Proceedings of the 28th IEEE
EMBS Annual International Conference
New York City, USA, Aug 30-Sept 3, 2006

Reinforcement Learning Interfaces for Biomedical Database Systems

I. Rudowsky, O. Kulyba, M. Kunin, S.Parsons, T. Raphan, *Member, IEEE*

Abstract— **Studies of neural function that are carried out in different laboratories and that address different questions use a wide range of descriptors for data storage, depending on the laboratory and the individuals that input the data. A common approach to describe non-textual data that are referenced through a relational database is to use metadata descriptors. We have recently designed such a prototype system, but to maintain efficiency and a manageable metadata table, free formatted fields were designed as table entries. The Database Interface Application utilizes an intelligent agent to improve integrity of operation. The purpose of this study was to investigate how reinforcement learning algorithms can assist the user in interacting with the Database Interface Application that has been developed to improve the performance of the system.**

I. Introduction

A common approach to describe non-textual data that are referenced through a relational database is to use metadata descriptors. Studies of neural function, for example, use a wide range of descriptors for data storage, depending on the laboratory and the individuals that input the data. These data potentially could be utilized in a shared environment to broaden understanding of brain function and lead to innovations in neuroscience, informatics and treatment of brain disorders [1-8]. However, no such global design has been suggested, nor has the feasibility of such a design been addressed in an organized fashion.

Classical database systems, which focused on textual data [9-11] are of limited utility in accessing files that use digital streams in non textual format [12, 13], representing analog and video information as input [14, 15]. Relational databases are capable of interacting with these data types, which have high storage and bandwidth requirements and have the capability to be queried and retrieved based on content [16-20]. More importantly, they have the flexibility to interact with application programs.

In this paper, we describe a methodology for incorporating an intelligent agent with learning capabilities into the system to insure relevant data retrieval when querying the database. Specifically, we describe the design and implementation of an interface agent (IA) to ease the access of data from incompletely specified or loosely associated keywords in the free formatted fields. This is accomplished by having the agent suggest more focused keywords related to the free-formatted entries. The suggestions are based on pre-established templates and enable the IA to learn how best to carry out its task. These algorithms learn to associate templates with user specified input when querying the database using the free-formatted fields. The input is matched against a set of fixed templates using a reward-based strategy to adapt the association rules.

II. Data Analysis and Interaction with a Relational Database

We have designed and implemented an Oracle9i database system, which can be accessed through a C/C++ front-end that links to an existing data acquisition and analysis program (VMF Application) [21-23] and data files. A graphical interface was designed so that it has the capability to index metadata that describe the experiments. Moreover, the submission of queries and retrieval of information from the database is user-friendly. The user interacts with the system through text fields and pre-filled, drop down combo boxes, which designate the locations for data entry. The criteria fields include items such as: (a) experiment number (a) subject species (human, monkey, rabbit, rat, mouse), (b) subject gender, (c) apparatus used (a number of rotation devices, linear movement device, human centrifuge device, treadmill information, etc), (d) date range of the experiments, (e) trials numbers within an experiment and (f) medical condition of the subject. Once the criteria have been supplied, an SQL query is generated and transmitted to the database engine. The resulting set of records is returned to the client machine and displayed in a graphical, tabular format along with key pieces of indexing information. This enables the researcher to (a) narrow down the possible candidates for viewing and (b) indicate additional experiments that might be considered for review. This organization enables the user to select an individual record, for visualization and analysis of the digitized analog channels and digital channels

Most of the metadata are in fixed fields, but we have intentionally included free formatted descriptors into a text field (COMMENT) that has been added to the database

Manuscript received April 3, 2006. This work was supported in part by the National Institutes of Health under Grants DC05222, DC05204, EY04148 and PSC CUNY Grant 36-295.

Ira Rudowsky is with the Department of Computer and Information Science, Brooklyn College of the City University of New York, USA. (phone: 718-951-5000 x2062; fax: 718-951-4489; e-mail: rudowsky@brooklyn.cuny.edu).

Olga Kulyba is with the Department of Computer and Information Science, Brooklyn College of the City University of New York, (e-mail: okulyba@yahoo.com).

Mikhail Kunin is with the Department of Computer and Information Science, Brooklyn College of the City University of New York, (e-mail: kunin@nsi.brooklyn.cuny.edu).

Simon Parosns is with the Department of Computer and Information Science, Brooklyn College of the City University of New York, (e-mail: parsons@sci.brooklyn.cuny.edu)

Theodore Raphan is with the Department of Computer and Information Science, Brooklyn College of the City University of New York and the Mount Sinai School of Medicine, (e-mail: raphan@nsi.brooklyn.cuny.edu).

table, which stores experimental data. This field holds trial related information that does not fit into the existing query categories and adds flexibility in describing experimental paradigms by the user. As opposed to the combo boxes that provide a drop down list of entries to choose from, the user is required to type specific text in the query text field – a minor inconvenience for the results obtained. It is also possible to combine key words in searching the COMMENT field by using OR and AND connectors. The program parses the input, constructs the appropriate SQL statement and displays the returned rows in the Results tab. Thus, the COMMENT field together with the ability to perform free-format searches on the field is a simple mechanism to provide scalability without adding overhead and complexity to the interface. It also simplifies the design and avoids numerous sparse fields.

One shortcoming of such a design is that it makes it incumbent on the researcher to make the free formatted descriptor as complete as possible for appropriately accessing the files to be analyzed. If users enter data in an incoherent (and, in an inter-user sense, inconsistent) way, it will make effective querying impossible. This flexible feature for a database user interface has raised the necessity of inclusion of an intelligent interface for entering data into the free-formatted fields. The design of an intelligent interface that minimizes ambiguities in free-formatted entries that could lead to incomplete searches or inaccurate data retrieval is the subject of this paper.

III. Template Design

A first step in the design to ensure that all necessary metadata descriptors are included in the free-formatted field is to have an "expert" who is familiar with the experiment being conducted design an appropriate template. Once entered into the template table, the researcher using the system will follow that pattern in describing all trials of that experiment. If additional trials are subsequently run by a laboratory assistant who is uncertain about what should be entered in the free-format field, the interface agent will be able to supply relevant templates based on keywords entered. The appropriate template can then be selected. This will ensure correct and consistent metadata for each trial for that particular experiment.

Subsequent queries of the database by entering keywords in the free-format field will activate the agent so that relevant templates are presented. An experiment, for example, may consist of a set of trials testing a subject's locomotion on a linear treadmill while fixating on a target x centimeters away. The domain of x is a finite set of distances. A second experiment may run a series of trials testing circular locomotion at x centimeters radius with treadmill rotation at y degrees/second. The parameters x and y are each varied over their own set of values with each (x, y) pair representing a unique trial. New experiments are continually being developed, each requiring a template description. Each new template may or may not have phrases in common with other templates. For example, the template "head shaking about an axis x for head orientation of (yaw, pitch)" has nothing in common with the two previously presented templates. Thus, templates are created by the person most familiar with the experiment and then used in both data entry and data querying to maintain a consistent presentation.

IV. Template Specification Using Reinforcement Learning

Once a group of templates have been designed, a key issue for interface agents is how they learn to extract the appropriate template– they need the ability to emulate the user behavior from user interaction with the agent. Research in adaptive interface agents focuses on building a profile of user interests which is then used to adapt future interaction to better meet the needs of the user. This adaptation includes making suggestions and/or performing queries/searches, transparent to the user, in order to provide information the agent determines that the user would want [24], adapting the content to better meet the user's knowledge and goals [25] or autonomously analyzing entries created in the course of routine work and classifying user knowledge to match the appropriate human expert [26].

Research on approaches such as rule induction [27] and naive Bayesian classification suggests that they may work well for this task, considering the text entries that we deal with. In this paper, we develop an agent that can learn by reinforcement, which template(s) to present to the user, and in which order, based on user supplied keywords entered in the free-format field. This should markedly improve the database search for relevant experiments and trials to enable the user's research. However, given the classification, the agent then has to learn how to process the entry, whether to let it stand, or to modify it in some way. If we consider that for each classification we have a small finite number of actions that the agent can carry out, which seems entirely reasonable for our domain, then we can use reinforcement learning [28] to do this. In reinforcement learning the agent will choose an action – a reorganization or not of the text – and the user will provide feedback, either agreeing with the action (leaving the reorganization alone) or disagreeing (reversing the reorganization). From this simple feedback, agents can use reinforcement learning to develop complex patterns of behavior.

The algorithm that we incorporated into the intelligent agent to suggest templates based on partial user specification was based on reinforcement learning ([29-34]), which is well suited for an interface agent interacting with a user. The algorithm is based on the idea that when a specific free-formatted query is made, it will be close to a template, which has been designed to be unambiguous. The association of the data entered with a particular template initiates a reward and influences future associations. It was expected that a template order would emerge based on the percentage of successful associations that have been made. Initially, we used sample average learning. That is, based on certain key

words in the free-formatted user input, the user was presented with a subset of templates in a given order that has been prioritized based on selection criteria.

V. Implementation of Reinforcement learning Algorithms

The reinforcement learning algorithm that was implemented was based on adding reward to a particular template choice based on the number of times it was selected in user interactions. The basis for the reward was implemented in two ways. In the first method, we chose five templates, which we refer to as T1 through T5 that we placed in a list with T1 at the top and T5 at the bottom. We chose twenty user inputs and arbitrarily pre-determined that the percentage of selection of these templates was as follows: T1=5%, T2 =70%, T3=10%, T4= 15% and T5 was not selected. When a template was selected, we added a reward to its value. The reward was also determined as a function of the number of times a particular template had been selected. Thus, if the first template selected were T3, a reward of 1 was added to its initial value, resulting in a current value of 1. For each template, we kept a running sum of the rewards earned and the number of times it was selected, determining the *value* of that template. We then sorted the templates by their value - largest first. There was no particular sort order applied to templates with equal values and they were just left in their relative positions.

When a template was selected again, the reward was obtained by subtracting the template position in the sorted list from 6, the number of templates +1. The position in the sorted list was utilized in the computation of the reward. Thus, if T3 was selected again, it received a reward of 5 (6-1), where 1 is its position in the list. The new value of T3 was computed as (1+5)/2 resulting in a value of 3, and continued to keep its position at the top of the list. The other template values were not updated, as no punishment rules have been implemented. The templates are now in order T3, T1, T2, T4, and T5. If the next template selected is T4, it now had a reward of 2 (6-4), where 4 is the position of the template in the list and its value was increased by 2 (2/1). The list was then reordered according to values and became T3, T4, T1, T2, and T5. This computation process continued for the remainder of selections.

VI. Results of Simulation

We ran simulations that made 10, 100, 1000, and 10,000 choices randomly based on the above percentages. The results of the simulation for the different test cases and runtimes are shown in Table 1. The simulations show that the system had approached the pre-determined percentages for 1000 and 10,000 runs, indicating that this simple reinforcement policy was reasonably effective.

In another reinforcement policy, the templates were not sorted, but the reward was increased according to the number of times a particular template was chosen. The results of the learning using this policy are shown in Table 2. These initial results suggest that the second approach converges faster than the first one.

	T1		T2		T3		T4		T5	
Runtimes	Computed Avg Reward	Expected Reward	Computed Avg Reward	Expected Reward	Computed Avg Reward	Expected Reward	Computed Avg Reward	Expected Reward	Computed Avg Reward	Expected Reward
10	0.00	5.00	88.64	70.00	6.82	10.00	4.55	15.00	0.00	0.00
100	1.78	5.00	76.67	70.00	14.22	10.00	7.33	15.00	0.00	0.00
1000	2.03	5.00	79.56	70.00	6.81	10.00	11.60	15.00	0.00	0.00
10000	1.96	5.00	77.71	70.00	6.98	10.00	13.35	15.00	0.00	0.00

Table 1. Computed Average Reward vs Expected Reward – Reinforcement Learning Method 1

	T1		T2		T3		T4		T5	
Runtimes	Computed Avg Reward	Expected Reward	Compt'd Avg Reward	Expected Reward	Computed Avg Reward	Expected Reward	Computed Avg Reward	Expected Reward	Computed Avg Reward	Expected Reward
10	20.00	5.00	60.00	70.00	10.00	10.00	10.00	15.00	0.00	0.00
100	7.00	5.00	74.00	70.00	11.00	10.00	8.00	15.00	0.00	0.00
1000	4.40	5.00	70.30	70.00	11.30	10.00	14.00	15.00	0.00	0.00
10000	4.21	5.00	70.45	70.00	10.09	10.00	15.26	15.00	0.00	0.00

Table 2. Computed Average Reward vs Expected Reward – Reinforcement Learning Method 2

These preliminary findings indicate that even a simple reinforcement learning algorithm has the ability to significantly assist users in free-formatted queries in database applications.

VII. Conclusions and Future Research

In previous work, we have developed a Database Interface Application, which utilizes Oracle as a means for data acquisition and analysis. We have also devised an interface, whereby an agent can communicate with the user and independently with the relational database. In the research reported on herein, we have designed and implemented an interface agent that interacts with the user in an intelligent manner to better define free formatted queries, which have simplified the overall database design and made the system more flexible. This is an important first step in removing ambiguities in the search algorithms needed to identify data for analysis. Future research will be directed at fully developing the intelligent agent and integrating it with the Database Interface Application.

There have been studies that emphasize the importance of unobtrusiveness as an important goal for interface agents [35]. However, the context of our interface agent is critical for maintaining data integrity as well as insuring proper searches. Therefore, the weight attached to maintenance of unobtrusiveness must be considered in relation to the context of the application. Another possibility for future research is to automatically put the highest ranked suggestion into the free-format field and then let the user decide if another choice is more appropriate for this search. In that case, the reinforcement algorithm will update the template rankings accordingly. In the case of expert users, an option can be provided that will turn off the suggestion routine and turn complete control over to the user.

Thus, we have developed a preliminary learning algorithm, based on techniques from reinforcement learning

(considering the problem as a simple n-armed bandit) to determine how templates can be ordered and recommended to users when there are free-formatted queries. The next step will involve the use of more sophisticated reinforcement learning algorithms to improve and monitor the performance of the system, and to allow it to adapt its behavior over time. More advanced techniques will display only one choice at a time to incorporate possible penalties in the reinforcement learning when a choice is declined. Other methods, such as randomly displaying low priority choices for a given free-formatted entry will enhance the system to better prioritize the selections [28]. The system might then be assessed using a number of metrics for evaluating intelligent agent performance.

REFERENCES

[1]S. Amari, et al, "Neuroinformatics: the integration of shared databases and tools towards integrative neuroscience," *J Integr Neurosci*, vol. 1, pp. 117-28, 2002.

[2]K. G. Becker, "The sharing of cDNA microarray data," *Nat Rev Neurosci*, vol. 2, pp. 438-440, 2001.

[3]D. H. Geschwind, "Sharing gene expression data: an array of options," *Nat Rev Neurosci*, vol. 2, pp. 435-8, 2001.

[4]S. H. Koslow, "Should the neuroscience community make a paradigm shift to sharing primary data?," *Nature Neuroscience*, vol. 7, pp. 473-481, 2000.

[5]S. H. Koslow, "Opinion: Sharing primary data: a threat or asset to discovery?," *Nat Rev Neurosci*, vol. 3, pp. 311-3, 2002.

[6]M. Miles, "Microarrays:Lost in a storm of data?," *Nat Rev Neurosci*, vol. 2, pp. 441-443, 2001.

[7]K. Mirnics, "Microarrays in Brain Research: The good, the bad and the ugly," *Nat Rev Neurosci*, vol. 2, pp. 444-447, 2001.

[8]J. D. Van Horn and M. S. Gazzaniga, "Opinion: Databasing fMRI studies towards a 'discovery science' of brain function," *Nat Rev Neurosci*, vol. 3, pp. 314-8, 2002.

[9]D. Knuth, *Sorting and Searching*, vol. 3, 2 ed. Reading, Mass: Addison Wesley, 1998.

[10] A. Silberschatz, H. Korth, and S. Sudarshan, *Database System Concepts*. New york: McGraw Hill, 2002.

[11] N. Wirth, *Algorithms + Data Structures = Programs*. Englewood Cliffs, NJ: Prentice Hall, 1976.

[12] G. Eisenhauer, "Portable Self-Describing Binary data Streams," College of Computing, Georgia Institute of Technology 1994.

[13] K. H. Foster, J. P. Gaska, M. Nagler, and D. A. Pollen, "Spatial and temporal frequency selectivity of neurones in visual cortical areas V1 and V2 of the macaque monkey," *J. Physiol. (Lond)*, vol. 365, pp. 331-363, 1985.

[14] C. Meghini, F. Sebastiani, and U. Straccia, "A model of Multimedia Information Retrieval," *Journal of the ACM*, vol. 48, pp. 909-970, 2001.

[15] R. Weber, J. Bolliger, T. Gross, and H.-J. Schek, "Architecture of a Networked Image Search and Retrieval Syestem," presented at 8th Intl Conference on Information and Knowlwedge Management, Kansas City, Missouri, 1999.

[16] S. Chaudhuri and L. Gravano, "Optimizing Queries over Multimedia Repositories," Proceedings of SIGMOD '96, Montreal, Canada, 1996.

[17] C. Faloutsos, et al, "Efficient and Effective querying by image content," *J. Intell. Inf. Syst.*, vol. 3, pp. 231-262, 1994.

[18] V. E. Ogle and M. Stonebraker, "Chabot: Retrieval from a Relational Database of Images," *IEEE Computer*, vol. 28, pp. 40-56, 1995.

[19] B. Ozden, R. Rastogi, and A. Silberschatz, "A Multimedia Support for Databases," Proc. 16th ACM SIGACT-SIGMOD-SIGART symposium on Principles of Database Systems, 1997.

[20] C. Traina Jr., et al, "Support to Content-Based Image Query in Object- Oriented Databases," Proc. of the ACM Symposium on Applied Computing, 1998.

[21] I. Rudowsky, O. Kulyba, M. Kunin, D. Ogarodnikov, and T. Raphan, "Relational Database Linkage of Scientific Applications and Their Data Files," presented at Proc of the IEEE International Workshop on Soft Computing in Industrial Applications, 2003.

[22] I. Rudowsky, O. Kulyba, M. Kunin, D. Ogarodnikov, and T. Raphan, "Flexible Security and Search Capability for a relational Database with Externally Linked Multimedia data Files," Proceedings of SCI2004 - The 8th World Multi-Conference on Systemics, Cybernetics and Informatics, Orlando, FL, 2004.

[23] I. Rudowsky, O. Kulyba, M. Kunin, D. Ogarodnikov, and T. Raphan, "Relational Database Integrity with Externally Linked Multimedia Data Files," *Journal of Integrative Neuroscience*, 2004.

[24] S. Schiaffino and A. Amandi, "An Interface Agent Approach to Personalize Users' Interaction with Databases," *J. Intelligent Information Systems*, vol. 25, pp. 251-273, 2005.

[25] P. Brusilovsky, "Methods and techniques of adaptive hypermedia" *User Modeling and User Adapted Interaction*, vol. 6, pp. 87-129, 1996.

[26] A. Vivacqua and H. Lieberman, "Agents to Assist in Finding Help," presented at CHI 2000 - Conference on Human factors in computing systems, The Hague, The Netherlands 2000.

[27] W. W. Cohen and Y. Singer, "Context-sensitive learning methods for text categorization," presented at 19th ACM International Conference on Research and Development in Information Retrieval, 1996.

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

[29] A. G. Barto, "Reinforcement learning and adaptive critic methods," in *Handbook of Intelligent Control: Neural, Fuzzy and Adaptive Approaches*, D. A. White and D. A. Sofge, Eds. New York: Van Nostrand Reinhold, 1992, pp. 469-491.

[30] A. G. Barto, "Reinforcement Learning " in *Handbook of Brain Theory and Neural Networks*, M. Arbib, Ed. Cambridge, MA: MIT Press, 1995, pp. 804-809.

[31] D. P. Bertsekas and J. N. Tsitsiklis, *Neural Dynamic Programming*. Belmont, MA: Athena Scientific, 1996.

[32] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement Learning: A survey," *J. of Artificial Intelligence Research*, vol. 4, 1996.

[33] S. S. Keerthi and B. Ravindran, "Reinforcement Learning," in *Handbook of Neural Computation*, E. Fiesler and R. Beale, Eds. Oxford: University Press, 1997.

[34] R. S. Sutton, *Reinforcement Learning*, vol. 8. Boston, MA: Academic Press, 1992.

[35] A. Jameson, "Adaptive Interfaces and Agents," in *The Human-Computer Interaction Handbook, Fundamentals, Evolving Technologies and Emerging Applications*, J. Jacko and A. Sears, Eds. Mahwah, NJ: Lawrence Erlbaum Associates, 2003.