CLASSIFICATION AND CLUSTERING METHODS IN THE DECREASING OF THE INTERNET COGNITIVE LOAD

Kristína MACHOVÁ, Ivan KLIMKO
Department of Cybernetics and Artificial Intelligence, Faculty of Electrical Engineering and Informatics,
Technical University of Košice, Letná 9, 042 00 Košice, Slovak Republic,
tel.: +421 55 602 4142, E-mail: Kristina.Machova@tuke.sk

SUMMARY

The paper focuses on solving the problem of Internet users' cognitive load decrease based on machine learning methods. It presents the AWS system designed to suggest Web pages. Users are provided with system's suggestions based on their models. The AWS system is based on both classification and clustering machine learning methods. In addition, the system is able to generate information about visitor stereotypes. The system offers information about visitor models for the purpose of server content management. With the help of this information it is possible to customise the content of a server to user needs. The AWS system is an advisory system with off-line learning capabilities. It enables both so individual adaptation and the support of global server content adaptation.

Keywords: *cognitive load, adaptive web, user model, machine learning, classification, clustering*

1. INDRODUCTION

At present, the Web represents one of the most used Internet based services. The number of accesses of various users is almost unbelievable. The Web consists of a vast number of web pages. It is not uncommon when a user stops its browsing through pages to be uninteresting (or unattractive) for him/her although the searched information is presented in these pages. Moreover, a huge number of links were accumulated among web pages. The basic feature of the Web – great number of hypertextual links representing relationships among pages – can be a source of difficulties (turning to a nightmare) when browsing the Web. These problems related to information search and retrieval can be measured by a user cognitive load.

The aim of this paper is to present the AWS system striving for decreasing the cognitive load of Internet users. The focus of the system is on supporting an adaptive web. The adaptive web is able to adapt itself to its visitors – the adaptation is based on an observation of users' activities (the behaviour of users) during users' visits of the Web. This approach is based on an idea of intelligent personal information assistants proposed in [4]. This concept of adaptive web represents an automatic change of documents driven by the behaviour of visitors. The automatic adaptation is performed by a server – and the adaptation can represent a change of the content of relevant documents, a change of a description of these documents, and/or a change of hypertextual links.

There are systems for the adaptive web which employ various techniques based on marking web pages by users (collaborative filtering) [6]. A user receives suggestions in concordance with interests of those users who are marked in the same way as the given user or who have the same or similar interests (e.g. Amazon.com portal). Other systems are able to perform page prediction (WebWatcher, AVANTI). A specific category is made up by systems inspired by neural nets based on Hebb learning [1]. Also, the system that enables a semantic search in a semantically annotated web domain was developed [7].

2. THE AWS SYSTEM

The AWS system (Adaptive Web System) employs supervised and unsupervised methods from the field of machine learning.

The AWS system focuses on the development of user models from users' requirements. Such a model type can be used to customise response to a user requirement – the user is provided only with those documents which are relevant to his/her profile (i.e. his/her model). User models are constructed using heuristic machine learning methods. The learning is based on logs of web servers. The AWS system represents an advisory system with off-line learning, individual adaptation (customisation for each particular user based on his/her individual model), and the support for global server adaptation (transforming pages into the form suitable for majority of visitors).

2.1. Used methods

The system employs two methods for heuristic search of concept space (namely HGS and HSG) which belong to supervised methods of machine learning [2], [3] and are applicable for solving classification tasks. In addition, one clustering method (CLUSTER/2) belonging to unsupervised methods was used.

Machine learning methods are generally based on a set of training examples and achieved results are tested using a set of test examples. Training and test examples constitute a set of typical examples. The typical examples are represented as a set of n attributes with their values. The last attribute can represent (in case of supervised learning) a class to which the given example belongs.

A set of typical examples is the most often given in the form of a table. An example is given in Table 1 presenting typical examples in the form used by the AWS system. This table contains typical examples characterised by an attribute *A* and belonging to a class *T*. The examples represent accesses to server pages. The attribute *A* (url) characterises those pages which were accessed (each page is stored on the server together with a set of key words which characterise the content of the page). The attribute *T* (user ID) identifies users who accessed the given pages and in this way it specifies the class to which the given access belongs. It is quite common, that several users visit the same page – and the same typical example is classified into more than one class at the same time.

Number	A (url)	T (userID)
1	/som.php	USER3eafc6cd8c98a
2	/maxnet.php	USER3eafc6cd8c98a
3	/ns_top.php	USER3eafc6cd8c98a
4	/cobweb.php	USER3eafc71274ad9
5	/id3.php	USER3eafc71274ad9
6	/c45.php	USER3eafc71274ad9
7	/pid1.php	USER3eafc7413857e
8	/psd1.php	USER3eafc7413857e
9	/plc.php	USER3eafc7413857e

Tab. 1 A set of typical examples obtained from a log of a www server

Classification represents the decision on a class of a new example (with unknown class value) based on definitions of classes which were constructed using some machine learning method.

The AWS system relies on using HGS (Heuristic General to Specific) and HSG (Heuristic Specific to General) methods [3], [4]. Both methods differ from exhaustive search of concept space – they do not search all concept space but the most promising hypotheses only. How promising particular hypotheses are, can be calculated by a score heuristic function. Each algorithm iteration considers only a limited number of hypotheses (with the highest score) – this number is defined as Beam Size (BS).

Both methods (HGS and HSG) use the principle of limiting the concept space to be searched. They differ in the direction of search. The HGS algorithm searches the concept space from more general concept descriptions to more specific (GS search direction). On the other hand, the HSG algorithm searches the concept space from more specific concept descriptions to more general ones (SG search direction).

Clustering methods can be applied when training examples do not contain any information about the class they belong to. In this case the examples can be grouped into natural groups or clusters using techniques of unsupervised learning (there is no feedback in the form of a class defined in advance). The clustering process starts with a set of objects – training examples. The aim is to create a set of clusters and distribute all available training examples over the set of clusters. In general, it is possible to distinguish several different approaches to clustering: iterative, conceptual, hierarchical, and probabilistic. The AWS system employs the CLUSTER/2 clustering method [5].

2.2. Structure of the AWS system

The AWS system was designed with the aim to enable suggestions of pages to a user based on his/her model and to carry out an individual adaptation. The user model is generated as a result of heuristic search of concept space using the HGS and HSG algorithms.

At the same time, the system provides information about models of server visitors and their interests in order to support server content management. In this way it contributes to customising the content of a server to users – it supports a global server adaptation. This feature of the system is backed up by the CLUSTER/2 clustering technique.

The AWS system can be characterised as an advisory system with off-line learning, individual customisation and supporting global adaptation of the server content.

As depicted in Figure 1, the system consists of two parts: on-line and off-line. The on-line part is responsible for the identification of visitors and subsequent generation of suggestions based on visitors' models. The off-line part of the system is responsible for development of user models and providing information vital for the adaptation of the server content. The learning itself is performed utilising information about the content of a server and a log of the given server. The application requires the server log in the NCSA Combined Log format.

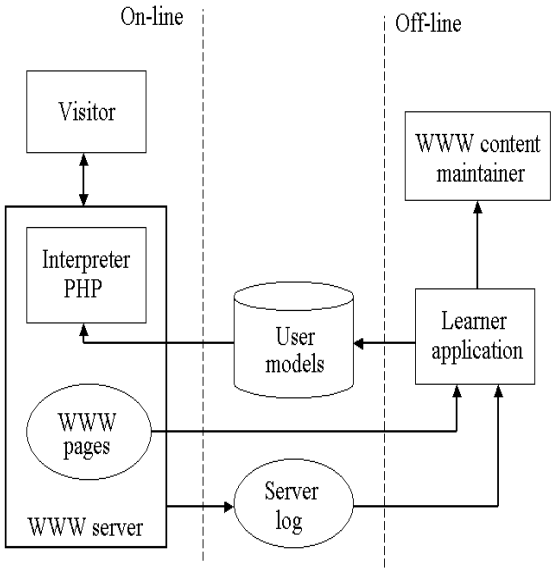

Fig. 1 Structure of the AWS system

The shared part of the system is represented by a database storing user models and server logs with information about processed user requirements. The AWS system enables to identify a visitor using his/her IP address or using cookies. The identification using cookies seems to be more suitable.

The on-line part consists of a www server, PHP interpreter, and a database of user models. A visitor sends his/her requirements on the www server. The server delivers these requirements to the PHP interpreter. The interpreter identifies the visitor and generates a query into the database. Using this query the interpreter retrieves addresses and names of pages to be suggested (based on the model of the identified visitor) and sends this suggestion to the visitor together with the page which was required by the visitor. If cookies are used to identify visitors, then in case that the visitor cannot be identified (e.g. because the user accesses the server for the first time or he/she deleted cookies in his/her web browser) a new unique identifier is generated. This identifier is sent to the client and stored on his/her disc in the form of a cookie for a subsequent identification.

The off-line part of the system represents a system mainstay. It consists of the AWS/Learner application. This application is responsible for learning (creation of user models using heuristic algorithms), populating the developed user models with relevant web pages, and for the support of the server content management based on clustering of the user models using a clustering algorithm.

2.3. Testing

The AWS system was preliminary tested on a set of 31 web pages drawn from fields of artificial intelligence and control. Activities of 8 visitors who accessed selected web pages were considered during the testing. The testing itself was divided into two phases. The first phase was devoted to tests relevant to generating suggestions for users. The focus was on time requirements of tested algorithms and the difference between concepts sets generated by these algorithms. The second phase aimed at tests relevant to clustering – fitness of generated cluster descriptions and their numbers of occurrence were examined.

The tests were performed using a machine with a 400 MHz AMD K6-2 processor, 152 MB RAM and Windows 2000 Professional operating system.

In order to compare time requirements of HGS and HSG (Figure 2), average processing times in seconds were used. From the graph it is obvious that HGS requires much more time to find a solution. The reason is that solutions to be found are on a rather low level of generality so the HGS algorithm has to traverse via more levels than the HSG algorithm. The graph also proves that the HGS "heuristically modified" is the slowest variant and its time requirements steeply increase with increasing BS. The rise of requirements of HGS "with deletion" is more moderate. Difference between HSG "heuristically modified" and HSG "with deletion" are not too relevant and time requirements of both algorithm versions increase only slightly with increasing BS value. Deletion of contradictory examples when using HGS represents the reduction of search space resulting in shortening average time necessary to find a solution. On the other hand, the deletion of contradictory examples enables the HSG algorithm to reach higher levels of generality which results in longer processing times.

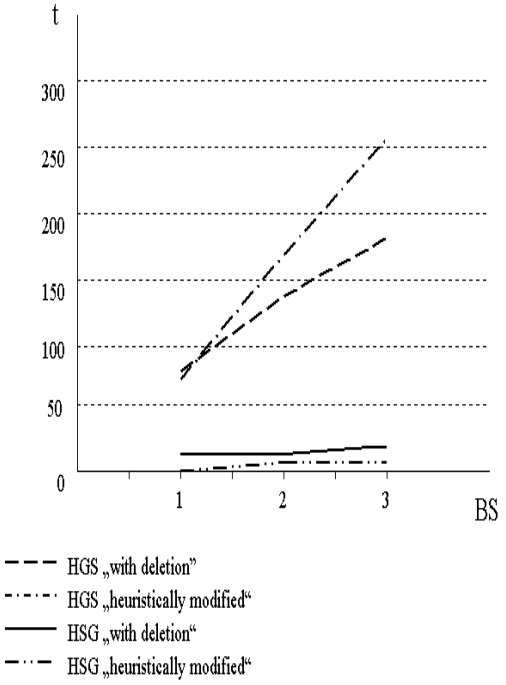

Fig. 2 Comparison of time requirements of HGS and HSG

The solutions which were found with the aid of HGS "heuristically modified" are illustrated in Table 2. And those which were found when using HSG "with deletion" are in Table 3.

User	Model
USER3eafc6cd8c98a	compet_learning, MAXNET, NN, SOM, topology
USER3eafc71274ad9	CN2, COBWEB, HCT, IWP, threshold concepts
USER3eafc7413857e	PLC, programming
USER3eafc77ec0dbc	CLUSTER/2, SOM, clustering
USER3eafc79999251	NEX, decision lists
USER3eafc7bca8371	Fuzzy controller
USER3eafc7dd6b312	Modules, programming, simatic, SLC
USER3eafc807499a2	C4.5, CN2, fuzzy controller, NEX, PLC, decision lists, simatic

Tab. 2 Models which were generated by HGS "heuristically modified"

User	Model
USER3eafc6cd8c98a	**AI, clustering**, compet_learning, MAXNET, NN, SOM, topology
USER3eafc71274ad9	(CN2), **AI**, **ML**, **decision making**, **decision trees**, **ID3**, COBWEB, HCT, IWP, threshold concepts
USER3eafc7413857e	**control**, **controllers**, step_less, **PID**, PLC, (programming)
USER3eafc77ec0dbc	**AI**, **ML**, CLUSTER./2, SOM, clustering
USER3eafc79999251	**AI**, **ML**, **decision making**, **decision trees**, **ID3**, NEX, decision lists
USER3eafc7bca8371	**control**, **fuzzy**, **controllers**, fuzzy controller
USER3eafc7dd6b312	**control**, **automats**, modules, programming, (simatic), SLC
USER3eafc807499a2	**AI**, **ML**, **decision making**, **decision trees**, **ID3**, C4.5, (CN2, fuzzy controller), NEX, (PLC, decision lists), simatic

Tab. 3 Models which were generated by HSG "with deletion"

Table 3 contains some keywords in bold (or in parentheses) – those keywords are superfluous (or missing) when comparing the table with the most specific results in Table 2. User models generated by HGS "heuristically modified" are more precise since they are composed of fewer keywords and therefore user interests are more sharply defined.

The presented results suggest that slower HGS algorithm provides more accurate results. The HGS algorithm performs best with classification threshold equal to 0.33 or 0.5. The HSG algorithm prefers values 0.5 and 0.6.

3. CLUSTERING

Clustering was based on the CLUSTER/2 method. Based on the structure of users, 2 and 4 clusters were considered. When generating two clusters, the following clusters were found:

Cluster 1: automats, fuzzy, fuzzy controller, modules, PID, PLC, programming, controllers, control, SLC, step_less. Three users with their primary interests in control were allocated in this cluster.

Cluster 2: C4.5, CLUSTER2, COBWEB, HCT, ID3, IWP, compet_learning, MAXNET, NEX, NN, threshold concepts, decision trees, decision lists, decision making, simatic, SOM, ML, topology, AI, clustering. This cluster consisted of five users primary interested in machine learning.

In a similar way, results for four clusters are illustrated in the Table 4.

Cluster	Cluster description	Number of visitors
1	C4.5, CLUSTER/2, COBWEB, HCT, ID3, IWP, NEX, threshold concepts, decision trees, decision lists, decision making, simatic, SOM, ML, AI	4
2	Automats, modules, programming, control, SLC	1
3	Fuzzy, fuzzy controller, PID, PLC, controllers, control, step_less	2
4	compet_learning, MAXNET, NN, SOM, topology, AI, clustering	1

Tab. 4 Structure of clusters generated by COBWEB/2

These four clusters are of less help than previous two clusters. The first and fourth clusters can be merged into one cluster, similarly can be merged the second and third clusters. This is backed by a weak support of the second and fourth clusters (measured in the number of users) – it preordains these two clusters to be merged with bigger ones.

4. CONCLUSION

The paper focuses on problems of adaptive web and machine learning. A description of the AWS system is presented – the system which was designed as an advisory system with off-line learning capabilities, possibility of an individual adaptation and with the support for a global content adaptation. The system was preliminary tested on a set of web pages. In the future, it would be worth validating usability of the system for large information spaces. Presented test results have proved the HGS "heuristically modified" algorithm as more accurate and the HSG algorithm as faster algorithm.

The presented approach can be further extended using an ontology for orientation and movement in the ordered concept space. Another improvement can be reached by employing a method for automatic extraction of key words from documents in order to replace the manual web page description/annotation.

The work presented in the paper was supported by the Slovak Grant Agency of Ministry of Education and Academy of Science of the Slovak Republic within the 1/1060/04 project "Document classification and annotation for the Semantic web".

REFERENCES

[1] Benko, P.: *Internet – heterogeneous distributed information system* (in Slovak), alf.fei.tuke.sk/publ/2202/InetHDIS.ps, FEI TU Košice, 2002, 60 pages.

[2] Berka, P.: Dobývání znalostí z databází. Academia – nakladatelství Akademie věd České republiky, Praha, 2003, 366 stran, ISBN 80-200-1062-9.

[3] Machová, K.: *Machine learning. Principles and algorithms* (in Slovak), ELFA s.r.o., 2002, Košice. ISBN 80-89066-51-8.

[4] Michalski, R.S.: Pattern Recognition as Rule-guided Inductive Inference. *IEEE Trans. on Pattern Analysis and Machine Intelligence, 2*, 1980, 349-361.

[5] Michalski, R.S., Stepp, R.: Automated Construction of Classification: Conceptual Clustering versus Numerical Taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence, No.5,* 1983, 219-243.

[6] Pareschi, R., Borghoff, U.M.: *Information Technology for Knowledge Management*. Springer-Verlang, Berlin, Heidelberg, 1998, ISBN 3-540-63764-8.

[7] Svátek, V., Vacura, M.: Automatic Composition of Web Analysis Tools: Simulation on Classification Templates. RAWS 2005, Proc. of the 1st International Workshop on representation and Analysis of Web Space, VŠB-Technical University of Ostrava, TiskServis, Ostrava, 2005, 78-84, ISBN 80-248-0864-1.

BIOGRAPHIES

Kristína Machová was born 16.02.1961. In 1985 she graduated (MSc.) with honours at the Department of Cybernetics of the Faculty of Electrical Engineering at Technical University in Košice. She defended her PhD. in the field of machine learning in 1996; her thesis title was "Knowledge Modification with the Aid of Explanation Based Learning". Since 1985 she is working as a lecturer at the Department of Cybernetics and AI of the Faculty of Electrical Engineering and Informatics at Technical University. Since 1988 she is working as an assistant professor at the same department. Her scientific research is focusing on knowledge based systems, machine learning and meta-learning, basic principles of the learning algorithms and so on. In addition to this, she also investigates the questions related to the adaptive web.

Ivan Klimko was born 7.7.1980. In 2003 he graduated (MSc.) with honours at the Department of Cybernetics and AI of the Faculty of Electrical Engineering at Technical University in Košice. He defended his MSc. in the field of machine learning in thesis "Adaptive WEB and Machine Learning Methods". Since 2003 he is working as a software manager for UNICOM Ltd. and his work is focused on PKI, electronic signatures, electronic registries and document management. In addition to this, he also investigates the questions related to the adaptive web.