OWL: A Recommender System for Organization-Wide Learning

Frank Linton, Andy Charron, & Debbie Joy
The MITRE Corporation
linton/asc/djoy@mitre.org

Abstract

We describe the use of a *recommender system* to enable continuous knowledge acquisition and individualized tutoring of application software across an organization. Installing such systems will result in the capture of evolving expertise and in organization-wide learning (OWL). We present the results of a year-long naturalistic inquiry into an application's usage patterns, based on logging users' actions. We analyze the data to develop user models, individualized expert models, confidence intervals, and instructional indicators. We show how this information could be used to tutor users.

Introduction

Recommender Systems typically help people select products, services, and information. A novel application of recommender systems is to help individuals select 'what to learn next' by recommending knowledge that their peers have found useful. For example, people typically utilize only a small portion of a software application's functionality (one study shows users applying less than 10% of Microsoft Word's commands). A recommender system can unobtrusively note which portions of an application's functionality that the members of an organization find useful, group the organization's members into sets of similar users, or peers (based on similar demographic factors such as job title, or similarities in command usage patterns), and produce recommendations for learning that are specific to the individual in the context of his/her organization, peers, and current activities.

This paper reports research on a recommender system (Resnick & Varian, 1997) intended to promote gradual but perpetual performance improvement in the use of application software. We present our rationale, an analysis of a year's collected data, and a vision of how users might learn from the system. We have worked with one commercial application, and believe our approach is generally applicable.

The research explores the potential of a new sort of user modeling based on summaries of logged user data. This method of user modeling enables the observation of a large number of users over a long period of time, enables concurrent development of student models and individualized expert models, and applies recommender system techniques to on-the-job instruction. Earlier work is reported in Linton (1990), and Linton (1996). Kay and Thomas (1995), and Thomas (1996) report on related work with a text editor in an academic environment.

A recommender system to enhance the organization-wide learning of application software is a means of promoting organizational learning (Senge, 1990). By pooling and sharing expertise, recommender systems augment and assist the natural social process of people learning from each other. This approach is quite distinct from systems, such as Microsoft's Office Assistant, which recommend new commands based on their logical equivalence to the less-efficient way a user may be performing a task. The system presented here will (1) capture evolving expertise from a community of practice (Lave & Wenger 1991), (2) support less-skilled members of the community in acquiring that expertise, and (3) serve as an organizational memory for the expertise it captures.

> In many workplaces ... mastery is in short supply and what is required is a kind of collaborative bootstrapping of expertise.
>
> (Eales & Welch, 1995, p. 100)

The main goal of the approach taken in this work is to continuously improve the performance of application users by providing individualized modeling and coaching based on the automated comparison of user models to expert models. The system described here would be applicable in any situation where a number of application users perform similar tasks on networked computers

In the remainder of this section we describe the logging process and make some initial remarks about modeling and coaching software users. We then present an analysis of the data we have logged and our process of creating individual models of expertise. In the final section we describe further work and close with a summary.

Each time a user issues a Word command such as Cut or Paste, the command is written to the log, together with a time stamp, and then executed. The logger, called OWL for Organization-Wide Learning, comes up when the user opens Word; it creates a separate log for each file the user edits, and when the user quits Word, it sends the logs to a server where they are periodically loaded into a database for analysis. A toolbar button labeled 'OWL is ON' (or OFF) informs users of OWL's state and gives them control.

Individual models of expertise

We have selected the Edit commands for further analysis. A similar analysis could be performed for each type of command. The first of the three tables in Figure 1 presents data on the Edit commands for each of our 16 users. In the table, each column contains data for one user and each row contains data for one command (Edit commands that were not used have been omitted). A cell then, contains the count of the number of times the individual has used the command. The columns have been sorted so that the person using the most commands is on the left and the person using the fewest is on the right. Similarly, the rows have been sorted so that the most frequently used command is in the top row and the least frequently used command is in the bottom row. Consequently the cells with the largest values are in the upper left corner and those with the smallest values are in the lower right corner. The table has been shaded to make the contours of the numbers visible: the largest numbers have the darkest shading and the smallest numbers have no shading, each shade indicates an order of magnitude.

Inspection of the first table reveals that users tend to acquire the Edit commands in a specific sequence, i.e., those that know fewer commands know a subset of the commands used by their more-knowledgeable peers. If instead, users acquired commands in an idiosyncratic order, the data would not sort as it does. And if they acquired commands in a manner that strongly reflected their job tasks or their writing tasks, there would be subgroups of users who shared common commands. Also, the more-knowledgeable users do not replace commands learned early on with more powerful commands, but instead keep adding new commands to their repertoire. Finally, the sequence of command acquisition corresponds to the commands' frequency of use. While this last point is not necessarily a surprise, neither is it a given.

There are some peaks and valleys in the data as sorted, and a fairly rough edge where commands transition from being used rarely to being used not at all. These peaks, valleys, and rough edges may represent periods of repetitive tasks or lack of data, respectively, or they may represent overdependence on some command that has a more powerful substitute or ignorance of a command or of a task (a sequence of commands) that uses the command. In other words, some of the peaks, valleys, and rough edges may represent opportunities to learn more effective use of the software.

In the second table in Figure 1 the data have been smoothed. The *observed* value in each cell has been replaced by an *expected* value, the most likely value for the cell, using a method taken from statistics, based on the row, column and grand totals for the table (Howell, 1982). In the case of software use, the row effect is the overall relative utility of the command (for all users) and the column effect is the usage of related commands by the individual user. The expected value is the usage the command would have if the individual used it in a manner consistent with his/her usage of related commands and consistent with his/her peers' usage of the command.

These expected values are a new kind of *expert model*, one that is unique to each individual and each moment in time; the expected value in each cell reflects the individual's use of related commands, and one's peers' use of the same command. The reason for differences between observed and expected values, between one's actual and expert model, might have several explanations such as the individual's tasks, preferences, experiences, or hardware, but we are most interested when the difference indicates the lack of knowledge or skill.

		Observed														
Count of COMMAND		EMPNMBR														
COMMAND	m300	m937	m242	m974	m375	m707	m118	m590	m926	m553	m600	m151	m000	m272	m068	m464
EditPaste	[illegible]	373	30	376	298	315	286	230	274	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	1
EditClear	[illegible]	393	799	340	233	81	74	2	0	0	[illegible]	[illegible]	0	6	1	0
EditCopy	[illegible]	277	27	104	101	47	152	145	17	120	26	9	36	20	17	0
EditCut	576	132	8	80	94	284	17	12	200	26	1	21	12	6	3	0
EditUndo	69	76	61	36	16	31	79	24	14	9	3	12	4	19	6	0
EditFind	217	4	1	12	2	4	8	83	1	1	0	0	1	0	2	0
EditSelectAll	147	20	11	4	2	3	8	3	12	5	20	3	0	[illegible]	1	0
EditDeleteWord	83	19	1	4	0	0	0	0	0	0	0	0	0	0	0	0
EditReplace	67	1	0	2	0	0	3	3	0	1	0	0	0	0	0	0
EditPasteSpeci	14	1	35	2	0	0	1	0	0	0	0	0	0	0	1	0
EditRedo	4	3	2	5	0	0	3	0	2	0	0	1	0	0	2	0
EditGoTo	11	0	4	2	2	0	0	0	0	0	0	0	0	0	0	0
EditGoToHeade	3	2	0	1	0	6	0	0	0	0	0	0	0	0	0	0
EditCopyAsPic	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0
EditAutoText	0	0	0	0	3	1	0	0	0	0	0	0	0	1	0	0
EditDeleteBack	0	0	0	3	1	0	0	0	0	0	0	0	0	0	0	0
EditBookmark	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0
EditGoBack	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

		Expected														
Count of COMMAND		EMPNMBR														
COMMAND	m300	m937	m242	m974	m375	m707	m118	m590	m926	m553	m600	m151	m000	m272	m068	m464
EditPaste	[illegible]	442	339	329	259	255	214	170	170	127	114	80	36	31	19	0
EditClear	[illegible]	338	258	251	197	194	163	129	128	85	67	61	27	23	14	0
EditCopy	[illegible]	278	213	207	160	161	135	107	102	80	73	61	23	19	13	0
EditCut	652	124	96	92	72	72	60	48	48	38	32	23	10	9	5	0
EditUndo	216	41	31	31	24	24	20	16	16	12	11	7	3	3	2	0
EditFind	159	26	23	22	18	17	16	12	12	9	8	5	2	2	1	0
EditSelectAll	118	22	17	17	13	13	11	9	9	6	6	4	2	2	1	0
EditDeleteWord	81	16	7	7	6	6	5	4	4	3	2	2	1	1	0	0
EditReplace	36	7	5	5	4	4	3	3	3	2	2	1	1	0	0	0
EditPasteSpeci	26	5	4	4	3	3	2	2	2	1	1	1	0	0	0	0
EditRedo	13	2	2	1	1	1	1	1	1	1	1	0	0	0	0	0
EditGoTo	9	2	1	1	1	1	1	1	1	0	0	0	0	0	0	0
EditGoToHeade	6	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
EditCopyAsPic	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
EditAutoText	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
EditDeleteBack	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
EditBookmark	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
EditGoBack	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

		Instruction														
Count of COMMAND		EMPNMBR														
COMMAND	m300	m937	m242	m974	m375	m707	m118	m590	m926	m553	m600	m151	m000	m272	m068	m464
EditPaste			More								More					
EditClear	_	_	_	_	_	_	_	More			_	_		_	[illegible]	
EditCopy			More									[illegible]				
EditCut			More			All					[illegible]					
EditUndo							All							All	All	
EditFind		More	More		More	More		All	More	More						
EditSelectAll	_	_	_	All	More	All	_	_	_	_	All	_		All		
EditDeleteWord			More													
EditReplace		More	[illegible]													
EditPasteSpeci		All	All													
EditRedo				All												
EditGoTo	_		All	_	_											
EditGoToHeade																
EditCopyAsPic																
EditAutoText																
EditDeleteBack																
EditBookmark																
EditGoBack																
1.13	0	2	5	0	2	1	0	1	1	1	2	1	0	0	1	0
0.87	0	1	2	2	0	2	1	1	0	0	1	0	0	2	1	0
2.27	4	1	1	0	4	4	1	2	4	3	4	3	2	1	0	0

Figure 1. Computing Tutorial Intervention

In the third table in Figure 1, each cell contains one of five symbols. The symbols are indicators of learning opportunities that could be used by an automated tutoring process. These indicators are data that, combined with domain and curriculum knowledge, would result in recommendations for learning. The five symbols are: "_" (underscore), " " (blank), "New," "More," and "Alt." "New" has the darkest shading, followed by "More" and "Alt." The underscore and blank have no shading.

A command whose expected value is zero need not be learned, and can be ignored; its indicator is a blank. A command that has an expected value, but is one the individual has never used, is a command the individual probably would find useful if s/he were to learn it; its indicator is "New." A command whose usage is within normal range of the expected value can also be ignored. The normal range was determined empirically so as to limit the amount of recommended learning, on average, to two New commands, one More command, and one Alt command. The indicator for a command within normal range of the expected value is an underscore. A command that is used less than expected may be a component of tasks unknown but potentially valuable to the user; its indicator is "More." A command that is used more than expected may indicate ignorance of more powerful ways of accomplishing some tasks; its indicator is "Alt."

Providing feedback to users: Intelligent Tips and Skillometer

We envision two methods the OWL system could use to provide coaching and feedback to users. An active method, Intelligent Tips, would coach passive learners; and a passive method, the Skillometer, would provide information to active learners. Both methods would make use of the data analysis processes discussed in the preceding section, but the Intelligent Tip interface would interpret the data conservatively, sending the user a tip only when it was rather certain of the utility of the tip to the user. For software like Word, the Intelligent Tip might appear when the user started up Word; it would inform the user of the nature of the tip and point the user to the appropriate information in the application's Help system. The goal of Intelligent Tips would be to provide the tip with perfect timing: after the user has become aware of his/her need for a command, but before s/he has made an effort to find the command.

The Skillometer (Figure 2), in contrast, would provide an interface for the user to explore his/her observed and expected models, and those of any demographic group in the organization that the user might be a member of, such as Engineer, Division 40, writer-of-long-documents, etc. The Skillometer would make it easy to compare and contrast one's actual usage with one's expert model, and with the synthesized expert models of various groups, enabling the

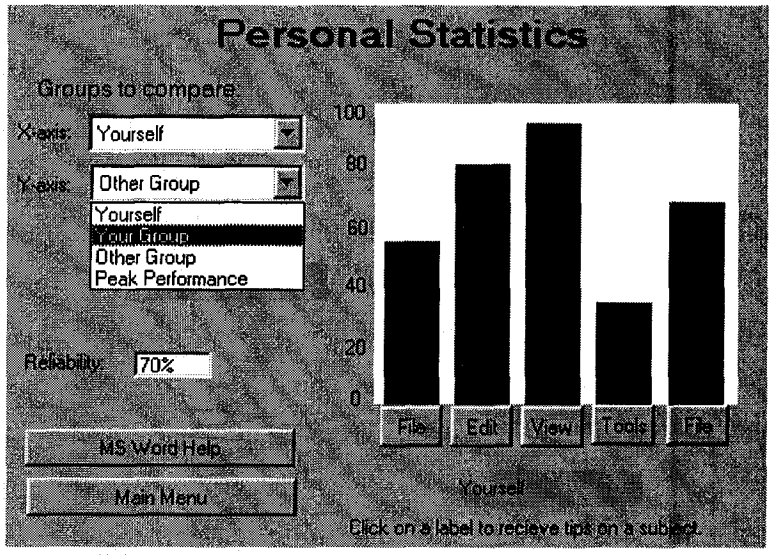

Figure 2. Skillometer: Skill Analysis

user to make decisions for himself or herself about the utility of the commands that others are using. The Skillometer should be useful to someone who anticipates performing a set of tasks and wants to become aware of which commands will be applicable.

Concluding remarks

We have proposed a recommender system as a means of helping a community of practice pool and share its expertise, where the community is a set of people who desire to become skilled users of their application software. Expertise evolves because users respond to changes in the work environment and to changes in the software, as well as continuously figuring out better ways to accomplish their tasks. The system will provide continuous performance improvement to those who do complex job tasks with application software. The system's expert models will serve as the organization's memory of expertise. Analysis of user log data collected over the past year revealed that editing command usage frequencies follow a power law. As a result of this power law of usage, careful work is required to distinguish apparent user ignorance or learning from effects due solely to the amount of information logged. A method was proposed for taking these effects into account. We further discovered that users tend to acquire commands in sequence. By combining the power law of usage with the acquisition sequence, we can determine the expected usage of each command by each individual: an individualized expert model; and take large deviations from this model as indicators of ineffective software use -- and learning opportunities. Finally, we proposed two methods of communicating these learning opportunities to users.

Bibliography

Eales, J., & Welch, J. (1995). Design for collaborative learnability. In J. Schnase and E. Cunnius (Eds.), *Proceedings of CSCL '95: The First International Conference on Computer Support for Collaborative Learning* (pp. 99-106). Erlbaum, Mawah NJ.

Howell. D., (1982). Statistical methods for psychology. Boston: Duxbury Press.

Kay, J., Thomas, R. (1995). Studying long-term system use; computers, end-user training and learning. Communications of the ACM Volume 38 number 7.

Lave, J., and Wenger, E. (1991). *Situated learning: Legitimate peripheral participation.* Cambridge: Cambridge University Press.

Linton, F. (1990). A coach for application software. *CBT Directions* (pp. 22-29), March 1990.

Linton, F. (1996). Promoting the Organization-Wide Learning of Application Software. ACM SIGOIS Bulletin. Vol 17 number 3.

Resnick, P., and Varian, H. (1997). Introduction to Special Section on Recommender Systems. Communications of the ACM Volume 40 number 3.

Senge, P. (1990). The fifth discipline: *The art & practice of the learning organization.* New York: Currency Doubleday.

Thomas, R. (1996). Long term exploration and use of a text editor. Unpublished doctoral dissertation. University of Western Australia.