The Anatomy of SnakeT: A Hierarchical Clustering Engine for Web-Page Snippets^{*}

Paolo Ferragina and Antonio Gulli

Dipartimento di Informatica, Università di Pisa
{ferragina,gulli}@di.unipi.it

The purpose of a search engine is to retrieve from a given textual collection the documents deemed *relevant* for a user query. Typically a user query is modeled as a set of keywords, and a document is a Web page, a pdf file or whichever file can be parsed into a set of tokens (words). Documents are *ranked* in a flat list according to some measure of *relevance to the user query*. That list contains hyperlinks to the relevant documents, their titles, and also the so called (*page or web*) *snippets*, namely document excerpts allowing the user to understand if a document is indeed relevant without accessing it.

Even if search engines are successfully used by millions of users to search the web everyday, it is well-known nowadays that the *flat ranked list* is getting obsolete, if not useless, due to the increasing complexity of the web retrieval problem. The key difficulty lies within the definition of what is “*relevant*” to the user issuing a query: Relevance is subjective and may change over the time.

Motivated by these considerations, many IR-tools [1] are currently being designed to help the users in their difficult search task by means of novel ways for reporting the query results. Two success stories are given by Vivisimo and Dogpile that were elected as the best meta-search engines of the last three years by a jury composed by about 500 web users, as reported by SearchEngineWatch.com¹. These IR-tools add to the flat list of relevant documents a *hierarchy of folders* built on-the-fly over the snippets returned by one (or more) search engine(s). Each folder is *labeled* via a meaningful sentence that captures the “theme” of the snippets (and, thus, of the corresponding documents, or web pages) clustered into it. As a result, users are provided with a concise, but intelligible, picture of numerous query results at various levels of details. Hence the user has no longer to scan the flat list, but may browse the labeled folders according to the “intent behind his/her query”. An example is provided in Figure 1.

Various scientific papers have recently investigated this challenging problem [8, 10, 7, 2, 6, 3, 5, 9, 4]. However, none of them achieved results comparable to Vivisimo or DogPile; and none of them provided a web interface with a detailed description of their architecture. The best results to date are “Microsoft and IBM products” [4, 9] not publicly accessible, as the authors communicated to us!

---

^{*} Partially supported by the Italian MIUR projects ALINWEB and ECD, and by the Italian Registry of ccTLD.it.

¹ This IR-tool was introduced in a primitive form by Northernlight, and then improved by many other commercial softwares such as Copernic, iBoogie, Kartoo and Groxis. Unfortunately, very little information is available about their underlying technology!

Fig. 1. VIVISMO and SNAKET tested on the query “Data Mining”.

In this paper we investigate the *web snippet hierarchical clustering* problem by devising an algorithmic solution, and a software prototype (called SnakeT, <http://roquefort.di.unipi.it/>), that offers some distinguishing features (see Figure 2 for details):

1. It draws the snippets from 16 search engines, the Amazon collection of books (a9.com) and the Google News.
2. In response to a user query, it builds *on-the-fly* the folders and labels them with sentences of variable length drawn from the snippets as *non* contiguous sequences of terms. This is crucial to ensure the readability of the labels (à la Vivismo), follows the “future directions of research” of Grouper [8], and extends the ideas in [6].
3. It uses some innovative ranking functions based on *two knowledge bases* (KB), built off-line and used at query time. One KB offers some *link and anchor-text information* derived by a crawling of 50 millions of Web pages done via the *Nutch open-source spider*. This information is used to provide “better candidate sentences”. The other KB consists of the *Open Directory Project (DMOZ)* and is used to rank the candidate sentences according to their frequency and position of occurrence within the DMOZ’s categories.
4. It organizes the folders into a *hierarchy*, and possibly let them overlap. This process is driven by the *set of (approximate) sentences* assigned to each folder and describing its content at various levels of details. The label eventually assigned to a folder will take into account the set of sentences assigned to its children, as well as some coverage and some label distinctiveness measures. As a result, the labels lying over any hierarchy’s path are *not necessarily one the substring of the other*, and thus they result much more descriptive. The overall time complexity is linear in the number of candidate labels, and takes few seconds. The picture offered by the hierarchy of labeled folders recalls a sort of knowledge extraction process, which is useful to drive the subsequent user browsing (see the Figure 1).

The web interface available for testing (see Figure 1) and some user studies are detailed in a companion paper published in this proceedings. Many issues remain to be further investigated: We are performing user studies on *free queries*,

and on a larger and more variegated set of users; we are investigating the application of *SnakeT* to other contexts, like *blogs*, news and books; and we are studying the use of *SnakeT* as a *personalized ranker*, in order to make the flat list of results more effective for resolving the user queries. Check the web page of *SnakeT* for the latest news.

Fig. 2. Architecture of SnakeT.

References

1. CNN.com. Better search results than Google? Next-generation sites help narrow internet searches. Associated Press, January 2004.
2. B. Fung, K. Wang, and M. Ester. Large hierarchical document clustering using frequent itemsets. *SIAM International Conference on Data Mining*, 2003.
3. F. Giannotti, M. Nanni, D. Pedreschi, and F. Samaritani. Webcat: Automatic categorization of web search results. In *SEBD*, pages 507–518, 2003.
4. K. Kumamuru, R. Lotlikar, S. Roy, K. Singal, and R. Krishnapuram. A hierarchical monothetic document clustering algorithm for summarization and browsing search results. In *WWW*, 2004.
5. D. J. Lawrie and W. B. Croft. Generating hierarchical summaries for web searches. In *ACM SIGIR*, pages 457–458, 2003.
6. Y. S. Maarek, R. Fagin, I. Z. Ben-Shaul, and D. Pelleg. Ephemeral document clustering for web applications. Technical Report RJ 10186, IBM Research, 2000.
7. D. Weiss and J. Stefanowski. Web search results clustering in polish: Experimental evaluation of Carrot. In *New Trends in I.I.P. and Web Mining Conference*, 2003.
8. O. Zamir and O. Etzioni. Grouper: a dynamic clustering interface to Web search results. *Computer Networks*, 31:1361–1374, 1999.
9. H. Zeng, Q. He, Z. Chen, and W. Ma. Learning to cluster web search results. In *ACM SIGIR*, 2004.
10. D. Zhang and Y. Dong. Semantic, hierarchical, online clustering of web search results. In *WIDM*, 2001.