# A Search Engine Development Utilizing Unsupervised Learning Approach

Mohd Noah Abdul Rahman[1], Afzaal H. Seyal[1], Mohd Saiful Omar[1], and Siti Aminah Maidin[2]

[1] School of Computing & Informatics, Institut Teknologi Brunei, Bandar Seri Begawan BE 1410, Brunei Darussalam
{noah.rahman,afzaal.seyal,saiful.omar}@itb.edu.bn
[2] Ministry of Defense, Bolkiah Garrison, Bandar Seri Begawan, Brunei Darussalam
ameena.alhajah@gmail.com

**Abstract.** This article reports a software development of a generic search engine utilizing an unsupervised learning approach. This learning approach has become apparently important due to the growth rate of data which has increased tremendously and challenge our capacity to write software algorithm and implementation around it. This was advocated as a mean to understand better the flow of algorithm in an uncontrolled environment setting. It uses the Depth-First-Search (DFS) algorithm retrieval strategy to retrieve pages with topical searching. Subsequently, an inverted indexing technique is applied to store mapping from contents to its location in a database. Subsequently, these techniques require proper approach to avoid flooding of irrelevant links which can constitute a poor design and constructed search engine to crash. The main idea of this research is to learn the concept of how to crawl, index, search and rank the output accordingly in an uncontrolled environment. This is a contrast as compared to a supervised learning conditions which could lead to information less overloading.

## 1 Introduction

The widespread use of internet have revolutionized the way people access to information. The advent of search engines have made these happenings [1,2,3] which have proliferated tremendously over the years. There are approximately 300 million Google users and 2 billion log-in searches per day [4] and few are valuable [5]. These figures are so stupendous which could easily lead to information overloading [6] and vocabulary differences [7].

The core of the searching capabilities is the availability of search engines which represent the user interface needed to permit users to query for information [8]. A basic search engine consists of a crawler, indexer and a searcher or query processor which will finally rank the results according to the required output. A crawler as the name implies, crawl the web site and follows all the links that are available. Developing and running a web crawler is a challenging task as it involves performance, reliability and social issues [9]. It also consumes resources belonging to other organizations as well [10].

R. Intan et al. (Eds.): ICSIIT 2015, CCIS 516, pp. 223–233, 2015.
DOI: 10.1007/978-3-662-46742-8_21

An unsupervised learning approach enables programmers the ability to apply machine learning paradigm to solve uncontrolled variables. This approach is multifaceted and it is incorrect to characterize unsupervised learning as stimulus driven, incremental and passive [11,12,13,14]. This learning method has become apparently important due to the rate of growth of data which has increased tremendously and challenge our capacity to write software algorithm around it. This information overloading has become one of the pressing research problems [15].

In a supervised learning methodology, a model is learned using a set of fully labeled items called the training sets [16]. Past research works have highlighted the importance of comparing multiple learning modes in order to develop more general theories of learning [17,18]. The earliest work by Shepard et al. [19] paved the way for quantitatively modeling of a supervised classification learning.

The unsupervised learning is synonym to Learning with a Teacher [20] or Inductive Machine Learning [21] is to build a generic search engine which can learn the mapping between the input and a non-determined set of outputs which is a total contrast as compared with the supervised learning approach [22]. The unsupervised learning is a multifaceted and performance varies with task conditions [39]. The implementation of this application will enable the next batch of students to clearly understand the working mechanics of a search engine and proceed to the next stage of their learning process.

The organization of this research is sequenced as follow: Firstly, a web page is constructed with an infinite number of links. Secondly, a crawler is developed to exercise the concept of crawling, indexing, searching and ranking. Thirdly, an experiment is conducted to test all searching activities and subsequently leads to discussions and a conclusion is made.

## 2 Proposed Approach

An uncontrolled environment to develop a search engine is proposed to enable search engineers to understand the algorithm of a generic search engine. A typical search engine is comprised of a world wide web (www), web crawling, indexing and searching [23].

Most of the search engines whether they are used in the industries which still remain as trade secrets for most of their algorithms or academia employ well established algorithms and techniques [24,25]. One of the fundamental problems with scientific uses of commercial search engines is that their results can be unstable [26,27,28,29]. The earliest and the de facto searching approaches provided by the search engine are based upon the Boolean retrieval model [30] so that the search results exceed the users' abilities to absorb [31,32]. Furthermore, there exists a plethora of classification algorithms that are used for supervised learning methodology.

Although most of the current retrieval models are still based upon keywords, many online users are inclined to establish searching process according to the high-level semantic concept [33]. This study utilises the Depth-First Search (DFS) algorithm retrieval strategy adapted from the graph theory to retrieve pages from sites which are topic-specific or topical searching. This algorithm involves the application of a stack, a data structure concept of last-in, first out (LIFO). This involves the stacking and destacking operations and link lists. The DFS processes each node of binary tree in level by level increasing fashion and left to right in each level [34].

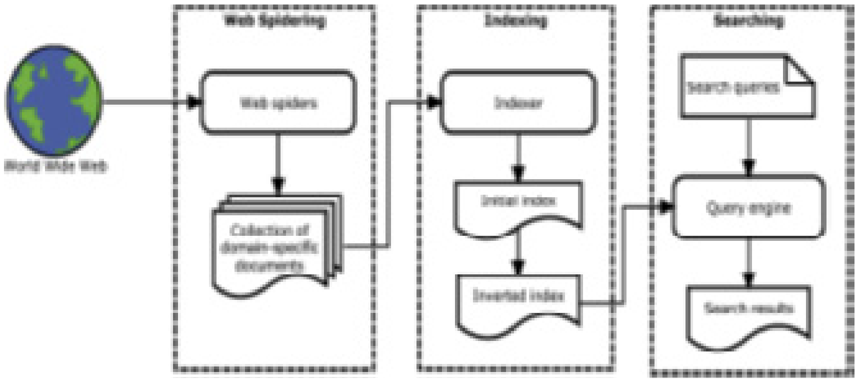


**Fig. 1.** Overall architecture of a search engine developmental tool [23]

The DFS is established by following hypertext links leading to pages which also allow for easier modification and enhancement of the codes in future. Using this function written in Python, the crawler can keep crawling continuously or deposit or save a list of un-indexed pages for later crawling avoiding the risk of overflowing the stack. This searching mechanism will definitely find a solution if it ever exists [35].

The generic algorithm of a DFS with recursive function is illustrated below:

1. Get the URLs seed.
2. Put them in the stack.
3. For every URL in the stack, download page and extract all URLs in the page.
4. Put those URLs in the stack.
5. Iterate step 3.

This function loops through the list of pages, calling a function *addtoindex* on each one. It then uses Beautiful Soup, which is a Python library available in [36] for pulling data out of HTML and XML files to get all links on that page and adds their URLs to a set called *newpages*. At the end of the loop*, newpages* becomes pages, and the process rolls over again. The function, *isindexed*, will determine if a page has been indexed recently before adding it to *newpages*.

Now that we have learnt how the indexing process works, next is to decide the best algorithm for the indexing process for this project. We will be using inverted indexing which stores a mapping from content such as words or numbers to its locations in a DBMS which offers many perceived benefits [37,40]. Theoretically, creating an inverted index for a set of documents D, each with unique ID *doc_id* involves all codes that are available in [36] written in Python programming language which is concise, easy to extend, multi-platform and it is free.

## 3 Proposed Approach

Users can used any search engines available to search for any materials or documents. This uniform resource locator (url) is often referred as the anchor tag or the initial url seed. The anchor tag is identified and the subsequent activities will commence until it reaches the final destination of displaying all the links.

In this experiment, we used a PC with a 4th generation Intel Core i5 with a 3.9 GHz processor with 4 GB RAM and 1TB hard drive. Notepad is used as the editor for the coding of HTML, PHP and a Python programming language and MySQL as the database management systems (DBMS) to store the links. For the search engine interface, there should be at least three pages; a crawler's page for crawling URLs and saving them into an index file, a search page for submitting queries and the result page for returning results of the search query. In relation to these, we will also need a website with lots of links to allow the search engine to work and of course, these will require browsers to view the web pages. Any supported browsers will do such as Microsoft IE 7 or above, Mozilla FireFox 3.0 or above, Google Chrome etc.

This section presents the results based on several test cases conducted to monitor the crawler's performance. The first testing is to make sure that the Rainbow website is connected to the internet. It then displays all external links and the results from the crawling process can be generated. These results are analyzed through the use of explanations and screenshots of system behavior and capabilities that forms the basis of the derive core results. The evaluation of the result is done by comparing system output with user expectations using a collection of data and queries.

Fig. 2. shows the buttons which form the basis in understanding the Links, Crawl, Index and Search concepts. The tests results from this figure is shown in the next section.

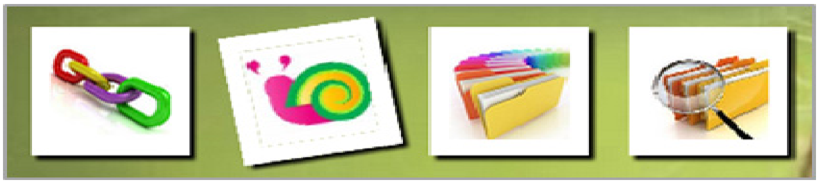


**Fig. 2.** Linking, Crawling, Indexing and Searching buttons

### 3.1 Test Results

● External links - Clicking any button in Fig. 2. navigates users to a webpage that provides links to external websites. Clicking on Email, Dine, Sports and News buttons will show four web pages respectively as shown in Fig. 3. providing contents, images and links to other webpages. This shows that the external links in Fig. 2. are functioning.

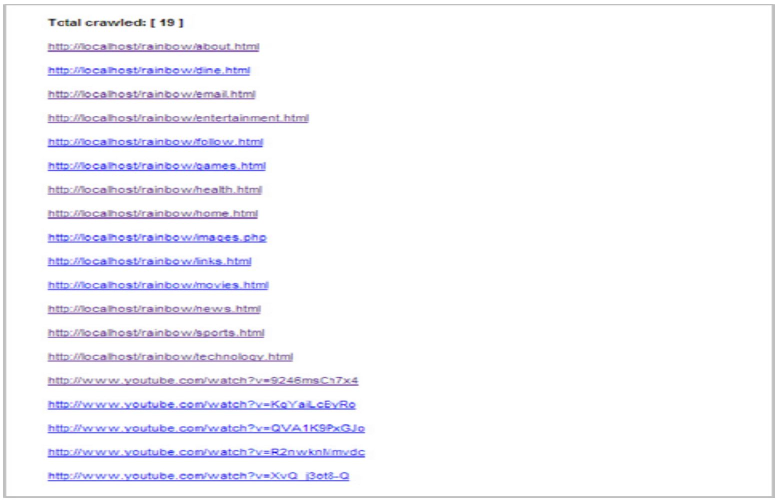
**Fig. 3.** External Links

• Link - The  (Link) button in Fig. 2. is used to show all internal links to pages. Unlike the external link, which requires internet connection to get its contents, this link does not require users to be connected to the internet. Clicking the button will bring user to a page providing internal links as in Fig. 4. which again, when clicked goes to another page as in Fig. 5. and goes on until the end of link is reached.

**Fig. 4.** Internal link level 1

**Fig. 5.** Internal link level 2

• **Crawled -** The (Crawled) button requires users to view the crawled URLs. When the button is clicked, the page will display URL list showing the total number of crawled URLs coming from the internal as well as the external sites. Testing showed that this function worked successfully because the crawler has managed to list out a total of 19 URLs crawled within 10 seconds as shown in Fig. 6. below.



**Total crawled: [ 19 ]**

http://localhost/rainbow/about.html

http://localhost/rainbow/dine.html

http://localhost/rainbow/email.html

http://localhost/rainbow/entertainment.html

http://localhost/rainbow/follow.html

http://localhost/rainbow/games.html

http://localhost/rainbow/health.html

http://localhost/rainbow/home.html

http://localhost/rainbow/images.php

http://localhost/rainbow/links.html

http://localhost/rainbow/movies.html

http://localhost/rainbow/news.html

http://localhost/rainbow/sports.html

http://localhost/rainbow/technology.html

http://www.youtube.com/watch?v=9246msC17x4

http://www.youtube.com/watch?v=KgYaiLcEyRo

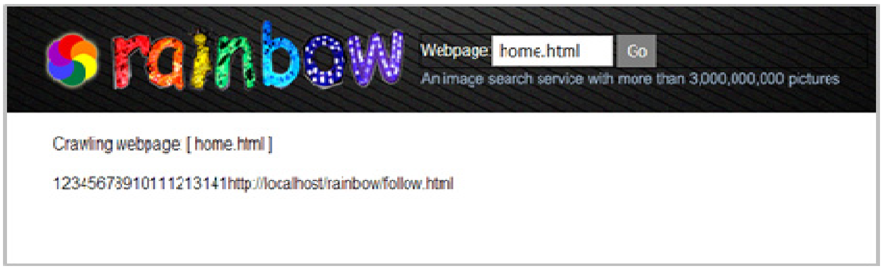
http://www.youtube.com/watch?v=QVA1K9PxGJo

http://www.youtube.com/watch?v=R2nwknMmvdc

http://www.youtube.com/watch?v=XvQ_j3ot8-Q

**Fig. 6.** Crawled URLs lists

● Index - The (Index) button requires users to crawl and index all internal and external websites from Rainbow. When this button is clicked, it will first prompt user to enter the webpage to be crawled. Having entered the starting page, the crawling and indexing process will start, taking URLs (no words, phrases, images etc.) from every page as shown in Fig. 7., because it was designed to index only links. The indexed URLs is then saved into a 'Pages' table in 'Rainbow' database as shown in Fig. 8., waiting for users to do search. Depending on how big the website is, there could have hundreds of thousands of web pages indexed and then they are all searchable on the website. Testing shows that crawling and indexing process is functioning because the URLs are added to the index table in the MySql database.

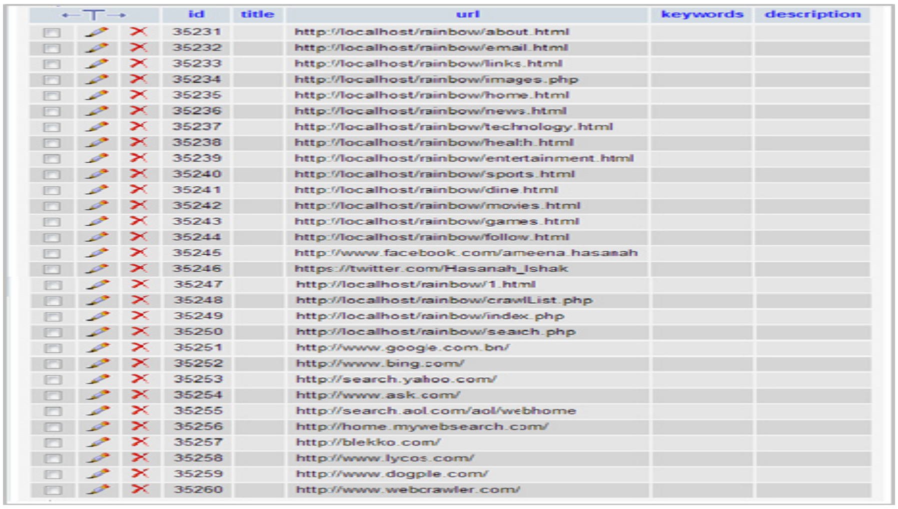

rainbow
Webpage: home.html
Go
An image search service with more than 3,000,000,000 pictures

Crawling webpage [ home.html ]

1234567891011121314http://localhost/rainbow/follow.html

**Fig. 7.** Crawling and indexing process

● Search - The (Search) button requires users to type in any keyword on the search text box corresponding to the URL name because the indexing process is design to index URLs as keywords. When users perform search, the words they entered in the search box are compared with the indexed URL stored in the database. Results from executing this query will return the following outputs as shown in Fig. 10.



| ←T→ | id | title | url | keywords | description |
|---|---|---|---|---|---|
| ☐ ✎ ✕ | 35231 | | http://localhost/rainbow/about.html | | |
| ☐ ✎ ✕ | 35232 | | http://localhost/rainbow/email.html | | |
| ☐ ✎ ✕ | 35233 | | http://localhost/rainbow/links.html | | |
| ☐ ✎ ✕ | 35234 | | http://localhost/rainbow/images.php | | |
| ☐ ✎ ✕ | 35235 | | http://localhost/rainbow/home.html | | |
| ☐ ✎ ✕ | 35236 | | http://localhost/rainbow/news.html | | |
| ☐ ✎ ✕ | 35237 | | http://localhost/rainbow/technology.html | | |
| ☐ ✎ ✕ | 35238 | | http://localhost/rainbow/health.html | | |
| ☐ ✎ ✕ | 35239 | | http://localhost/rainbow/entertainment.html | | |
| ☐ ✎ ✕ | 35240 | | http://localhost/rainbow/sports.html | | |
| ☐ ✎ ✕ | 35241 | | http://localhost/rainbow/dine.html | | |
| ☐ ✎ ✕ | 35242 | | http://localhost/rainbow/movies.html | | |
| ☐ ✎ ✕ | 35243 | | http://localhost/rainbow/games.html | | |
| ☐ ✎ ✕ | 35244 | | http://localhost/rainbow/follow.html | | |
| ☐ ✎ ✕ | 35245 | | http://www.facebook.com/ameena.hasanah | | |
| ☐ ✎ ✕ | 35246 | | https://twitter.com/Hasanah_Ishak | | |
| ☐ ✎ ✕ | 35247 | | http://localhost/rainbow/1.html | | |
| ☐ ✎ ✕ | 35248 | | http://localhost/rainbow/crawlList.php | | |
| ☐ ✎ ✕ | 35249 | | http://localhost/rainbow/index.php | | |
| ☐ ✎ ✕ | 35250 | | http://localhost/rainbow/search.php | | |
| ☐ ✎ ✕ | 35251 | | http://www.google.com.bn/ | | |
| ☐ ✎ ✕ | 35252 | | http://www.bing.com/ | | |
| ☐ ✎ ✕ | 35253 | | http://search.yahoo.com/ | | |
| ☐ ✎ ✕ | 35254 | | http://www.ask.com/ | | |
| ☐ ✎ ✕ | 35255 | | http://search.aol.com/aol/webhome | | |
| ☐ ✎ ✕ | 35256 | | http://home.mywebsearch.com/ | | |
| ☐ ✎ ✕ | 35257 | | http://blekko.com/ | | |
| ☐ ✎ ✕ | 35258 | | http://www.lycos.com/ | | |
| ☐ ✎ ✕ | 35259 | | http://www.dogpile.com/ | | |
| ☐ ✎ ✕ | 35260 | | http://www.webcrawler.com/ | | |

**Fig. 8.** Crawled and indexed URLs saved in database

Based on several outputs from the system, this search engine has proved to be functioning successfully generating outputs matching queries submitted by users with data stored in the index tables. Although this is just a simple search engine with a few basic functions, the crawling, indexing and searching process have produced information to user when searching process is called. In some situations, during searching process, some pages failed to open file contents deriving from the external sites. These problems have affects the effectiveness of the search engine in which the quality of output is not up to the standard of any search engine.

## 4 Discussion

Based on few test cases, the results show that the quality of output was not up to the standard of effectiveness due to the output quality. This is because the current version of this search engine was designed to index only URLs (links or pages) no words, titles, images, videos or anything else to be used as keywords for the search process. Using URL's resource name alone is not good enough as they are poor descriptions of actual information needs. This could be improved and there are many ways for improvement to produce better results and one way is to have the document indexed on its contents, titles, metadata, images, links etc. so that user can have a broad view of the keywords to be searched. Since this current version of search engine is only used for learning process, in future it can be utilized further to enable document to be indexed on title, images or words, links etc. on the page.

Processing time is the time taken to crawl and index websites whereas response time is a delay between a user submitting a query and the time when results are received, usually measured in milliseconds (ms). Processing and response time are again determined by few things such as the speed of the processor and how huge the index file is. For this research project, we have tested the search engine using a 3.96 GHz Intel processor with 4 GB RAM and it came out that the processing time for the crawling and indexing processes are good enough as compared to our previous testing using the supervised learning approach [18]. This is because the indexing algorithms for this crawler's version are made simple by indexing on multiple fields. If the indexing is designed to do a full-text search or other types of indexing such as titles, images, videos, links etc. this would take longer time which again depends on the size of the site contents.

This search engine is able to crawl the web pages for 35 minutes and downloaded 29853 links from the Rainbow website. In order to maintain a search engine corpus of say, ten billion web pages, in a reasonable state of freshness, say with pages being refreshed every 4 weeks on average, the crawler must download over 4,000 pages per second [38]. This can be achieved using distributed crawlers over multiple computers and each crawling machine must pursue multiple downloads in parallel and this would likelihood overload and crash that web server. Therefore, the politeness policies are suggested to be implemented to limit the amount of traffic directed to the web server.

The performance of the Rainbow search engine is far slower because it was not designed for a distributed crawling. If it is to crawl continuously until the end, it would probably take hours or even days to finish crawling because it involves crawling to external sites as well. Performance will only be slow in areas where indexing images, videos or full-text are involved but this crawler only use URLs as the index field. Therefore, this search engine should perform faster than expected. On the other hand, the response time taken when submitting queries and receiving results is quite fast because the table has been indexed which makes it easy to retrieve specified URLs.

The strength of this extended research of a search engine can be seen in its capability to accomplish the crawling, indexing and searching process for information. Based on our previous discussions, there are rooms for improvement in order to make the search engine more powerful and functional by first implementing the ranking algorithm which was not included in the system. One of the most important steps in improving the website ranking in any search engine is to ensure that it contains plenty of rich information that includes relevant keywords that indicate the subject matter of the page's content. Subsequently, we can suggest to improve the performance of the search engine by implementing certain strategies.

For system effectiveness, that is improving the quality of output, we used headings to summarize page contents and include accurate description of page content in metadata. Apart from these, full-text search can also be used that is breaking a block of unstructured text into individual words, phrases, and special tokens. With these, a wide category of indexed keywords are stored in the index file to enable users to find relevant searches which are far better than using only descriptive URLs as keyword as in this crawler. Besides splitting documents into keywords, ranking can also improve the search quality. This can be done in many ways and the most common ones are using the location and frequency of keywords on the web page. Those with higher frequency are deemed to be more relevant than those with lower frequency.

## 5 Conclusion

This study has demonstrated a generic search engine which forms the core of applications that facilitate information retrieval process in an unsupervised learning environment. It employs a DFS algorithm for retrieval strategy which allows future code modifications and enhancements. This proposed approach uses all available computing resources efficiently as compared to most tasks performed by high end servers.

The development for this prototype of a search engine will enable students to develop and enhance their programming language and algorithms analytical skills to the next level of their learning process. This will further enhance their abilities to produce new algorithms and laid a ground-work in innovation and creativity.

One of the recommendations can now be made about this search engine is to update regularly the algorithm and need to be checked and revised periodically. Future direction of this study can be further extended using other learning paradigm such as semi-supervised learning or other hybrid learning approaches. A majority of human learning activity can be classified as unsupervised.

## References

1. Yahoo! Inc., http://www.yahoo.com
2. Bing Inc., http://www.bing.com
3. Google Inc., http://www.google.com
4. Mishra, A.A., Kamat, C.: Migration of Search Engine Process into the Cloud. International Journal of Computer Application 19(1) (April 2011)
5. Glover, E.J., Lawrence, S., Gordon, M.D., Birmingham, W.P., Giiles, C.L.: Web Search Your Way. Communications of the ACM 44(12), 97–102 (2001)
6. Chen, H., Buntin, P., Sutjahjo, S., Sommer, C., Neely, D.: Expert, prediction, symbolic learning and neural networks: An experiment on Greyhound racing. IEEE Expert. 9(21), 21–27 (1994)
7. Fumas, G.W., Landauer, T.K., Gomez, L.M., Dumais, S.T.: The Vocabulary Problem in Human-System Communication. Communications of the ACM 30(11), 964–971 (1987)
8. Kumar, G., Duhan, N., Sharma, A.K.: Page Ranking Based on Number of Visits of Links of Web Pages. In: Computer & Communication Technology (ICCT), pp. 11–14 (2011)
9. Brin, S., Page, L.: The anatomy of a Large-Scale Hypertextual Web Search Engine. In: World Wide Web Conference (WWW 1998), pp. 107–117 (1998)
10. Koster, M.: Robots in the web: threat or treat. ConneXions 4(4) (April 1995)
11. Berry, D.C., Dienes, Z.: Implicit learning: Theoretical and empirical issues. Erlbaum, Hillsdale (1993)
12. Hayes, N., Broadbent, D.E.: Two modes of learning for interactive tasks. Cognition 24, 249–276 (1988)
13. Hock, H.S., Malcus, L., Hasher, L.: Frequency discrimination: Assessing global and elemental letter units in memory. Journal of Experimental Psychology, Learning, Memory & Cognition 12, 232–240 (1986)
14. Kellog, R.T.: When can we introspect accurately about mental processes. Memory & Cognition 10, 141–144 (1982)
15. Bowman, C.M., Danzig, P.B., Manber, U., Schwartz, F.: Scalable Internet Resource Discovery: Research Problems and Approaches. Communications of the ACM 37(8), 98–107 (1994)
16. Croft, W.B., Metzler, D., Strohman, T.: Search Engines: Information Retrieval in Practice, p. 344. Pearson Education Inc. (2010)
17. Love, B.C., Markman, A.B., Yamauchi, T.: Modeling classification and inference learning. In: Fifteenth National Conference on Artificial Intelligence, pp. 136–141. MIT Press, MA (2000)
18. Yamauchi, T., Love, B.C., Markman, A.B.: Learning non-linearly separable categories by inference and classification. Journal of Experimental Psychology: Leraning, Memory & Cognition 28, 585–593 (2002)
19. Sheperd, R.N., Hoyland, C.L., Jenkims, J.M.: Learning and memorization of classifications. Psychological Monographs 75(13, Whole No. 517)
20. Haykin, S.: Neural Networks: A Comprehensive Foundation, 2nd edn. Prentice Hall PTR (1998)
21. Kotsiantis, S.: Supervised Machine Learning: A Review of Classification Techniques. Informatica Journal 31, 249–268 (2007)
22. Rahman, M.N., Seyal, A.H., Maidin, S.A.: Search engine development: Adaptation from supervised learning methodology. In: Fourth International Conference on Digital Information Processing & Communications, March 18 -20, pp. 35–42 (2014)

23. Chau, M., Wong, C.H.: Designing the user interface and functions of a search engine development tool. Decision Support Systems 48, 369–382 (2010)
24. Salton, G.: Automatic Text Processing. Addison-Wesley, Reading (1989)
25. Faloutsos, C.: Access Methods for Text. ACM Computing Surveys 17(1), 48–74 (1985)
26. Bar-Ilan, J.: Search engine results over time: A case study on search engine stability, http://www.cindoc.csis.es/cybermetrics/articles/v2i1p1.html (retrieved January 26, 2014)
27. Bar-Ilan, J.: Search engine ability to cope with the changing web. In: Levene, M., Poulovasilis, A. (eds.) Web Dynamics. Springer, Berlin (2004)
28. Mettrop, W., Nieuwenhuysen, P.: Internet search engines - fluctuations in document accessibility. Journal of Documentation 57(5), 623–651 (2001)
29. Rousseau, R.: Daily time series of common single word searches in AltaVista and NorthernLight. Cybermetrics 2(3) (1999)
30. Frants, V.I., Shapiro, J., Taksa, I., Voiskunskii, V.G.: Boolean Search: Current State and Perspectives. Journal of the American Society of Information Science 50(1), 86–95 (1999)
31. Jung, S., Herlocker, J.L., Webster, J.: Click Data as Implicit Relevance Feedback in Web Search. Information Processing & Management 33, 791–807 (2007)
32. Chau, M., Chen, H., Qin, J., Zhou, J.Y., Qin, Y., Sung, W.K., McDonald, D.: Comparison of two approaches to building a vertical search tool: a case study in the nanotechnology domain. In: 2nd ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 135–144. ACM (July 2002)
33. Yi, J.: The Research of Search Engine Based on Semantic Web. In: International Symposium on Intelligent Information Technology Workshop (IITAW), pp. 360–363 (2008)
34. Brassard, G., Bratley, P.: Fundamentals of Algorithms, 1st edn., pp. 303–305. PHI Publications, New Delhi (2008)
35. Russel, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Prentice Hall, Upper Saddle River (1995)
36. Segaran, T.: Programming Collective Intelligence: Building Smart Web 2.0 Applications, 1st edn. O'Reilly Media Inc. (2007)
37. Rahman, M.N., Seyal, A.H., Mohamed, H.Y., Mashud, I.: A theoretical framework on the use of database management systems. Journal of Technology & Management 5(1), 36–48 (2007)
38. Najork, M.: Web Crawler Architecture. Encyclopedia of Database Systems (2009)
39. Love, B.C.: Comparing supervised and unsupervised category learning. Psychonomic Bulletin & Review 9(4), 829–835 (2002)
40. Rahman, M.N., Seyal, A.H., Mohamed, H.A.Y.: An empirical framework of DBMS usage in Brunei Darussalam. In: The Fifth Annual Global Information Technology Management World Conference (GITM), San Diego, California, USA, June 13-15, pp. 189–192 (2004)