.

WILLETT, P. 1988. Recent trends in hierarchic document clustering: a critical review. *Information Processing and Management 24*, 5, 577–597.

WISE, J. A., THOMAS, J. J., PENNOCK, K., LANTRIP, D., POTTIER, M., AND SCHUR, A. 1995. Visualizing the non-visual: Spatial analysis and interaction with information from text documents. In *Proceedings of IEEE Information Visualization*. 51–58.

ZAJIC, D., DORR, B., AND SCHWARTZ, R. 2002. Automatic headline generation for newspaper stories. In *Proceedings of the ACL-02 Workshop on Text Summarization*. Philadelphia, PA.

ZAMIR, O. AND ETZIONI, O. 1998. Web document clustering: a feasibility demonstration. In *Proceedings of ACM SIGIR*. Melbourne, Australia, 46–54.

ZHOU, L. AND HOVY, E. 2003. Headline summarization at ISI. In *Document Understanding Conference (DUC-03)*. Edmonton, AB, Canada.