Personalizing Search Based on User Search Histories

by

Mirco Speretta

B.Sc. , Udine University, Udine, Italy 2000

Submitted to the Department of Electrical Engineering and Computer Science and the Faculty of the Graduate School of the University of Kansas in partial fulfillment of the requirements for the degree of Master of Science in Computer Science.

____

Dr. Susan Gauch, Chairperson

____

Dr. Arvin Agah, Commitee Member

____

Dr. Perry Alexander, Commitee Member

____

Date Project Accepted

Abstract

User profiles, descriptions of user interests, can be used by search engines to provide personalized search results. Many approaches to creating user profiles collect user information through proxy servers (to capture browsing histories) or desktop bots (to capture activities on a personal computer). Both these techniques require participation of the user to install the proxy server or the bot. In this study, we explore the use of a less-invasive means of gathering user information for personalized search. In particular, we build user profiles based on activity at the search site itself and study the use of these profiles to provide personalized search results. By implementing a wrapper around the Google search engine, we were able to collect information about individual user search activities. In particular, we collected the queries for which at least one search result was examined, and the snippets (titles and summaries) for each examined result.

User profiles were created by classifying the collected information (queries or snippets) into concepts in a reference concept hierarchy. These profiles were then used to re-rank the search results and the rank-order of the user-examined results before and after re-ranking were compared. Our study found that user profiles based on queries were as effective as those based on snippets. We also found that our personalized re-ranking resulted in a 34% improvement in the rank-order of the user-selected results.

Acknowledgments

Developing this project has been a challenging and remarkable experience; for this and for introducing me to the field of information retrieval I would like to deeply thank Dr. Susan Gauch. She is an insightful professor as well as a very supportive and patient advisor. I have learned more from the interesting conversations we have had than from the many classes that I have taken.

I would like to thank Dr. Arvin Agah and Dr. Perry Alexander for serving as members of my thesis committee.

I am also very grateful to my wife, Janet, for putting up with me all the time and for her constant encouragement.

A special thanks goes to my friend Vishnu Challam whose comments helped me to improve the quality of this work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Companies that provide marketing data report that search engines are used more and more as referrals to web sites, rather than direct navigation via hyperlinks [30]. As search engines perform a larger role in commercial applications, the desire to increase their effectiveness grows. However, search engines order their results based on the small amount of information available in the user's queries and by web site popularity, rather than individual user interests. Thus, all users see the same results for the same query, even if they have wildly different interests and backgrounds. To address this issue, interest in personalized search had grown in the last several years, and user profile construction is an important component of any personalization system. Explicit customization has been widely used to personalize the look and content of many web sites, but we concentrate on personalized search approaches that focus on implicitly building and exploiting user profiles.

Another issue facing search engines is that natural language queries are inherently ambiguous. For example, consider a user issuing the query ″canon book.″ Due to the ambiguity of the query terms, we will obtain results that are either related to religion or photography. According to an analysis of their log file data conducted by OneStat.com [24] over a 2 month period of time, the most common query length submitted to a search engine (32.6%) was only two words long and 77.2% of all queries were three words long or less. These short queries are often ambiguous, providing little information to a search engine on which to base its selection of the most relevant Web pages among millions.

A user profile that represents the interests of a specific user can be used to supplement information about the search that, currently, is represented only by the query itself. This information could be used to narrow down the number of topics considered when retrieving the results, increasing the likelihood of including the most interesting results from the user′s perspective. For the user in our example, if we knew that she had a strong interest in photography but little or none in religion, the photography-related results could be preferentially presented to the user.

Many approaches create user profiles by capturing browsing histories through proxy servers or desktop activities through the installation of bots on a personal computer. These require the participation of the user in order to install the proxy server or the bot. In this study, we explore the use of a less-invasive means of gathering user information for personalized search. Our goal is to show that user profiles can be implicitly created out of short phrases such as queries and snippets collected by the search engine itself. We demonstrate that profiles created from this information can be used to identify, and promote, relevant results for individual users.

1.2 Current Technologies and Problems

In general, personalization can be applied to search in two different ways:

1. by providing tools that help users organizing their own past searches, preferences, and visited URLs;
2. by creating and maintaining sets of user's interests, stored in profiles, that can be used by retrieval process of a search engine to provide better results.

The first approach is applied by many new toolbars and browser add-ons. The Seruku Toolbar [28] and the SurfSaver [3] are examples of tools that try to help users to organize their search histories in a repository of URLs and web pages visited. Furl [11] is another personalization tool that stores web pages including topics which users are interested in, however it was developed as a server-side technology rather than a desktop toolbar.

Recently, search engines have been improved with personalization features. One of the most innovative is a9 [1], recently launched by amazon.com [2] . Users are identified through a login + cookie technology. All queries submitted can be viewed, organized and reused in future searches. Submitted queries are also used to do a full text search on the books available at amazon.com [2] to locate and suggest the best books related to the query topic. ujiko.com [32] is also a new interesting search engine that identifies users through cookies and has an appealing interface that allows users

- to give explicit judgments about specific results;
- to store submitted queries;
- to organize browsed results;
- to be helped in "refining" their searches augmenting queries with special terms suggested.

All these systems have interesting features that can guide users to find better information but they represent the user with overall profile rather than trying to identify simple specific topics of interest. Our study focuses on personalization in search based on implicit feedback. Many implicit feedback systems captures browsing histories through proxy servers or desktop activities through the installation of bots on a personal computer. These technologies require the direct participation of the user in order to install the proxy server or the bot.

In this study, we explore the use of cookies as non-invasive means of gathering user information for personalized search. Desktop bots can capture all activity whereas proxy servers can capture all Web activity. In contrast, cookies can only capture the activity at one specific site, the one that issues the cookie. Our goal is to show that user profiles can be implicitly created out of the limited amount of information available to the search engine itself; the queries submitted and snippets of user-selected results. We demonstrate that profiles created from this information can be used to identify, and promote, relevant results for individual users.

1.3 Overview

Our approach builds user profiles based on the user's interactions with a particular search engine. Among all search engines available, we decided to adopt Google [13] for the following reasons:

- it maintains one of the biggest collections of web pages;
- it provides a special APIs (Google APIs [12]) that allows users to write programs that submits queries to Google using a web service based on the SOAP protocol [35]. The results retrieved are returned in a structured XML file that can be easily processed;

- it is very popular, so users feel comfortable using it via a new interface rather than relying on a completely different search engine altogether.

For our system, we implemented GoogleWrapper: a wrapper around the Google search engine [13] that logs the queries, search results, and clicks on a per user basis. This information was then used to create user profiles and these profiles were used in a controlled study to determine their effectiveness for providing personalized search results. In order to capture unbiased data, Google's results were randomized before presentation to the user. The study was conducted in three phases:

1. Phase 1. Collecting information from users. All searches for which at least one of the results was clicked were logged per user;

2. Phase 2. Creation of user profiles; two different sources of information were used for this purpose: all queries submitted for which at least one of the results was visited; and all user-selected snippets (along with corresponding title.) Thus, two profiles were created one from queries and another from snippets;

3. Phase 3. Evaluation. The profiles created were used to calculate a new rank for the search results provided by Google. The average of this rank was compared with Google's rank.

1.4 Privacy

In general, in order to provide personalized search, the system needs some information from which to build a profile whether it be allocated by the server or by a client-side bot. A commercial server-side approach could store just the profile rather than the raw data. However, since we need to run and evaluate a variety of algorithms, we stored data for the duration of the experiment. This raises several privacy issues. First, how securely was the data protected from hacking and second, do users want to share their data at all.

To address the first issue, users were identified using an alphanumeric ID stored in a cookie. No data on personal identity was exchanged except during the initial registration process. This information was stored separately in order to reset a cookie in case it was lost. The log files were stored in a directory that was not world accessible. The log with queries and snippets was separated from the file maintaining the identities of users. The mapping between the two files was created by means of IDs.

To address the second concern, we made GoogleWrapper available to volunteers only and clearly described what data was collected and how it was used.

Chapter 2

Background

2.1 Personalization

Personalization is the process of presenting the right information to the right user at the right moment. In order to learn about a user, systems must collect personal information, analyze it, and store the results of the analysis in a user profile. Information can be collected from users in two ways: explicitly, for example asking for feedback such as preferences or ratings; or implicitly, for example observing user behaviors such as the time spent reading an on-line document.

Commercial systems tend to focus on personalized search using an explicitly defined profile. In Google's beta version [14], for example, users are asked to select the categories of topics which they are interested in and the search engine applies this information during the retrieval process.

Explicit construction of user profiles has some drawbacks. The users may provide inconsistent or incorrect information, the profile is static whereas the user's interests may change over time, and the construction of the profile places a burden on the user that she may not wish to accept. On the other hand, implicitly created user profiles do not place any burden on the user and they provide an unbiased way to collect information. Thus, many research efforts are underway to implicitly create accurate user profiles [6] [9] [10] [25].

User profiles can also be divided in other two groups: the ones representing user's preferences (e.g., search engines preferred, types of documents) and the ones representing user's interests (e.g., sports, photography.)

We focus our attention on maintaining user's interests. To achieve effective personalization, these profiles should be able to distinguish between long-term and short-term interests. They should also include a model of the user's context, e.g., the task in which the user is currently engaged and the environment in which they are situated [22]. Several systems have attempted to provide personalized search based upon user profiles that capture one or more of these aspects and, in the following sections, we discuss some examples.

A classification of interest-based user profiles is provided by Kuflik and Shoval [21]. In their study, they consider different ways of creating and maintaining user profiles, however only long-term interests profile are considered. In general they split the definition of user profiles into 3 classes:

1. *content-based profiles* (i.e. instance using vector of terms);
2. *collaborative profiles* (i.e. grouping users who use similar patterns);
3. *rule-based profiles* in which rules are created from answers provided by users on questions about information usage and filtering behavior.

In their future work, they plan to compare the various approaches discussed in the paper.They also plan to identify how much data is necessary to tune a profile for effective information filtering.

2.1.1 Profiles Based on User's Preferences

Van Gils and Schabell [7] discuss the application of user profiles representing user's preferences rather than user's interests. The type of profile they investigate is explicitly defined by the user and captures the way results are retrieved and displayed. Users express their preferences using specific criteria defined by the authors. The criteria provided are grouped into

- *representation format*: pdf, HTML or Web services;
- *structural format*: abstracts or complete documents;
- *relevance*: resources are relevant to a given query when they meet the criteria specified by the user

A profile is a list of preferences, selected by the user and maintained in an XML file.

These profiles can be applied to two areas of the retrieval process:

1. post-processing the results retrieved, converting resources into the specified format;
2. verifying that the search engine performs the tasks selected by the user.

All profiles can be stored in a repository and be used by various search engines that can interpret the structure of the XML file. Vimes is a meta-search engine that applies this mechanism. Its main component is a "broker", responsible to transform resources into the format specified by users, to interact with search engines, to retrieve results and to be able to interact with many user profiles maintained in either local or remote repositories. At the time the paper was written, this system was still under development so no results about performance were provided.

Another example of user profile, based on on user's preference, is provided by the Personal Search Assistant [26], a meta-search engine that can run as a background process on the user's machine. The application can retrieve results immediately after a query has been submitted or can keep searching for a specified amount of time before showing the results to the user. The agent running in the background should help users to reduce the amount of time spent on a search. In this case, the profile is supplied to an agent that can automatically gather information on behalf of the user.

A user can view the results retrieved and manually select the ones to be stored into the database. Stored results can be viewed at any time and applied to successive searches. A personal agent creates and updates user profiles. The user profile is built throughout 2 phases:

1. before submitting the first query, the user must select some preferences such as preferred search engines, a rank between the search engines selected, the time to spend for each search, etc.;

2. the behavior of the user, browsing the results from the repository, is observed by the agent that tries to identify patterns to be stored into the *conceptual database*.

In this example, the burden of work seems to lie with the user. Thus we believe that the approach of creating user profiles implicitly should be adopted as much as possible.

2.1.2 Profiles Based on User's Interests

Interests-based profiles are more ambitious than preferences-based profile, because they try to extract from documents topics and subjects that match user's needs. User browsing histories are the most frequently used source of information to create interest profiles. An example of this approach is the OBIWAN project [27] that focuses on interactive, personalized search. In this system, user profiles are implicitly created based on browsing histories rather than explicitly created from user input. These user profiles are represented as weighted concept hierarchies where the concepts are defined by a reference ontology. Search results from a conventional search engine are then classified with respect to the same reference ontology based upon the snippets summarizing the retrieved documents. Documents are re-ranked based upon how well their concepts match those that appear highly weighted in the user profile.

Server side techniques are also applied to create interests-based profiles. Kim and Chan [20] build user profiles from server access logs. Unlike the OBIWAN project [27], which is based on a classification of known concepts, the user interest hierarchy is created by applying a clustering method. One of their main assumptions that we also adopted in our study, is that the action of visiting a document indicates an interest in the content of the page. Web pages are first collected by crawlers and proxies and then grouped by the user. All groups of documents are pre-processed through a list of stop words and then stemmed. Each pair of words is analyzed to find possible relationships that are ranked using a weight value. All words are then subdivided into groups (clusters) of related words. Various approaches such as Augmented Expected Mutual Information (AEMI), thresholds, window size (that has the advantage of keeping the distance between words) are investigated to identify correlated words. Once the cluster structure has been defined, all documents in the original group are associated with the corresponding clusters.

Internal nodes, which hold more words than leaves do, represent long-term interests while leaves describe a more specific interest that has a temporary value (short-term interest).

The accuracy of the cluster was considered as a measure of the effectiveness of the algorithm. Each cluster was manually reviewed and classified as *good*, *fair* and *low* depending on meaningfulness of all its words. The best result (61% of *good* clusters). was obtained using the *MaxChildren* method: the threshold selected is such that the maximum of child clusters is generated. Contrasting results were found experimenting with the size of the window.

In this study, they also address the problem of identifying long-term interests versus short-term interests. All terms listed in a given cluster are also present in the cluster's parent. This establish a relation between parents and siblings and also distinguish between long-term versus short-term interests.

Another interesting study was conducted by Soltysiak and CrabTree [29]. They also tried to classify user's interests automatically and, similarly to Kim and Chan [20], the hierarchy of user's interests is created using a clustering approach. The set of interests represents a user profile which can be used to automatically search for information, for filtering or to personalize the way of showing information. The first problem they tackled was to find the best approach to extract a document description that is as accurate as possible. The analysis is based on documents collected from web pages visited and emails received or sent. Each document is identified by a vector of keywords represented by their $tf \times idf$ value.

The learning mechanism used to create user profiles is divided into 2 phases:

- the *primary learner* collected documents, extracted keywords, and applied a clustering algorithm to these documents in order to produce interest clusters;

- the *secondary learner* classified the clusters from the previous phase into a predefined interests (clusters).

The final set of clusters represents a user profile. Users were asked to give a feedback on the output of the first phase. As time progressed the profile become more accurate and the user's feedback decreases its influence in the cluster calculation. Many variables were investigated in the experiments described. The most significant results were that the number of documents used to define a cluster can highly influence the profile. Using the best values for the variables investigated, they found that 70% of the new defined clusters were automatically classified.

2.2 Ontologies and Semantic Web

For our study, based on previous research work from Trajkova and Gauch [18], we decided to represent user profiles as a hierarchy of weighted concepts that are defined in a reference ontology. According to Gruber [15], an ontology is a "specification of a conceptualization." Ontologies can be defined in different ways, but they all represent a taxonomy of concepts along with the relations between them. In the context of the World Wide Web, ontologies are important because they formally define terms shared between any type of agents without ambiguity, allowing information to be processed automatically and accurately.

OntoSeek [16] is an example of an information retrieval system based on ontologies. The main assumption is that precision and recall would improve if we used "sense matching" instead of "word matching." The domains in which the system operates are catalogues of either heterogeneous or homogeneous products. The description of each product in the catalog is translated into a *lexical conceptual graph*; i.e., a tree structure where nodes are nouns from the description and arcs are concepts inferred by the corresponding nouns. All graphs, one for each product, are stored in a repository. A special user interface is provided to submit queries. When a query is issued, the user is required to disambiguate its meaning. This process is carried out by the user interface that tries to identify the concept provided and asks the user to choose between potential solutions. The main challenge in developing this system was to find an automatic way to create the *lexical conceptual graphs*. The thesaurus provided by WordNet [33] and the ontology provided by Sensus [17] were used to accomplish this task.

Ontology is often conducted in support of the Semantic Web. The expression "Semantic Web" [34] was introduced by ETAI (Electronic Transactions on Artificial Intelligence) in 2000 to describe the extension of the Web to deal with the meaning of available content rather than just its syntactic form. Many XML-based projects such as Resource Descriptor Framework (RDF) [36], Notation 3 (N3) [5], and OWL [37] contribute to this goal by defining a syntax capable of describing and/or manipulating ontologies. One of the main bottlenecks in the evolution of the Web is the amount of manual effort usually required to create, maintain, and use ontologies. Our approach shares many of the same goals as the Semantic Web, however we focus on automatic techniques, wherever possible, for conceptual extraction and matching.

2.3 Evaluation of User Profiles

Our study requires relevance feedback from users in order to construct profiles. We decided to infer user's interests adopting an implicit technique. The main advantages to implicit feedback collection over explicit methods are that users:

- are not influenced by the relevance feedback interface;

- do not have to spend time in providing personal judgments.

Although explicitly collected feedback seems to be more accurate than implicitly collected data, this problem can be addressed by collecting a larger amount of data from the users. Kelly and Teevan [19] give an interesting overview of the most popular techniques used to collect implicit feedback. The user behavior collected is represented by a 2-dimensional table in which the axes are labeled with “behavior Category” and “minimum scope.” The “behavior category” may be one of the following user actions: examine, retain, reference, annotate or create. The “minimum scope” refers the smallest possible scope of the item being acted upon, where possible item scopes are: segments of text (whitin a snippet), objects or classes (within web pages). The goal of this categorization is to create a deterministic description of the type of user action feedback systems could capture. Problems raise when actions are ambiguous and it becomes necessary to interpret the behavior. For example, the action of a user saving a new document could be considered as a “retain” rather than a “create.”

They summarized the research on implicit feedback collection, as compared to the accuracy of explicit feedback, concluding that

- the time spent on a page, the action of scrolling the page and the combination of these variables were correlated with the explicit rating; while the mouse clicks seem not to have a correlation with explicit rating;

- no significant correlation was found between length of document and reading time or document readability and reading time;
- there seems to be a relation between reading time and user interest but it is very difficult to find a threshold values that eliminate spurious reading time;
- contextual search gave encouraging improvements in search result quality;
- link analysis, pioneered by Google, seems to be very effective.

Chapter 3

Approach and System Architecture

3.1 Approach

Our study investigates the effectiveness of personalized search based upon user profiles constructed from user search histories. GoogleWrapper is used to monitor user activities on the search site itself. Individual user information such as queries submitted, results returned (titles and snippets), and Web pages selected from results retrieved is collected. This per-user information is classified into a concept hierarchy based upon the Open Directory Project [23], producing conceptual user profiles. Search results are also classified into the same concept hierarchy, and the match between the user profile concepts and result concepts are used to re-rank the search results.

We believe this approach has several advantages. User interests are collected in a completely non-invasive way and search personalization is based upon data readily available to the search engine. Unlike other approaches, we do not require the user to install a bot or use a proxy server to collect and share their browsing histories. Finally, the system effectiveness can be evaluated by monitoring user activities rather than requiring explicit judgments or feedback.

3.2 System Architecture

The architecture of our system consists of three modules:

1. GoogleWrapper: a wrapper for Google that implicitly collects information from users. Google APIs [12] and the nusoap library [4] were used for the implementation. Users register with their email addresses in order to create a cookie that stores their userID on their local machines. If the cookie is lost, GoogleWrapper notifies the user and they can login to reset the cookie.

   When queries are submitted by users, GoogleWrapper forwards the query to the Google search engine. It intercepts the search engine results, logs them along with the query and the userID, re-ranks them, and then displays them to the user. When users click on a result, the system logs the selected document along with the user ID before redirecting the browser to the appropriate Web page.

2. The classifier from KeyConcept [27], a conceptual search engine, is used

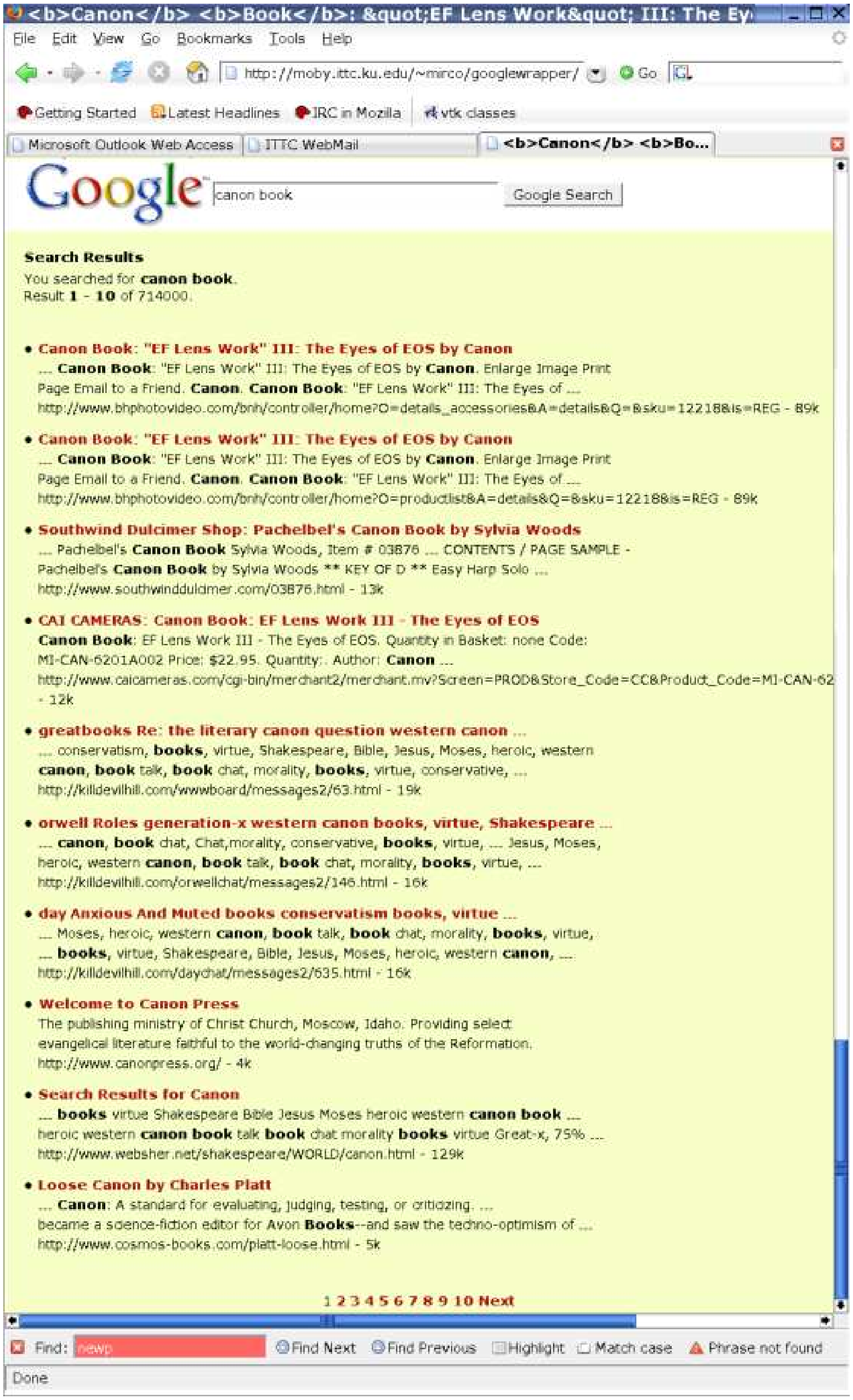
to classify queries and snippets for each user, as well as the search engine results. This KeyConcept classifier is based on the vector-space model.

3. A set of scripts that processes the log files and evaluates the per-user and overall performance. The log file is split between users and, for each user, further divided into training and testing sets.

3.3 User Profiles

User profiles are represented as a weighted concept hierarchy. The concepts hierarchy is created from 1,869 concepts selected from the top three levels of the Open Directory Project, and the weights represent the amount of user interest in the concept. The concept weights are assigned by classifying textual content collected from the user into the appropriate concepts using a vector space classifier. The weights assigned by the classifier are accumulated over the text submitted.

In earlier work [31], we constructed user profiles from Web pages browsed by the user. However, this study focused on using the user's search history rather than their browsing history, information more easily available to search engines. We evaluate the effectiveness of profiles built from user queries with those built from snippets (titles plus the textual summaries) of user-selected results.


<b>Canon</b> <b>Book</b>: &quot;EF Lens Work&quot; III: The Ey
File Edit View Go Bookmarks Tools Help
http://moby.ittc.ku.edu/~mirco/googlewrapper/
Go
Getting Started
Latest Headlines
IRC in Mozilla
vtk classes
Microsoft Outlook Web Access
ITTC WebMail
<b>Canon</b> <b>Bo...
Google
canon book
Google Search

Search Results

You searched for **canon book**.
Result **1** - **10** of 714000.

- **Canon Book: "EF Lens Work" III: The Eyes of EOS by Canon**
  ... **Canon Book**: "EF Lens Work" III: The Eyes of EOS by **Canon**. Enlarge Image Print Page Email to a Friend. **Canon**. **Canon Book**: "EF Lens Work" III: The Eyes of ...
  http://www.bhphotovideo.com/bnh/controller/home?O=details_accessories&A=details&Q=&sku=12218&is=REG - 89k
- **Canon Book: "EF Lens Work" III: The Eyes of EOS by Canon**
  ... **Canon Book**: "EF Lens Work" III: The Eyes of EOS by **Canon**. Enlarge Image Print Page Email to a Friend. **Canon**. **Canon Book**: "EF Lens Work" III: The Eyes of ...
  http://www.bhphotovideo.com/bnh/controller/home?O=productlist&A=details&Q=&sku=12218&is=REG - 89k
- **Southwind Dulcimer Shop: Pachelbel's Canon Book by Sylvia Woods**
  ... Pachelbel's **Canon Book** Sylvia Woods, Item # 03876 ... CONTENTS / PAGE SAMPLE - Pachelbel's **Canon Book** by Sylvia Woods ** KEY OF D ** Easy Harp Solo ...
  http://www.southwinddulcimer.com/03876.html - 13k
- **CAI CAMERAS: Canon Book: EF Lens Work III - The Eyes of EOS**
  Canon Book: EF Lens Work III - The Eyes of EOS. Quantity in Basket: none Code: MI-CAN-6201A002 Price: $22.95. Quantity:. Author: **Canon** ...
  http://www.caicameras.com/cgi-bin/merchant2/merchant.mv?Screen=PROD&Store_Code=CC&Product_Code=MI-CAN-62 - 12k
- **greatbooks Re: the literary canon question western canon ...**
  ... conservatism, **books**, virtue, Shakespeare, Bible, Jesus, Moses, heroic, western **canon**, **book** talk, **book** chat, morality, **books**, virtue, conservative, ...
  http://killdevilhill.com/wwwboard/messages2/63.html - 19k
- **orwell Roles generation-x western canon books, virtue, Shakespeare ...**
  ... **canon**, **book** chat, Chat,morality, conservative, **books**, virtue, ... Jesus, Moses, heroic, western **canon**, **book** talk, **book** chat, morality, **books**, virtue, ...
  http://killdevilhill.com/orwellchat/messages2/146.html - 16k
- **day Anxious And Muted books conservatism books, virtue ...**
  ... Moses, heroic, western **canon**, **book** talk, **book** chat, morality, **books**, virtue, ... **books**, virtue, Shakespeare, Bible, Jesus, Moses, heroic, western **canon**, ...
  http://killdevilhill.com/daychat/messages2/635.html - 16k
- **Welcome to Canon Press**
  The publishing ministry of Christ Church, Moscow, Idaho. Providing select evangelical literature faithful to the world-changing truths of the Reformation.
  http://www.canonpress.org/ - 4k
- **Search Results for Canon**
  ... **books** virtue Shakespeare Bible Jesus Moses heroic western **canon book** ... heroic western **canon book** talk **book** chat morality **books** virtue Great-x, 75% ...
  http://www.websher.net/shakespeare/WORLD/canon.html - 129k
- **Loose Canon by Charles Platt**
  ... **Canon**: A standard for evaluating, judging, testing, or criticizing. ... became a science-fiction editor for Avon **Books**--and saw the techno-optimism of ...
  http://www.cosmos-books.com/platt-loose.html - 5k

1 2 3 4 5 6 7 8 9 10 Next

Figure 3.1: Screenshot of GoogleWrapper where the query "canon book" has been submitted

3.4 Google Wrapper

GoogleWrapper was developed in two different releases. The first version helped us identify parameter settings to be used later during evaluation. In this version, GoogleWrapper allowed users

- to submit queries to Google;
- to display results using Google rank;
- to store queries issued along with the first 10 results displayed. Each entry of the log file had the following format: (*timestamp, user_ID, query, URL, user_click, Google_rank, random_rank, title, snippet*).

Our first version of GoogleWrapper displayed all results retrieved from Google, ten results per page, presented in the original order. We used this version to collect data about how users normally interact with a search engine. After collecting 576 queries for which at least one result was selected, we randomly picked a sample set of 100 queries for detailed analysis. From this sample, we found that 94% of the user-selected results occurred in the first 3 Google-ranked results, and no result after the tenth result, i.e., on the second page, was ever selected. Figure 3.2 shows that the top-ranked result was by far the most frequently selected (60%), followed by the second (20%), and the third (14%). From these observations we concluded that users rarely, if ever, look for results beyond the first page displayed by the search system. Thus, for our later

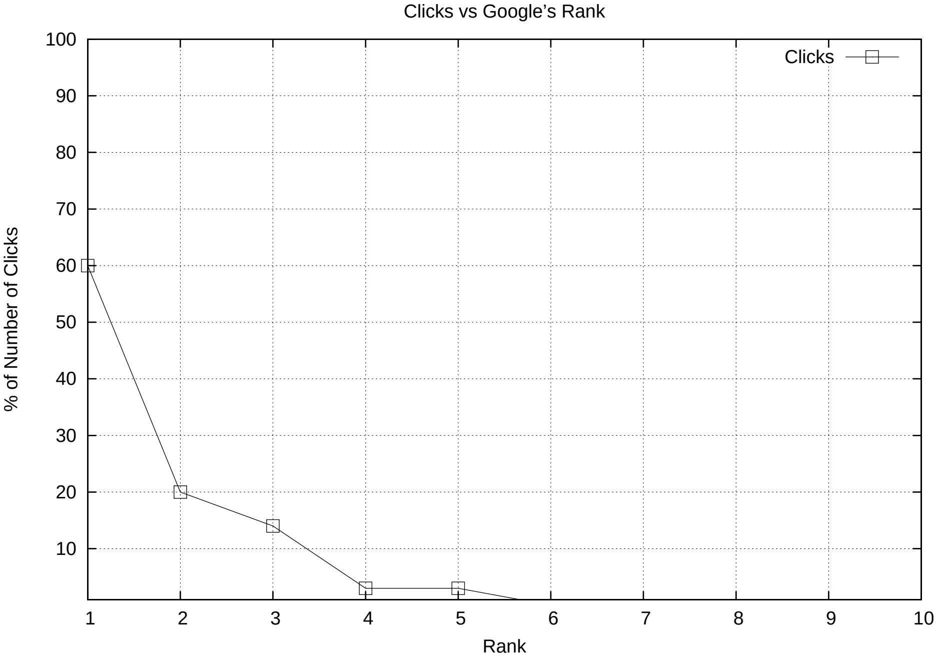


Figure 3.2: Percentage of user selections versus rank for the top 10 results from Google.

experiments, we process only the top 10 results retrieved from Google.

Our second conclusion was that users may be influenced by the rank-order of the result presentation. To verify this hypothesis, we modified GoogleWrapper so that it displayed the top ten results from Google in random order. We randomly selected another sample set of 100 queries and performed the same analysis. This time, the original rank of the user-selected results was more uniformly distributed across the 10 results displayed. Figure 3.3 shows that the top three results as ranked by Google accounted for 46% when results were presented in random order versus 94% when they were shown in the original order. Google's top result was selected only 15% of the time when it was shown

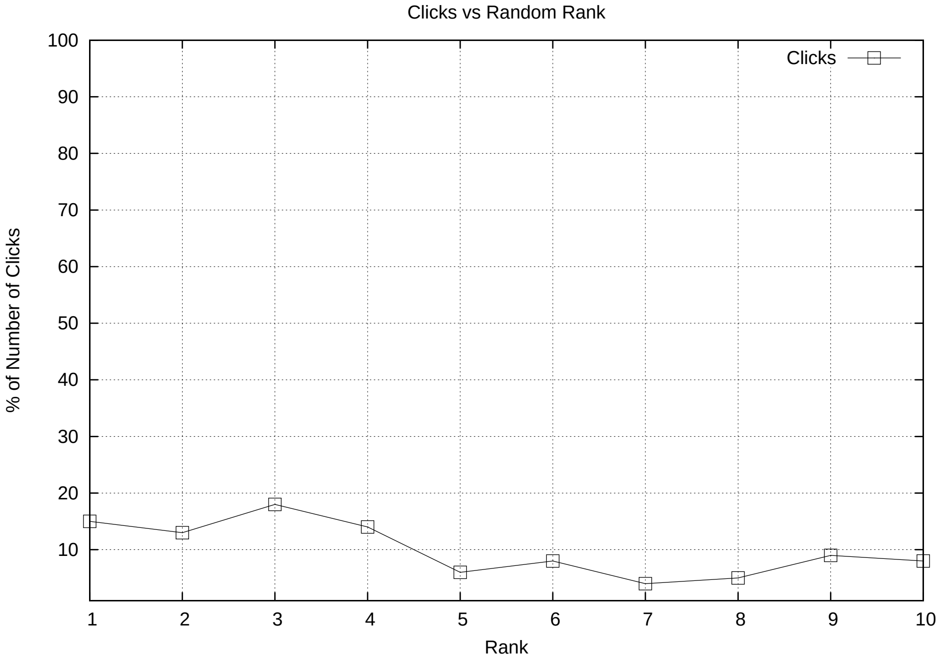


Figure 3.3: Percentage of user selections versus rank for the top 10 results from Google displayed in random order.

in a random position versus 60% when it is presented at the top of the page. From this, we concluded that user judgments are affected by presentation order, so we continued to randomize the search engine results before presenting them to the user in later experiments.

3.5 Personalized Search

The log file generated by GoogleWrapper allowed us to perform an retrospective analysis on user's selected results. The top 10 results, viewed by the user, were re-ranked using a combination of their original rank and their conceptual similarity to the user′s profile. The search result titles and summaries are classified to create a document profile in the same format as the user profile. The document profile is then compared to the user profile to calculate the conceptual similarity between each document and the user′s interests. The similarity between the document profile and the user profile is calculated using the cosine similarity function:

$$conceptual\_match(user_i, doc_j) = \sum_{k=1}^{N} cwt_{ik} \times cwt_{jk}$$

where

$cwt_{ik} =$ Weight of Concept_k in UserProfile_i

$cwt_{jk} =$ Weight of Concept_k in DocumentProfile_j

$N =$ Number of Concepts

The documents are re-ranked by their conceptual similarity to produce their conceptual rank. The final rank of the document is calculated by combining the conceptual rank with Google's original rank using the following weighting scheme:

$$FinalRank = \alpha \times ConceptualRank + (1 - \alpha) \times GoogleRank$$

where α has a value between 0 and 1. When α has a value of 0, conceptual rank is not given any weight, and it is equivalent to the original rank assigned by Google. If α has a value of 1, the search engine ranking is ignored and pure conceptual rank is considered. The conceptual and search engine based rankings can be combined in different proportions by varying the value of α.

Chapter 4

Experiments and Validation

4.1 Experiments

This chapter describes our evaluation experiments in detail. Section 4.1.1 reports on the volunteers involved in the study and the criteria adopted to divide collected data into training data (used to create the user profiles) and testing data (used to test the effectiveness of the user profiles). Section 4.1.2 explains the preliminary study based on which we selected the parameter settings applied in the later experiments.

4.1.1 Experimental Validation

GoogleWrapper was used by six volunteers for a period of almost 6 months. These users included faculty members and graduate students from different departments at the University of Kansas, i.e., Electrical Engineering and Computer Science, Mathematics, and Pharmaceutical Chemistry.

The final version of GoogleWrapper (see Section 3.4), used to collect the user information for our study of personalized search, presented the top 10 results for each query in random order. Using this system, we collected 609 queries for which at least one result was selected. We removed duplicate queries for each user and, from this collection, we selected 47 queries per user (282 total) divided into the following sets:

- 240 (40 per user) queries were used for training the 2 user profiles (query-based and snippet-based);
- 30 (5 per user) queries were used for testing personalized search parameters;
- 12 (2 per user) queries were used for validating the selected parameters.

In the following sections, we present the experiments in which we investigated the effects of user profiles built out of queries and snippets. We measured the accuracy of such profiles by comparing, for user-selected results, Google's original rank with the conceptual rank based on the profile. Because our goal was to evaluate the quality of the user profiles, not produce the best possible search results, we set α to 1 so that Google's original rank did not affect the FinalRank. Once the best conceptual match was determined, we conducted further experiments to evaluate the effect of varying α to produce a final ranking.

4.1.2 Preliminary Study

As described in Section 3.3, a user profile was created by categorizing each training query and accumulating the returned concepts and weights. One question that needed to be resolved was, since the categorizer returns an ordered list of concepts and weights, how many of these concepts per query should be used to create the profile and then update it.

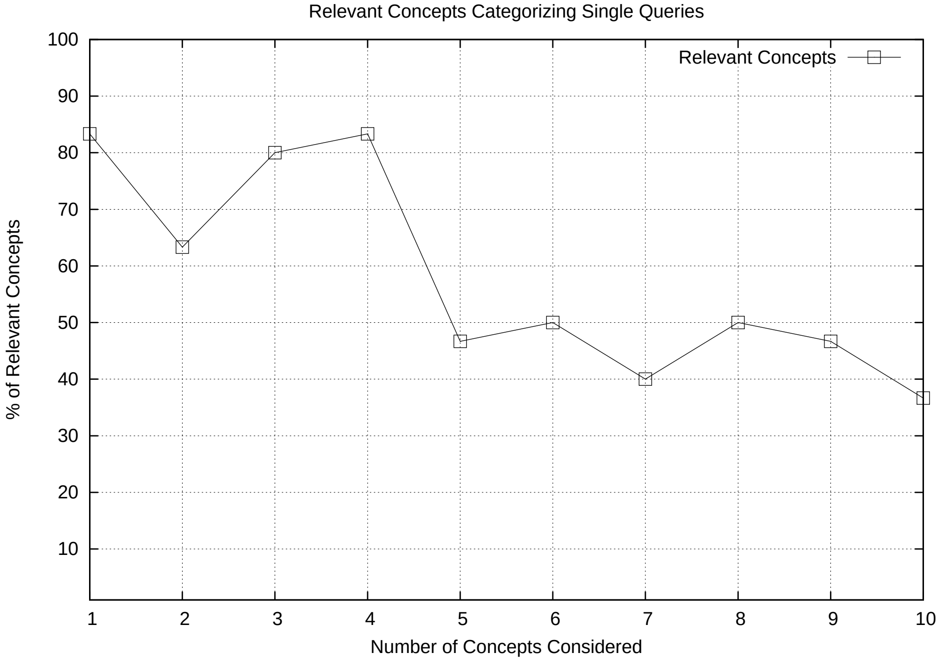


Figure 4.1: Percentage of relevant concepts versus number of categories considered per single query.

To investigate this question, we randomly selected 30 queries (5 per user) and performed a detailed analysis of the classification results. For each query, the top 10 concepts returned by the classifier were manually judged as relevant

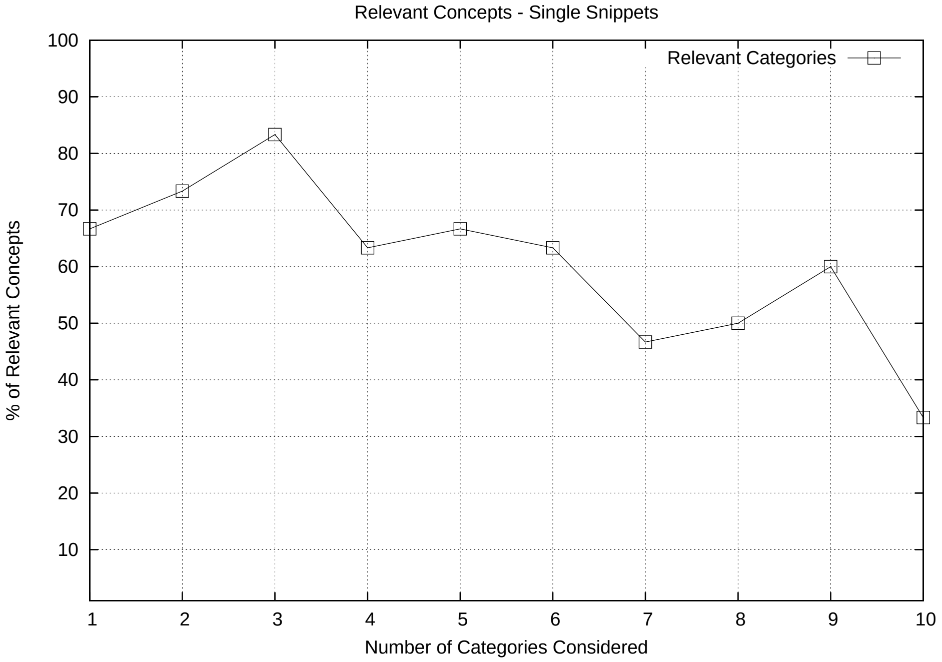


Figure 4.2: Percentage of relevant concepts versus number of categories considered per single snippet.

or not. Figure 4.1 shows that the top 4 concepts assigned per query were relevant 78% of the time, and that the accuracy dropped dramatically further down the list. Thus, in the experiments that follow, we built the query-based profiles considering only the top 4 classification results per query.

A similar analysis was conducted for profiles built out user-selected snippets (titles along with respective summaries). As with query-based profiles, we needed to determine the number of classification results to add to the user profile. Once again, we performed an analysis of the classifier accuracy on 30 randomly chosen user-selected snippets (5 per user). Since the snippets contain more text than an average query, the classifier seems to be able to identify more valid concepts per snippet. Compared to the query classification results, the accuracy does not drop as precipitously as we moved down the list of matching concepts. As shown in Figure 4.2, the top 5 classified results are accurate 71% of the time and, further down the list, the accuracy begins to steadily decrease. Based on this analysis, the snippet-based profiles used in this study were built using the top 5 concepts for each snippet returned by the classifier.

Based on these results, all the experiments reported in this study adopt user profiles with the following characteristics:

- query-based profiles are built using the top 4 concepts returned from the classifier;
- snippet-based profiles are constructed using the top 5 concepts returned from the classifier.

Comparing Figure 4.1 and Figure 4.2, we observe that the accuracy is distributed more uniformly when snippets are used. Over all the 10 concepts considered, the average accuracy is 58% when queries are classified and 61% when snippets are used. Altough the difference in accuracy is quite small, snippets are classified into more concepts than queries (174 versus 182). This increase in the number of accurate concepts is likely due to the fact that snippets contain more text than queries and thus can be classified more accurately and/or fit in more concepts.

4.1.3 Experiment 1 - Effect of training queries and conceptual rank on classifier accuracy

The first factor we investigated was the number of training queries necessary to create an accurate user profile using the query text alone. As mentioned in Section 4.1.2, we used the top 4 concepts returned by the classifier for each query. We created user profiles using training sets of 5, 10, 20, 30, and 40 queries.

# Training Queries	*Average Rank*	*Average Improvement*	*Concepts Considered*
0 (baseline)	4.4	–	–
5	3.7	16%	4
10	3.3	24%	10
20	3.1	29%	5
30	2.9	33%	4
40	3	31%	5

Table 4.1: Average rank compared between Google's original rank and conceptual rank where user's profile was built using different number of training queries.

A second factor studied was the number of concepts to use from the resulting profile when calculating the similarity between the profile and the search results. We varied this number from 1 through 20.

The document profile, as described in Section 3.5, was constructed using the top 7 concepts returned by the classifier for each search result. This value is based on earlier experiments [8] in which the same methodology was used to verify the effectiveness of user profiles applied to contextual search.

The resulting profiles were evaluated based upon the rank of the user -

selected results returned by the concept ranking algorithm alone, without any contribution from Google's original ranking. This conceptual rank was compared to Google's original rank of the user-selected results to see if there was any improvement. Table 4.1 shows the average rank comparison for different number of training queries per user. The results are averaged over 1 through 20 profile concepts. The average Google rank for the 30 testing queries (5 per user) is shown in the first row while the rows further down the list show the average conceptual rank for the same 30 testing queries when the profile is built varying the number of training queries used (5, 10, 20, 30, 40). See Appendix A for detailed charts. The third column of the table reports the percentage improvement calculated by comparing Google's original rank with the conceptual rank. The last column shows the number of concepts from the user profile used when calculating the conceptual match that provided the best improvement. From Table 4.1, we see that the best result occurs when 30 training queries per user are used to create the profile.

Figure 4.3 shows the case in which we obtained the best improvement in more detail: user profiles built using 30 training queries. The average original Google rank for the 30 testing queries is 4.4. In contrast, when 30 training queries are used to build the profile, and 4 concepts from that profile are used during conceptual match, the average conceptual rank is 2.9. Using a paired, two-tailed t-test, this improvement of 33% was found to be statistically signif-

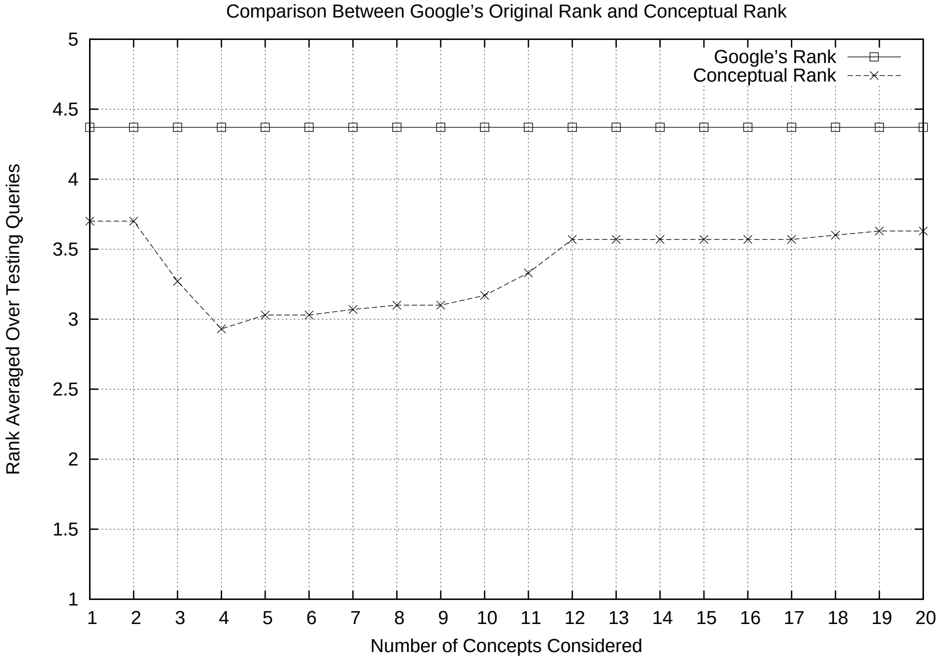


Figure 4.3: Google's original rank and conceptual rank averaged over all testing queries. User profiles are built using 30 training queries

icant (p = 0.002). We see a steady improvement in the search result ranking as concepts 2 through 4 are used from the user profile. Then, there is a gradual degradation as more concepts are used. After 12 concepts, the performance plateaus. It is worth noting that including 1 or more concepts consistency improves the rank by at least one place. We also examined the results to investigate their effect on the individual testing queries. The query-based profile ranked the selected results higher for 13 queries, hurt the ranking for 3, and left 14 unchanged. Therefore, the re-ranking process helped far more testing queries than it hurt.

4.1.4 Experiment 2 - Effect of training snippets and conceptual rank on classifier accuracy

This experiment repeats Experiment 1, the only difference being that the profiles were built using snippets rather than queries. Since the text classified is, on average, longer than queries, we expected an improvement. Once again, the conceptual rank was used for evaluation. We trained on user-selected sets of 5, 10, 20, 30, and 40 snippets per user for 6 users.

# Training Snippets	*Average Rank*	*Average Improvement*	*Concepts Considered*
0 (baseline)	4.4	–	–
5	3.2	27%	5
10	3.2	27%	9
20	3.1	29%	7
30	2.9	34%	20
40	3	31%	18

Table 4.2: Average rank compared between Google's original rank and conceptual rank where user's profile was built using different number of user-selected snippets.

Table 4.2 shows the average rank comparison for different numbers of training snippets. The results were evaluated for 1 through 20 user profile concepts used for profile. The average Google rank for the 30 testing queries (5 per user) is shown in the first row while the rows further down the list show the average conceptual rank for the same queries when the profile is built varying the number of user-selected training snippets (5, 10, 20, 30, 40). See Appendix A for detailed charts. The third column shows the percentage improvement calculated by comparing Google's original rank with the conceptual rank. The last column of the table reports the number of concepts from the user profile used when calculating the conceptual match that provided the best improvement. From Table 4.2, we observe that the best result occurs when 30 user-selected snippets per user are used to create the profile.

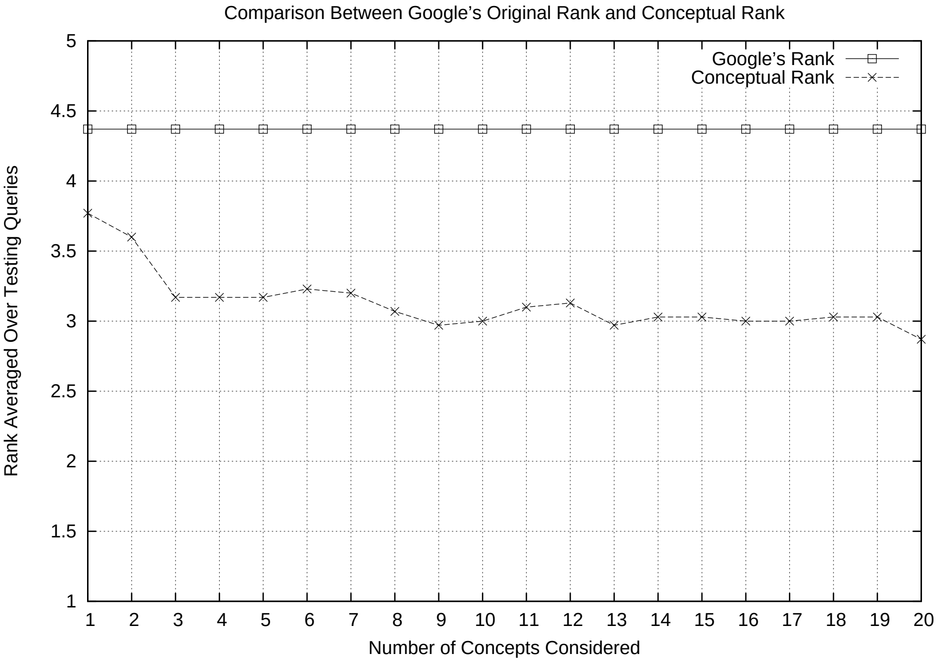


Figure 4.4: Google's original oank and oonceptual oank overaged over oll testing queries. User profiles are built using 30 training snippets

Figure 4.4 shows the case in which we obtained the best improvement in more detail: user profiles built using 30 user-selected training snippets. The rank improves quickly as the first 3 concepts are used, then a gradual increase is seen up until 9 profile concepts are used for conceptual match. After that, the curve remains flat, showing that the results are somewhat stable beyond that point. Using a paired, two-tailed t-test, the improvement of 34% was found to be statistically significant (p = 0.007). The snippet-based profile improved the ranking for 11 queries, hurt 4, and left 15 unchanged. Once again, the conceptual ranking helped far more queries than it hurt.

4.1.5 Experiment 3 - Effect of training queries and final rank on classifier accuracy

In this experiment, we wanted to see if including the original rank returned by the search engine in the calculation of the FinalRank (as described in Section 3.5) could further improve the overall results. Based on the results of Experiment 1, the query-based profile adopted in this experiment was built using 30 training queries and 4 concepts were used during conceptual ranking. The final rank for a search result was calculated by varying the value of α from 0.0 to 1.0 with a 0.1 step so as to modify the relative contributions of the conceptual and original ranks.

Figure 4.5 shows the average rank comparison between Google's original rank and the FinalRank. The best results are obtained when α is 1.0, i.e., when the original search engine ranking is ignored altogether. This is likely due to the fact that the top 10 results are all good matches for the keywords in the query and, therefore, the distinguishing feature between the results is how well they match the user's interests.

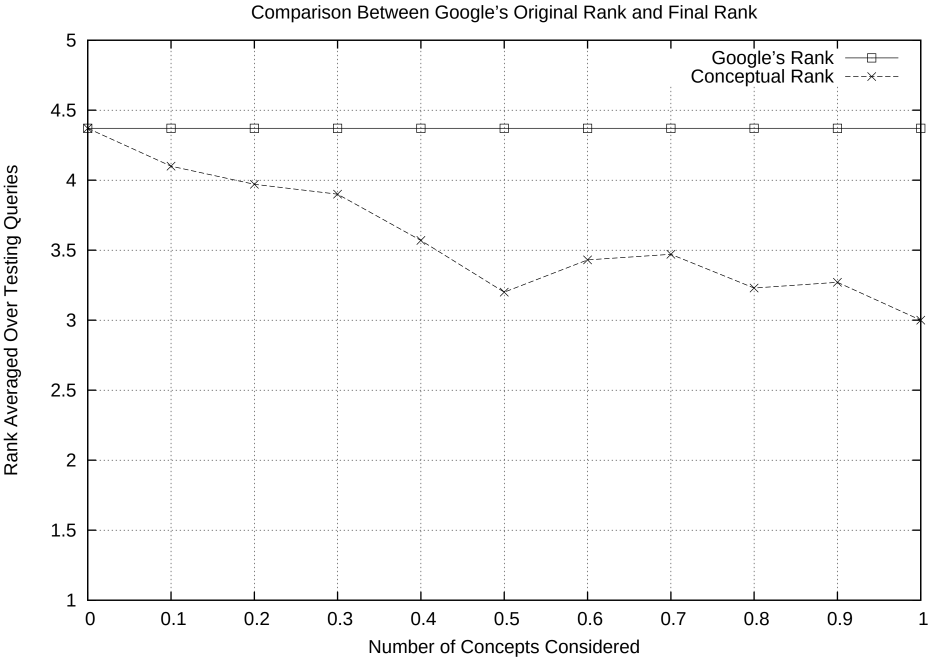


Figure 4.5: Google's original rank and final rank averaged over all testing queries. User profiles are built using just queries.

4.1.6 Experiment 4 - Effect of training snippets and final rank on classifier accuracy

Similar to Experiment 3, we examined the effect of combining the search engine's original rank with the conceptual rank when calculating the FinalRank (as described in Section 3.5) of a result. Based on the results of Experiment 2, we used a snippet-based profile built from 30 training snippets and used 20 concepts from the profile for conceptual ranking.

Fig 4.6 shows the comparison between Google's rank and the final rank as α is varied from 0.0 to 1.0 with a 0.1 step. Once again, the best results occur when

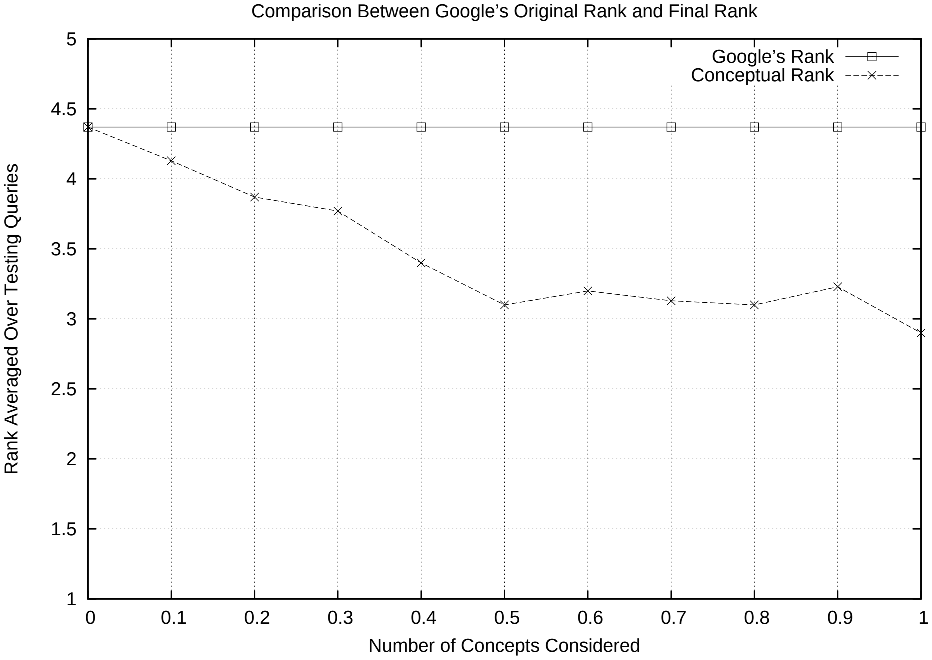


Figure 4.6: Google's original rank and final rank averaged over all testing queries. User profiles are built using snippets.

α is 1.0, i.e., when the original search engine rankings are ignored altogether.

4.1.7 Experiment 5 - Validation

To verify that the user profiles created are able to improve queries that were not used to tune the profile construction algorithms, we conducted a validation experiment with 12 new testing queries (2 per user). Based on Experiments 3 4.1.5 and 4 4.1.6, we compared Google's original rank with the conceptual rank alone.

We first calculated the conceptual rank using the query-based profile and

Ranking Based On	*Average Rank*	*Percent Improvement*
Google (Original)	4.8	–
(30) Testing Queries	2.9	33 %
(30) Testing Snippets	2.9	34 %
(12) Validation Queries	1.8	37%
(12) Validation Snippets	3.5	27%

Table 4.3: Comparison of average rank for validation queries.

compared this rank to Google's original rank. The average rank of the user-selected results was 4.8 according to Google's rank. The average value for conceptual rank was 1.8. The query-based profile produced a 37% improvement on the validation queries.

The same analysis was conducted using the snippet-based profiles. The average conceptual rank was 3.5, a 27% improvement on the validation queries. Table 4.3 summarizes these results and verifies that we see comparable improvements for the validation queries as observed for the original test queries used to tune the profile creation algorithms.

Chapter 5

Conclusions

5.1 Conclusion and Future Work

We built a system that creates user profiles based on implicitly collected information, specifically the queries submitted and snippets of user-selected results. We were able to demonstrate that information readily available to search engines is sufficient to provide significantly improved personalized rankings. We found that using a profile built from 30 queries produced an improvement of 33% in the rank of the selected result. A user profile built from snippets of 30 user-selected results showed a similar improvement of 34%. The snippet-based profile improved more queries (11 versus 10) and hurt fewer (2 versus 3), so there is some indication that it is a slightly more accurate profile.

Our best results occurred when conceptual ranking considered only four concepts from the query-based profile, and twenty from the snippet-based profile. The ranking improvements for the query-based profile seems to hold across the range of concepts between 3 and 10 while the snippet-based profile has a fairly steady improvement in the range 3 - 9.

The user profiles we used to build were based on a three-level deep concept hierarchy. We would like to examine the effect of using fewer or more levels of the ODP hierarchy as our profile representation. Also, the current concept hierarchy is static, and we would like to evaluate algorithms to dynamically adapt the hierarchy for specific users by merging and/or splitting concepts based upon the amount of user interest. Finally, we would like to combine the user profiles with the document selection process, not just the document re-ranking, to provide a wider set of relevant results to the user rather than just reorganizing the existing results.

Appendix A

Results in detail

In this appendix we show more in detail some of the results that we mentioned in Experiment 1 (Section 4.1.3) and Experiment 2 (Section 4.1.4).

Figures A.1, A.2, A.3, A.4 show the comparison between Google's average rank and conceptual rank in the case user profiles where built using training queries.

Figures A.5, A.6, A.7, A.8 show the comparison between Google's average rank and conceptual rank in the case user profiles where built using training snippets.

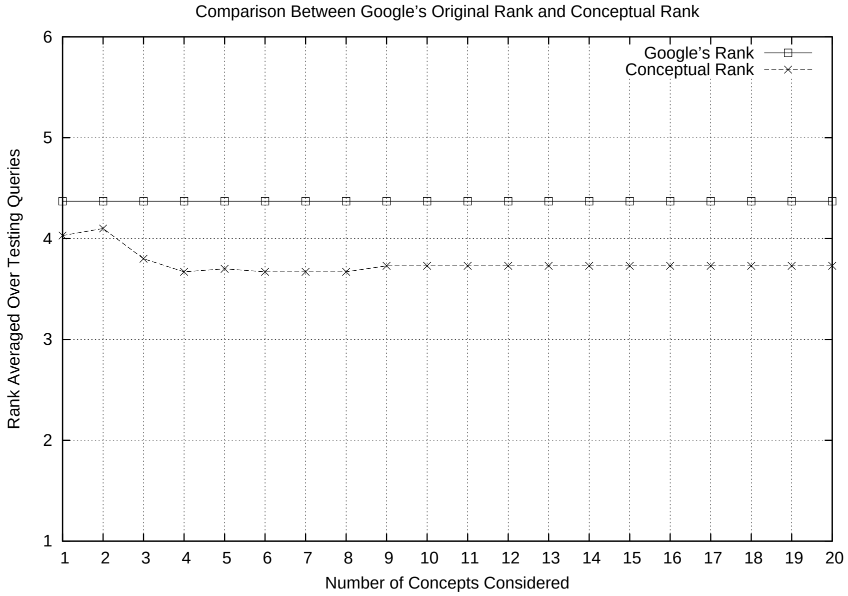


Figure A.1: Google's original rank and conceptual rank averaged over all testing queries. User profiles are built using 5 training queries

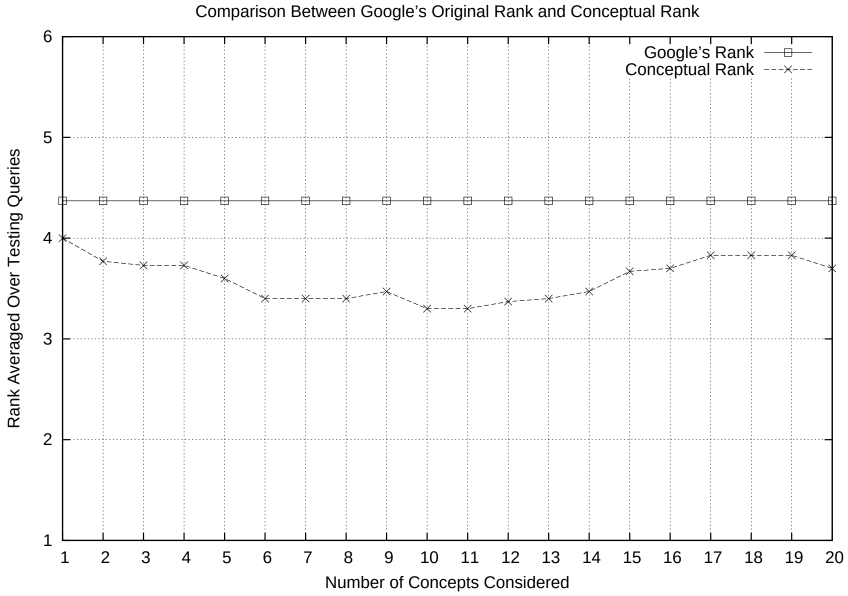


Figure A.2: Google's original rank and conceptual rank averaged over all testing queries. User profiles are built using 10 training queries

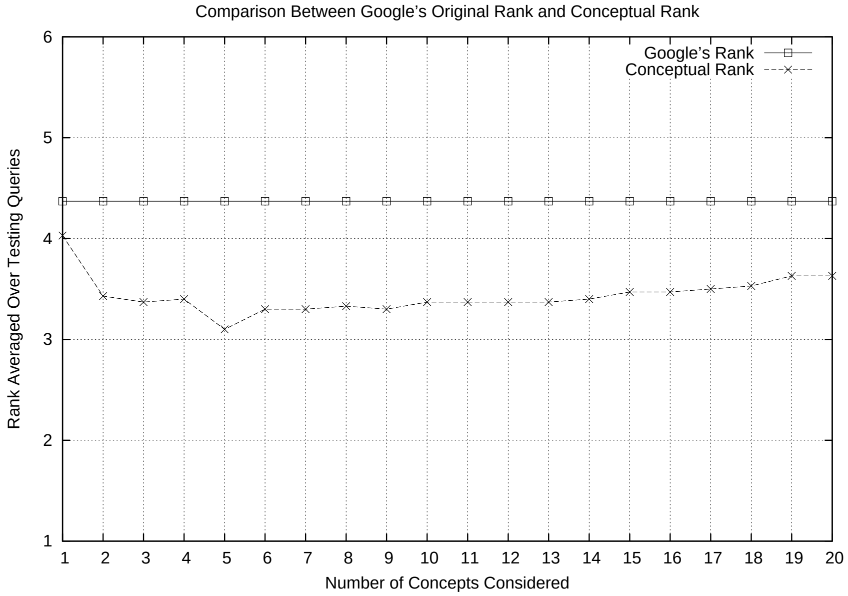


Figure A.3: Google's original rank and conceptual rank averaged over all testing queries. User profiles are built using 20 training queries

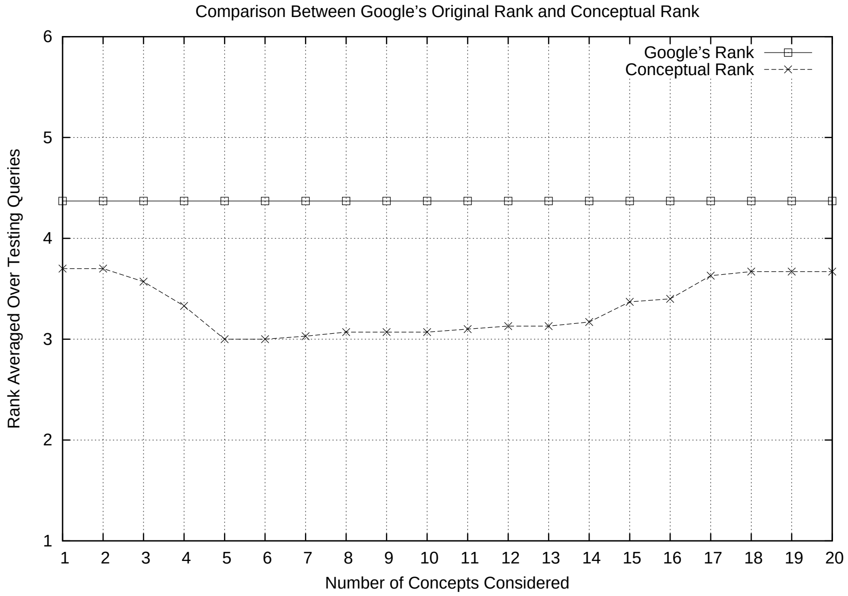


Figure A.4: Google's original rank and conceptual rank averaged over all testing queries. User profiles are built using 40 training queries

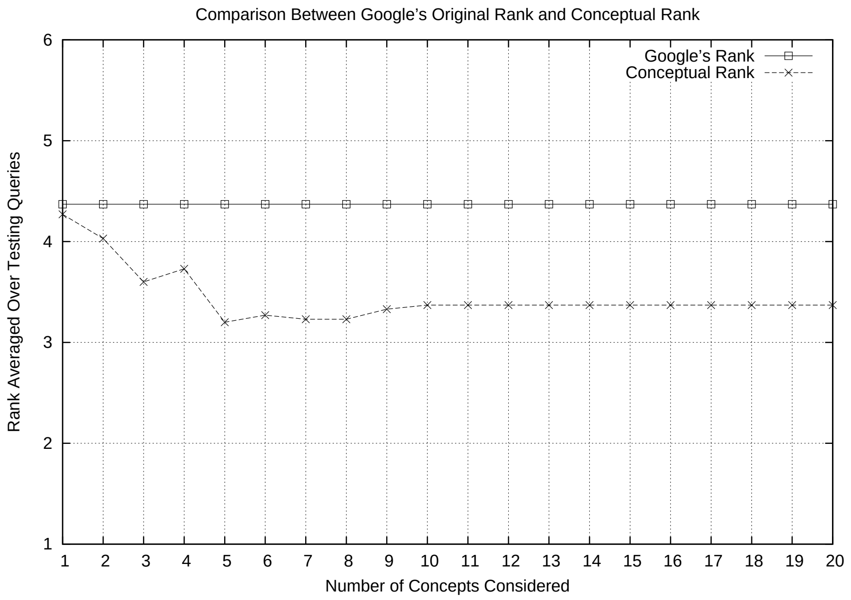


Figure A.5: Google's original rank and conceptual rank averaged over all testing queries. User profiles are built using 5 training snippets

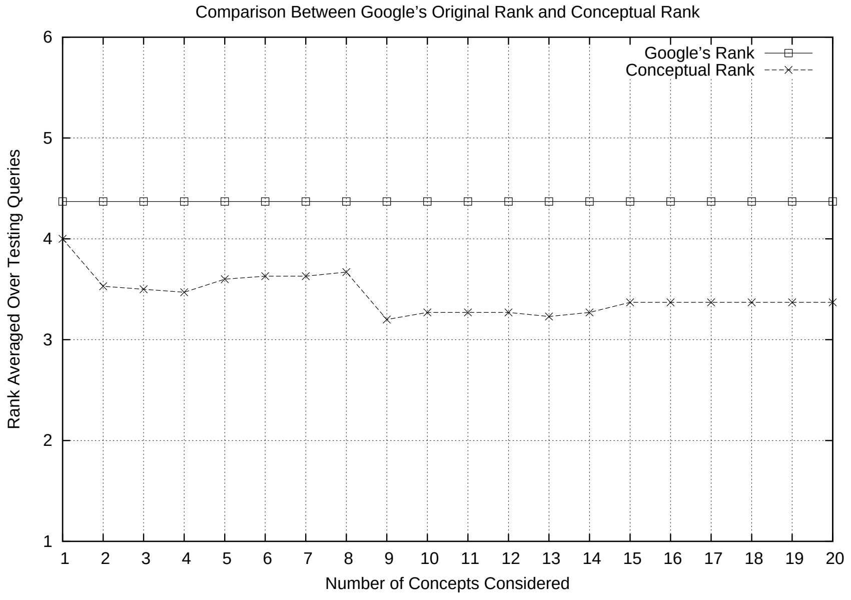


Figure A.6: Google's original rank and conceptual rank averaged over all testing queries. User profiles are built using 10 training snippets

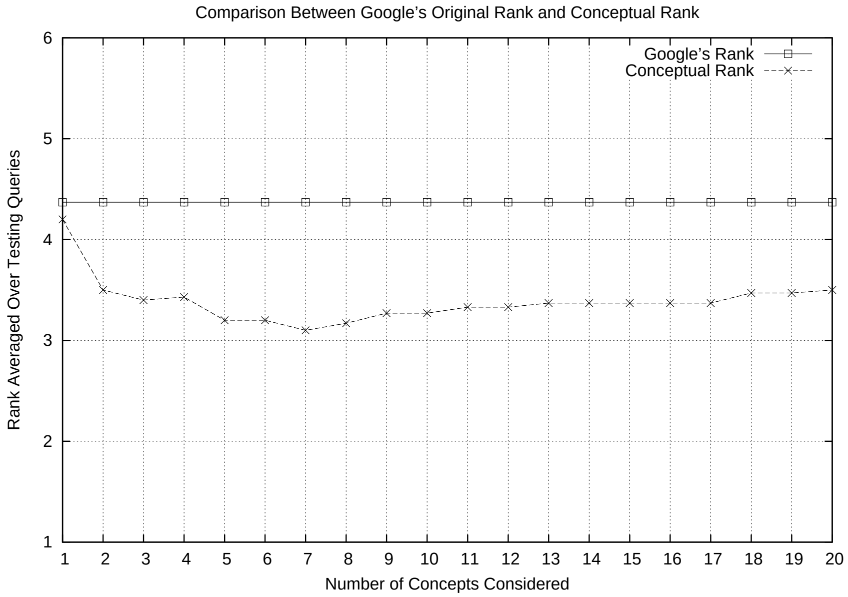


Figure A.7: Google's original rank and conceptual rank averaged over all testing queries. User profiles are built using 20 training snippets

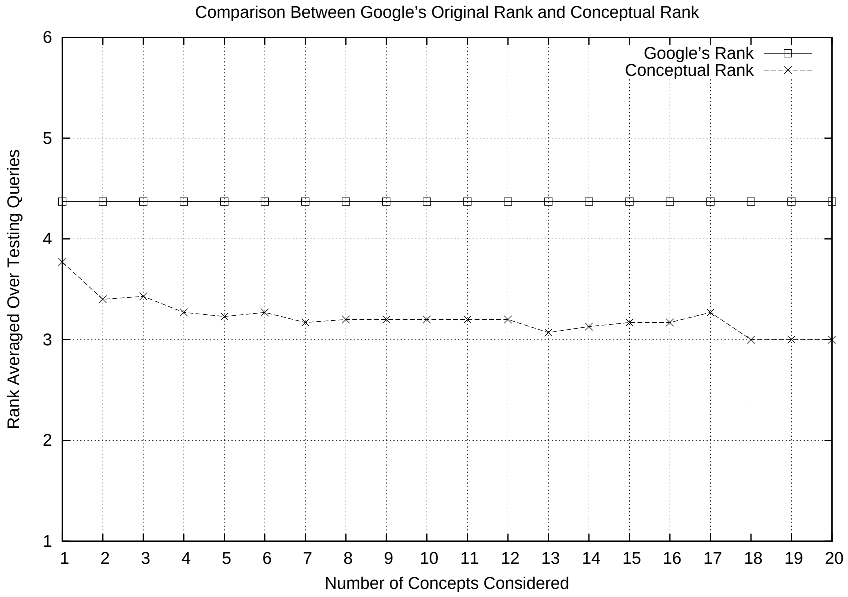


Figure A.8: Google's original rank and conceptual rank averaged over all testing queries. User profiles are built using 40 training snippets

Bibliography

[1] amazon.com. A9, 2005. http://www.a9.com.

[2] amazon.com. amazon.com, 2005. http://www.amazon.com.

[3] askSam Systems. Surfsaver, 2005. http://www.surfsaver.com.

[4] Dietrich Ayala. Php library for google apis, 2005. http://dietrich.ganx4.com/nusoap/.

[5] Tim Berners-Lee. Notation3 - an rdf language for the semantic web, 2001. http://www.w3.org/DesignIssues/Notation3.html.

[6] D. Billsus and M. Pazzani. A hybrid user model for news story classification, 1999. citeseer.ist.psu.edu/billsus99hybrid.html.

[7] E.D. Schabell B.V. Gils. User-profiles for information retrieval. In *BNAIC'03: Proceedings of the 15th Belgian-Dutch Conference on Artificial Intelligence*, 2003. http://osiris.cs.kun.nl/pubmngr/Data/2003/VanGils/UserProfiles/2003-VanGils-UserProfiles.pdf.

[8] V. Challam. Contextual information retrieval using ontology based user profile. Master's thesis, University of Kansas, 2004.

[9] Chien Chin Chen, Meng Chang Chen, and Yeali Sun. Pva: a self-adaptive personal view agent system. In *KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 257–262. ACM Press, 2001. http://doi.acm.org/10.1145/502512.502548.

[10] Mark Claypool, Phong Le, Makoto Wased, and David Brown. Implicit interest indicators. In *IUI '01: Proceedings of the 6th international conference on Intelligent user interfaces*, pages 33–40. ACM Press, 2001. http://doi.acm.org/10.1145/359784.359836.

[11] Mike Giles. Furl, 2003. http://www.furl.net.

[12] google.com. The google apis, 2005. http://www.google.com/apis.

[13] google.com. google.com, 2005. http://www.google.com/.

[14] google.com. google.com, 2005. http://labs.google.com/personalized.

[15] Thomas R. Gruber. A translation approach to portable ontology specifications. *Knowl. Acquis.*, 5(2):199–220, 1993. http://dx.doi.org/10.1006/knac.1993.1008.

[16] N. Guarino, C. Masolo, and G. Vetere. Ontoseek: Content-based access to the web, 1999. citeseer.ist.psu.edu/guarino99ontoseek.html.

[17] E. Hovy. The sensus ontology, mid-1990's. http://www.isi.edu/natural-language/projects/ONTOLOGIES.html.

[18] S. Gauch J. Trajkova. Improving ontology-based user profiles. In *RIAO '04: Proceedings of RIAO (Computer assisted information retrieval)*, pages 380–389, 2004.

[19] Diane Kelly and Jaime Teevan. Implicit feedback for inferring user preference: a bibliography. *SIGIR Forum*, 37(2):18–28, 2003. http://doi.acm.org/10.1145/959258.959260.

[20] Hyoung R. Kim and Philip K. Chan. Learning implicit user interest hierarchy for context in personalization. In *IUI '03: Proceedings of the 8th international conference on Intelligent user interfaces*, pages 101–108. ACM Press, 2003. http://doi.acm.org/10.1145/604045.604064.

[21] Tsvi Kuflik and Peretz Shoval. Generation of user profiles for information filtering – research agenda (poster session). In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 313–315, New York, NY, USA, 2000. ACM Press. http://doi.acm.org/10.1145/345508.345615.

[22] S. Mizzaro and C. Tasso. Ephemeral and persistent personalization in adaptive information access to scholary publications on the web, 2002. citeseer.ist.psu.edu/mizzaro02ephemeral.html.

[23] ODP. The open directory project (odp), 2005. http://dmoz.org.

[24] OnStat. Most people use 2 word phrases in search engines according to onestat.com, February 2004. http://www.onstat.com.

[25] Michael J. Pazzani, Jack Muramatsu, and Daniel Billsus. Syskill webert: Identifying interesting web sites. In *AAAI/IAAI, Vol. 1*, pages 54–61, 1996. citeseer.ist.psu.edu/pazzani98syskill.html.

[26] Dr. K. Narayana Murthy P.R.Kaushik. Personal search assistant: A configurable personal meta search engine. In *AUSWEB99: Proceedings of the 5th Australian World Wide Web Conference. Southern Cross University.*, 1999. http://ausweb.scu.edu.au/aw99/papers/murthy/.

[27] S. Induri at al. S. Gauch, D. Ravindran. Ttc-fy2004-tr-8646-37. Technical report, Information and Telecommunication Technology Center, University Of Kansas, 2004.

[28] Seruku. Seruku toolbar, 2004. http://www.seruku.com/index.html.

[29] S. J. Soltysiak and I. B. Crabtree. Automatic learning of user profiles – towards the personalisation of agent services. *BT Technology Journal*, 16(3):110–117, 1998. http://dx.doi.org/10.1023/A:1009690117684.

[30] StatMarket. Search guiding more web activity, 2003. http://www.statmarket.com.

[31] J. Trajkova. Improving ontology-based user profiles, 2003. citeseer.ist.psu.edu/trajkova04improving.html.

[32] ujiko.com. ujiko.com, 2005. http://www.ujiko.com.

[33] Princeton University. Wordnet a lexical database for the english language, 2005. http://wordnet.princeton.edu/.

[34] W3C. Semantic web, 2001. http://www.w3.org/2001/sw/.

[35] W3C. Soap - simple object access protocol, 2003. http://www.w3.org/TR/soap/.

[36] W3C. Resource description framework, 2004. http://www.w3.org/RDF/.

[37] W3C. Web ontology language, 2004. http://www.w3.org/2004/OWL/.