WebInEssence: A Personalized Web-Based Multi-Document Summarization and Recommendation System

Dragomir R. Radev†‡ and **Weiguo Fan§** and **Zhu Zhang†**
†School of Information
‡Department of Electrical Engineering and Computer Science
§Business School
University of Michigan
Ann Arbor, MI 48103
radev,wfan,zhuzhang@umich.edu

Abstract

In this paper, we present our recent work on the development of a scalable personalized web-based multi-document summarization and recommendation system: WebInEssence. WebInEssence is designed to help end users effectively search for useful information and automatically summarize selected documents based on the users' personal profiles. We address some of the design issues to improve the scalability and readability of our multi-document summarizer included in WebInEssence. Some evaluation results with different configurations are also presented.

1 Introduction

These days, people are overwhelmed by the large amount of information on the Web. The information overload problem is worsening at an unprecedented speed. Take the size of the Internet as an example. The number of web pages on the Internet was 320 million pages in Dec. 1997 as reported by (Lawrence and Giles, 1998), 800 million in Feb. 1999 (Lawrence and Giles, 1999), and more than 1.2 billion in March, 2000 (Censorware, 2000). The number of pages available on the Internet almost doubles every year. Similar observations can be made on non-digital media. A recent study (Lyman et al., 2000) showed that the world's annual output of content is roughly 1.5 million terabytes.

To help alleviate the information overload problem and help users find the information they need, many search engines have emerged (e.g. Google, Fast, AltaVista, etc.). These search engines commonly build a very large centralized database to index a portion of the Internet: ranging from 50 million to more than 1.2 billion web pages[1]. Search engines do help reduce the information overload problem by allowing a user to do a centralized search, but they also bring up another problem: too many web pages are returned for a single query. To find out which documents are useful, users often have to sift through hundreds of pages to find out that only a few of them are relevant. Research results have shown that search engine users often give up their search in the first try, examining no more than 10 documents, which is what most search engines output on the first page (Jansen et al. 2000). We will introduce an effective search engine to summarize clusters of related web pages which provide more contextual and summary information to help users explore the retrieval result more efficiently.

We describe in this paper a system, WebInEssence, which blends the traditional information retrieval technology with advanced document clustering, document recommendation, and multi-document summarization technology in an integrated framework.

Text summarization is the process of selecting the most salient information in one or more textual documents. When the input consists of more than one document, we talk about multi-document summarization. Summarization relies on the principles of text redundancy and unevenly distributed information content to extract highly informative snippets (often keywords or sentences) from a document which can be read by the user *in lieu* of the original document(s). In WebInEssence, we have adopted an approach to summarization of multiple documents called *centroid-based summarization* (Radev et al., 2000). Centroid-based summarization of multi-document clusters puts special emphasis on portions of these documents that are most central to the topic of the entire cluster. WebInEssence. Its goal is to provide the user with the essence or gist of what is on the Web without overwhelming them with unnecessary detail. NewsInEssence combines multiple strategies to deal with the information overload: search, summarization, clustering, and recommendation. We will describe these in the rest of the paper.

1.1 Design Goals

The goal of the system is to provide users every opportunity to reduce the information overload and do simplified yet effective search and navigation. The system is designed to possess the following features:

- Easy to use and control

[1] According to http://www.searchenginewatch.com

- Easy to customize and personalize
- High quality search
- Scalable to handle multiple users

In this paper we will discuss how we achieve the above goals. We also discuss in detail the different components of our system, presented briefly in the following Table.

WebInEssence is highly flexible and can be used in at least four major modes of operation, as shown below.

Generic Search

The system can be used like other search engines to support generic search. After the user enters a query, the system will return a hit list of documents that match the query.

Generic Search + Summarization

The system can also enable users to hand select the documents (URLs) that they are interested in and perform single-document or multi-document summarization. This feature allows users to quickly appreciate the contents without the further endeavor to browse through each document.

Generic Search + Clustering (+ Recommendation) + Summarization

Besides mode 2, this system also gives a user more flexibility to navigate the hit list. If the user feels that a hit list for a given query is too long and he/she is interested in seeing only a few of them more interesting to him/her, the user can request the system to perform a clustering operation on the hit list. The number of items on the hit list will be grouped according to its coherent semantics and the user should be able to quickly identify the interesting clusters he/she may want to look at in more detail. In addition, WebInEssence produces cluster-based summaries of all clusters, and uses the contents of each of the clusters to recommend additional documents to the user.

Personalized Mode

The above three modes give users ample control on the search and browse process. However, users still need to change the parameters every time for different search sessions. The personalized mode, besides having all the features of the above three modes, enables users to do search, clustering, and summarization just one click away. All of the parameters for the three processes have been stored in a user's personal profile. Once the user logs in to the system, he/she is right ready to go.

2 Related Work

Our work is related to researches on clustering and summarizing of web search results. We briefly describe some of the related works in this section. Table 2 summarizes the overall features of their systems. The list is by no means exhaustive, although it highlights the strengths of WebInEssence.

Weiss et al. presented their research on a framework for the design of a hierarchical network search engine that exploits content-link hypertext clustering (Weiss et al., 1996). Their engine can cluster hypertext based on the semantic information embedded in hyperlink structures and document contents and group these hypertexts into hierarchies. Their engine also provides an abstraction function that can summarize cluster contents to support scalable query processing. However, the summaries produced by their system are simply in the form of salient keywords/terms for a cluster with the goal of reducing the network engine storage space, not a set of natural language sentences as we have in our system.

Zamir described his experience in developing a dynamic clustering interface (called Grouper) to web search results (Zamir, 1999), which is essentially a web-based text categorization system. His work shares similar idea with the recent meta-search engine Vivísimo, although may differ in the implementation details. He utilized a fast clustering algorithm called STC to dynamically cluster post retrieval results from a meta-search engine into clusters labeled by phrases extracted from the search result. The evaluation based on the query log demonstrated that there is a substantial difference in the number of documents followed and in the amount of time and effort expended by users accessing search results between the grouper clustered interface and traditional unclustered interface. No summarization is provided, except some key phrases are provided for the display of clusters. Our system supports both pre-computed clusters and post retrieval clustering.

Chen, Hearst et al. described a web-based search system called "Cha-Cha" for an Intranet environment (Chen et al., 1999) . It organizes web search results in hierarchies so as to reflect the underlying structure of the Intranet. An "outline" or "table of contents" is created by first recording the shortest paths in hyperlinks from root pages to every page within the web intranet. After the user issues a query, these shortest paths are dynamically combined to form a hierarchical outline of the context in which the search results occur. Cha-Cha can also provide the page summary for each URL. The summary is basically a keyword-in-context (KWIC) summary, which consists of a sentence from the beginning of the document and three sentences containing the query terms. KWIC has been shown to be very helpful to provide the contextual informa-

Components	Functions
Search engine	Search the indexed collection of the web for a given query and return a hit list
Summarization engine	Perform single-document and multi-document summarization
Cluster engine	Cluster the document collection into groups around a central theme
Recommendation engine	Suggest documents from the same clusters in which the documents returned in the hit list are located
Web interface	Provide the interaction mechanism between the system with the user

Table 1: System components and their functions

System	Clustering of documents		Summarization		Personalization	Recommendation of documents
	Before retrieval	After retrieval	Single-document	Multi-document		
eSseNSe (our work)	Yes	Yes	Yes	Yes	Yes	Yes
Hypersuit (Weiss et al. 1996)	Yes	No	Keywords	No	No	No
Grouper (Zamir 1999)	No	Yes	Keywords	No	No	No
Vivísimo[3]	No	Yes	Yes	No	No	No
Cha-Cha (Chen et al. 1999)	No	No	Yes	No	No	No
InCommonSense (Amitay, 2000)	No	No	Yes	No	No	No
Neto (Neto et al. 2000)	Yes	No	Yes	No	No	No
Extractor[4]	No	No	Yes	No	No	No

Table 2: System feature comparison

tion. No clustering is done in the system. Multi-document summarization is also not supported by their system.

Amitay comes up with a different idea in terms of presenting the search results. Instead of using sentences or text from the original document as the summary, he uses the human annotations collected from the web as the summary for a particular URL. In other words, for a given URL in the search hit list, the links which point to this URL are downloaded and analyzed following some patterns and rules that have been identified a prior. Some of the human annotations that related to this URL are extracted and filtered. The best one is selected and used as the summary for this URL. A prototype system called InCommonSense is implemented to illustrate his idea (Amitay, 2000). The summary generation process is very unique. But it suffered from the fact that many URLs may not have any human annotations. No clustering is done in his prototype system.

Neto et al. describes a text mining tool that performs document clustering and text summarization. They used the Autoclass algorithm to perform document clustering and used TF*ISF (an adaptation of TF*IDF) to perform sentence ranking and generate the summarization output (Neto et al., 2000). Our work is different from theirs in that we perform personalized summarization based on the retrieval result from a generic personalized web-based search

engine. A more complicated sentence ranking function is employed to boost the ranking performance. The compression ratio for the summary is customizable by a user. Both single-document summarization for a single URL and multiple-document summarization for a cluster of URLs are supported in our system.

More related work can be found on the Extractor web site[2]. They use MetaCrawler to perform web-based search and automatically generate summaries for each URL retrieved. They only support single document summarization in their engine and the compression rate of the summarizer is also non-customizable. We not only support both single and multiple document summarization, but also allow the user to specify the summarization compression ratio as well as to get per-cluster summaries of automatically generated clusters, which, we believe, are more valuable to online users and give them more flexibility and control of the summarization results.

3 WebInEssence System Architecture

3.1 Overall architecture

The overall structure of WebInEssence can be seen in Figure 1. The two main stages of operation are shown on the left and right hand side of the figure, respectively. During the offline stage, the system behaves as a Web-based spider that collects URLs from the Web. These URLs are represented by hollow circles. The clustering component groups these URLs into clusters of lexically related documents (Radev et al., 1999). From each of the clusters, a query-independent multi-document summary in the form of a sentence extract is produced using the MEAD centroid-based algorithm (Radev et al., 2000). During the online stage, a user query produces a hit list consisting of documents related to the query. In a dynamic fashion, all pre-computed clusters in which hits (solid circles) appear are extracted and shown to the user along with the rest of the documents from these clusters. In addition to these documents, WebInEssence displays the summaries of each of the pre-computed clusters plus a dynamically-produced summary of the documents selected by the user from the hit list. The shaded square boxes indicate which pre-computed clusters contain hits from the search engine's results. All URLs in these clusters that are not returned by the search engine are shown to the user as recommendations.

3.2 Search engine

The search component of WebInEssence is a personalized search engine called MySearch. MySearch utilizes a centralized relational database to store all the URL indexes and other related URL information. Spiders are used to fetch URLs from the Internet. Currently, we query multiple search engines such as Google, AltaVista, and Yahoo for URLs related to "electronic commerce", "data mining", "text mining", "search engine", "online banking" and other internet related keywords. As a result, we have collected, indexed, and clustered 22,988 URLs.

The contents of URLs are indexed based on the locations of the keywords: Anchor, Title, and Body. This allows weighted retrieval based on different word positions. For example, a user can specify that he'd like to give a weight 5 for the keyword appearing in the title, 4 for anchor, and 2 for body. This information can be saved in his personal profile and used for later weighted ranking.

Besides the weighted search, MySearch also supports Boolean search and Vector Space search. For the vector space model, TF*IDF is used for ranking purpose. We used a modified version of TF*IDF: $\log(tf+0.5)*\log(N/df)$, where tf means the number of times a term appeared in the content of a URL, N is the total number of documents in the text collection, and df stands for the number of unique URLs in which a term appears in the entire collection.

A user can choose which search method he wants to use. He/she can also combine Boolean search with Vector Space search. These options are provided to give users more flexibility to control the retrieval results as past research indicated that different ranking functions give different performances (Salton, 1989).

3.3 Clustering

WebInEssence uses two types of clustered input – either the set of hits that the user has selected or the output of our own clustering engine – CIDR (Radev et al., 1999). It uses an iterative algorithm that creates as a side product the so-called "document centroids". The centroids contain the most highly relevant words to the entire cluster. We use these words to find the most salient "themes" in the cluster of documents.

3.4 Finding themes within clusters

One of the underlying assumptions behind WebInEssence is that when a user selects a set of hits after reading the single-document summaries from the hit list retrieved by the system, he or she performs a cognitive activity whereby he or she picks documents which appear to be related to one or more common themes. The multi-document summarization algorithm attempts to identify these themes and to identify the most salient passages from the selected documents using a cluster centroid which is computed automatically from the entire list of hits selected by the user.

[2]http://extractor.iit.nrc.ca/

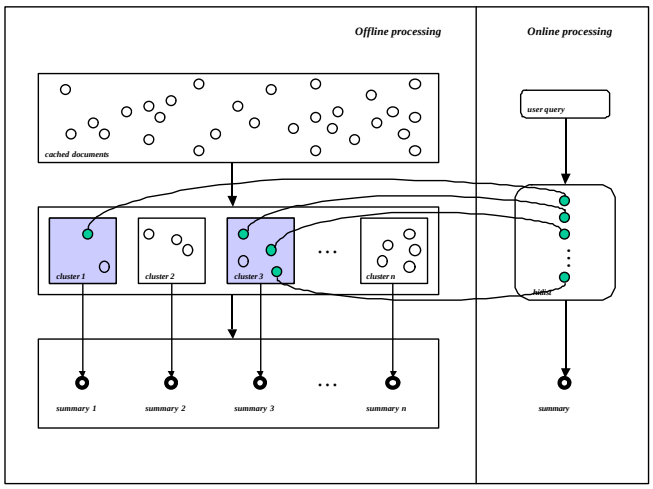

Figure 1: Overall WebInEssence architecture

Figure 2 shows the top 6 clusters (and their centroids), associated with the query "intelligent & agents". Each word in a centroid is associated with a salience score. As an illustration, the salience scores for Cluster 00766 are as follows: user (140.53), interface (123.89), users (52.71), ariadne (51.41), hypertext (36.19), search (32.34), computer (28.60), etc.

3.5 Recommendations

The recommendation heuristic that we use is based on the observation that certain documents that are related to the user's query may not be ranked high by the search engine for lack of appropriate keywords. Instead, such hits may be lexically related to the documents retrieved by the search engine. As a result, for each of the documents retrieved on a given page by the search engine, we display all other documents that are included in the same pre-computed cluster as the hits themselves. The following table illustrates this component of the system. To illustrate the recommendation process, suppose that the user's query is "Linguistics". Some important documents about Linguistics such as http://www.acm.org/sigir and http://www.departments.bucknell.edu/russian/language don't contain the word "linguistics" but, being in the same cluster as documents containing these keywords, are recommended as relevant.

If we use "Linguistics" as the query, and try out different values for the number of hits returned, we notice that the number of clusters increases slowly with the size of the hit list. For example, when the top 40 hits from the search engine are considered, the recommendation system extracts 9 of the pre-computed clusters. These clusters contain a total of 323 documents, of which 40 are already in the hit list. The remaining 283 are sorted and suggested to the user. Clusters which contain a larger number of documents that are also on the current hit list are shown first.

Number of hits	Number of clusters	Number of URLs in clusters
10	3	180
20	5	194
30	6	197
40	9	323
50	11	332
60	13	335
70	15	640
80	18	779
90	22	1092
100	23	1094
...	...	...
397 (all hits)	87	3841

Table 3: Recommended URLs as a function of hit list size

3.6 Summarization

The main technique that we use for summarization is centroid-based sentence extraction. We score individually each sentence within a cluster and output those that score the highest. A more detailed description of the summarizer can be found in (Radev et al., 2000). The input to the summarization component is a cluster of documents. These documents can be either the result of a user query or the output of CIDR or a pre-computed cluster.

The summarizer takes as input a cluster of d documents with a total of n sentences as well as a compression ratio parameter r which indicates how much of the original cluster to preserve. The output consists of a sequence of $[n * r]$ sentences from the orig-

Cluster ID	Number of URLs	Centroid words
00085	63	information retrieval systems university workshop papers research library science submission chair text edu conference computer applications language processing data libraries
00044	167	web internet site design hosting search online sites commerce page business meta your information content server marketing you electronic pages
00086	115	retrieval information university ir text research systems science semantic evaluation document pp library
00657	135	vol pp no retrieval proceedings conference user ir information query
00766	6	user interface users ariadne hypertext search computer interfaces we designer system laurel information nelson knowledge collaboration representation interaction process systems
00127	4	neural systems computational data evolutionary intelligent networks learning artificial knowledge intelligence

Figure 2: Sample centroids

inal documents in the same order as the input documents. The highest-ranking sentences are included according to the scoring formula below:

$$S_i = w_c C_i + w_p P_i + w_f F_i$$

In the formula, w_c, w_p, w_f are weights. C_i is the centroid score of the sentence, P_i is the positional score of the sentence, and F_i is the score of the sentence according to the overlap with the first sentence of the document.

3.7 Centroid value

The centroid value C_i for sentence S_i is computed as the sum of the centroid values C_w of all words in the sentence i. For example, the phrase "hypertext user interface" gets a score of 300.61 which is the sum of the individual centroid values of the words (hypertext = 36.19, user = 140.53, and interface = 123.89).

$$C_i = \sum_w c_w$$

3.8 Positional value

The positional value is computed as follows: the first sentence in a document gets the same score C_{max} as the highest-ranking sentence in the document according to the centroid value. The score for all sentences within a document is computed according to the following formula:

$$P_i = \frac{(n-i+1)}{n} * \max_i(C_i)$$

3.8.1 First-sentence overlap

The overlap value is computed as the inner product of the sentence vectors for the current sentence i and the first sentence of the document. The sentence vectors are the n-dimensional representations of the words in each sentence whereby the value at position i of a sentence vector indicates the number of occurrences of that word in the sentence.

$$F_i = \vec{S}_1 \vec{S}_i$$

3.8.2 Combining the three parameters

A discussion of the ways to combine the different parameters is included in (Radev and Fan, 2000).

4 User Interface

We describe in this section the interface design issues. A sample search query "intelligent & agents" will be used to illustrate the overall search, clustering and summarization process.

The search process begins with the user's formulation of information need (request) as a query. The search result is displayed in order of decreasing relevance to the original query. Since the search result interface means to provide a user with more contextual information about the relevancy information for the hit list, the following information is provided by WebInEssence to accomplish this:

- The number of URLs found to be potentially relevant
- The number of URLs shown on each page
- The relevance score for each URL in the hit list
- The contextual information (keywords, abstract, summary) for each URL
- The choice of different ranking criteria

A sample user query and search result is shown in Figure 3.

After the user gets back the search hit list, he/she may find some of the URLs very interesting and would like to know further about the central theme for these URLs. The user could click on each of the URLs and go through them individually. But this

process will be very time-consuming when the number of interesting URLs is large. For the above situation, the summarization engine comes into play. After a user selects the documents that are of interest to him/her, he/she can submit his/her summarization request, the URLs selected by a user for summarization will be highlighted in a separate table. Meanwhile, our system also provides more control options for users to choose in terms of the summarization process:

- Summary length: the percentage of the size of the original URLs.
- Suppress repeated information: sentences from the same or different documents that are too close lexically are shown only once.
- Different order scheme: the sentences used for the produce of summary can be ordered by different schemes, such as, position, time sequence, relevance to the query, etc.

The final summary generated by the summarization system preserves the contextual information, e.g., the URLs for which the summary is generated for. Similarly as in the search process, it should also give the user more detailed information about the summarization process. Such information includes:

- How many sentences are contained in the URLs selected
- The summary length
- The relevance or ordering score for each sentence

Summarization is very helpful for users to absorb large quantities information. But it may still contain too much information (too many URLs) or few relevant URLs (they expect more similar relevant URLs). Take the "intelligent & agents" query as an example. There are total 369 results returned. It's very tedious for a user to navigate page by page to go through each of the URLs and select those for summarization. Instead, he/she may like to group these URLs into disjoint groups according to their semantics. This could not only reduce the information navigation space, but also bring up more semantically related URLs. The clustering engine is deigned to address this issue. Two options, for the moment, are provided for the user to control the clustering process:

- Cluster only the current page
- Cluster the entire search result

The first option is designed for those users who have time constraints to go through the entire search result but still want more relevant URLs similar to ones in the current page. The number of clusters returned for this option is normally very low, varying from 1 to 4 clusters for a result set of 10 URLs. For the rest of the users, option 2 is provided to dramatically reduce the navigation space for the entire search result. For example, the total number of clusters generated for those 369 URLs for the query "intelligent & agents" is only 73, much smaller than the total number of 369.

To help users quickly identify clusters that are more relevant to them, various contextual information is provided for each cluster as shown in Figure 4. These information include:

- **Keyword**s: the keywords are extracted from the centroid of each cluster as described in Section 3.4. These are the first-tier context information to help judge the relevance.
- **Summary**: multi-document summarization is automatically applied to each cluster to generate the summary. These are the second-tier context information for relevance judgment. Note that the number of the sentences for each summary may be large, only the top three sentences are displayed for each cluster. The user can click on "full summary" to view the full contents of the summary.
- **URLs within a cluster**: the top 5 URLs from each cluster is shown for the user. Some of them are also shown in the original hit list, these URLs are marked as "original from search hit list". Others that are not in the original hit list are marked as "new from cluster". These new URLs serve as recommendations from the cluster engine and may help users to find more relevant information from within a cluster. The user can click on "more from this cluster" to examine the full list of the URLs within a cluster.
- **Cluster score**
  In order to help users quickly glimpse the size of each cluster, we also provide a cluster score (third-tier context information) for each cluster. This is defined as

$$\frac{\text{The number of URLs in a cluster that are also in the original search hit list}}{\text{The total number of URLs in a cluster}}$$

  For example, a cluster score "19/63 = 0.30" means that there are total 63 URLs in this cluster, among which 19 are also in the original search hit list. That is 30 percent of the URLs in the cluster are in the original hit list. This may help users to decide which cluster is more worthwhile to examine further.

One of the unique features of WebInEssence is in the personalization. Most of the control parameters

can be stored in a user's personal profile and are reusable throughout the search and summarization process.

5 Discussion

5.1 Statistics on the different components of the system

From the entire collection of 22,998 documents, we extracted 6,113 clusters ranging from 1 to 676 documents. For each cluster, we computed a 5% multi-document summary. These summaries range from 1 to 13 sentences in length.

5.2 Sample summary produced by the system

To illustrate the types of summaries produced by WebInEssence, we enclose the first four sentences of the summary for cluster 00018. The sentences in the summary are sorted according to the order in which they appear in the input documents, that is, they are not sorted by decreasing score. Cluster 00018 contains 122 documents and its centroid contains the following words: mining (84.54), data (64.13), knowledge (14.25), discovery (11.98), advertised (11.20), databases (9.69), information (6.98), research (6.96), analysis (6.95), text (6.05), patterns (5.30), algorithms (4.39).

5.3 Scalability

To improve the scalability of the search engine, we have taken several measures to make it more scalable to handle realistic multiple user access.

- Caching documents

  All documents retrieved and indexed by WebInEssence are cached locally and the local copies are used in the summarization process unless these documents have changed on the Web.

- Multithreading

  WebInEssence employs a fast multi-threading web server to handle multiple user simultaneous access. We monitored the memory consumption of server thread and optimized the total number of server threading processes based on the current constraints of our hardware. Depends on the system load, and traffic of our website, server replication scheme may also be employed to boost the performance.

- Caching query results

  For each search query submitted to the search engine, the search result is automatically cached. When a new query comes in, we first check the cache for the availability of the search result. This works for our current system status where the document collection is relatively static. New schemes must be designed, like reducing the caching time to 24 hours, to deal with the dynamic document collection.

- Compressed data representation

  We employed compressed data representation similar to the implementation of Google (Brin and Page, 1998) to store the inverted index information. For the 22,988 documents for a total size of 320MB that we downloaded, we used only 70 MB for the inverted indexing. The auxiliary information about the URLs is about 67MB.

- Caching clustering results

  Although we give the user an option to cluster selected URLs on the fly, we pre-compute all clusters from the entire collection of indexed URLs. As a result, if the user chooses one of the options "cluster all hits" or "cluster the current page", we can also provide the subsets of the precomputed clusters that correspond to the current hit list.

- Caching summaries

  All summaries of the pre-computed clusters are similarly pre-computed and stored. They are retrieved as cluster summaries after the retrieval stage.

5.4 Improving Readability

Several measures were taken to improve the readability of the search result:

- Keyword in context

  The keyword used for the query is automatically highlighted on all the contextual information: URLs, abstract. This will help the user to quickly decide the possibility of the relevancy of each URL.

- Different color schemes for different types of information

  Different colors were used for display of various types of information. For example, the file size is shown in light green, hyperlinks are shown in blue.

5.5 Improving Usability by Personalization

Personalization is employed throughout the system to improve the usability of our system. A personal profile is created for each user only after the user chooses to register. The registered user will have more flexibility to control their entire information seeking process.

For example, a user can specify his own criteria of weighted information retrieval. Recall that in weighted information retrieval, different weights are assigned for different types of tags: <title> <anchor> <body>. A weight of 5 for title, 2 for anchor, 1

Figure 3: Sample user query and resulting hit list

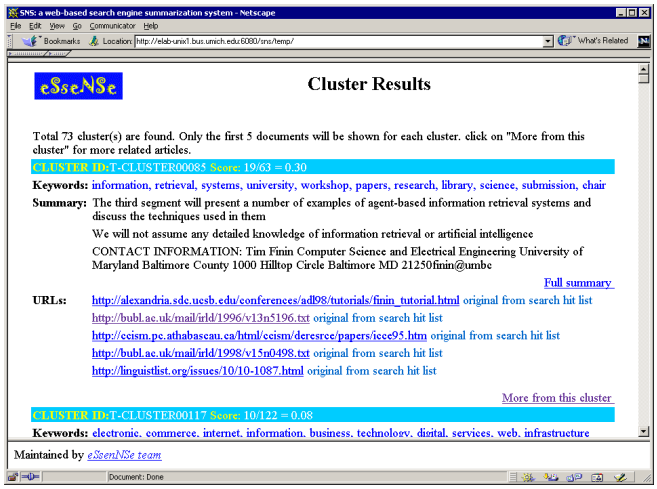

Figure 4: Clustering output

for body means that the keywords appeared in the title is 4 times more important than those in the body. Similarly, the keywords appeared in the anchor is 1 times more important than those in the body. As shown in the previous experiment results section, different ranking function gives totally different search results. The personal profile enables a user to store his/her preference into the profile and enjoy more accurate search results tailored to his/her own need.

6 Future work and Conclusion

6.1 Future work

The main limitations in our system will be addressed in a future release of WebInEssence. These include the personalization of the multi-document summarization component to take into account the user query and the uniformization of the user interfaces used by the different components.

6.2 Conclusion

In this paper, we presented what we believe is the first multi-document summarization and document recommendation system (called WebInEssence) deployed on the Web. We discussed how clusters of related Web pages are produced and summarized as a group. We concluded with an empirical evaluation of our system.

References

E. Amitay. 2000. Incommonsense - rethinking web search results. In *Proceedings of IEEE International Conference on Multimedia and Expo (ICME 2000)*, New York City, NY, USA.

Text Mining is a new and exciting research area that tries to solve the information overload problem by using techniques from data mining machine learning information retrieval natural-language understanding case-based reasoning statistics and knowledge management to help people gain insight into large quantities of semi-structured or unstructured text	183.73
Current Projects - Aims The Data Mining Program has two projects: DMITL - Data Mining in the Large Conducting practical case studies for clients involving the analysis of large and complex data sets ParAlg – Parallel Algorithms The main computing facility for these projects consists of a secure multiprocessor Sun E4000	156.56
The DMITL project aims to develop knowledge and techniques relevant to the data mining of large and complex datasets using high performance computers	163.78
Megaputer develops software and solutions for data mining text analysis and knowledge discovery in databases	188.96

Figure 5: Sample summary produced WebInEssence.

http://www.ics.mq.edu.au/ einat/publications/-ieee_multimedia2000.pdf.

S. Brin and L. Page. 1998. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the WWW7 conference*, Australia. http://www7.scu.edu.au/programme/full-papers/1921/com1921.htm.

Censorware. 2000. http://www.censorware.org/web_size.

M. Chen, M. Hearst, J. Hong, and J. Lin. 1999. Cha-cha: A system for organizing intranet search results. In *Proceedings of the 2nd USENIX Symposium on Internet Technologies and SYSTEMS (USITS)*, Boulder, CO, October 11-14.

S. Lawrence and C. L. Giles. 1998. Searching the world wide web. *Science*, 280(3):98–100.

S. Lawrence and C. L. Giles. 1999. Accessibility of information on the web. *Nature*, 400:107–109.

P. Lyman, H. R. Varian, J. Dunn, A. Strygin, and K. Swearingen. 2000. How much information. http://www.sims.berkeley.edu/how-much-info/.

J. L. Neto, A. D. Santos, C. A. A. Kaestner, and A. A.Freitas. 2000. Document clustering and text summarization. In *Proceedings of 4th Int. Conference on Practical Applications of Knowledge Discovery and Data Mining (PADD-2000)*, pages 41–55. London: The Practical Application Company.

Dragomir Radev and Weiguo Fan. 2000. Automatic summarization of search engine hit lists. In *Proceedings, ACL Workshop on Recent Advances in NLP and IR*, Hong Kong, October.

Dragomir R. Radev, Vasileios Hatzivassiloglou, and Kathleen R. McKeown. 1999. A description of the CIDR system as used for TDT-2. In *DARPA Broadcast News Workshop*, Herndon, VA, February.

Dragomir R. Radev, Hongyan Jing, and Malgorzata Budzikowska. 2000. Summarization of multiple documents: clustering, sentence extraction, and evaluation. In *ANLP/NAACL Workshop on Summarization*, Seattle, WA, April.

R. Weiss, B. Velez, M. A .Sheldon, C. Manprempre, P. Szilagyi, A. Duda, and D. K. Gifford. 1996. Hypursuit: A hierarchical network search engine that exploits content-link hypertext clustering. In *Proceedings of the Seventh ACM Conference on Hypertext*, Washington, DC. http://www.psrg.lcs.mit.edu/publications/-Papers/hypertabs.htm.

O. E. Zamir. 1999. *Clustering Web Documents: A Phrase-Based Method for Grouping Search Engine Results*. Ph.D. thesis, University of Washington, http://www.cs.washington.edu/homes/zamir/.