DATA MINING FOR BROWSING PATTERNS IN WEBLOG DATA BY ART2 NEURAL NETWORKS

A. Nachev, I. Ganchev

Abstract: Categorising visitors based on their interaction with a website is a key problem in Web content usage. The clickstreams generated by various users often follow distinct patterns, the knowledge of which may help in providing customised content. This paper proposes an approach to clustering weblog data, based on ART2 neural networks. Due to the characteristics of the ART2 neural network model, the proposed approach can be used for unsupervised and self-learning data mining, which makes it adaptable to dynamically changing websites.

Keywords: data mining, weblog, neural networks, adaptive resonance theory.

Introduction

With the rapid increase in Web traffic and e-commerce, understanding user's behaviour based on their interaction with a website is becoming more and more important for website owners. There are many commercially available Web server log analysis tools [15], [16], but generally they have limited abilities for reporting user activity and use statistical techniques to provide frequency of accesses to individual files or times of visits. Such tools, however, are not designed to operate properly with data from very high traffic Web servers, and usually provide little analysis of data relationships among accessed files, which is essential to fully utilise the data from the server logs [1]. The complexity of tasks such as designing a Web site, designing Web server in terms of physical data layout, and ease of navigation through a Web site have increased rapidly. Identifying the user behaviour may enable to provide customised content for the users, thereby making it more adaptive to user experience. It can be used also to restructure a Web site in order to serve better the needs of users of a site and to help site navigation by providing a list of popular destinations from a particular Web page.

A lot of research has been done in the area of Web usage clustering, which directly or indirectly addresses the issues involved in data mining for extraction of web navigational patterns [2], ordering relationships [3], prediction of web surfing behaviour [4], and clustering of web usage sessions [5] based on web logs. Many research studies [12], [13], [14] have looked at capturing users' web access patterns and store them in log files for different purposes.

Some techniques of weblog data mining use cookies to identify site users and user sessions. Cookies play role of markers that are used to tag and track site users automatically. Another approach to identify users is to use a remote agent, as described in [7], uses Java agents that is run on the client side in order to send back accurate usage information to the Web server. The major disadvantage of both techniques is that they rely on implicit user cooperation, which doesn't exist in many cases. There is a constant conflict between the Web user's desire for privacy and the Web provider's desire for collecting information about the visitors. In fact many users disable the browser features that allow storage of cookies or Java agents, which makes such techniques impractical. We assume that user identification is not facilitated by the techniques described above.

Data Preprocessing

The first stage of the Web usage analysis, called preprocessing, aims to extract data from the weblog files of a Web server in order to form input files for the next stage of processing. The preprocessing stage includes several substages: data filtering, user identification, session identification, transaction identification, and vector preparation. Data filtering is the task of extracting only those records of weblog files, which are essential for the analysis, thus reducing significantly data necessary for further processing. User identification is the task of identifying users and grouping page references into user clickstreams. Session identification is the task of identifying logically self-contained sub-clickstreams, which can be considered as separate visits by the same user. Transaction identification creates meaningful clusters of page references for each session, reducing the size of the processed data. Vector formation is the task of representing transactions in a form of vectors, used for clustering by a neural network simulator.

1. Data Filtering

Data filtering aims to extract page views rather than page hits. When a user requests a Web page, he does not explicitly request all of the graphics, audio, video, etc. files which the client automatically downloads due to the HTML tags in the Web page. The HTTP protocol establishes a separate connection with the Web server for every requested file. A user's request to a page view often results in several log entries, some of which are not relevant to the Web usage analysis. Those entries correspond to requested URLs with extensions *gif*, *jpeg*, *jpg*, *css*, *swf*, etc. which should be filtered out by the system. Moreover there are users requests, which cannot be processed by the Web server and produce entries with error codes. Such entries are not of interest for the Web usage analysis and should be eliminated. Also, the weblog entries corresponding to call of *cgi-scripts* and those corresponding to *POST* and *OPTIONS* operations are difficult to analyse since they don't have any static existence in the web site hierarchy. Such entries should also be discarded at this preprocessing stage.

2. User Identification

In order to analyse users' behaviour based on weblog data, all entries originating from a particular user should be identified and grouped together. This is the task performed by the second stage of preprocessing, called user identification. All entries representing requests from registered users contain their IDs. If the user is anonymous, a hyphen (-) represents the username. Manipulating the usernames enables the system to group together all log entries of a particular registered user. The entries which originate from anonymous users, however, should be grouped in other way. Many works [6] suggest client's IP address to be used as a primary criterion for user identification, because presumably at a moment a single user works with a particular machine, therefore he/she can be associated with the machine's IP address. Pirolli et al. [8] consider client's agent type and operating system type as indicative for the user identity. They suggest using those data as a secondary input for the user identification. A change in either the client's agent type can be considered as change of the user even when the IP address does not changed.

3. Session identification

It is very likely that a user visits a Web site more that once, each time searching different information. Logically each search can be distinguished from the others and considered as a separate session. Extracting user sessions from weblog files uses a time-out heuristic. The basic idea of this approach is that if there are no hits from a particular user for a time-out period τ, then session is assumed to have ended. Any hit after the time-out will be ascribed to a new session. The new session will continue until the dormancy period between successive hits from that user is more than the time-out threshold τ. In [9] Catledge and Pitkow suggest a timeout value 25.5 minutes as acceptable for session identification.

All considerations above assume that only one user works at a time with a particular client at a particular IP address.

4. Transaction identification

Some entries of the weblog files correspond to references to content-oriented Web pages which provide meaningful information to the user. Other Web pages can be considered as auxiliary, i.e. they contain rather links to other pages than useful information. Majority of the identified sessions inevitably contain references to auxiliary pages, but such information is not indicative for the user's behaviour and interests. Next stage of preprocessing, called transaction identification, aims to filter the sessions from auxiliary page references. The outcome of this preprocessing is a set of meaningful clusters of references, called transactions.

The transaction identification involves the assumption that the amount of time a user spends on a page correlates to whether the page should be classified as an auxiliary or content-oriented page for that user. It is expected that the times spent on the auxiliary pages are smaller than those spent on content-oriented pages. The time spent on each reference is estimated by taking the difference between the time of the next reference and the current reference. Since the last reference in each transaction has no "next" time to use in estimating the reference length, this approach uses the assumption that all of the last references are content-oriented references.

In order to distinguish between auxiliary and content-oriented page references, a reasonable estimate of cutoff reference length must be made. It requires analysing of the reference length of all considered weblog

entries for a particular time period. Such estimate strongly depends on the specifics and content of a particular Web site, which makes impossible deriving it by universally applicable rules or criteria. In order to simplify the analysis, a set of 3000 randomly selected entries from the weblog files comprising a 45-days period was considered. Figure 1 shows the distribution of the lengths of page references between 0 and 450 seconds. Analysis of the histogram shows that the distribution of the lengths of references can be approximated well by an exponential distribution.

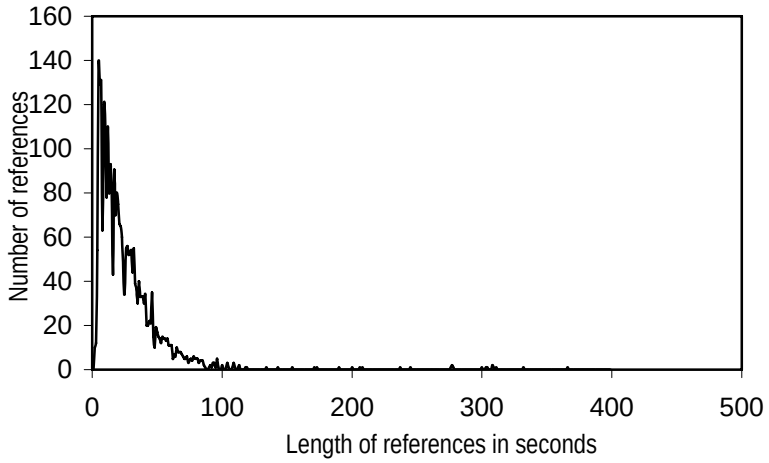

Figure1. Distribution of lengths of page references of 3000 randomly selected weblog entries.

The content-oriented references are longer than the auxiliary ones and make up the upper tail that extends out to the longest reference. The lower end of the curve represents the auxiliary references. For such case of distribution Cooley et al. [6] suggest calculating the cutoff time σ by equation (1), where γ is percentage of the auxiliary references in the log and η is reciprocal of observed mean reference length.

$$\sigma = \frac{-\ln(1-\gamma)}{\eta} \tag{1}$$

The definition of (1) comes from integrating the formula for an exponential distribution from γ to zero.

The initial input to the transaction identification process consists of all of the page references for a user session. The data processing on this stage consists of filtering out the auxiliary page references using the calculated cutoff time, thus converting each user session into a transaction.

Let Ω be a set of all user transaction entries in a weblog file and let $t \in \Omega$ is a user entry that consists of several components, namely the user IP address $t.uid$, the user identifier $t.uid$, the user agent type $t.uag$, the reference URL $t.url$, the time of access $t.time$, and the length of reference $t.length$. Formally a transaction T can be defined as a quadruple

$$\begin{aligned} T =< uip_T, uid_T, uag_T, \{(t_T^1.url, t_T^1.time, t_T^1.length), \\ \cdots, (t_T^m.url, t_T^m.time, t_T^m.length)\} > \end{aligned} \tag{2}$$

where for all $k : 1 \le k \le m$ $t_T^k.uip = uip_T$, $t_T^k.uid = uid_T$, and $t_T^k.uag = uag_T$.

Although the first three components of (2) uniquely identify the corresponding transaction, the system assigns an additional unique identifier id_T to each transaction in order to simplify the following processing stages.

The transaction identification stage also maps each URL of a transaction entry $t_T^k.url$ into a unique URL identifier $t_T^k.urlid$ (positive integer). The mapping enables preparing numeric input vectors for the neural network, necessary for the next stage of processing. Including those identifiers in (2), a transaction can be defined by a quintuple, as shown in (3)

$$\begin{aligned} T =< id_T, uip_T, uid_T, uag_T, \{(t_T^1.url, t_T^1.urlid, t_T^1.time, t_T^1.length), \\ \cdots, (t_T^m.url, t_T^m urlid, t_T^m.time, t_T^m.length)\} > \end{aligned} \tag{3}$$

Finally, the outcome of this preprocessing stage is a transaction file that consists of five-component records, following the format of definition (3).

5. Vector preparation

Transaction files contain indicative data for the user behaviour, however the format of data is improper for clustering by an ART2 neural network simulator. It can use an input of fixed size patterns, each of which is a numeric vector of rank *M* [9]. The vector size *M* can vary depending on the application requirements, but it must be fixed prior to the moment when the network performs learning.

The task of the next stage of preprocessing, called vector preparation, is to convert the transaction records into input patterns applicable to an ART2 neural network simulator. Since the aim of clustering is to group together transactions that represent similar user behaviour, the input patterns for the neural network should contain those transaction components that indicate such behaviour. Two of the transaction components are significant for the clustering: URL identifier $t_T^k.urlid$ and length of reference $t_T^k.length$. The former represents the visited page, while the later shows how long time the user has spent viewing the page. Thus, an input pattern can be defined as a vector of rank *n*, where the *k*-th component corresponds to Web page of URL identifier $t_T^k.urlid = k$. The component *k* has a value that is a total of all length references for that page. Thus, transactions, which have similar Web page references, will be represented by input vectors with non-zero components on similar slots.

Formally an input vector I can be represented by (4)

$$I = (u_1, \ldots, u_N),$$

$$u_k = \sum_{\substack{i \\ t_T^i.urlid = k}} t_T^i.length, \quad 1 \le k \le N \tag{4}$$

Clustering

The ART2 (Adaptive Resonance Theory) artificial neural network (ANN) is one of a hierarchy of neural architectures capable of organising arbitrary sequences of input patterns [9]. The ART 2 ANN is self-organising and unsupervised, designed for the processing of analog, as well as binary input patterns. Its major feature is the ability to overcome the stability-plasticity dilemma, according to which a neural network should be able to learn new input patterns (plasticity) without affecting the storage and recalling of the previously learned ones (stability). ART2 achieves this by relying on a vigilance parameter that determines whether the new input pattern should be accepted as a member of an existing category or, on the contrary, adopted as the prototype of a new category. If this is the case, but the network storage capacity does not allow the creation of new categories, the new pattern is simply rejected, thus preserving the information the network has already acquired. ART networks operation follows naturally on their own architectures and their dynamics can be fully described by differential equations. Due to their unsupervised learning method and fast computations, they provide an efficient way of observing the natural clustering tendencies of the data [11].

ART2 consists of a layer of M input nodes, called F_1, and a layer of N output nodes, called F_2, which are fully connected by two sets of bottom-up and top-down adaptive weights and an orienting subsystem which incorporates the reset mechanism (see Figure 2).

Briefly, the algorithm runs in the following manner: input pattern I is normalised, thresholded (to cut out any part of the pattern falling below the set threshold θ and then renormalised. F_2 node activation is calculated by performing a dot product of the input pattern with the bottom-up weights connecting the input layer to each F_2 node. The F_2 layer is subject to lateral inhibition, the winning node being the one with the highest activity. The vigilance test then checks if the activity of the winning F_2 node (i.e. the match between the input pattern and the bottom-up weights) is greater than the vigilance parameter, set beforehand. If it is, learning takes place at the winning F_2 node, otherwise the winning node is reset and the search continues for a suitable F_2 node. The algorithm continues in this way for all input patterns, and is then cycled a number of times depending on the amount of learning required. The described approach of clustering makes sure that the network always converges to a stable state and still maintains plasticity.

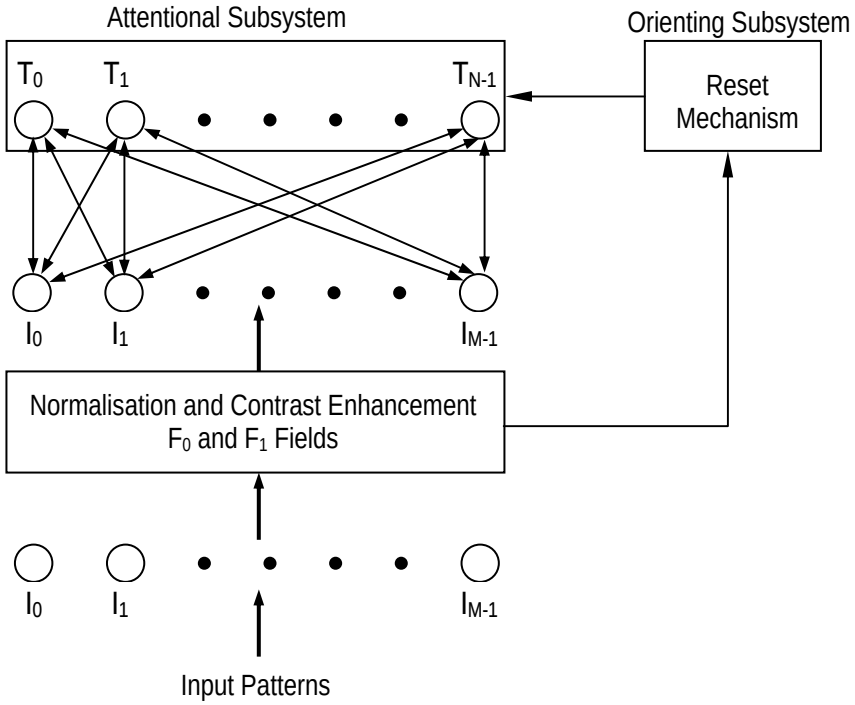

Figure 2. Topology of the ART 2 ANN model. It consists of two input preprocessing fields F0 and F1 and an output field F2 fully connected by a set of bottom-up and top-down weights.

Experiments

This section summarises the experimental results from simulations, which aimed to explore how a neural network simulator based on ART2 architecture clusters weblog entries.

For the purposes of the experiments were used weblog files collected over a 45-days period. Source of the weblog files was the Web server *www.is.nuigalway.ie*, which was set up to log its activity on daily basis using W3CELFF file format, recording only the fields listed in Table 1.

The total size of raw log files involved in the experiments was 162 MB, which included 252430 weblog entries originating from 1990 different IP addresses.

Field	Appears As	Description
Date	*date*	The date that the activity occurred.
Time	*time*	The time that the activity occurred.
Client IP	*c-ip*	The IP address of the client that accessed the server.
User ID	*cs-username*	The registered name of authenticated user who accessed the server or (-) for anonymous users.
Method	*cs-method*	The action the client was trying to perform (e.g. GET).
URL	*cs-uri-stem*	The URL of the accessed resource.
Status	*sc-status*	The status of the action in HTTP terms.
Agent	*cs (User-Agent)*	The client browser

Table 1. Fields of the W3C Extended Log File Format used for the experiments.

At data filtering stage, first the log files were cleared from system messages (see Table 2), and then explored for URL's filename extensions (field *cs-uri-stem* of the weblog records). All entries containing filename extensions, but *htm, html, doc, pdf, txt,* and *xls* were filtered out. Major part of the filtered entries contained extensions *gif, jpeg, jpg, swf, cgi, asp, css, mov, ico, dll, exe,* etc. The number of entries was reduced to 43368 entries, which is approximately 82% reduction rate (see Table 2).

At this stage the information available from the field *cs-method* was used to filter out all entries containing method of access either *HEAD* or *OPTIONS* (see Table 2).

Filtering	Number of entries before filtering	Number of entries after filtering	Reduction in %
System messages	252,430	252,258	0.068
URL extensions	252,258	43,368	82.8
Method of access	43,368	36,266	16.4
Returned code	36,266	29,638	18.2

Table 2. Number of entries before and after filtering of weblog data.

At the end of this preprocessing stage all weblog entries containing HTTP return code different than either 200 (OK) or 206 (fulfilled partial GET request) were filtered out (see Table 2). W3CELFF stores the HTTP return codes in the field *cs-status*.

The next preprocessing stage, user identification, checked the entries for their client IP address (field *c-ip*) and client agent type (field *cs (User-Agent)*) in order to identify users and to group together all entries of each of the users. If either changes, a new user was registered. Table 3 shows a sample of user log data. Each log entry originates from anonymous user (user ID is not recorded) at the same IP address. However, the last three entries have different agent type, which is indicative that the user has changed.

Date	Time	Client IP	User ID	Method	URL	Status	Agent (OS)
2003-03-11	10:07:12	141.204.18.54	-	GET	/default.htm	200	Mozilla/5.0
2003-03-11	10:07:31	141.204.18.54	-	GET	/inks/default.htm	200	Mozilla/5.0
2003-03-11	10:16:10	141.204.18.54	-	GET	/inks/l2.htm	200	Mozilla/5.0
2003-03-11	10:46:28	141.204.18.54	-	GET	/info/default.htm	200	Mozilla/5.0
2003-03-11	10:46:41	141.204.18.54	-	GET	/info/archive/d.htm	200	Mozilla/5.0
2003-03-11	10:47:02	141.204.18.54	-	GET	/contacts/default.htm	200	Mozilla/5.0
2003-03-11	11:25:00	141.204.18.54	-	GET	/sdh/a1/default.htm	200	Mozilla/4.0
2003-03-11	11:25:11	141.204.18.54	-	GET	/sdh/a1/frame1.htm	200	Mozilla/4.0
2003-03-11	11:25:58	141.204.18.54	-	GET	/sdh/a1/frame2.htm	200	Mozilla/4.0

Table 3. Sample information from a W3CELFF file.

After the preprocessing, 3990 users were identified.

At the session identification preprocessing stage each group of weblog entries originating from a particular user were subdivided into sessions using the timeout heuristic. Duration of each reference (excluding the last one) was calculated taking the difference between the time of the next reference and that of the current one. If the duration exceeds the timeout limit of τ=25.5 minutes, that entry was marked as the end of that session. After the session extraction using timeout heuristic, a total of 5698 sessions were identified. The average number of references per session was 13.

At the transaction identification preprocessing stage all weblog entries of duration less than the cutoff time σ=25 were filtered out as auxiliary pages. The value of the cutoff time was calculated using equation (1) with parameters η=0.017 and γ=0.35. After transaction identification, a total of 3346 non-empty transactions were extracted. Average number of references per transaction was 5.78.

At the vector preparation preprocessing stage using (4) were created a total of 3346 input vectors of rank M=277 which formed the input space for the neural network.

The experiments used a neural network simulator based on ART2 architecture with preprocessing field F_0. The differential equations representing the learning rules were solved using the fourth order Runge-Kutta method. The simulator was adjusted for stable performance by proper values of the network parameters according the requirements of the original ART2 architecture described in [9] and [10].

Four series of simulations with different values of vigilance parameter were performed. Figure 4 shows how the parameter ρ influences on the number of clusters created by the neural network simulator. For each simulation the input set of 3346 was presented to the neural network and the number of generated clusters was counted after each 500 input patterns. It was observed that the network creates new clusters rather at the beginning of the series than at the end, which means that the network has an initial period of learning, after which it stabilises. As the vigilance parameter is higher, as the stabilisation period is longer.

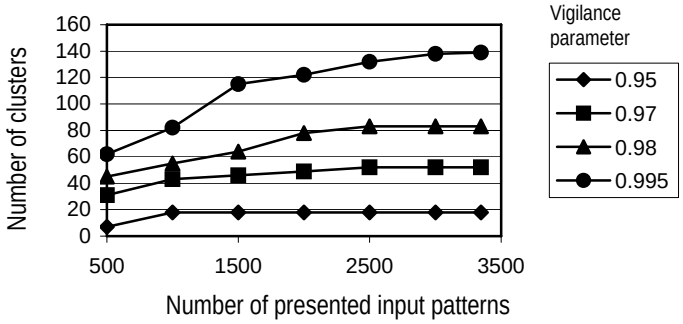

Figure 4. Four series of clustering 3346 weblog transactions by ART2 neural network using different values of the vigilance parameter: 0.9; 0.97, 0.98, and 0.995. The curves represent number of clusters created by the network.

A detailed analysis of the experimental result showed that some of the clusters were overpopulated, whereas others contained small number of transactions. Most of the transaction of the overpopulated clusters contained transactions, which originated from IP addresses local to the Web server.

Conclusion

We have presented an approach to weblog data mining, which uses an ART2 neural network to group together similar Web navigation patterns in clusters of similarity. Such analysis of data relationships among weblog files could be useful for applications that predict Web surfing behaviour. The paper discusses the preprocessing stages necessary to extract meaningful information from the raw weblog data, as well as to prepare input patterns for an ART2 neural network. Due to the characteristics of the ART2 neural network model the proposed approach of clustering offers unsupervised self-learning data mining, which makes the analysing system adaptive to dynamically changing websites.

Bibliography

[1] T. Yan, M. Jacobsen, H. Garcia-Molina, U. Dayal. From User Access Patterns to Dynamic Hypertext Linking. In Proc. of the Fifth International WWW Conference, Paris, France, 1996.

[2] M. Spiliopoulou and L. C. Faulstich. WUM: A Tool For Web Utilization Analysis. In Proc. of the EDBT Workshop WebDB'98, pages 184–203. Springer Verlag, 1999.

[3] H. Mannila and C. Meek. Global partial orders from sequential data. In Proc. of the 6th Intl. Conf. on Knowledge Discovery and Data Mining (KDD2000), pages 161–168, Aug 2000.

[4] J. Pitkow and P. Pirolli. Mining Longest Repeating Subsequences To Predict World Wide Web Surfing. In Proc of the 2nd USENIX Symposium on Internet Technologies & Systems (USITS'99), Oct 1999.

[5] Y. Fu, K. Sandhu, and M. Shih. A Generalization-Based Approach To Clustering Of Web Usage Sessions. In M. Spiliopoulou B. Masand, editor, Web Usage Analysis and User Profiling, pages 21–38. Springer, 2000.

[6] R. Cooley, B. Mobasher, J. Srivastava. Data Preparation for Mining World Wide Web Browsing Patterns. Knowledge and Information systems 1, Springer-Verlag 1999.

[7] C. Shahabi, A. Zarkesh, J. Adibi, V. Shah. Knowledge Discovery From Users Web-Page Navigation. In: Workshop on Research Issues in Data Engineering, Birmingham, England, 1997.

[8] P. Pirolli, J. Pitkow, R. Rao. Silk From A Sow's Ear: Extracting Usable Structures From The Web. In: Proc. of the Conference on Human Factors in Computing Systems (CHI-96), Vancouver, British Columbia, Canada, 1996.

[9] G. Carpenter, S. Grossberg, ART2: Self-Organization of Stable Category Recognition Codes for Analog Input Patterns, Applied Optics,26: p4916:4930,1987

[10] Nachev, A. D. Dimitrova, Parameters of ART2 Neural Networks. In Proc. of the Conference Automatics and Informatics'98, vol.2, Sofia, 1998.

[11] S. Grossberg, Linking Mind to Brain: The Mathematics of Biological Intelligence. Notices of the American Mathematical Society, vol. 47, pp. 1361-1372, 2000.

[12] G. Piatetski-Shapiro, R. Braachman, T. Khabaza, W. Kloesden, and E. Simoudis. An Overview of Issues in Developing Industrial Data Mining and Knowledge Discovery Applications. In Proc. of the Second International Conference on Knowledge Discovery and Data Mining, pp.98-95, 1996

[13] http://netpressence.com/accesswatch/.

[14] http://www.interse.com/.

[15] http://www.netcount.com/.

Author information

Anatoli Nachev - IS, Dept. of Acc. & Finance, NUI Galway, Ireland; e-mail: anatoli.nachev@nuigalway.ie
Ivan Ganchev - ECE, University of Limerick, Limerick, Ireland; e-mail: ivan.ganchev@ul.ie