An Effective Recommender System for Highly Dynamic and Large Web Sites

Ranieri Baraglia[1], Francesco Merlo[2], and Fabrizio Silvestri[1]

[1] Istituto di Scienze e Tecnologie dell'Informazione (A. Faedo)
ISTI–CNR – Pisa, Italy
{ranieri.baraglia,fabrizio.silvestri}@isti.cnr.it
[2] LIASES – Laboratorio di Informatica Applicata alle Scienze Economiche e Sociali (G. Rota)
Universitá degli studi di Torino
Facoltá di Economia
merlo@econ.unito.it

Abstract. In this demo we show a recommender system, called *SUGGEST*, that dynamically generates links to pages that have not yet been visited by a user and might be of his potential interest. Usually other recommender systems exploit a kind of two-phase architecture composed by an off-line component that analyzes Web server access logs and generates information used by a successive online component that generates recommendations. *SUGGEST* collapse the two-phase into a single online Apache module. The component is able to manage very large Web sites made up of dinamically generated pages by means of an efficient LRU-based database management strategy. The demo will show the way *SUGGEST* is able to anticipate users' requests that will be made farther in the future, introducing a limited overhead on the Web server activity[1].

1 Introduction

The continuous and rapid growth of the Web has led to the development of new methods and tools in the Web recommender or personalization domain [1], [2]. Web Usage Mining (WUM) is the process of extracting knowledge from Web users access data (or clikstream) by exploiting Data Mining (DM) technologies.

In this demo, we present a recommender system, called *SUGGEST*, which is designed to dynamically generated personalized content of potential interest for users of a Web Site. It is based on an incremental personalization procedure, tightly coupled with the Web server. It is able to update incrementally and automatically the knowledge base obtained from historical usage data and to generate a list of page links (*suggestions*). The suggestions are used to personalize *on the fly* the HTML page requested. Moreover, the adoption of a LRU-based algorithm to manage the knowledge base permits us to use *SUGGEST* also on

[1] This work was funded by the Italian MIUR as part of the National Project Legge 449/97, 1999, settore Società dell'Informazione: Technologies and Services for Enhanced Contents Delivery (2002-2004)

J.-F. Boulicaut et al. (Eds.): PKDD 2004, LNAI 3202, pp. 531–533, 2004.

large Web sites made up of dynamically generated pages. Furthermore the system is able to evaluate the importance of the pages composing the underlying Web site by adopting a sort of PageRank estimation based on the maintained usage information. Furthermore, the system proposed was evaluated by adopting the new quality metric we introduced in [3].

2 *SUGGEST*

Schematically, *SUGGEST* works as follows: once a new request arrives at the server, the URL requested and the session to which the user belongs are identified, the underlying knowledge base is updated, and a list of suggestions is appended to the requested page. To catch information about navigational patterns, *SUGGEST* models the page accesses information as a undirected graph $G = (V, E)$. The set V of vertices contains the identifiers of the different pages hosted on the Web server. Based on the fact that the interest in a page depends on its content and not on the order a page is visited during a session [4], we assign to each edge E a weight computed as: $W_{ij} = N_{ij}/max\{N_i, N_j\}$ where N_{ij} is the number of sessions containing both pages i and j, N_i and N_j are respectively the number of sessions containing only page i or page j. Dividing by the maximum between single occurrences of the two pages has the effect of reducing the relative importance of links involving index pages. Such pages are those that, generally, do not contain useful content and are used only as a starting point for a browsing session. Index pages are very likely to be visited with any other page and nevertheless are of little interest as potential suggestions. The data structure we used to store the weights is an adjacency matrix M where each entry M_{ij} contains the value W_{ij} computed according to Formula above. To further reduce the impact of insignificant links we filtered out links whose weigths are under a predetermined threshold *minfreq*. As in the previous version, *SUGGEST* finds groups of strongly correlated pages by partitioning the graph according to its connected components. *SUGGEST* actually uses a modified version of the well known incremental connected components algorithm. After the clustering step, *SUGGEST* has to construct the suggestions list for the current user request. This is done in a straightforward manner by finding the cluster that has the largest intersection with the *PageWindow* related to the current session. The final suggestions are composed by the most relevant pages in the cluster, according to the order determined by a sort of PageRank algorithm applied to the Web site's usage graph.

3 Evaluation and Conclusions

In order to evaluate both the effectiveness (i.e. the quality of the suggestions) and efficiency (i.e. overhead introduced on the overall performance of the Web server) of *SUGGEST* several tests were conducted. All tests were run on a processor Intel Celeron 2,4 GHz with 256 MBytes of RAM, an ATA 100 disk with 30 GBytes, and operating system Linux 2.4.20. The *SUGGEST* effectiveness was

evaluated by using a performance parameter that takes into account the distance of the suggestions generated with the actual pages visited during the session. To evaluate the *SUGGEST* effectiveness experimental evaluation was conducted by using three real life access log files of public domains: Berkeley, NASA, USASK [3] For each dataset we measured the quality of the suggestions (Ω) varying the *minfreq* parameter. Figure 1 shows the results obtained. Moreover, we also plotted the curve relative to the suggestions generated by a random suggestion generator (labelled rnd in Fig. 1). As it was expected, the random generator performs poorly and the intersection between a random suggestion and a real session is almost null. On the other hand, suggestions generated by *SUGGEST* show a higher quality, that, in all the datasets, reaches a maximum for *minfreq*=0.2. For low values of the *minfreq* parameter, good values are obtained for

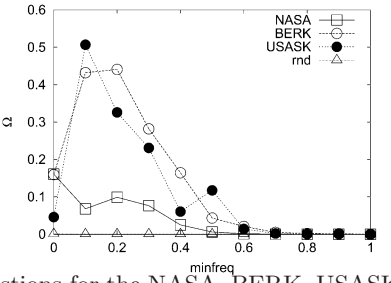

Fig. 1. Coverage of suggestions for the NASA, BERK, USASK access log files, varying *minfreq*.

the quality of the suggestions. In the demo we are going to show the working of *SUGGEST* on a real production Web Site of the Economy Faculty of the University of Turin. Goal of the demo will be showing the ability of *SUGGEST* to generate links to "*useful*" pages.

References

1. Magdalini, E., Vazirgiannis, M.: Web mining for web personalization. ACM Trans. on Internet Technology **3** (2003) 1–27
2. Deshpande, M., Karypis, G.: Item-based top-n recommendation algorithms. ACM Trans. on Information Systems **22** (2004) 143–177
3. Silvestri, F., Baraglia, R., Palmerini, P., M., S.: On-line generation of suggestions for web users. In: Proc. of IEEE Int'l Conf. on Information Technology: Coding and Computing. (2004)
4. Baraglia, R., Palmerini, P.: Suggest: A web usage mining system. In: Proc. of IEEE Int'l Conf. on Information Technology: Coding and Computing. (2002)