

Enhancing Search Engine Performance Using Expert Systems

Stan Lovic¹, Meiliu Lu², and Du Zhang²

¹Intel Corp., 1900 Prairie City Road, Folsom, CA 95630, stan.lovric@intel.com

²Computer Science Department, California State University, Sacramento, CA 95819
{mei, zhangd}@ecs.csus.edu

Abstract

Search engines of today do a great job of sifting through billions of pages of Internet content and returning search results highly relevant to user queries. However, in localized implementations (a local university search or an intranet search of a private company), the same search engine technology usually has less than satisfactory performance. The technology that works well on billions of pages of general content doesn't work well on a much smaller scale of closely related content. In this paper, we analyze the performance problem in localized search engine implementations and identify specific performance issues through examining search logs. Our proposed solutions to those issues are based on utilizing an expert system where the fixes to the search issues are defined as a set of rules. We conduct experiments with California State University, Sacramento web site, and the preliminary results indicate that when applying those rules to search engine queries and search results, search engine performance and user satisfaction are improved.

Keywords: search engines, localized search, JESS rules.

1. Introduction

In the last decade search engines have improved their performance to the point of becoming a tool of everyday use for most Internet users [3]. The leading search engine companies and many smaller players continue to invest in improving performance of their engines and inventing new methods of indexing and searching for content. While the web search arena is very saturated, the localized (enterprise) search technology is still developing and has room for improvement. This paper's focus is improving search engine performance in localized search engine implementations.

There are many public and private localized searches on the Internet and on corporate intranets, and most of them use the same or similar search engine technology that is used in global web search.

1.1. Scope-limited web search

Most global Web search engines allow users to use their technology for localized searches. By limiting search scope to a locality of interest, global Web search engines can be used as local search engines too. This only works

for public environments. Private intranets are not indexed by global Web search engines and cannot use this solution. Using a global search engine for a local search implementation allows little or no configuration of search features.

1.2. Enterprise search engine products

There are many companies [4] that specialize in enterprise search and provide search engine products that can be installed and configured in local environments.

Google Appliance. Besides providing a global web search, Google also makes a localized search engine product called Google Search Appliance [6]. The Appliance is a hardware and software platform that can be set up on corporate intranets and provide Google search technology in a private environment.

Ultraseek, Verity. Infoseek was one of the search engine pioneers which used its Ultraseek technology for both Web search and for an enterprise search engine product. Ultraseek enterprise search product was acquired by Verity, another search engine pioneer in the enterprise search arena. In 2005 Verity was acquired by Autonomy, which continues to market both Verity and Ultraseek software search engine products for enterprise search engine installations.

Convera, Coveo. Convera (formerly Excalibur) and Coveo (formerly Copernic) are software enterprise search engine products with advanced categorization/summarization features which make them useful for research and discovery.

Microsoft SharePoint. Not so configurable and not as powerful as the aforementioned search engines, Microsoft's enterprise search engine is referred to as a new and promising product.

All enterprise search products provide an administration interface, usually web-based, for configuring the search engine and tuning its performance. They differ in the number of configuration options and performance.

Because the search engine technology is tuned to work well on the World Wide Web, there is room for its improvements in localized implementations. The objective of this work is to analyze performance of localized search engine implementations and to find ways of enhancing that performance using expert system technology.

Local environments have their own properties and

characteristics that can be compiled and described by human experts. A person knowledgeable about an environment can be more effective in pointing users where to look for information than a search engine that uses algorithms tuned to work well on the vast amounts of information on the Web. In short, better results can be obtained by users of localized searches when the search engine technology is supplemented with experience and knowledge of human experts for a particular domain.

This paper is organized as follows. Section 2 describes typical problems in localized search engine implementations. Section 3 provides a methodology overview of the expert system solution. Search engine enhancement techniques are discussed in Section 4. Performance metrics are given in Section 5. Finally Section 6 concludes the paper with remark on future work.

2. Localized Search Problems

To find out how well a search engine performs in a localized setting, we use California State University, Sacramento (CSUS, a.k.a. Sacramento State) web site and Google search engine in our case study. CSUS web site uses Google's University search. Because CSUS search results are provided by Google and served from Google web servers, it is not possible to analyze web server logs directly. Instead, we captured user queries for a period of one year (July 1, 2004 to June 30, 2005) and sorted them to a list of most frequent queries. We also collected statistics on how many search result pages opened per query, and users who browsed to the second or third page of search results because they did not find what they were looking for on the first page. The number of "next-page" clicks per query is a good indicator of search problems. While many queries had less than 1% of "next-page" clicks, some had significantly more.

We find six common search issues through our examining the search logs and "next-page" queries. For each issue, there are two possible remedies: (1) A web page can be fixed or improved so that search engine gives it a higher ranking in search results. This remedy should be done by content owners. (2) Search engine can be tuned to provide more relevant results for specific queries. This remedy can be done by the search engine owner.

This paper focuses on remedy 2: how search results can be improved by tuning the search engine. Remedy 1 is preferred in some cases, but most often search administrators do not have control over content and they can only provide general suggestions to content owners on how to improve their content. Even if those suggestions are implemented by content owners, that still does not guarantee that search results will be satisfactory because search engine relevancy algorithms are highly unpredictable when it comes to how individual pages rank against the rest of the indexed content.

(1) Errors in Search Terms. The top user query on csus.edu web site was "webct" (course management web site) with 0.07% "next-page" clicks. The top 15th query on the list was "web ct". 12% of users spelled WebCT with a space between "web" and "ct". As a result, Google did not bring up "WebCT" home page on the first page of search results. Because of this, 6% of users went on to the second or third page of search results. What did others do? They did not find what they were looking for on the first page. Maybe they modified their query or they found WebCT site in some other way. Similarly, 28% of users spelled "casper web" when searching for "CasperWeb" (student and faculty service web site) and they could not find it among the first five pages of search results.

A search engine should know that when users type in "web ct" and "casper web" in the domain of csus.edu, they mean "webct" and "casperweb", respectively. The search engine should fix these minor user errors automatically, or point out the problem to the user and suggest how to fix it.

(2) Spelling Errors. Similar to the previous examples, these errors are also made by users when they enter search query terms, but these mistakes are made against the rules in standard English words.

(3) Synonyms. Users searching for "tuition" are not going to find the "fees" page. Similarly, users searching for "athletics" will not find the main "sports" site. In these cases, the most obvious remedy would be to somehow indicate to the search engine that "tuition" and "fees" are synonymous and if someone searches for "tuition" it should also retrieve results for "fees".

(4) Query syntax errors. One of the most frequent user errors is incorrect use of query syntax. Early search engines allowed Boolean expressions for building complex queries. They interpreted key words AND, OR, and NOT as Boolean operators. This syntax is too advanced for most users, so a simplified Boolean-like syntax, sometimes called "search engine math" [13], was introduced. The main principle of this syntax is that most operators need to be entered as prefixes to the query term. Incorrect use of query syntax is very common and can be observed in search logs. One of the most common errors is inserting space characters between the prefix operator and search term, for example "registration - graduate" instead of "registration-graduate". Some query syntax errors can be fixed automatically. It is also useful to provide suggestions to users on how to use supported query syntax correctly.

(5) Relevancy issues. Even if users do not make any obvious errors, search results can be unsatisfactory. The most relevant pages do not always show up at the top of the results, and the cause can usually be attributed to content owners who failed to design their pages with search engines in mind. In these cases, the best thing to do is manually force relevant results to the top of the search results list. Items placed on the top of the list regardless of the search engine algorithm ranking are sometimes called

“key results”, “key matches” or “best bets” [14].

(6) **Missing content.** If some content is not indexed and included in the search engine catalog, it cannot be found by users. There are various reasons why content may be missing. Search engine cannot index web sites that are not linked from other sites. A web site may require authentication and a search crawler does not have permission to index the site. A search engine cannot follow links on a web site because links are built dynamically using client-side script, usually JavaScript. Most search crawlers do not execute client side scripts and thus cannot find these links. In any of these scenarios, we can manually insert a missing web site or a missing web page in search results before they are displayed to the user.

3. Methodology

To address the aforementioned problem scenarios, we can use an expert system to enhance the search engine performance. The expert system can analyze and modify user queries before they are submitted to the search engine. It can also analyze and modify search results before they are returned to the user.

The expert system needs to be implemented between the end user and Google’s web site. Since all queries are channeled through CSUS web site, this is a logical place to implement the expert system (Figure 1).

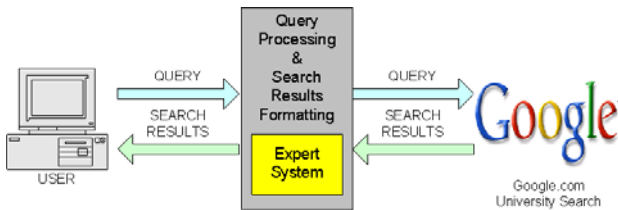


Figure 1. An expert system solution to enhance search engine results.

Google provides a set of APIs [7] to allow developers to programmatically access its search engine. Using Google APIs, it is easy to modify users’ queries before they are sent to the search engine, and it is also easy to modify search results that are returned in XML format. The APIs would be a perfect way to add an expert system between the user and the search engine, if it weren’t for a few important limitations. The APIs are still in “beta” phase, and at the time of this writing, they have been in “beta” for several years. The main limitation of the “beta” release is that the number of queries per day is limited to 1000 and there is no way to increase this limit. Because of this limit, and since the future of Google’s web search API is unclear, we decided not to use it for this work. Instead, we chose to use simple http requests. A query to Google is easy to send using query string parameters, and even advanced syntax can be included. Search response is received in HTML format. Parsing and modifying HTML are not hard, but there is a danger that search results

formatting can change which will cause HTML parsing algorithms to fail. XML schema of Google search results is not going to change easily, and this is where Google API would be very useful because it can retrieve results in XML format. Without Google API, we had to take a risk at parsing HTML and make sure the program fails gracefully if HTML format changes.

JESS [8] is a tool for building expert systems. Expert systems [1,2,9] have sets of rules that can be applied to analyze information provided by the user and return some useful information or recommend a course of actions. In this case, the expert system will be developed with knowledge (in the form of rules) about CSUS searches, and it will provide either automatic fixes to problems or suggestions to users on how to fix problems.

JESS was originally conceived as a Java clone of CLIPS [10], another expert system building tool, but nowadays has many features that differentiate it from its parent. CLIPS shell was used in this work to build and validate rules before they were implemented in JESS.

Java Studio Creator [11] is an integrated development environment from Sun Corporation. It includes JavaServer Faces [12] technology that simplifies building of user interfaces for web applications. Web applications can be quickly assembled from reusable UI components by dragging and dropping components in a page; connecting these components to an application data source; and wiring client-generated events to server-side event handlers. Using these technologies, an interface was built to process user queries and search results using an expert system implemented in JESS.

4. Search Engine Enhancement Techniques

Search engine enhancements implemented in this work can be divided into three main categories: (1) *Query modification*, a method of rewriting a user’s query using expert system knowledge before the query is sent to the search engine. (2) *Query suggestion*, a method of providing instructions or tips to users based on their query input. (3) *Inserting key results*, a method of providing additional search results items that are not found by the search engine. These methods are illustrated in Figure 2.

4.1. Query modification

Changing a user’s query behind the scenes is the most intrusive of all methods for enhancing search engine performance. This method can be useful, but the practice is rare because it can lead to usability problems. A query that is revised behind the scenes may cause user confusion, which can be counter-productive. This is why web search engines do not employ this method on a global Internet scale. However, in a smaller controlled environment, the method can be effective in fixing small user query errors provided that usability is not severely affected.

For example, the following JESS rule fixes a user’s

query “web ct” as described earlier:

```
(defrule webct
  (phase QueryModifications)
  ?q <- (query (text $?before
    "web" "ct" $?after))
  =>
  (modify ?q (text $?before webct $?after)))
```

Query modification can also be used to fix spelling errors. Even though spelling suggestions are provided by most modern search engines, they will not fix those errors automatically to avoid problems with user experience. An incorrect spelling fix by the search engine can cause a lot of user dissatisfaction, especially if users are not able to run their query at all because the engine keeps converting it to what it thinks is the correct spelling.

Query modification is most useful for fixing query syntax errors. Expert system rules can automatically remove spaces between prefixes (“+”, “-”, site:) and query terms.



Figure 2. Three methods for enhancing search engine results.

4.2. Query suggestion

Changing a user’s query behind the scenes can be useful but it can also lead to user confusion. To avoid usability problems, we can simply provide suggestions instead of forcefully fixing user’s queries. Providing suggestions is the least intrusive of all methods for enhancing search results, and it is still very effective. A simple and clear message displayed directly between search box and search results communicates directly to a user about what may be wrong with a query and how it can be fixed.

Suggestions can be given in many different scenarios. Instead of fixing users’ spelling errors, most search engines suggest a spelling correction that can be clicked to immediately re-run the corrected search. Google search engine provides this help automatically for most standard words. However, Google will conservatively suggest spelling corrections. If a misspelled word appears on many pages, Google will conclude that it may be a valid word that does not need intervention. For example, the misspelled word “calender” occurs often enough on csus.edu web pages that Google considers it a legitimate word. Using the query suggestion method we can provide

suggestion for the query “calender” using the following expert system rule:

```
(defrule calender
  (phase QuerySuggestions)
  (query (text $?before "calender" $?after))
  ?t <- (tip (text empty))
  =>
  (modify ?t (text Did you mean:
    $?before calendar $?after)))
```

Query suggestion can also be used for suggesting fixes for query syntax errors, rather than correcting errors behind the scenes. However, suggestions are most often used for synonyms. For example, if a user searches for “jobs”, query suggestion can remind them to also try “careers”.

4.3. Inserting key results

Rather than modifying or suggesting query enhancements, this method inserts key search results directly at the top of the search results list. A user query is not affected and no additional user action is required to see the added results.

For example, a link to “Career Center” can be inserted for a synonymous query “jobs” which otherwise would not find the “Career Center” site.

```
(defrule career-center
  (phase KeyResults)
  (query (text $? "jobs" $?))
  ?k <- (keymatch (url empty)
    (title empty) (description empty))
  =>
  (modify ?k
    (url http://www.csus.edu/careercenter/)
    (title CSUS - Career Center)
    (description The Career Center provides
      career services to students through
      career development, experiential
      learning, and employer relations.)))
```

Synonyms do not have to be regular dictionary words. They can be domain-specific key words that are easily mistaken and used in the way not intended by their creators. An example from the *csus.edu* domain is the query “casper” which fails to retrieve a web site titled “CasperWeb”. A key result would be useful to make sure “CasperWeb” site shows up when users search for “casper”.

Key matches are especially useful for web sites that are excluded from search results because they are not part of a localized search. In our example, any site that is not part of the domain *csus.edu* will never show up in search results. However, there are external sites that should be included, for example, the “State Horner” newspaper web site that has its own domain *statehorner.com*. This site cannot be included in Google’s University search, the only way to add it is through a method like key result insertion.

Key matches are also useful for internal web sites that cannot be indexed by the search engine for various reasons described in Section 2. Web site owners who notice that

their web sites are not included in search results usually report such issues. Search administrators can then identify the cause, suggest a fix to the web site owner, or solve their problem by creating a key result.

4.4. Other methods

In addition to the three common methods for enhancing search engine results, there are other methods which are not as common. One of them is rewriting or expanding search engine query behind the scenes, but hiding extra processing from users. For example, the query “registration” can be expanded to include related terms such as “admissions” and “application”. The rationale of this expansion is that search results will include information that is more useful. By hiding the expansion from users, there are no usability issues except that users may wonder why additional terms that they did not search for are showing up in search results. However, even though we hide it from users, the expansion is intrusive and can lead to unexpected results.

5. Performance Evaluation

In Section 2 we discussed how a number of “next-page” queries could be used to find queries that return less-than-satisfactory results. Can clicking the “Next” button to retrieve the next page of search results really be called unsatisfactory?

User surveys indicate that users expect the search engine to return what they are looking for at the top of the search results list [15,5]. Most users do not like to browse through more than one page of search results, and some do not even like to scroll down the page to look at all 10 results on the first page. Expecting the first few results to provide a perfect match in every scenario is very unrealistic. However, users have learned to expect this because of their general experience with modern search engines that perform very well in most search cases.

Therefore, the number of next-page searches is a valid indicator of user dissatisfaction. Measuring the number of next-page searches before and after search enhancements implementation gives us good indication of performance improvements. This metrics should be done in a real-world environment with real users.

We created a sample expert system in JESS with 25 rules fixing 25 most frequent problems. Each rule was tested using a Java application that modified search queries and search results based on expert system output as described in Section 4. Based on our evaluation, the expert system rules would eliminate next-page searches in these 25 test cases.

We can assess the performance improvements by calculating how many searches an expert system of a given size would potentially fix. The most recent CSUS web

server logs (1.78 million searches during the period from May 2005 and April 2006) show that among the top 100 queries run every day, about 8% are “next-page” queries. This gives us a large number of queries we can potentially improve. We want to focus on areas where we will make the most impact, so we limited our analysis to queries run 50 or more times in a year. We sorted this list by percentage of “next-page” clicks and calculated how many “next-page” queries we can fix with an expert system with 100, 200 and 300 rules. By implementing an expert-system with 100 rules, we can eliminate about 7.9% of problem queries. Unfortunately, this relationship is not linear because we selected 100 queries with the most impact. Adding 100 more rules will result in smaller improvements as shown in Figure 3.

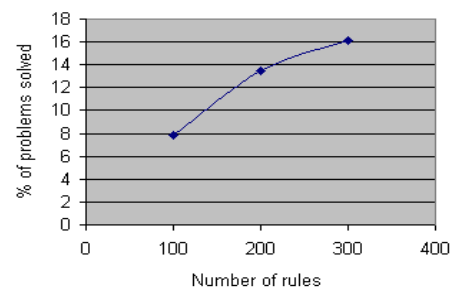


Figure 3. Expert system size and its effectiveness.

Another technique for quantifying user satisfaction and search engine effectiveness is measuring the time needed to find a specific result. If the page we are looking for is on the second or third page of search results, it takes time to load those pages from a web server, and it takes more time to browse through results and identify the one we are looking for. Web server statistics provide information on how long users spend looking at web pages. For search results pages, this time should be as short as possible. Implementing search engine enhancements should show a measurable time decrease. This time is usually measured in seconds; however, if we multiply that by the number of searches run and the number of users who are having the same kind of issues, the collective loss in productivity can be quite big. This measure can be used as a justification or return-on-investment for implementing search engine enhancements for enterprise web sites.

There are other tools such as email feedbacks and user surveys for performance assessment. All of these methods need to be used in a real-world environment to get a true measure of performance improvement. Statistics gathered before and after an expert system implementation should show measurable improvement.

6. Conclusion

The discussion of various problems in localized search

engine implementations shows that even the best search engines of the Web can have less than satisfactory performance in localized implementations. This is where human expert knowledge can make a lot of difference. Our work demonstrated how an expert system can be built to compile expert knowledge and apply it to fixing search engine issues in localized environments.

Examples are provided to demonstrate the performance improvement of search results through the use of an expert system. We also made several suggestions on how to measure the improvements of search result quality, including an estimation of problems resolved as a function of the rule base size. However, the level of actual performance improvements would be a domain-specific measure and would have to be gauged with regard to different real-world enterprise environments.

Most commercial enterprise search engine products described in Section 1 support synonyms and key results. Some provide spelling suggestions. However, they do not support processing and modifying a user's query before it is sent to the search engine. Neither do they allow complex processing of a user's query like detecting incorrect query syntax and providing remedies for it. An expert system enables these advanced processing features and adds much needed value to commercial products.

An expert system provides the most benefits in circumstances where global Web search engines are used for localized searches, as in the case with *csus.edu* web site. In these scenarios, the Web search does not allow any customization for a local environment. This is where an expert system brings the most value by providing spelling suggestions, synonyms, query processing, key results, and other methods, as described in this work.

Some possible future work includes:

(1) *Web usage data analysis*. The Web is a dynamic medium that changes over time, and so do search engine catalogs and search results. Ongoing maintenance of this system is required to keep search results updated and performance satisfactory. A simple and easy to use application can be built to track user queries over time. Content experts can use this application to analyze user queries, find common problems, and provide solutions to search administrators.

(2) *Web based knowledge acquisition interface for search administrator*. Search administrators need an easy way to update rules in the JESS knowledge base. A web based tool can be built for acquiring and validating rules and putting them into production once their performance enhancement is tested and confirmed using the testing application developed in this work.

(3) *Search knowledge management tool for local content experts*. All of these tools can be combined into

one for content experts to use. They could then do all the tasks themselves: analyze search problems, determine solutions, implement solutions in a knowledge base, test solutions and implement them in production. For this process to work, content experts either need to have some adequate web technology background, or need to be provided with a very intuitive and easy to use interface so that they do not have to worry about technology behind the system and can use the application without any knowledge of how search engines and expert systems work.

References

- [1] Stuart Russell and Peter Norvig, *Artificial Intelligence, A Modern Approach*, Prentice Hall, 2003.
- [2] Joseph Giarratano and Gary Riley, *Expert Systems, Principles and Programming*, PWS Publishing, 1998
- [3] Search Engine Watch, the source for search engine marketing, <http://searchenginewatch.com/>
- [4] Search Tools for Web Sites and Intranets, <http://www.searchtools.com/>
- [5] Nielsen NetRatings, http://www.netratings.com/mktg.jsp?section=ps_mp
- [6] Google Search Appliance, an enterprise search solution, <http://www.google.com/enterprise/gsa/index.html>
- [7] Google Web Search API: Develop your own applications using Google, <http://www.google.com/apis/>
- [8] Jess, the Rule Engine for the Java Platform, <http://herzberg.ca.sandia.gov/jess/>
- [9] Expert System, article from Wiki, http://en.wikipedia.org/wiki/Expert_system
- [10] CLIPS, a tool for building expert systems, <http://www.ghg.net/clips/CLIPS.html>
- [11] Sun Java Studio Creator, integrated development environment, <http://developers.sun.com/prodtech/javatools/jscreator/>
- [12] Sun JavaServer Faces, technology for rapid web application development, <http://java.sun.com/javaee/javaserverfaces/>
- [13] Search engine math, <http://searchenginewatch.com/facts/article.php/2156021>
- [14] Search engine 'best bets' http://www.steptwo.com.au/papers/cmb_bestbets/
- [15] iProspect Search Engine User Behavior Study http://www.iprospect.com/premiumPDFs/WhitePaper_2006_SearchEngineUserBehavior.pdf