Value Estimation Based Computer-Assisted Data Mining for Surfing the Internet

Bálint Gábor
Department of Information Systems,
Eötvös Loránd University,
Pázmány Péter sétány 1/C,
Budapest, Hungary H-1117.
E-mail: gbalint@caesar.elte.hu

Zsolt Palotai
Department of Information Systems,
Eötvös Loránd University,
Pázmány Péter sétány 1/C,
Budapest, Hungary H-1117.
E-mail: zspalotai@vnet.hu

András Lőrincz*
Department of Information Systems,
Eötvös Loránd University,
Pázmány Péter sétány 1/C,
Budapest, Hungary H-1117.
E-mail: lorincz@inf.elte.hu

Abstract— **Gathering of novel information from the WWW constitutes a real challenge for artificial intelligence (AI) methods. Large search engines do not offer a satisfactory solution, their indexing cycle is long and they may offer a huge amount of documents. An AI-based link-highlighting procedure designed to assist surfing is studied here. It makes use of (i) 'experts', i.e. pre-trained classifiers, forming the long-term memory of the system, (ii) relative values of experts and value estimation of documents based on recent choices of the users. Value estimation adapts fast and forms the short-term memory of the system. All experiments show that surfing based filtering can efficiently highlight 10-20% of the documents in about 5 steps, or less.**

[*] Corresponding author.

I. Introduction

The number of documents on the World Wide Web is increasing quickly; it has passed 1 billion in 2001 and might reach 10 billions soon. The number of new documents published on the WWW is much over 1 million per day. The number of documents that change on a daily basis, e.g. documents about news, business, entertainment, etc., is even larger. The ever-increasing growth presents a considerable challenge in finding, gathering and ordering information on the web. Gathering of novel information is not efficient with conventional search engines. First, the information they present is not up-to-date. Second, these engines may offer hundreds or even thousands of documents, many of which are not really relevant. Other documents could be 'traps', e.g., by making use of particular (sometimes fake) keywords, or being simply collections of documents indexed in every 'dimension' of the web.

Specialized crawlers, possibly intelligent personalized crawlers may circumvent these problems. The development of such crawlers represents a real challenge for methods of artificial intelligence and has been attempted by several research groups [1], [2], [3], [4]. Intelligent crawlers can serve as pre-filtering methods. Our question is how an intelligent crawler could serve human intelligence to search and find information.

Our aim was to establish a connection between human intelligence and artificial intelligence. While the user surfs the web, our system tries to choose the links which are likely the most relevant to him. The level of relevance – the *value* – of a document is estimated by the previous decisions of the user. With this method, the best links can be highlighted in the browser (for example shown in a different color), so as to make the choice between the links easier.

To this end, we developed a set of artificial 'experts' (classifiers) with different methods. Our solution is an alternative to other expert methods, such as the method of mixture of experts and the method of product of experts. Mixture of experts represent a collection (union) of example sets [5]. Product of experts, on the other hand, makes use of multiplicative probability estimation [6].

The method studied here is favored because of its fast adaptation. It has the following main characteristics. We used a set of text classifiers which had rather sharp decision surfaces. In turn, the output of the classifiers is a vector with values close to $+1$ or -1. Output $+1$ (-1) means that the input belongs (does not belong) to the classifier. In a given decision problem either the positive (within class) or the negative (not within class) outputs of the classifiers could be of importance. It is also possible that neither of these outputs has any relevance to the actual decision making problem. In turn, an expert should have a weight, which can assume positive, zero or negative values and these weights can serve *value estimation* for each document. Value estimation can be improved by reinforcing feedback provided by the user [7].

The paper is organized as follows. First, our methods are described in Section II. This section contains the description of the databases used for testing as well as the description of the adaptation algorithms studied. Section III describes the results of the investigations. This section is followed by the discussion on the found properties of different methods (Section IV). Conclusions are drawn in Section V.

II. Methods

A. Value estimation and user modelling

The task can be interpreted as a prediction of the next decision to be made by the user. Two types of questions can be asked during the process. (i) Which is the next document to be chosen by the user? (ii) If we rank the links, how good is this ranking? We shall say that the goodness of our ranking is 90% on average, if on average only 10% of the documents

0-7803-8359-1/04/$20.00 ©2004 IEEE

has been given better ranks by the learning algorithm than the document selected by the user model. That is, if goodness of ranking is above 90% then about 10% of the documents could be highlighted, i.e., reordered, or colored.

To allow fast adaptation we assumed that a surfer follows a – possibly changing – goal and that this goal can be characterized by weighted mixtures of classifiers. Our assumption allows us – in principle – to tolerate a large number of possible goals. To give a rough estimation, let us restrict the choice of weights to ± 1. Then the number of possible goals is 2^{50} for 50 classifiers formed by the 50 clusters. In our experiments we made several restrictions on the possible goals, but the type of restrictions were not known for the learning algorithm.

The outputs of the classifiers were continuous, mostly falling in the neighborhood of -1 and $+1$ values.[1] The output of a classifier can be interpreted as 'reporting' ('expressing opinion') with some uncertainty that a document belongs to the class or not.

We conducted our experiments using user models. These models were simple weight vectors, with several restrictions on their values (see Section II-D). In some tests the model user could change the topic of interest in order to test the speed of adaptation: a random change of the weight vector was used to test this property.

We asked two questions. The first question was how fast and how precisely the weights of the user model can be estimated during surfing. The second question was whether different classifiers can describe the same goal.

So, in our approach, it is assumed that the behavior of any user can be approximated by a relevance (or weight) vector $\mathbf{W}^T = (W_1, \ldots, W_n)$. For a model user, the value of document S is the scalar product of the output vector ($\mathbf{S}^T = (S_1, \ldots, S_n)$, concatenated from the outputs of the classifiers) and the weight vector, where superscript T denotes transposition. In turn, the value of document S is estimated by our model as

$$V(S) = \sum_n W_n S_n = \mathbf{W}^T \mathbf{S} \qquad (1)$$

Figure 1 depicts value estimation using this weight vector method. User selects the document with the highest value. The goal of the user identification algorithm is to identify the weights of the user.

B. Databases

To study the problem, first, a model user and a model database were created and studied. The third case concerned highlighting during real Internet searches performed by our user model. In this section databases are described.

1) Artificial database generation: A model problem made use of a database defined by an algorithm. The state of a document was represented by an n-dimensional random number ($n = 50$), whose components were chosen from the set of $\{-1, +1\}$, and with an additive superimposed real number chosen from a Gaussian distribution of 0.1 variance. This n-dimensional number was regarded as an output vector of the classifiers for a 'document'. In each trial, we generated data of 100 'documents', available at the actual step of user model.

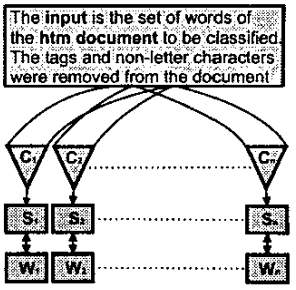

Fig. 1. **Value estimation**
There are n different classifiers ($C_i, i = 1, \ldots, n$). If a new page is downloaded then each classifier provides an output, which is often close to $+1$ or to -1. These real numbers form the classifiers' output vector $\mathbf{S}^T = (S_1, \ldots, S_n)$. The value of each document is estimated by means of the weight vector $\mathbf{W}^T = (W_1, \ldots, W_n)$. The value is given as a scalar product $V(document) = \mathbf{S}^T \mathbf{W}$. Users are modelled by their weight vectors.

2) Tests on the Internet: Highlighting experiments were conducted on the Internet. The starting node was the main page of Geocities. No other restriction was applied; the user could surf the whole web.

In the Geocities and the Internet tests we assumed that the user could use the 'back' button or the 'history' utility, so any link seen so far is available at each step.

C. Clustering and classification

The downloaded portion of the Geocities database (i.e. 90,000 html documents) were separated into 50 basic clusters by Boley's classification method [8] known to perform well on texts.

For classification of documents in the clusters, the probabilistic term frequency inverse document frequency (PrTFIDF) classification method [9] was used. The term frequency vector had 4,000 components.

D. Restrictions on user models and highlighting methods

In all experiments users could not return to documents, which had been visited previously.

1) The simplest user model on the model database: The user was modelled by a vector with two non-zero components. The values of these components could be either $+1$ or -1.

2) User models and highlighting systems on the Internet: Three different types of experiments were conducted on the internet, which are depicted in Fig. 2. Each subfigure contains the classifier used to model the user and the classifiers used for highlighting.[2]

In the first type of experiments – from now on it will be referred to as *scheme fullC* (Fig. 2) – the user was defined

[1] On our collection from the *Geocities* database, about 5% of the classifier outputs fell into the domain $(-0.9, +0.9)$, whereas 95% of the outputs were close to either -1 or $+1$.

[2] Note, that the two-classifier user model identification problem is not depicted in the figure.

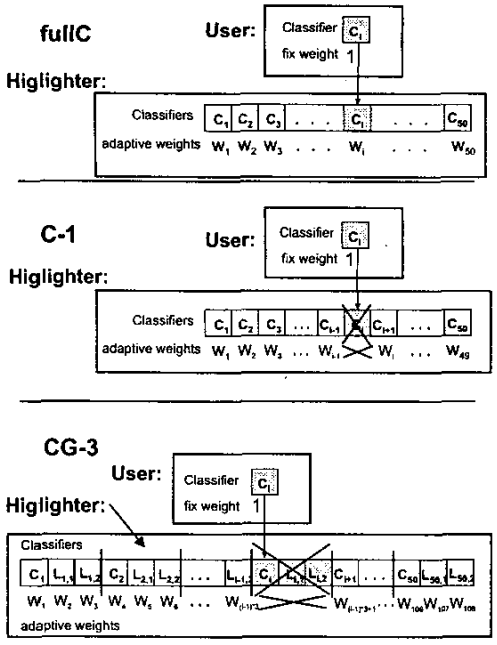

Fig. 2. **Schemes of the studies.**
In all cases the user was one of the Boley classifiers. Top (scheme fullC): every Boley classifier takes part in highlighting, middle (scheme C-1): every Boley classifier except the user's classifier takes part in highlighting, bottom (scheme CG-3): all Boley classifiers and the belonging context graph classifiers but those of the user take part in highlighting. Note, that the two-classifier user model identification problem is not depicted in the figure.

by one Boley classifier, with weight value 1. The highlighting classifier system contained all the Boley classifiers, like in the previous case.

In the second type of experiments, *scheme C-1*, the same kind of user was assumed (i.e., user was defined by one cluster), but the classifier which represented it was taken off from the set of classifiers in the highlighting system. In turn, value estimation could not use that classifier. The user did not belong to the convex hull of the classifiers and the learning system had to approximate the class, which defined the user. This second test concerns the case when our system tries to predict a user, 'who' does not match any of our classifiers.

Note that the dimension of the term frequency vector is 4,000. In our tests each classifier (out of the 50 classifiers) was tried as a user. Single value decomposition (see Fig. 3) demonstrates that the frequency vectors of the 50 classifiers are not linear combinations of each other, and therefore, perfect identification is not possible.

In the third type of experiments, moves of the user were predicted by classifiers developed in a different manner. A novel and efficient approach in Internet crawling [10], [7], [11] makes use of *context graphs* (CG) of relevant documents and we applied such context graph classifiers. CGs are constructed as follows: the nodes are documents on the Internet, the oriented edges are links from one site to another. A CG is a tree, its 0th level (the 'root' level) contains documents from

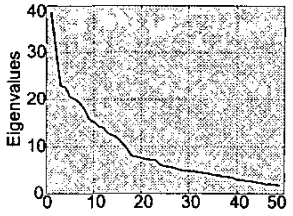

Fig. 3. **Single value decomposition of a matrix made of 50 Boley classifiers**
The size of the matrix is 50 × 4000, it's rank is 50. Taking out 1 of the classifiers, the rest can not reconstruct that vector by linear combination of the others.

the cluster to which it belongs. Its 1st level contains the documents which have links pointing to a document at the 0th level. Its 2nd level includes documents which have links to a document at the 1st level, but do not have any link to a 0th level document, and so on.

Context graphs were developed for each Boley class. The centers of a Boley class (as provided by Boley's method [8]) were chosen as root nodes of the corresponding context graph. Our search for documents to build the context graph was successful for 37 classes; in these cases we found more then 100 documents for 2 CG levels around the Boley clusters. To each of the 37 × 3 classes, a classifier was built using the cited PrTFIDF classification method. A classifier represented a decision surface between documents of a cluster and the other documents, which did not belong to this cluster.

The user was modelled by one of the Boley clusters, and the classifiers belonging to the context graph of this Boley cluster were removed from the classifier vector of the highlighting system. The rest of the PrTFIDF classifiers (i.e, 36 × 3 classifiers) were used to predict the choices of the model user (scheme CG-3 of Fig. 2).

E. Adaptation

1) On-line reinforcement learning (RL): RL is the state-of-the-art method in value estimation and it can deal with function approximators.[3] The RL algorithm for learning linear weighting is relatively simple:

$$\delta(t+1) = r(t+1) + \gamma \mathbf{W}^T \mathbf{S}(t+1) - \mathbf{W}^T \mathbf{S}(t) \quad (2)$$

$$\mathbf{W}(t+1) = \mathbf{W}(t) + \alpha \delta \mathbf{S}(t+1) \quad (3)$$

where $\mathbf{W}(t) = (W_1(t), \ldots, W_n(t))$ is the estimated weight vector (or preference vector) of the user ($n = 50$) at surf step t, $\mathbf{S}(t) = (S_1(t), \ldots, S_n(t))$ is the output vector of the classifiers for the document visited at surf step t, γ is the discount factor, $r(t+1)$ is the immediate reward after the $(t+1)^{th}$ step is made by the user, $\delta(t+1)$ is the error of the value estimation, α is the learning rate, which may depend on time, and Eq. (3) is the update rule of the weight vector. The immediate reward can be determined by the designer of the learning system. In our case the immediate reward was

[3]For an excellent recent introduction see [12]. Utilization of RL principles in internet surfing can be found in [13], [7].

zero if $\mathbf{W}^T(t)\,\mathbf{S}(t+1)$ was the maximum value among all of the neighboring documents. Otherwise, $r(t+1) = -1$ was utilized. Different discount factors, including no discount ($\gamma = 1.0$) were tried. Under certain conditions, RL warrants that the estimation of the value function will reach the global optimum [12]. Conditions of such theorems include that the states are known and are Markovian, reward could be stochastic but the no probability distribution function may change during time. Notice that these conditions are not met in web surfing.

2) Learning with moving window technique: The second method utilized an exponentially weighted update rule:

$$\mathbf{W}(t+1) = (1-\alpha)\mathbf{W}(t) + \alpha\mathbf{S}(t+1)$$

Here α was decreased with $1/(\kappa * t)$ and $\kappa = .1$ was chosen. This rule minimizes the mean square error $J = \frac{1}{2}||\mathbf{W} - \mathbf{S}||^2$ according to the Robbins-Monro criterion. The problem with this rule is as follows: weights adapt to the input even when the weights are perfect. In turn, internet regions on different topics may spoil the perfect weight vector.

3) Learning with value estimation error modulated moving window (MW value estimation, MWV) technique: The third method avoids uncertainties about the learning rate. The advantage of the method is that the learning rate becomes zero automatically if no learning is necessary. This third method is a self-consistent combination of value estimation and the moving window method. The algorithm is as follows:

$$\delta(t+1) = \mathbf{W}^T(t)\,(\mathbf{S}^*(t+1) - \mathbf{S}(t+1)) \qquad (4)$$

$$\mathbf{W}(t+1) = \frac{(1-\alpha\delta(t+1))\mathbf{W}(t) + \alpha\delta(t+1)\mathbf{S}(t+1)}{|(1-\alpha\delta(t+1))\mathbf{W}(t) + \alpha\delta(t+1)\mathbf{S}(t+1)|}, \qquad (5)$$

where α was set to constant to follow online changes, $\mathbf{S}^*(t+1)$ is the best selection according to the value estimation, $\mathbf{S}(t+1)$ is the selection of the user, and $\delta(t+1)$ is the value estimation error computed at surf step $t+1$. It is easy to see that the estimated value of the user selected link will be increased by Eq. (5), provided that the S vectors at each step are normalized $|\mathbf{S}(t)| = 1$ for $t = 1, 2, \ldots$. In this case, $\mathbf{W}(t+1)^T\mathbf{S}(t+1) \geq \mathbf{W}(t)^T\mathbf{S}(t+1)$.

If the user is properly represented by the learned weights, no further weight tuning may occur under this learning rule. Although the algorithm estimates the value, it is not making use of cumulated long-term value estimations, thus it is not an exact RL method.

III. Results

A. Results on artificial 'documents' and 'links'

Computer runs were averaged over 230 independent simulations. In each simulation 200 steps were executed by the model user. Averaged results are shown in Fig. 4. In each computer run we assumed that the user's behavior changes at discrete time steps. For the sake of visual inspection, the occurrences of changes were restricted to the 50th, 100th, and 150th time steps. The figure depicts the average goodness of ranking.

Reinforcement learning performs rather poorly (dotted 'line'). Performance of RL improves rather slowly if there is any improvement at all (Fig. 4). Note, that a goodness value of 50% means that on average, half of documents have better ranking. In turn, random choice from the two options (i) highlight and (ii) do not highlight has a goodness value of 50%.

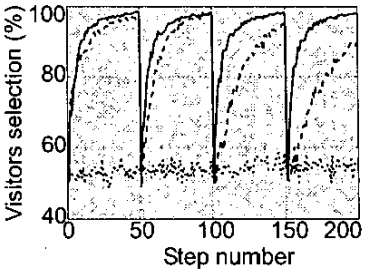

Fig. 4. **Comparison of different methods.**
Choice of model user is averaged over 230 generation of 200 link-sets with 100 links in each set. The graph shows the goodness of ranking. The higher the number, the better the highlighting. Solid line: update using MWV, dashed line: moving window update, dotted line: reinforcement learning. User switched weights randomly in every 50 steps. In the moving window update experiment the learning rate was proportional to $1/t$.

The moving window estimation is relatively good (dashed line). This method is sensitive to the choice of the learning rate. This can be seen clearly in Fig. 4, because the $1/t$ behavior of the learning rate provides striking deterioration of performance. The moving window method falls below the MWV method (solid line). Of particular importance is the first 25 steps of our experiments. Within this time domain, the learning rate of the moving window method changes quickly. According to the figure, the initial rise of the moving window method is faster than that of MWV method. This indicates that the learning rate chosen for the MWV method is suboptimal. The sharp rise of the moving window saturates and falls below that of the method using value estimation error modulation.

One can argue that an optimal choice of the learning rate is not possible. The choice of large rates allows fast adaptation but prohibits fine tuning. On the other hand, small rates allow for fine tuning but may deteriorate performance when changes are fast – as it is demonstrated by Fig. 4. Moreover, there is a pitfall for this moving window estimation: this method modifies the weights in regions where all estimated values are low (i.e.,where all neighboring documents have low values for the user). In such regions highlighting is not relevant. Also, no change of weights is necessary in regions where the error of value estimation is small. The MWV adaptation (Eq. 4) avoids these problems by construction.

Value-estimation error has other advantages, e.g., it can serve as an indicator to measure changes of user's behavior. This is demonstrated in Fig. 5

B. Results on the Internet

Four different studies were conducted on the Internet (see Section II-D.2), each of them used the MWV technique.

In Study 1 and in Study 2 schemes fullC and C-1 were applied, respectively. Complete tests were made, all classifiers

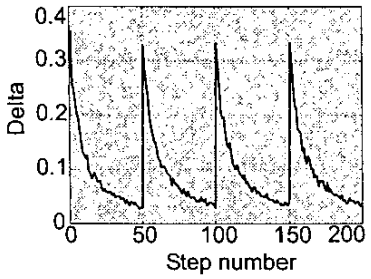

Fig. 5. **Value-estimation error as a function of time.**
Changes of the weight vector of the user model occur in every 50th step in the experiment. Value estimation error can be used to monitor changes of the behavior of the user.

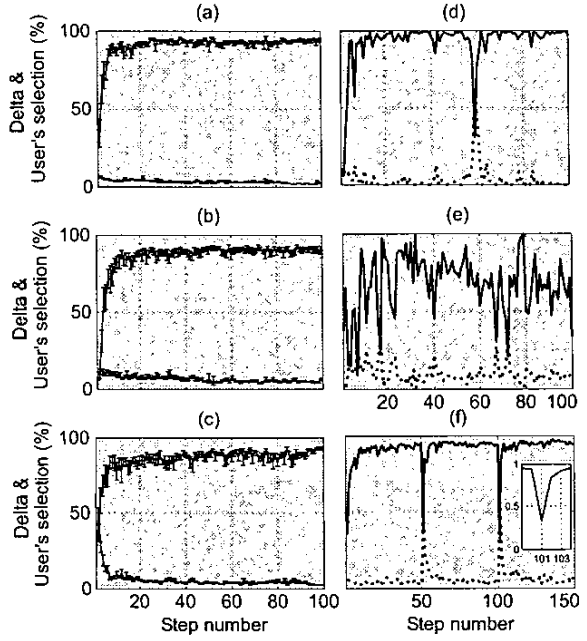

Fig. 6. **Performances and individual cases.** (a): user is from the convex hull of the Boley classifiers, (b): the classifier of the user is excluded from Boley classifiers used for estimation, and (c): the set of classifiers is extended by classifiers belonging to levels of context graphs. (d): The best highlighting result on the Internet for case (b) of this figure, (e): highlighting with the worst result on the Internet for case (b) of this figure, (f): case (a) but with 'user' who changes behaviors. Inset: fast adaptation at step number 100. Upper curves: performances. lower curves: value estimation errors. Error bars denote variances.

belonging to the Boley clusters were tried as user models in each study. Averaged results (with error bars representing variances) are depicted in Fig. 6(a) and (b). In Study 3, scheme CF-3 was tested. All the 37 classifiers which had CF clusters were tried as users. Averaged results are shown in Fig. 6(c).

Variances are smaller than standard deviations, which – if centered around the average values – would represent a region sometimes beyond 100%. This occurs, because performance, which is good on average, was heavily corrupted from time to time. To show this, performances and value-estimation errors are shown in Figs. 6(d) and (e) for the best and for the worst cases of Study 2.

Subfigures (a-c) of Fig. 6 demonstrate that performances are similar for the three different user types. However, – according to subfigures (d) and (e) – performance changes from user to user.

In Study 4 we repeated Study 1, but the user was randomly changed in every 50th steps, to study the speed of the adaptation. Results are shown in Fig. 6(f). Note that the initial transient is relatively long at start, but it becomes much shorter (2 steps) at later changes of the user, as shown by the inset of Fig. 6(f): at the beginning there are not enough good links to choose from. Continuous improvement could be reached only by assuming that links of visited documents are *available* at each instant (Fig. 6(f)). Without this assumption, performance was compromised given the small average branching ratio (about 7) of the web. Small branching ratio *and* a wrong starting region together prohibit the estimation of the interest of the user: selections could be meaningless under these conditions. The 'back' button (or the 'history') makes a difference: there is a clear improvement of learning speed of estimations beyond the 50th and the 100th step. In turn, a good highlighting 'agent' provides easy access to the (highlighted) history of surfing.

IV. Discussion

A. Evaluation of the results

Performance was similar for the users constructed from Boley classifier either when all Boley classifiers, the other Boley classifiers and the other Boley classifiers together with many additional CG classifiers were available. In case when the approximation had the potential to fully identify the user (i.e., when linear combination of Boley classifiers were approximated by *all* Boley classifiers) performance was somewhat better.

The increase of the number of classifiers did not improve performance. Moreover, the weights and the changes of the weights were similar for the 50 Boley classifiers and for the case when CG classifiers were included (Fig. 7). In turn, the set of 50 Boley classifiers seems a reasonable choice.

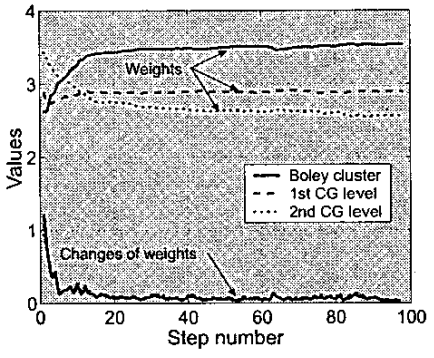

Fig. 7. **Weights and changes of weights**
The sum of the absolute values of the weights (upper curves) and the sum of the absolute values of the changes of the weights are depicted for the three different experiments. (CG: context graph)

Performance was never perfect, although it could have been, at least for the case of scheme fullC. This may be due to noise and/or to changing local minima in parameter space, which may arise in a particular domain of documents.

Another point concerns the large differences between different Boley clusters. For a uniform and improved performance the contents of these clusters need to be uncovered. We have noticed that there is a strong topic specificity whether a given user model can be followed or not. This issue requires further studies. Pre-clustering of users could be a solution here.

B. Differences among highlighting, monitoring and crawling

Highlighting can help to review an internet site and its immediate environment quickly. As such, it can decrease search time. Moreover, it may decrease traffic at portals or gateways and could serve as the basis of dynamical home pages. Highlighting can be very efficient but may miss information. Highlighting requires fast adaptation because it serves the user during an exploration. During surfing, both the topic of interest of the user and place of valuable information could be unknown.

Monitoring is different. Monitoring assumes that the place of information is known. It pays constant attention to a given site or a particular set of links. Monitoring collects recently published ('breaking news' type) information. In turn, monitoring will not miss information at the known places. However, monitoring may increase internet traffic and may miss information, which appear at new locations. Monitoring may not require adaptation.

Intelligent *crawling* is somewhere in between highlighting and monitoring. In this case, the topic of the search is more or less known, whereas the place of novel information is considered unknown. Adaptation is necessary under this condition, because different regions of the internet require different evaluations, as it has been demonstrated in the literature. [7], [11].

Highlighting, crawling and monitoring are all important ingredients for finding novel information. On the other hand, such tools are not to replace but to complement classical browsers, search tools and databases. The importance of these novel tools is striking when fresh information is searched for. Combination of search engines, crawlers, site monitors and highlighting techniques are all needed for efficient fast information access.

V. Conclusions

Users surfing on the Internet can be assisted in various ways. Intelligent crawlers can perform pre-filtering. On-line assistance, however, requires fast adaptation of such pre-filtering methods. In this paper a pre-trained, modifiable and extendable classification scheme was suggested for on-line guidance using exploration of the neighborhood of user-selected documents. It was shown that on-line reinforcement learning performs poorly relative to moving window methods. Value estimation error modulated moving window, which is closely related to reinforcement learning, however, showed improved efficiency. Both the artificial and for the Internet studies, highlighting could filter out about 80% of the documents in less than 10 steps. Sometimes, this number was as low as 2. Future research is needed to discover the 'human factor', when the highlighting agent interacts with real users.

Acknowledgments

This material is based upon work supported by the European Office of Aerospace Research and Development, Air Force Office of Scientific Research, Air Force Research Laboratory, under Contract No. F61775-00-WE065. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the European Office of Aerospace Research and Development, Air Force Office of Scientific Research, Air Force Research Laboratory. Careful reading of the manuscript is gratefully acknowledged to Attila Meretei.

References

[1] S. Chakrabarti, D. Gibson, and K. McCurley, "Surfing the Web backwards," in *8th World Wide Web Conference*, Toronto, Canada, 1999, http://www8.org/w8-papers/ 5b-hypertext-media/ surfing/.

[2] A. McCallum, K. Nigam, J. Rennie, and K. Seymore, "Building domain-specific search engines with machine learning techniques," in *AAAI-99 Spring Symposium on Intelligent Agents in Cyberspace*, 1999, http://www.cs.cmu.edu/~mccallum/ papers/ cora-aaaiss98.ps.

[3] S. Lawrence, "Context in web search," *IEEE Data Engineering Bulletin*, vol. 23, pp. 25–32, 2000.

[4] S. Mukherjea, "WTMS: A system for collecting and analyzing topic-specific web information," in *9th World Wide Web Conference*, 2000, http://www9.org/w9cdrom/293/293.html.

[5] G. Hinton, B. Sallans, and Z. Ghahramani, *Learning in graphical models, Ed.: M.I. Jordan.* Cambridge, MA: MIT Press, 1999, ch. Hierarchical community of experts, pp. 479–494.

[6] G. Hinton, "Training products of experts by minimizing contrastive divergence," Gatsby Computational Neuroscience Unit, University College London, Technical Report TR-2000-004, 2000, http://www.gatsby.ucl.ac.uk/.

[7] A. Lőrincz, I. Kókai, and A. Meretei, "Intelligent high-performance crawlers used to reveal topic-specific structure of the WWW," *Int. J. of Founds. of Comp. Sci.*, no. 13, pp. 477–495, 2002.

[8] D. Boley, "Principal direction division partitioning," *Data Mining and Knowledge Discovery*, vol. 2, pp. 325–244, 1998.

[9] T. Joachims, "A probabilistic analysis of the rocchio algorithm with tfidf for text categorization," 1996, http:// www-users.cs.umn.edu/ boley/ Distribution/PDDP.html.

[10] M. Diligenti, F. Coetzee, S. Lawrence, C. L. Giles, and M. Gori, "Focused crawling using context graphs," in *26th International Conference on Very Large Databases, VLDB 2000*, Cairo, Egypt, 10–14 September 2000, http://www.neci.nec.com/ lawrence/papers/ focus-vldb00/ focus-vldb00.ps.gz.

[11] I. Kókai and A. Lőrincz, "Fast adapting value estimation based hybrid architecture for searching the world-wide web," *Applied Soft Computing*, vol. 2, pp. 11–23, 2002.

[12] R. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction.* MIT Press, Cambridge, 1998.

[13] J. Rennie, K. Nigam, and A. McCallum, "Using reinforcement learning to spider the web efficiently," in *Proceedings of ICML-99, 16th International Conference on Machine Learning*, I. Bratko and S. Dzeroski, Eds. Bled, SL: Morgan Kaufmann Publishers, San Francisco, US, 1999, pp. 335–343. [Online]. Available: citeseer.nj.nec.com/rennie99using.html