advanced searching techniques

chapter twelve

Advanced text searching tries to mimic the way humans remember and find things. These techniques try to overcome the "yes/no" and "on/off" nature of computer information. Databases require the user to know exactly how a word or term appears in the data, and usually he also must know which field the data appears in. The nature of the information in document management systems will often not be bent to such narrow indexes.

For this reason, libraries of digital information are expected to offer this human-like accessibility. The user thinks: "If I'm going to a digital library, I expect to be able to search every single word in every book in the whole library. Otherwise, I could just go to the old library with the paper card catalog."

This is perhaps a primal reaction to computers: "Okay, if you're so smart, prove it." If the computer is hard to use, or if the user can't find what he is looking for, the computer is the dumb partner, not the user.

example

People expect to be able to search for ideas much the way they do in common conversation:

DAVE: I want to read about that inventor who made those great spy planes...

HAL: Did he build the Blackbird spy planes?

DAVE: Right, and he built the P-38 and the U-2.

HAL: Did he work for Lockheed?

DAVE: Right, Lockheed Skunkworks.

HAL: Kelly Johnson appears in most articles on the Skunkworks and Blackbird, so you must be looking for Kelly Johnson.

In the above example, we used celebrity stand-ins from the Stanley Kubrick film 2001: A Space Odyssey. The human star was Dave, and the famous movie star computer was HAL9000. Advanced text-searching systems provide the kind of access that was fantasized about in this 1969 sci-fi epic and Academy Award-winning movie.

Methods For Advanced Searching

In a recent interview, Phillippe Courtot, CEO of Verity, Inc., one of the world's leading text search vendors, explained the goals of text-searching systems: "When the user asks the question, 'Who is the president?,' they want the answer! They don't want documents. They want 'Bill Clinton.' "

Traditional text-search engines bring back a series of articles. The user then reads the articles and arrives at the answer, or the information he sought. Courtot is referring to the next step of automation, where the computer reads the articles and brings back the answer. Here again, we are asking the computer to perform a human function; we are trying to mimic or automate thinking. This is a fundamental key of true instant access. This section goes beyond the basic searching capabilities of the Acrobat family by introducing compatible search features by other developers: natural query language, concept searching, fuzzy logic, intelligent agents and more.

There's a great article on this subject called "Chatterbots, Tinymuds, and the Turing Test: Entering the Loebner Prize Competition," by Michael L. Mauldin of the Center for Machine Translation at Carnegie-Mellon University. The content areas include: natural language processing, believable interactive characters, the Turing test, and the philosophy of AI.

Find this article on the Web via the URL, or search for the author and title!

http://fuzine.mt.cs.cmu.edu/mlm/aaai94.html

tip

Two levels of query expansion:

Lexical: Word stemming, wildcards, fuzzy, pattern recognition

Logical: Word thesaurus, dictionary, concept relations

For example, if you were searching for the "inventor of copiers," a lexical expansion would modify the individual search terms by narrowing them down to root words and then adding pieces to that root. The term "inventor" would at least be reduced to "invent" and all versions such as invented, invention and so on. With right and left truncation, or word stemming, other terms such as vented, inventory, prevention and so on might be created from the core form of the word. These types of query expansions may or may not help you find the "inventor of copiers."

On the other hand, a lexical expansion might offer you additional meanings of the query term. Perhaps a thesaurus or dictionary would suggest the word "xerography" in place of "copier" and lead to Chester Carlson.

In a nutshell, lexical expands the specific query term or word, while logical expands the meaning of the original query.

Natural Language Query

A natural language query capability allows you to "speak" to the computer in the same language commonly used to speak to humans. This is usually accomplished by a program that "parses" the user input query by stripping out stopwords and inferring relationships between the words in the query.

Stopwords are words that are purposely ignored in many text search engines; they usually include prepositions (of, in, to) and articles (the, a, an). The reason they are ignored is that they are considered too common in most collections to be useful, and the index can be made smaller and faster by ignoring them.

As an alternative to the more formal language of Boolean queries, natural language queries are especially helpful for new and unfamiliar users. With the widespread adoption of text-searching applications on the Web and everywhere else, the ability to form Boolean queries may one day become as common as the ability to type on a keyboard, but in the meantime, natural language query capability will be highly desirable.

For efficiency, experienced users may always rely on the more structured Boolean query language because the results are highly predictable and the terms can be controlled and modified with great precision.

For a taste of how a natural language query text-searching system performs, the Excite for Web Servers software runs on many Web sites, including the Adobe site. The Excite Web search engine is also available to search the Web itself.

The Excite Web searcher is available at:

http://www.excite.com

Excite is designed to help the user who may not know exact topics or keywords.

Concept Searching

If a computer is going to think like a human, then it should be able to handle many related ideas as if they are all part of one big concept. This is the quantum leap where advanced text searching loses the surly bonds of conventional computer databases.

On a very mechanical level, a form of concept searching can be handled by simple, brute force techniques. Wildcards are the original brute force technique. Searching with wildcards can extensively plumb the depths of text collections. Many text searchers use word stemming as a built-in wildcard function, and simple word or term entries are automatically expanded.

What wildcards and stemming do on an individual word level, concept searching does on the entire query level. In a concept search, the user can enter a number of words or terms of interest, and the software will expand these particular terms in a number of logical ways.

example

Building on the Dave and Hal interaction:

The term “spy planes” might expand via a thesaurus to include the most popular terms such as Blackbird, U-2, Stealth and so on. The advanced text-search software serves up Blackbird as a potential “expanded” search topic.

By having the computer continue the conversation, by sensitively mentioning relevant topics, the text search proceeds in a way that intuitively blends the user’s intentions with the computer’s ability.

For a look at Vector Space Similarity Searching, try Fulcrum’s Web site:

http://www.fultext.com

Automatic Summary

When talking to another person, large bodies of information can be assumed in just a few words.

example

DAVE: Do you know the fastest planes that were ever built?

HAL: Yes, I'm a big fan of all forms of supersonic flight. Which ones are you most interested in? The experimental planes, the military planes or other high-performance aircraft?

DAVE: I'd like to know about the fastest plane that takes off and lands like a regular plane and gets the fastest speed records.

HAL: Do you mean sustained speed, operational speed or some special post-to-post race speed ?

DAVE: I mean the plane that is the fastest overall. The fastest plane in my definition has the highest number of common, day-in and day-out missions that register the highest speeds and altitudes.

HAL: The SR-71 Blackbird appears in the majority of highest absolute speed records for conventional aircraft.

Automatic summary provides assistance in understanding the contents of the hit articles. In a way, this can be considered a form of advanced browsing rather than searching. Rather than just providing a ranked list of hit terms in response to a query, automatic summary will write a small report or abstract of each hit.

For some applications, these summaries may provide all of the information necessary from the hit articles and save you time retrieving and reading each article. In the example above, each of the iterations of the computer's response in the form of HAL's conversation reveals progressive summaries of the information available. Based on these summaries, Dave can quickly move to the articles that are of primary interest.

Fuzzy Logic

example

DAVE: I wonder what is the fastest plane that I could fly on?

HAL: Fly on as a regular passenger, or a custom flight on some special deal, or just what is physically possible ?

DAVE: What plane flies faster than anything else, that I personally, a non-pilot, could actually fly in?

HAL: Of a number of two-seater supersonic aircraft, the fastest is the SR-71B trainer. It can exceed Mach 3 in normal flight.

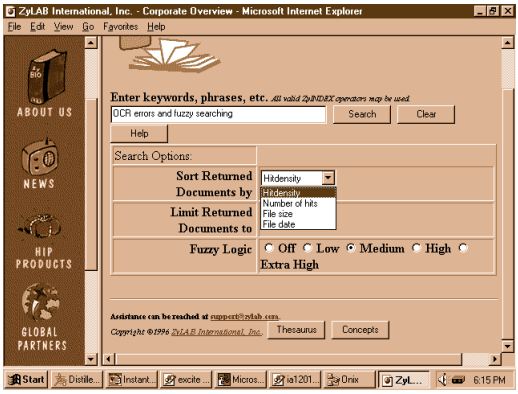
Fuzzy logic can be thought of as an “Expanded Or” search, where you want to find as many of your query terms as possible, but not necessarily all of them. Fuzzy logic attempts to perform the rough grouping that a human mind can.

In the example, the original query has to do with the fastest plane ride theoretically available. After one iteration, the query is narrowed down to a more focused search. This second search effectively finds the fastest planes and then narrows that down to planes with at least one passenger seat.

Another form of fuzzy logic that is relevant to text searching is the ability to recognize incomplete hits in the text. This can be very important where documents contain errors or variables in the words and terms. For example, documents that are converted through optical character recognition (OCR) may have many misrecognized characters within words.

These types of errors can not be completely compensated for by wildcards and word stemming because there is no way to predict where the errors may occur. If the error occurs in the stem word, these methods are ineffective.

Fuzzy logic in this case effectively does a wildcard-type search, where any of the letters in the words may be the wildcard. The fuzzy logic determines that if the hit term has at least some of the characters of the query term, it may be a valid hit.



To take a tour of fuzzy logic text-searching capability with a specific adaptation to overcome OCR errors, try ZyLab's Web site:

http://www.zylab.com

ZyLab offers ZyIndex, one of the venerable and perhaps most popular of traditional text search engines. Since Jon Karlin added document-imaging capability and many other updated features, a new product ZyImage has emerged as a uniquely well-integrated solution for scanning, recognition and text retrieval.

Semantic Word Relationships

In the previous examples, we discussed using a thesaurus to perform a concept search. For example, rather than searching for a "shoe" we could search for many kinds of shoes simultaneously. The query for "shoe" would also search for "boot," "sandal," "sneaker" and so on because all of the terms might occur in a thesaurus listing of the term "shoe."

To get a feel for semantic searching that employs dictionaries as well as thesauri and gives you the opportunity to choose the expanded query terms, try the Dataware Technologies or Excalibur Technologies Web sites:

In semantic searching, the terms can be expanded to a whole class of objects of the original query. In this type of concept search, or expanded term searching, "shoe" could be any article of clothing, not just something worn on the feet. Now the query "shoe" may also search for "hat," "gloves," "coat" and so on.

This prosaic example of "shoe" is overly simple. Loftier examples might include searching for "passion" in Shakespeare's plays, where the query "passion" is expanded to include all the array of emotions between love and hate, fear and joy, and the words and concepts that portray passion.

http://www.dataware.com

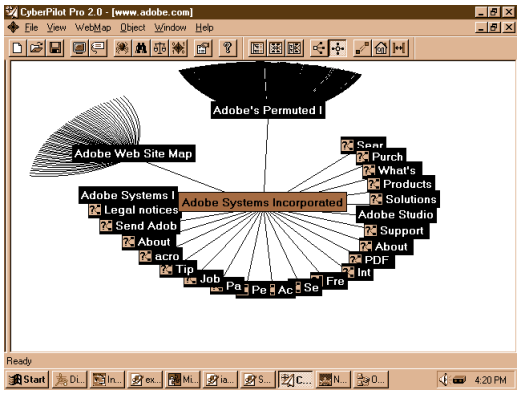
http://www.excalib.com

Intelligent Agents

Intelligent agents are also known as persistent searches, which continuously or periodically perform the same queries, which may employ any of the methods discussed previously.

For example, the intelligent agent could be constructed by a user to collect specific information from any number of sources. This is particularly relevant in constantly updating pools of new information. A very simple example of this functionality is offered by the automated clipping services that will constantly watch any number of news feeds and immediately tag any article that matches the query, just as traditional clipping services have done on paper.

Web robots and spiders are specialized forms of intelligent agents that constantly roam the Web and bring back indexes of the information available at certain URLs. Many of the large Web search engines employ these Web robots and spiders to continuously search and map the World Wide Web. In effect, the spider replaces a large number of humans doing manual searches and creating infinite bookmarks pointing to interesting information on the Web. Every user of the search engine enjoys the beneficial results of the labor of these tireless bots.



For the searcher who doesn't want to surf, a personal info-gathering bot like NetCarta's CyberPilot Pro maps Web sites for enhanced insight into contents.

Relevance Ranking And Term Weighting

Relevance ranking refers to the order in which the hit list is presented to the user. The feature is designed to save the user time by presenting the most "relevant" documents first. Some search engines offer relevance ranking on the fly, such as Acrobat Search's "Rank By" option.

Term weighting allows the user to assign more or less importance to each term in the query, thereby directly modifying the relevance of each term.

Relevance ranking refers to the way the results of a query are returned to the user. There are many ways to determine relevance, and each vendor tends to offer a variation on the theme.

In general, the purpose of relevance ranking is to present the user with the documents that are most likely to satisfy the intent of the query. Because there are many methods of determination, and several search engines actually use more than one method, the rank is often determined numerically.

Relevance rank may be determined by:

Total number of hits in the document;

Hit density which is determined by hits vs. total terms in the document;

Hit clustering which is determined by adjacency of the hits;

All of the query terms appear, rather than many instances of one or two terms.

There are other ways of determining relevancy, and each vendor's approach to the question is described in the vendor profiles at my Intelligent Imaging site:

http://www.onix.com/tonymck/ocrlab.htm

Term weighting is a way for the user to add emphasis to certain terms in the query, to add relative importance to particular terms or words. In effect, term weighting allows the user to influence or completely determine the relevance ranking of hit documents.

For example, if a user is searching for information on the SR-71 Blackbird reconnaissance plane, he might use term weighting to tailor the search to return only the most likely relevant documents. If the following three query terms are used, each could have separate weighting:

example

Query: SR-71 + + + , Blackbird + , Beatles---

In this case, three (+) or three (-) would be maximum positive and negative weighting for relevance:

SR-71 + + +	Most important term; return every document with this term.
Blackbird +	Neutral importance term; may be relevant; return hits.
Beatles---	Least important term; not likely to be relevant to the query.

tip

Always take at least a quick look at the user guide whenever using a new search engine. They all perform similar functions, but they all have their idiosyncrasies. For adding emphasis to a term, they may agree to " + " as the identifying operator, but that doesn't mean they agree that the " + " appears at the beginning or end of the term. For example, + agree or agree + ?

Advanced Text Searching In Action

In this section, we'll show advanced searching in action to achieve instant access. You should concentrate on the contents of your electronic document system while considering the usefulness of the features discussed here.

At the heart of this discussion, consider simple and advanced text searching to have a relationship comparable to that between arithmetic and advanced mathematics. While the truth of arithmetic operations is never compromised by advanced calculus, the formulae of calculus can arrive at solutions that would be extremely tedious or

Text searching comparison to math:

Boolean text search:
Arithmetic with associative nesting, algebra

Advanced text search:
Advanced algebra, geometry and calculus

even impossible to arrive at by extensive arithmetic operations. In a similar way, many of the advanced text-search features described here are often actually automated or combined operations of simple Boolean text-search operations.

Exploiting Verity Search: A Boolean Primer

In 1961, the late Gerard Salton received a grant funding his group's research into information retrieval (IR). His test bed, SMART, is the most widely used research tool in the field in the 35 years since its inception. Dr. Salton is one of the leaders, if not the leader, in the field of IR.

Boolean search can be complicated for the uninitiated but worth the effort for improving search results. The following techniques can be used alone or in combination with other search methods.

Expressions Within Fields

All of the Boolean functions described earlier in this book operate within Document Info fields the same way as they do in text searching. This of course means that the And, Or, Not operators can be used singly or in combination to retrieve a group of documents.

For example, for a system managing software documentation, the user might enter "Adobe **Or** Microsoft **Or** Oracle" in the Author field to retrieve all files produced by those three vendors.

For more information on the pioneering work in the field of pattern recognition in general, and specifically in text searching, visit Cornell on the Web. Start at this page, and then backspace to /Info and resume your browsing:

`http://www.cs.cornell.edu/Info/Faculty/Gerard_Salton.html`

Special Operators For Document Info Fields

The following operators are designed to be used only with the Document Info fields:

Operator	Semantics
=	matches exactly (for text, numeric and date values)
~	contains (for text values)
!=	does not contain (for text, numeric and date values)
<	is less than (for date or numeric values)
<=	is less than or equal to (for date or numeric values)
>	is greater than (for date or numeric values)
>=	is greater than or equal to (for date and numeric values)

The comparison operators "<," "<=," ">" and ">=" function just like the mathematical relations that they symbolize, namely less than, less than or equal to, greater than and greater than or equal to. These operators can be used only with values of the same type because, as everyone knows, you can't compare apples and oranges. Therefore, these operators can be used only with date or numeric values.

In terms of text, the new operators just refine earlier Boolean operators. The = operator provides for an exact text match for the entire field, while the ~ (tilde) operator simply requires that the search term be contained somewhere within the field. The != for text is equivalent to the Not operator for exact matches.

Expressions Within Multiple Fields

A query can be designed to use Boolean operators and multiple Document Info fields. As mentioned earlier, the large size of the Find text box suggests the extended queries that can be built in Acrobat Exchange. The Find text box is also used to enter query arguments that can include multiple Document Info fields combined by multiple operators.

By using the name of each Document Info field as part of each query term, the Verity search engine will look for the term only in that field. If we extend the mathematical explanation of the process one more step, we can say that this capability adds a "value" to each query term. However, unlike the simple positive and negative values of numbers in arithmetic, this system offers many "values" for a term.

Instead of just plus or minus, a query term may have a "value" of Title, Subject, Author, Keywords, Date Created or Date Modified. This capability of assigning a value to each query term allows a skilled user to perform very powerful combined text and database searching.

example

The example given in the Acrobat Exchange Help file follows:

"You can build a Boolean expression that uses more than one field by combining the field expression with the search expression in the Find text box. For example, if you enter:

("Sixteen to one project" **Or** "16 to 1 project") **And** (Author ~ Raskin) **And** (Keywords ~ "slide show" **Or** keywords ~ presentation **Or** keywords ~ spreadsheet)

in the Find text box, the search returns only documents that contain either the phrase "sixteen to one project" **or** "16 to 1 project" **and** have an Author field that contains "Raskin," **and** have a Keywords field that contains **either** the phrase "slide show" **or** the word "presentation" or "spreadsheet."

The first part of the query is the argument defined by parentheses

("Sixteen to one project" **Or** "16 to 1 project")

which demonstrates the use of double quotes to define a phrase rather than a word search. The reason that double quotes are required is that the stopword "to" is a part of the phrase, and without the quotations the "to" would be ignored during the search.

The use of the Boolean operator Or within the first set of parentheses makes this a query argument rather than a term because the contents within the parentheses are determined by the operation of the Or. That means that this "term" is actually one of a set of terms.

The first argument is then combined with the second argument via the And operator

(Author ~ Raskin)

which demonstrates the Document Info field search capability. Within this argument, which is once again the result of an operation rather than a single term, the ~ operator is used to select only those documents that contain "Raskin" in the Author field.

Both of these arguments are combined via the And operator with a selective search of the Keywords Document Info field. At this point in the math-like operation of the query, the user has defined two versions of a phrase that must have been published by one author, and the user has finely focused the search operation.

No matter how large this collection, the user will retrieve only one author's mention of the key phrases "sixteen to one project" or "16 to 1 project." The next argument will retrieve only such mentions as appear in slide shows, presentations or spreadsheets.

In this example, you would be excluding all other potential mentions of these query terms in the database that might have popped up in e-mail, proposals, training documentation, and other material that may not be useful for the purpose at hand.

Perhaps you are preparing a paper or presentation and need only the author's finished materials for reference. All other materials would be less meaningful because they may have just been part of the preparations for the documents represented by slide shows, presentations or spreadsheets.

The final argument

(Keywords ~ "slide show" **Or** keywords ~ presentation **Or** keywords ~ spreadsheet)

assigns the "value" of keywords to every term in this inclusive Or statement, which means that these terms must appear in the Keywords Document Info field.

Summary

As stated at the beginning of this section, the builder of an electronic document system should consider all of these approaches in terms of the content he intends to offer. In most cases, there are ways to adapt text-search engines to accomplish these functions once the desired functionality has been identified.

There is an entire education in this field freely available over the World Wide Web. The engines are all running out there, ready for test drives on the road to instant access.

The Boolean query translation project mentioned in Chapter 3 is one response to the opportunities and challenges of the new global library. It seems logical that one commonly accepted set of query grammar and syntax rules will become standardized, if only informally. Concept-search software will follow the same imperatives of market demand. Most users come to the Web hoping to find an easier way of accessing information. The concept-search software will interact with the user to refine expanded concept queries by suggesting options and techniques.