USING MACHINE LEARNING TO IMPROVE INFORMATION ACCESS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By
Mehran Sahami
September 1998

Chapter 9

SONIA – A Complete System

9.1 Introduction

Having described the major technical components that comprise SONIA in the previous chapters, we can now see how all these pieces fit together to form a complete system. To return to our original motivation for building SONIA, recall that the enormous amount of information available on the World Wide Web and other networked information sources, such as Digital Libraries, has created an urgently pressing need to provide users with tools to navigate these information spaces. The initial attempts at addressing this problem have led to the development of a number of information finding tools such as Web-based search engines (e.g., *Alta Vista*) and hierarchical directory services (e.g., *Yahoo!*). However, as we explained previously (in Chapter 1), such methods for information access are quickly being rendered inadequate due to the tremendous growth in the number of documents available. Also worth noting is the fact that networked information can often come from a number of heterogeneous sources (i.e., the World Wide Web, different Digital Libraries, proprietary databases, etc.), whereas many existing information finding tools are only implemented to work with one information source.

We seek to address these problems with SONIA, a system for *topical* information space navigation that combines both the query-based and taxonomic approaches. SONIA employs the machine learning techniques described earlier (i.e., feature selection, clustering, and hierarchical classification) to create dynamic document categorizations based on the full-text of articles that are retrieved in response to users' queries. In this way, users can explicitly specify their information needs as queries while also having the ability to browse the results of their queries at a topical, rather than document, level. Moreover, the hierarchical organization scheme imposed on a set of documents may be used to classify new documents that become available to the system (e.g., documents returned in response to follow-up queries by the user).

The ability to automatically create meaningful document groupings relies on the validity of the Cluster Hypothesis. Recall that this hypothesis states that "closely associated documents tend to be relevant to the same requests" [165]. As we described in Chapter 1, a good deal of previous work [166, 124, 70, 71, 2] has provided significant support for the cluster hypothesis, as have our more recent results reported in Chapter 5. Thus, we have strong reason to believe that clustering may be used as an effective tool to help organize documents.

While our system embodies some similar elements to those in previous work on document clustering, it uses entirely different, and in many cases improved, technologies to realize this functionality. More importantly, however, SONIA provides significant new extensions in terms of technical functionality and broader applicability. Operating in the dynamic context of networked information, SONIA makes use of a number of methods for relevant feature extraction from documents through a multi-tiered feature selection process that is customized to each user query. Furthermore, since our system exists as part of a general architecture within the Stanford Digital Libraries Testbed [157], it has the ability to simultaneously retrieve information from a number of heterogeneous sources, thereby making our system maximally flexible. SONIA was also designed with efficiency in mind, thereby facilitating real-time user interactivity even when accessing diverse, distributed document collections.

The most significant extension of SONIA beyond existing systems, however, is the ability to save various document clusterings (i.e., topical partitionings) as hierarchical classification schemes that can be used to automatically categorize the results of subsequent, but related, queries (using the hierarchical classification method presented in Chapter 8). This combination of clustering and classification allows users to not only navigate a given document collection more easily, but enables them to quickly construct and maintain their own organizational structures for the vast quantities of information available to them. In this way, we hope to elevate user interaction with large information sources (e.g., the Web, Digital Libraries, etc.) beyond simple one-shot queries and move to addressing users' more persistent information needs. Moreover, we also show that the basic ideas implemented in SONIA are also quite applicable to personal information management, such as helping a user organize the file system on his or her personal computer desktop. In this context, SONIA can be very effective at helping automatically generate and maintain portions of a user's hierarchical file directory structure.

In the remainder of this chapter we present the technical details of SONIA. In Section 9.2, we describe the architecture in which SONIA is embedded and how it interacts with a variety of heterogeneous information sources. There, we also give a brief description of the Stanford Digital Libraries InfoBus Architecture, showing how SONIA is situated within a larger distributed systems context. Section 9.3 provides a more detailed account of how the various machine learning methods employed in SONIA are incorporated together in the system. Then, in Section 9.4, we show anecdotal examples of the system in use, discussing its efficacy in information browsing and classification. Finally, Section 9.5 gives a summary of this work and its future directions.

9.2 SONIA on the InfoBus

The focus of the Stanford Digital Libraries project is on providing interoperability among heterogeneous, distributed information sources, services and interfaces. To this end, the InfoBus architecture [10] shown in Figure 9.1 has been developed. In brief, the InfoBus is comprised of network proxies that encapsulate the protocols used by disparate interfaces, information sources, and information services. These proxies allow for communication among the different entities connected to the InfoBus by translating their communications into a common language.

SONIA exists within this architecture as an information service with a number

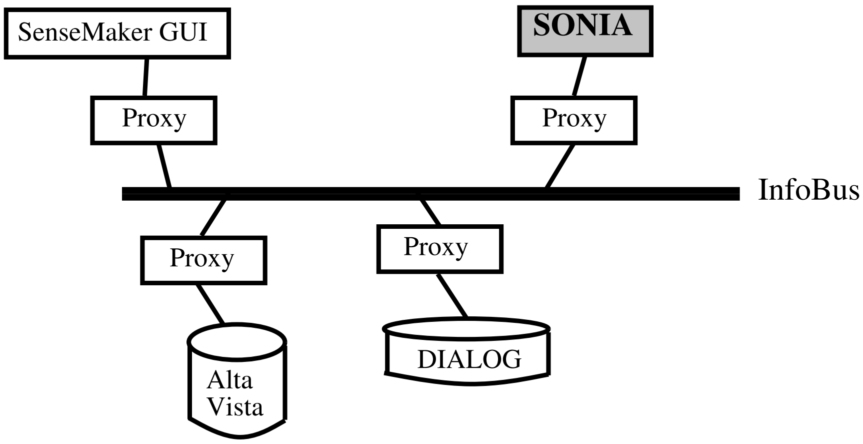

Figure 9.1: The InfoBus architecture.

of capabilities. First, it allows for the clustering of collections of documents to help extract subtopics that may be present. This allows users to quickly home in on sub-collections of documents that satisfy their information needs, and thus ignore much of the irrelevant material often returned by simple queries. Furthermore, SONIA also allows for such document groupings to be stored as persistent hierarchical categorization schemes. Each such hierarchy is simply a multi-level partitioning of documents into a number of semantically meaningful groups. These hierarchies may then be updated by automatically classifying additional documents into them. In this way, new query results can be integrated into an existing topic hierarchy derived from previous query results. This allows the user to build up a large collection of results spanning multiple related queries within the same organizational scheme. Moreover, SONIA allows a single user to save several distinct hierarchies to reflect each of his or her diverse information needs.

Previously, the functionality in SONIA was accessible through the Java-based *SenseMaker* interface [11]. SenseMaker allows users to simultaneously query multiple heterogeneous information sources and then organize the retrieved documents by matching titles, matching URLs (for Web documents), and the like, or it can utilize SONIA to cluster documents by their full-text content [140, 141]. However, since the the focus of the SenseMaker interface was not on the formation of topic hierarchies,

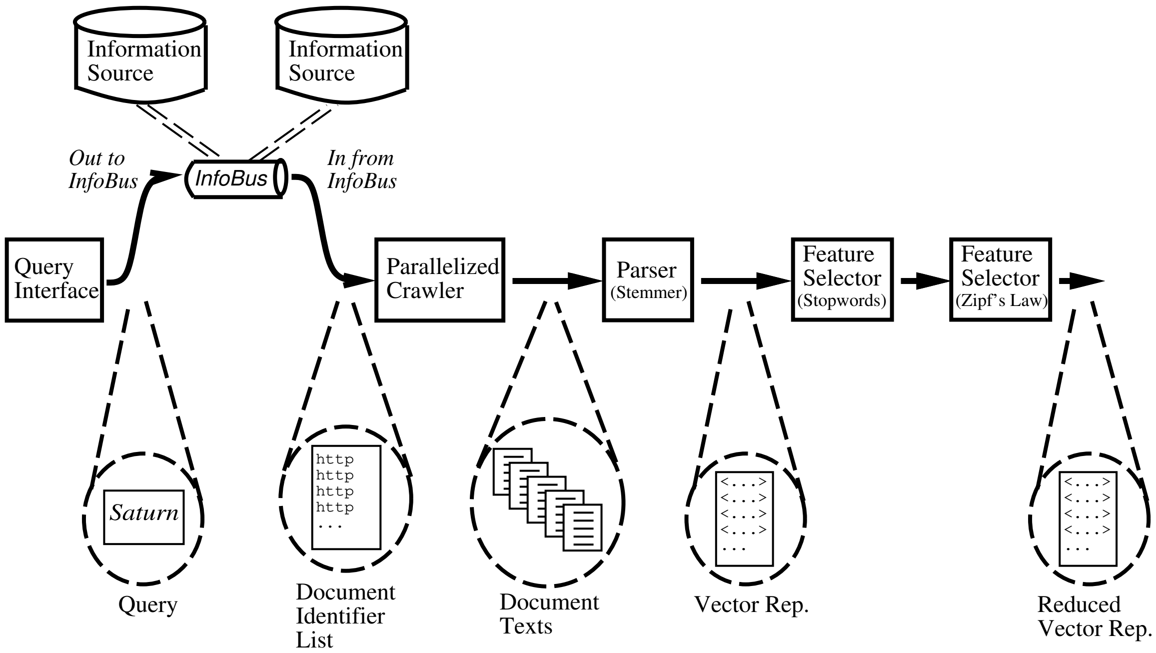

Figure 9.2: Initial querying and document processing stages in the SONIA system.

we have more recently developed our own custom interface for SONIA. In this new interface (which we show examples of in Section 9.4), users can still query a variety of heterogeneous information sources, but can now form hierarchical organization schemes to capture the richer topical structures that exist in many document collections. The InfoBus is still used as the primary means of communicating with these disparate information sources. Thus, SONIA can effectively query virtually any information sources which are, or will be, connected to the InfoBus. To better understand the technologies incorporated within the system, we presently turn our attention to the components that comprise SONIA.

9.3 A Component View SONIA

It is simplest to view SONIA as a series of modules, each of which is responsible for a data transformation procedure. We begin by examining the initial querying and document processing stages in SONIA, which are represented in the block diagram in

Figure 9.2. Here, the user begins by using SONIA to issue a query to various information sources linked to the InfoBus. A list of potentially relevant document identifiers (e.g, URLs for Web documents, ID numbers for DIALOG, etc.) are then returned to the system in response to the query. At this point, it becomes necessary to represent these documents in a manner suitable for processing by subsequent machine learning algorithms (i.e., a vector representation). The processing stages which produce this vector representation very closely follow our treatment of document representation given in Chapter 2. Once a suitable representation has been obtained, the documents are then analyzed by different machine learning tools, depending on whether the user is choosing to organize the documents according to an existing hierarchical classification scheme or not. The machine learning components of SONIA are depicted in Figure 9.3. We explain the details of all of these processing stages, as implemented in SONIA, below.

9.3.1 Document retrieval and parsing

Since on-line information sources are rapidly changing, SONIA does not attempt to maintain its own (possibly outdated) inverted index of documents, but rather treats networked information as a massive digital library from which it can dynamically retrieve documents. As a result, SONIA only requires that it receive a list of document identifiers (and not the actual documents) from the information sources that it queries. For example, this makes it possible to employ SONIA in conjunction with any of the existing well-known search engines on the Web.

A highly parallelized document retrieval module (sometimes called a network *crawler* or *spider*) is employed to retrieve the full text of the corresponding documents. This module does not present a timing bottleneck in real-time interaction as it is capable of robustly retrieving as many as 250 document texts in parallel, and utilizes a time-out condition (currently set at 30 seconds) to prevent needlessly long waits for documents. Moreover, as the infrastructure of the Internet continues to improve, and localized repositories become more prevalent, this document retrieval stage will become even less of a time issue.

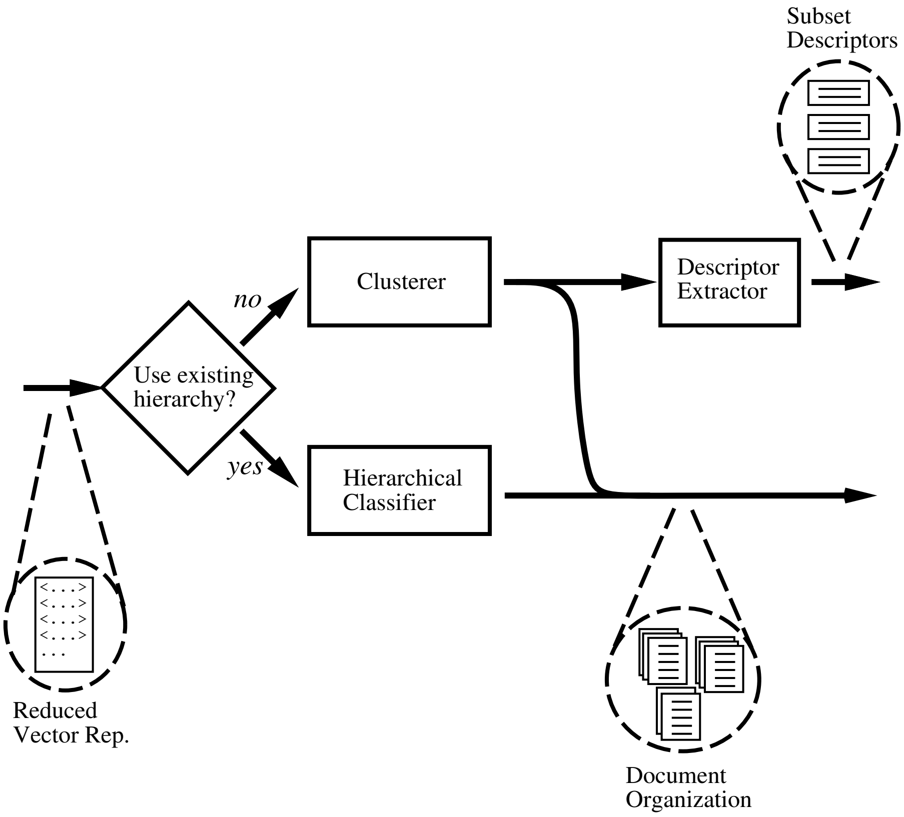

Figure 9.3: The machine learning components in the SONIA system.

The retrieved document texts are then parsed into a series of alphanumeric terms (i.e., words). Optionally, these terms may be stemmed to their root as SONIA's parser includes a standard word stemming scheme [125]. We note that we currently do not make use of such stemming in the examples of system usage provided in later sections. Empirically, we have not found stemming to create much of a difference in the results obtained with the system.

Each term then forms a dimension in a high-dimensional space in which the documents can now be represented as vectors. That is, the vector representing a document contains in the dimension for each term, the count of how many times that term appeared in the document. Since we now have the term counts for each document, SONIA is capable of transforming the vector representation of documents to different term weighting schemes, such as a Boolean representation (as in Eq. 2.6). Such different representations are easily generated when needed by different modules within SONIA.

9.3.2 Initial feature selection

As we have mentioned on a number of occasions, the number of distinct terms in unrestricted text is very large, so feature selection is generally necessary. SONIA uses a multi-tier feature selection process, using both stop word elimination as well as statistical techniques to reduce the feature space drastically. The system currently incorporates two initial forms of feature selection, each of which operates on the vector-space representation of the documents. Initially, dimensions representing stop words are eliminated from the document vectors. These stop words are determined using a standard English stop word list of 570 words (for example, see Table 2.4), as well as a hand-crafted list of approximately 100 Web stop words (such as "html" and "url").

In the second tier of feature selection, a Zipf's Law analysis of term occurrence over the collection is used. This process eliminates terms that appear fewer than three or greater than 1000 times in the entire collection as not having adequate resolving power to differentiate sub-collections of documents. These threshold values implemented in SONIA were chosen heuristically since they appear to work quite well in practice across the range of collection sizes typically used with the system—a few hundred documents. We note that we have explored methods for tuning these thresholds to the characteristics of more varied collections in previous work [139]. However, in the current implementation of SONIA we have not observed much empirical difference through the use of these more sophisticated techniques, and thus do not employ them here to save processing time.

After these first two stages of feature selection, the system reaches a branching point (see Figure 9.3) depending on the user's choice to organize the current set of documents with respect to an existing hierarchy or not. If an existing topic hierarchy is not being employed, we are working in the context of *unsupervised* learning, and consequently allow the user to create a new document organization from scratch using the clustering method described in Chapter 5. If an existing hierarchy is being used, then we have a *supervised* learning problem, in which case we can employ the hierarchical classification scheme presented in Chapter 8. We describe the clustering and classification modules in more detail in Sections 9.3.3 and 9.3.4, respectively.

9.3.3 Clustering

We first consider the case where an existing hierarchy is not being employed and the user chooses to create a new organization scheme. Since we have found that the clustering techniques presented in Chapter 5 are quite effective without the use of any additional feature selection beyond that which we have already applied, SONIA currently does not make use of the unsupervised feature selection methods presents in Chapter 4. This not only gives us an important computational savings, but we believe that the use of additional feature selection at this point would have little influence on the efficacy of the clustering algorithm. The reason for this is that we have no direct objective function tying the additional feature selection to the subsequent discovery of a good organizational scheme (as we do when a class hierarchy is present). Thus, we choose to be conservative by keeping more terms. Moreover, the clustering algorithms we employ are less computationally intensive than those used for classification. Hence the fact that we keep more features during clustering is not a serious computational hindrance.

In the clustering method implemented in SONIA, we use the document similarity measure based on the scaled term overlap between two documents, given in Eq. 5.25. Moreover, to compute the probability of a word appearing in a document, we used the normalized geometric mean (NGM) estimate, defined in Eq. 5.18, as this provided the best results in our previous comparative work. Nevertheless, we note that any reasonable clustering method can be used in this module of SONIA. In fact, SONIA's modular architecture makes it easy to upgrade any individual component as advances are made in a particular technology.

In conjunction with this similarity measure, SONIA makes use of a two step clustering procedure. As explained in Chapter 5, this clustering procedure first uses hierarchical agglomerative clustering to form an initial set of seed clusters. These clusters are then further optimized using an iterative clustering technique (with a maximum of 10 iterations). We chose this two step approach since it produced the best results in our previous experiments. Moreover, this approach is deterministic, which is an important characteristic from the end-user standpoint. On this point, our belief is that reproducibility of results is important to users, especially since they may find it disconcerting to see different results when applying a clustering operation to the same collection of documents.

Both of the steps in the clustering procedure employed here require not only a definition of a similarity score between pairs of documents d_i and d_j (which we denote $Sim(d_i, d_j)$), but also a similarity score between each pair of *clusters* c and c' (which we denote $Sim(c, c')$). This score is computed (as in our previous clustering experiments) using the *group-average* similarity between every pair of documents in those clusters (where one document comes from each cluster). More formally,

$$Sim(c, c') = \sum_{d_i \in c, d_j \in c'} \frac{1}{|c| \cdot |c'|} Sim(d_i, d_j). \tag{9.1}$$

Finally, note that the clustering methods we employ currently require that the user specify a priori the number of clusters into which a collection of documents should be grouped. SONIA provides an interface which easily allows users to direct the system to produce anywhere from two to 10 clusters. Moreover, since SONIA also allows users to *aggregate*, or undo a clustering of documents, it is simple for users to repeatedly try clustering a collection of documents into different numbers of subgroups and then select the one that best suits their needs.

9.3.4 Classification

Alternatively, a user may be employing an existing topic hierarchy to classify an incoming stream of documents (for example, in response to a follow-up query by the user). In this case, SONIA directly applies the hierarchical classification scheme (i.e., the *pachinko machine*) described in the previous chapter. To this end, we use a combination of feature selection and Bayesian classification at each node in the user's existing topic hierarchy. Here, the documents in the existing hierarchy simply become the training data and a hierarchy of classifiers is built that can then be used to classify the new incoming documents. In generating the hierarchy of classifiers, we first apply the feature selection algorithm introduced in Chapter 6 to the documents at each node of the classification hierarchy. Note that we use no conditioning information (i.e., Markov blanket size = 0) during this feature selection process. This allows us to obtain fast computational speed, without much degradation in classification performance (as evidenced by the results in Chapters 7 and 8). Currently, we select the 50 most informative features at each node, since the results from Chapter 6 show that 50 features are often sufficient for accurate classification. Moreover, using so few features gives us a big win in terms of computation time when subsequently inducing k-dependence Bayesian classifiers (which require time quadratic in the number of features).

Note that SONIA provides full generality to use any classification algorithm. However, we have chosen to focus on techniques based on k-dependence Bayesian classifiers presented in Chapter 7. More specifically, SONIA makes use of 1-dependence models, as these seem to provide the best tradeoff between expressivity and parameter robustness (i.e., bias and variance). Finally, once the hierarchy of classifiers is built, we simply classify the new documents in accordance with this resulting pachinko machine.

After the new documents are classified, they are then displayed in the appropriate hierarchy nodes with asterisks appended to the start of their titles to readily differentiate them from the documents previously existing in the hierarchy. We show an example of this in the usage scenarios presented later in this chapter. Thus, the topic hierarchy can be updated with minimal effort by the user.

9.3.5 Descriptor extraction

Finally, whenever the user clusters a set of documents, he or she not only produces a partitioning of the documents, but also causes SONIA's final module to extract *descriptors* from the document subsets. These descriptor words provide a description of the subtopics found in the document collection, and are presented to the user as initial labels for each cluster. More precisely, SONIA generates both a grouping of the documents based on the results of clustering as well as a set of automatically generated topical descriptors that are extracted from each such subset of documents.

We have informally compared a few methods for extracting these descriptors. The first such method is a probabilistic *odds* scheme in which, for each cluster c_j, we compute the probabilistic odds $O_j(x_i)$ of a term x_i appearing in a document in c_j versus appearing in a document in any other sibling cluster c_k. More formally, we have

$$O_j(x_i) = \frac{\mathrm{P}(x_i \mid c_j)}{\sum_{c_k \neq c_j} \mathrm{P}(x_i \mid c_k)} \,. \tag{9.2}$$

We then select some number, κ, of terms with the highest O_j values as the descriptor for document subset c_j.

Alternatively, we have also considered a simple *centroid*-based approach for descriptor extraction. Here, we simply compute the Euclidean centroid of all documents assigned to each group c (using the term frequency vector representation for each document). As before, we simply take the κ terms corresponding to the dimensions with highest value in the centroid vector as the descriptor for that group. Currently, we use $\kappa = 12$, as this value appears to achieves a good balance between brevity and descriptiveness.

In practice, we have found that the centroid-based approach appears to yield words that are much more indicative of the topic of a given document subset. It should be noted, however, that part of the success of the centroid-based approach relies on the efficacy of prior stop word elimination to prevent common meaningless words from appearing in the descriptor lists, since these words will be very common and hence have high frequency counts in all document subsets. In contrast, the problem with the odds based approach is that it seems to favor very rare (and hence not particularly descriptive) terms that may appear a few times in one document subset, but not in any of the others. As a result, these terms get a much higher *odds* score than more common terms that may appear even a few times in the other document subsets.

We also make use of the descriptor extraction module in SONIA to help *suggest query terms* pertaining to a particular document subset to the user. Here, the user simply selects any topic node (i.e., document subset) in a hierarchy and asks the system to supply terms which may be useful in future queries aimed at finding more documents related to the selected topic. We simply employ the same centroid-based descriptor extraction scheme, and list for the user the top 50 words (in order) in the resulting centroid vector. As will be seen in the example usage scenarios below, we have found this scheme to work quite well in practice.

9.4 Examples of System Usage

Having detailed the myriad components that comprise SONIA, we presently give detailed examples of the complete system in action. Since it is difficult to provide an objective measure by which to evaluate such a system as a whole, we attempt to elucidate on the particularly salient aspects of each of the examples presented here, so as to highlight both the strengths and weaknesses of the system.

9.4.1 Usage Scenario One

In the first usage scenario, let us consider the situation in which a user is interested in finding out more information about the ringed planet Saturn and its recently discovered new moons. The user begins by using SONIA to issue the query "Saturn" to the *Excite* Web search service, asking for the top 200 matching URLs. The query dialog box in which this request is issued is shown in Figure 9.4. Note that the user has a uniform interface regardless of which information sources are being queried.

SONIA sends this query via the InfoBus to *Excite* and receives the top 200 matching URLs back from the search engine. At this point, the parallel crawler module of SONIA is invoked to simultaneously retrieve all 200 Web pages pointed to by these

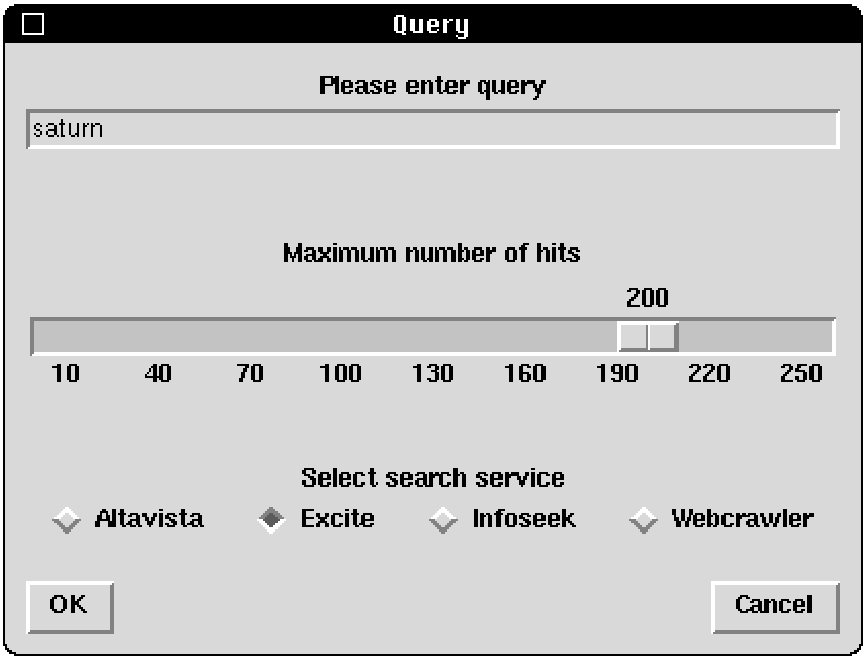

Figure 9.4: Using SONIA to issue the query "Saturn" to *Excite*.

URLs. Of the 200 original URLs, the crawler is able to retrieve 150 valid documents, since may links on the Web are outdated and point to non-existent documents, or the servers on which some of these documents reside may not be currently operational. These documents are then parsed into a vector representation, which initially has approximately 4000 features. The application of both Zipf's Law-based feature selection and stop word elimination reduces these vectors to 1872 features. The entire process of document retrieval, parsing and initial feature selection takes approximately 1.5 minutes of wall clock time on a heavily loaded Sun SPARC ULTRA 2.

The user then chooses to cluster this initial group of documents into four clusters in order to see what sorts of topics are contained in this collection. Applying clustering to the entire collection and then extracting descriptors from each resulting cluster takes roughly 1 minute. While we believe that the time taken to cluster is quite reasonable for an interactive system, we note that it would be possible to speed-up this process in many ways if desired (e.g., using an approximation to the similarity measure that is faster to compute, performing fewer clustering iterations, etc.), but

Extracted Descriptors	Sample Document Titles	Feasible Topics
saturn sega games sale fighter game virtua world nhl www ii $5	Sega Online: Strategy Guides Sega Saturn with 6 games for sale Sega Saturn Links	Sega Saturn Video Game
saturn car club price saturn's market higher san 000 service money diego	Saturn: A Case Study of How to Grow Market Award Winning Saturn Falling On Hard Times Saturn of Honolulu	Saturn Car Businesses
saturn talk car www sc2 fl kind mail 4a4 home dark sl2	Saturn, Let's Talk Saturn New Car Sales Saturn is different! New cars and used	Saturn Car Talk and Info
saturn saturn's rings moons ring jupiter moon planet system hydrogen interior earth	Saturn's Small Moons The 1995-6 Saturn Ring Plane Crossings Science Tip - Saturn	Planet Saturn

Table 9.1: Initial clustering results on documents matching the query "Saturn".

such speed-ups may have a negative impact on the quality of the resulting clusters. This is a classic example of the trade-off between speed and quality. Still it may be possible to tune our system for different user contexts, as appropriate.

An overview of the results of this clustering are given in Table 9.1, which includes the descriptors automatically generated for each cluster, a sample of the titles of documents contained in each cluster, and a human-generated description of the likely "feasible topic" for each cluster. Figure 9.5 shows the SONIA interface after this clustering, reflecting the first level of the topic hierarchy being generated by the user. Furthermore, Figure 9.6 shows this hierarchy after the user has renamed each cluster with its feasible topic.

The results in Table 9.1 reveal that SONIA is quite capable of discovering meaningful subtopics within the given collection, readily distinguishing those documents about the planet Saturn with those about the car company, as well as the Sega Saturn video game. Moreover, these results also highlight how query results may contain many documents which are about topics entirely different than what the user is truly interested in. By using clustering to quickly identify and sort out these non-relevant documents, the user is able to quickly hone in on just those documents relevant to him or her. This is especially important when we note that some of the web pages

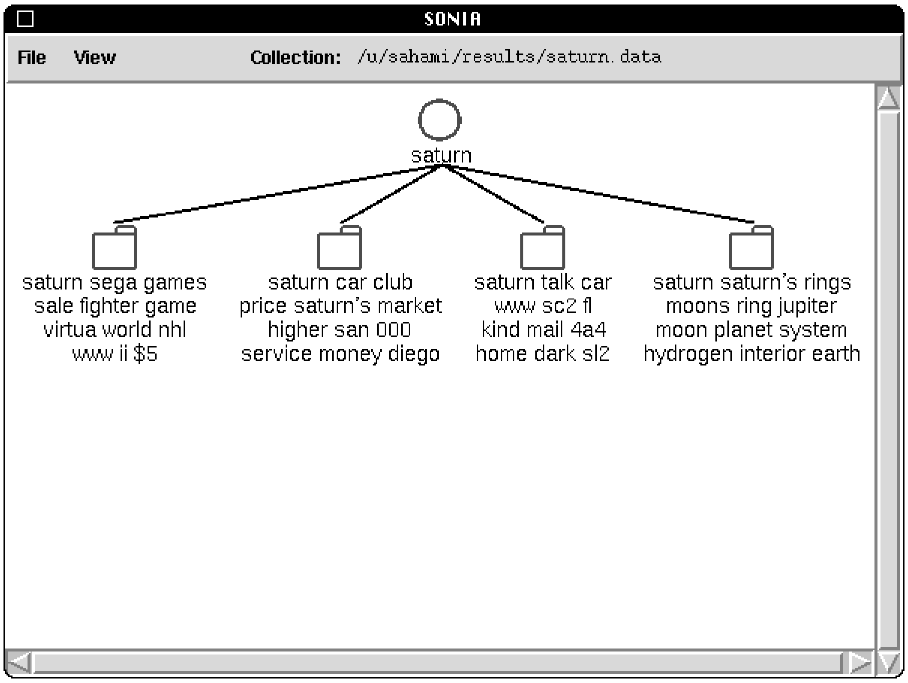

Figure 9.5: SONIA display after initial clustering of "Saturn" collection.

of Saturn car enthusiasts have such vague titles as "Saturn Page" and "Saturn" that could be misconstrued as pages about the planet if only titles were available (as is the case with simple Web searches that provide no categorization mechanism). Indeed, a detailed analysis of this document collection reveals that only 38 of 150 documents are about the planet Saturn, and all of then are correctly grouped together into one cluster. Similarly, the collection contains 23 documents about the Sega Saturn video game, which are all placed in a single cluster. The remaining two clusters contain documents exclusively about the Saturn car company. Thus, it appears that SONIA's clustering algorithm is very effective at keeping each subgroup of documents relatively coherent.

At this point, the user wishes to find more structure in the subcollection of document specific to the planet Saturn. Consequently, SONIA is used to cluster this subgroup into three clusters. A brief overview of these clusters is given in Table 9.2.

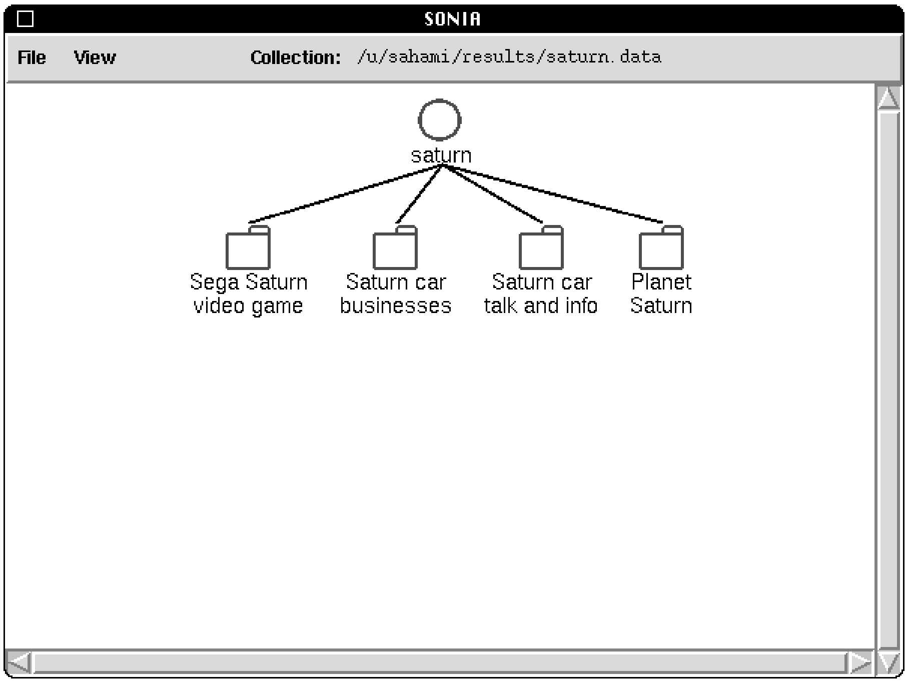

Figure 9.6: SONIA display after naming of the document clusters.

A closer examination of the contents of each cluster reveals that one contains documents on general information about the planet Saturn itself, another contains documents primarily about Saturn's rings and moons (i.e., satellites), and the final cluster contains documents pertaining to the observation of Saturn. We do note, however, that the descriptors automatically extracted from each cluster tend to have several overlapping terms. Thus, using these descriptors alone may be insufficient for the user to properly discern the true topic of each subcollection. Still, by reading just a few of the document titles in each subcollection, the contents of each subgroup becomes much more clear.

A close manual examination of this clustering shows that only two documents are questionably placed in a cluster which may not best match their topical theme. These documents both have a fair deal of information about the planet Saturn itself, but also contain a good deal of information on Saturn's rings and virtually all of its moons. These documents are both clustered into the "Moons and Rings", which seems quite acceptable. However, they could have been equally reasonably grouped into the

Extracted Descriptors	Sample Document Titles	Feasible Topics
saturn hydrogen planet interior saturn's jupiter ice layer rings composition system core	Saturn Facts Composition of Saturn's Interior Saturn - What We Know	General information
saturn saturn's moons rings ring moon plane jupiter system voyager image planet	7 (Saturn) Moons, compare Saturn's Small Moons The 1995-6 Saturn Ring Plane Crossings	Moons and Rings
saturn jupiter moon venus mars mercury day rings side earth picture visible	Best times to observe Hourly Cycle of Solar System Objects APOD: July 5, 1995 - The Night Side of Saturn	Observation

Table 9.2: Results of clustering the "Planet Saturn" subcollection.

documents containing general Saturn information. While their current grouping is very reasonable, this example does show that allowing documents to belong to *multiple* clusters would be a desirable property. In previous work [139] we have taken some initial steps in this direction, but a good solution to this problem still remains an open question.

Still, if the user believes that some documents are improperly categorized by the system, SONIA's interactive interface allows the user to move such documents to other folders in the hierarchy if they choose. Thus, the clustering need not be 100% accurate in order to still be effective at partitioning the document collection into meaningful subgroups.

Returning to the usage scenario, the user now decides to rename the subcollections about the planet Saturn for easier identifiability. Moreover, the user also restructures part of the hierarchy dealing with Saturn cars to create one subtree which encompasses both of the document groups related to this topic. This final hierarchy, as displayed in SONIA, is shown in Figure 9.7.

Focusing on the subtree dealing with the planet, Figure 9.8 shows a view of the interface with two of the document folders open. This figure displays how users are able to browse document titles (and accompanying URLs) easily within the SONIA interface. Thus, when automatically extracted descriptors may be inadequate to characterize the topic of a group of documents, it is simple enough to peruse the

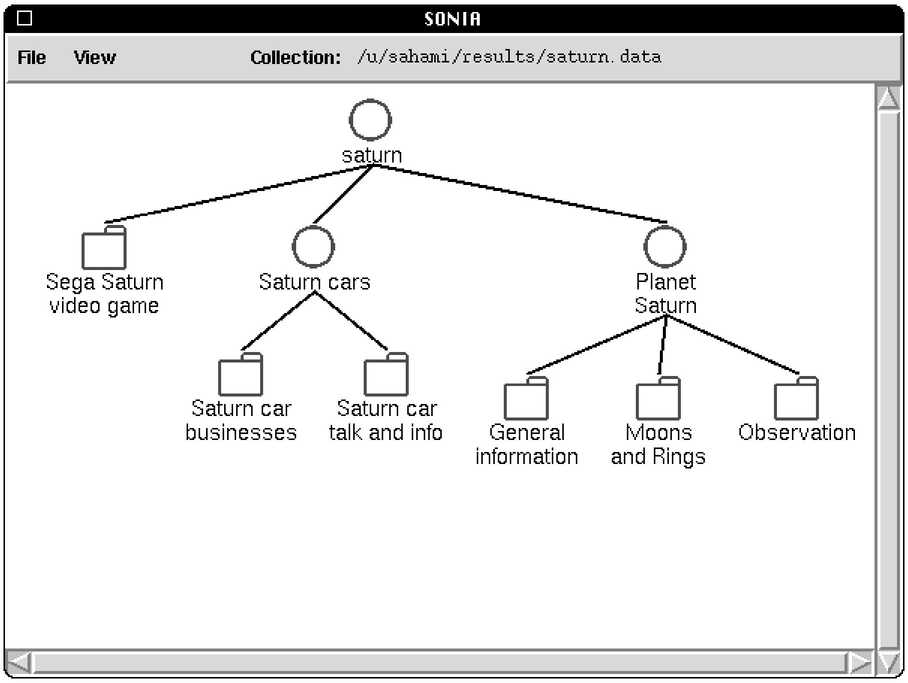

Figure 9.7: SONIA display showing the hierarchical organization produced by the user.

document titles within SONIA.

Nevertheless, a user may wish to browse the actual Web pages in a given group of documents, as opposed to just their titles. To this end, SONIA is capable of dynamically generating Web pages which contain links to the actual documents contained in each node of the hierarchy. Moreover, a browser is launched in parallel with SONIA to display such Web pages to the user. Consequently, the user may organize documents with SONIA and also browse the documents in any subcollection generated with the system *at the same time*. Hence, the ability to actually browse documents is well integrated with the tools for organization that SONIA provides, helping to enable a more topical level of document browsing. For example, Figure 9.9 shows one such Web page generated by SONIA for the "Moons and Rings" subcollection.

Say that the user now wishes to find more articles related to the "Moons and Rings" of Saturn. By selecting this node in the hierarchy and asking the system to suggest query terms, the user invokes the descriptor extraction module of SONIA.

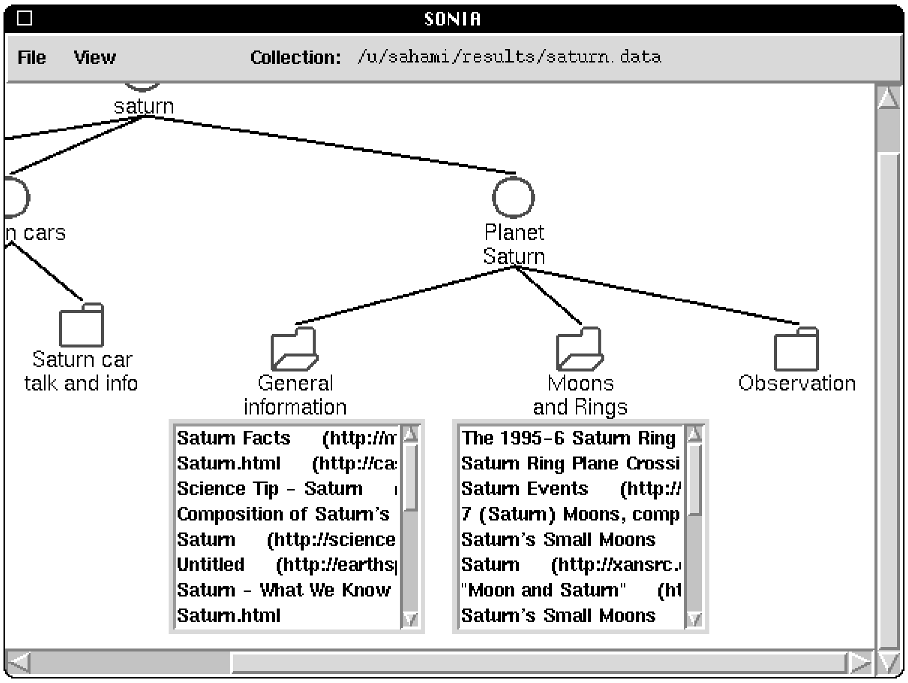

Figure 9.8: SONIA display focusing on the sub-clusters of the "Planet Saturn" node.

This module then displays the top 50 terms related to this collection of documents. We show the top 24 of these terms in Table 9.3. Note that this list of terms includes the specific names of several of Saturn's moons (e.g., pan, dione, tethys, atlas, and titan), as well as more general terms for referring to such objects, such as "moons" and "satellites".

With these suggested query terms in hand, the user then issues the follow-up query "saturn satellites and moons" to the system, asking for 50 URLs from the *Excite* search service. As before, this request is serviced using the InfoBus protocol. The crawler in SONIA is capable of retrieving 40 actual web pages from the 50 URLs returned by *Excite*. Rather than organizing these documents from scratch, however, the user makes use of the currently existing topic hierarchy to classify the new incoming documents. These documents are classified into the topic hierarchy by inducing a pachinko machine classifier, as detailed in Section 9.3.4. The process of inducing a hierarchical classification scheme and classifying the new documents takes approximately 20 seconds.

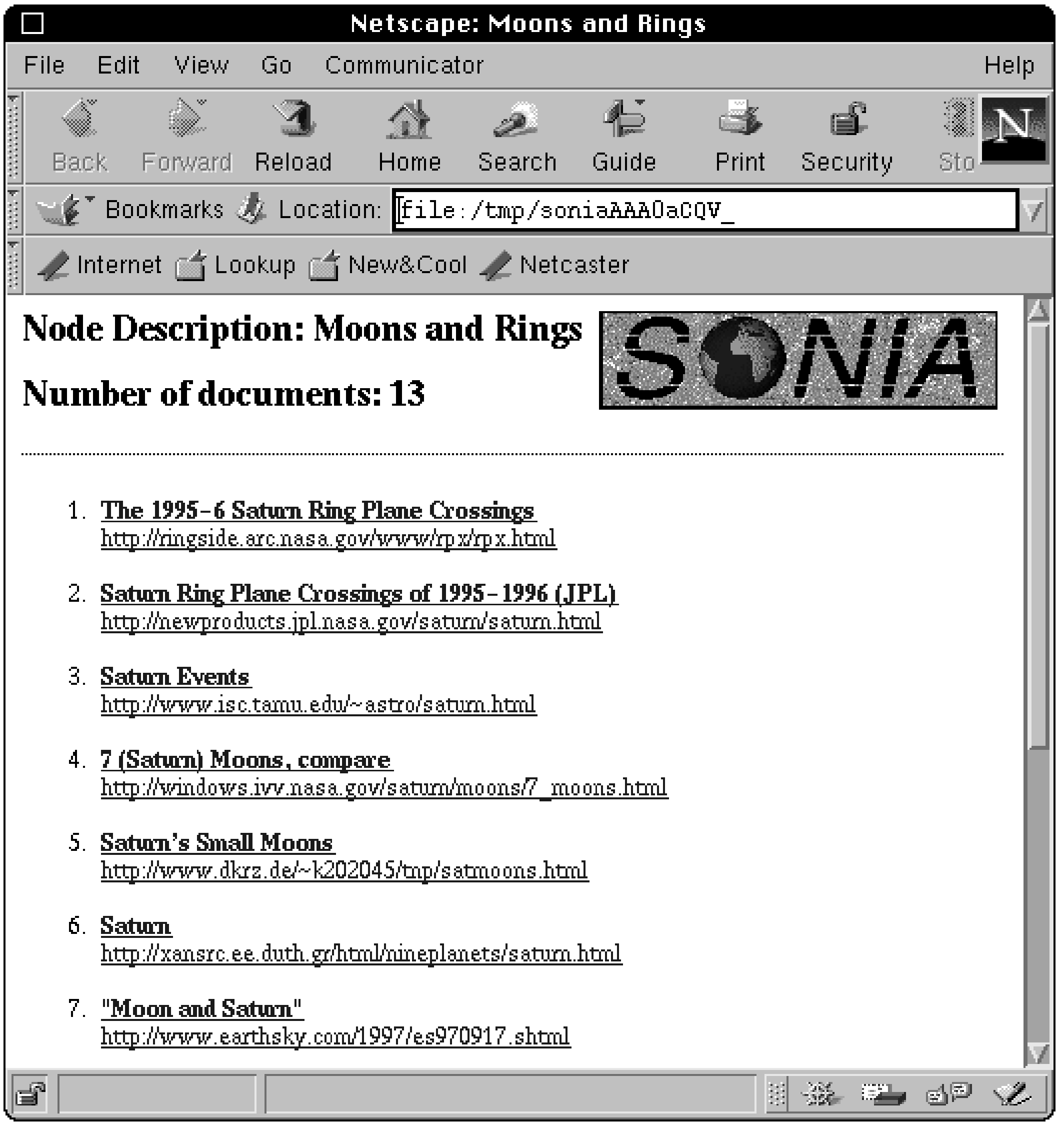

Figure 9.9: A Web page, dynamically generated by SONIA, containing links to documents in the node labeled “Moons and Rings”.

saturn	system	solar
saturn's	voyager	tethys
moons	image	cassini
rings	planet	km
ring	pan	atlas
moon	earth	titan
plane	dione	telescope
jupiter	satellites	gif

Table 9.3: Top 24 suggested query terms for "Moons and Rings" subcollection.

A close inspection of the results of this classification reveals that only one document is not placed somewhere in the "Planet Saturn" subtree. This Web page, which contains a problem set from an astronomy class, would most likely belong in the "Moons and Rings" topic. However, it is mistakenly placed in the "Saturn Car Talk and Info" node, since it contains little actual information about Saturn's moons and rings, and uses very conversational language (akin to many of the "car talk" documents).

In the "Planet Saturn" subtree, three documents are placed in the "General Information" topic, and all these documents do in fact appear to belong in this category, as they contain information predominantly about the planet Saturn itself and only in passing refer to its satellites. The remaining 36 documents retrieved in response to the query are placed in the "Moons and Rings" class. Not surprisingly, 33 of these documents are in fact discussing the moons and rings of Saturn. Two of the documents are about the moons of the planet Jupiter and are arguably classified into the best topic in the hierarchy for them. The last document contains information on the moons of both Jupiter and Saturn, and again seems to be appropriately classified, given the classes that exist in the hierarchy. Thus, it seems that SONIA is quite effective at classifying documents into a hierarchy formed via clustering, making only one error in classifying 40 documents in the example above—a 97.5% accuracy rate. Still, we point out that it would be useful for a system such as SONIA to detect when incoming documents many not fit nicely into any existing topic. This is one venue we believe would be especially promising for future work.

If more documents were misclassified in the example above (for example, into nodes not in the "Planet Saturn" subtree), it might be argued that the user would not look at these articles if he or she only focused on the results of the "Planet Saturn" subtree. This point becomes less significant, however, when we recognize the vast quantity of relevant documents that a user would never see on a subject because they are not in digital format, have not been indexed, etc. In the context of large information repositories, such as Digital Libraries and the Web, the ability to get query results with high *precision* is generally much more important that being able to *recall* all possibly relevant documents. Thus, for classification, it is arguable that we should generally care more about filtering out non-relevant information than making sure we properly classify all relevant documents. However, we do not pursue this conjecture further here, especially since SONIA's success at classification appears to make this point almost moot.

9.4.2 Usage Scenario Two

In the second usage scenario, we consider the case where a user makes use of SONIA to help him organize the files in his computer's file system[1]. The user begins with 66 text files that he wishes to organize. Being a graduate student in Computer Science who has been engaged in a recent job search, this user's document set contains both job-related documents (cover letters, various resumes, and letters of recommendation written for former students who are also conducting job searches), as well as some material related to courses (which have recently been taken or taught by the user).

In loading these raw text files into SONIA, the documents are first parsed to produce a vector representation, and then both Zipf's Law-based feature selection and stop word elimination are applied. Since the documents are stored on local disk, there are no network delays in loading this data. Thus, the process of parsing all the documents and applying feature selection, which reduces the feature set from

[1]In fact, the particular user in this study is the author of this dissertation, who used SONIA to organize some of the actual documents that had been stored on his personal computer.

Extracted Descriptors	Sample Document Titles	Feasible Topics
stanford computer science university teaching programming research department ca learning program interests	New_Resume cover-letter-education Andy-Reference	Job related
user error minor system • program users time problem command major model	CS147-paper1 psych251-week3 GradingCriteria2	Class related

Table 9.4: Results of initially clustering the file system collection.

approximately 3500 to just under 900 features, is performed in 20 seconds (again, on a heavily loaded Sun Sparc Ultra 2).

To initially organize these documents, the user decides to cluster the collection into two group, since there are at least two major themes present. With such a small collection, clustering takes less than 10 seconds to perform. In Table 9.4, we give an overview of these clusters. Also, the SONIA interface showing these two initial clusters is presented in Figure 9.10.

From the results seen in Table 9.4, it is apparent that SONIA can effectively distinguish the job-related document from those pertaining to academic courses. For example, the job-related cluster is characterized by descriptive words such as "stanford", "computer", "science", "teaching", and "research", which are very indicative of the user's background as a graduate student in Computer Science at Stanford University, and his desire to obtain a job involving teaching and research. Also, descriptive terms from the class-related cluster, such as "user" and "model" are reflective of the fact that the user's classes, CS 147 and Psychology 251, are about human-computer interaction and computational models of human learning, respectively. Terms such as "program", "major", "minor", and "error" are also seen in the class-related cluster, reflecting the fact that the grading criteria for the classes the user is teaching measure students' performance by the number of major or minor errors that are made on their programming assignments.

Moreover, a close examination of the results reveals that the only categorization errors made by the clustering algorithm are two letters of reference (which are clearly

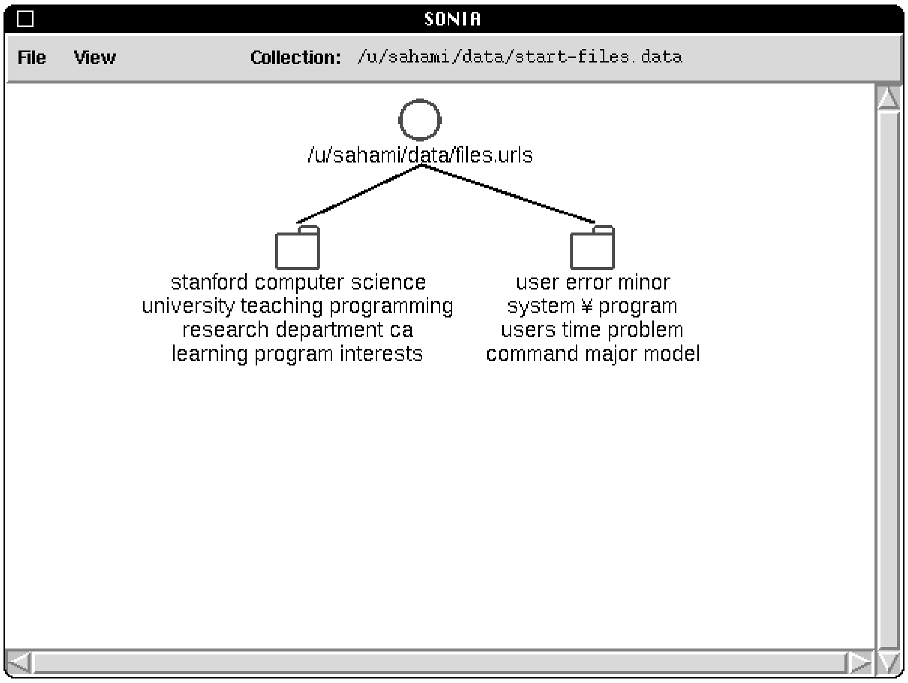

Figure 9.10: SONIA display after initial clustering of the user's file system.

job-related) that are incorrectly grouped with the class-related materials. Still, even with these errors, SONIA achieves quite an acceptable level of performance—97.0% accuracy. This result is also notable since the document collection is very skewed toward job-related information, containing 51 such documents, whereas there are only 15 class-related ones. Still, SONIA finds these two major themes, rather than simply trying to produce two convoluted clusters that are of more equal size. Furthermore, the interactive nature of SONIA's interface makes it quite easy for the user to simply move the two errant documents into the proper folder. The user can then name these two subgroups of documents "Job related" and "Classes", respectively, for easier identifiability. This naming is reflected in the interface shown in Figure 9.11.

Now, say, the user wishes to further organize the folder of "Job related" documents, so he clusters this subcollection into three groups. An brief overview of the results of this clustering are given in Table 9.5, where it appears that SONIA has once again successfully identified substructure in this document collection. Here, we find that the three sub-themes of the job-related documents become clear: resumes,

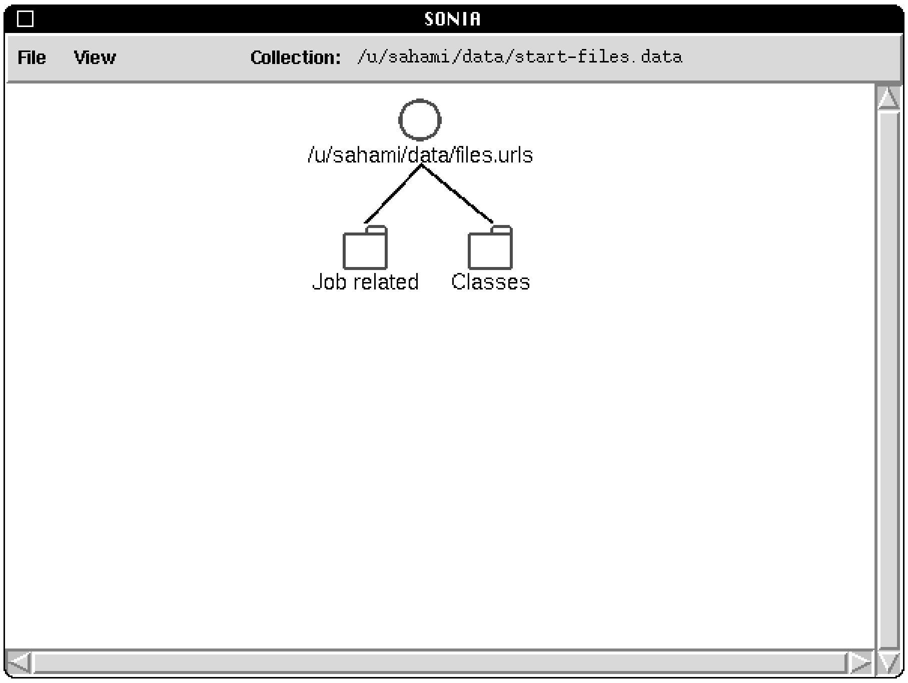

Figure 9.11: SONIA display after naming of the initial file system clusters.

cover letters, and job references written for former students. As in the first usage scenario, however, we see that the descriptors automatically extracted for these subgroups contain several overlapping terms, and it is only by viewing the document titles that we can distinguish the major themes of each subgroup. Thus, it seems that our descriptor extraction scheme, while working well at the high levels of the hierarchy, is of more limited utility at finer-grained subtopics lower in the hierarchy.

Performing our usual follow-up examination of the document clustering reveals that only a single reference letter is erroneously included in the same cluster as the cover letters. This yields an accuracy of over 98.0%. Again, the user can use SONIA to simply move this single misclassified document into the correct grouping with minimal effort.

Similarly, the user also decides to cluster the "Classes" subcollection into two groups, trying to separate the documents pertaining to the classes in which he enrolled from those that he taught. The results of this clustering are presented in Table 9.6. Here, the grouping is completely accurate at separating those documents written by

Extracted Descriptors	Sample Document Titles	Feasible Topics
computer stanford science programming university software ca teaching learning resident responsibilities dormitory	NewResume Resume-newest CurriculumVitae	Resumes
stanford computer research university science interests department information teaching consideration learning machine	letter-MIT coverletter-Brown cover-_U_Michigan	Cover Letters
stanford computer program science class programming university teaching student section students eric	Ref-Kleper Phil.rec referenceJen	References for others

Table 9.5: Results of clustering the "Job related" subcollection.

Extracted Descriptors	Sample Document Titles	Feasible Topics
user system • users command problem model information printer task provided knowledge	CS147-paper1 psych251-week3 GroupPaper	My Courses
error minor major errors properly program time array grading face smiley bar	GradingCriteria2 GC1 error-criteria	Grading Criteria

Table 9.6: Results of clustering the "Classes" subcollection.

the user for classes in which he was enrolled (CS 147 and Psychology 251) from the grading criteria for assignments in the programming classes he taught.

After giving more easily identifiable names to the new document groups formed via clustering, the user produces the hierarchy depicted in Figure 9.12. Note that this hierarchy is essentially a directory structure for helping the user manage this portion of his file system. Indeed, after some time, the user may write several more related documents and it would be natural to use SONIA to classify these newly written documents into the correct place into the file system hierarchy. In accordance with this idea, the user classifies a set of nine new documents (including two cover letters, three new papers for courses he is in, two reference letters for former students of his, an updated resume, and the last grading criteria for the class he is teaching this quarter) into the current hierarchy. SONIA is successful at correctly classifying 100%

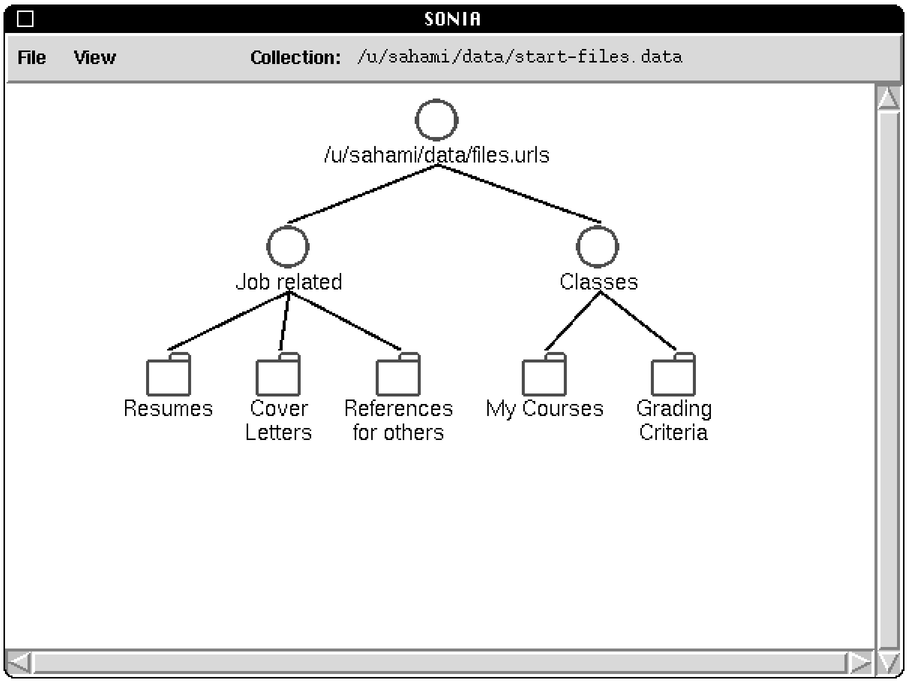

Figure 9.12: SONIA display showing the hierarchy initially constructed by the user.

of these documents into the appropriate classes in the hierarchy using the pachinko machine classification scheme. This shows that SONIA is also effective at helping a user maintain his hierarchical file organization as new documents are being introduced into the system.

After classifying these new documents, the user finally wishes to refine the hierarchy further by differentiating between the different classes he is taking. Thus, he chooses to cluster the “My Courses” folder into two groups (reflecting the number of courses he’s been working on most recently). In this case, SONIA distinguishes the two classes with complete accuracy, correctly grouping together all CS 147 papers in one folder and all Psychology 251 papers in another. This final hierarchy (after the “CS147” and “Psych251” folders are named by the user) is shown in Figure 9.13. Note that this figure also shows some of the document titles contained in the “Cover Letters” folder. Here, SONIA marks the newly classified documents with an asterisk (*) at the beginning of their titles to differentiate them from the documents which

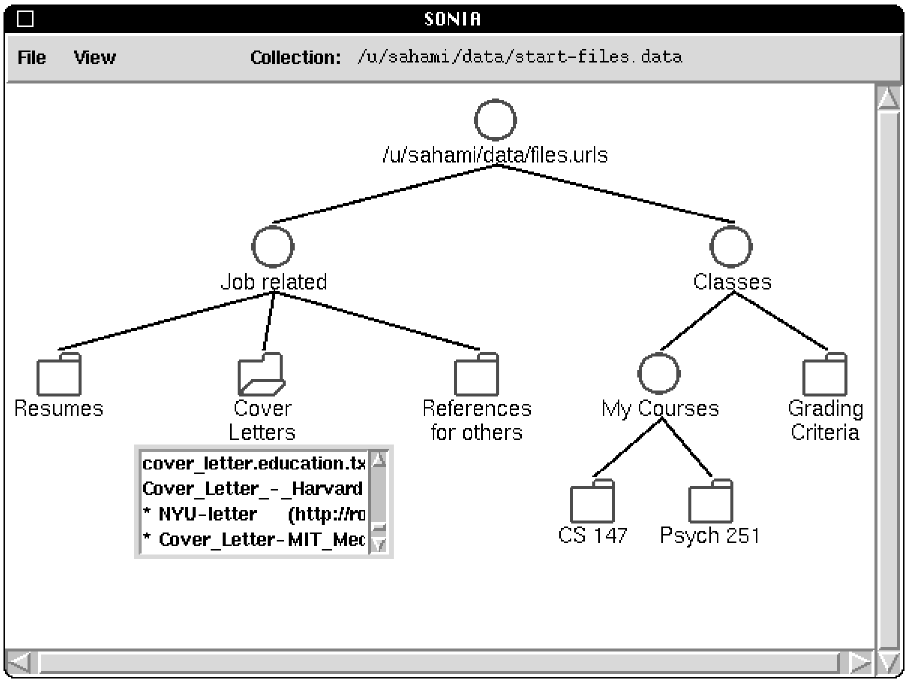

Figure 9.13: SONIA display highlighting the newly classified documents (marked with asterisks) in the "Cover Letters" folder.

previously populated the hierarchy. In this way, the user can quickly visually distinguish the new documents which have been automatically classified into a hierarchy from those that were there previously.

This usage scenario shows that SONIA is effective not only for helping users search for and manage information on the World Wide Web, but also for helping users organize information in much more personal environments, such as the documents on their desktop. Thus, we believe that a system such as SONIA has the potential to be an important component in an intelligent operating system that provides methods for helping users automate their file system management.

9.5 Conclusions

We have presented SONIA, a system that provides the ability to organize document collections into hierarchical categorization schemes, using a variety of machine learning techniques. SONIA is an operational system currently integrated into the Stanford Digital Libraries Project testbed via the InfoBus communications protocol. We have shown that SONIA can effectively help users find and keep track of relevant information in large information spaces by utilizing its automated organizational capabilities. Moreover, by emphasizing the use of hierarchies in document organization, we can leverage users' familiarity with existing hierarchical topical organization schemes used on the World Wide Web (e.g., *Yahoo!*) and personal computer file systems. In this way, we hope to allow users to quickly construct their own personalized and extensible hierarchies of categories, as well as apply SONIA in a variety of other related applications, such as organizing Web page bookmarks.

In future work, we hope to further develop several of the technical modules in SONIA. Most notably, we seek to find better means for descriptor extraction which are robust across a variety of granularity levels in the document hierarchy. Also, we would like to address the issue of clustering or classifying documents into multiple topics in the hierarchy, when appropriate. We believe that basing our work on probability theory can help take a first step in this direction by allowing documents to be included in multiple topics whose probability is close to the maximally probably category. Still, much work is needed to further formalize and implement this idea.