Improving Intelligent Assistants for Desktop Activities

Simone Stumpf, Margaret Burnett, Tom Dietterich

School of Electrical Engineering and Computer Science
Oregon State University
Corvallis, OR
{stumpf, burnett, tgd}@eecs.oregonstate.edu

Abstract

Tasks have been identified as playing an important role to knowledge workers as high-level units for organizing their information. TaskTracer is a task-aware desktop system that leverages the semi-automatic association between tasks and information to provide intelligent assistance to the user. TaskPredictor is a component that attempts to automatically predict the current task of the user. Our experience with TaskPredictor tells us that there still remain challenges to increasing acceptance by users. We describe our approaches to reduce the number of interruptions to the user, and to improve the accuracy of predictions.

Projects, Activities, and Tasks

Projects, tasks, activities, and to-do items have been identified as playing an important role to knowledge workers, i.e. computer users who create and process information as part of their work [2, 5, 7, 9]. These activities are conceptual structuring devices – they are the units into which a person divides their work, and the names associated with those units. Some tasks correspond to concrete projects like "Conduct user study" or "Create end of year report". Other tasks may be ongoing, with no start date, end date, or deliverables, such as "Advise students". More intuitively, tasks might correspond to the folders that you would find in a physical filing cabinet, or to a bulleted list that a person might write describing the projects and tasks for which they are responsible. To take advantage of tasks, a few recent task-centric solutions have been developed for email [4], or for desktop activity [12, 3, 16]. However, these systems have so far not exploited the support that could be provided by intelligent assistants to increase the productivity of knowledge workers.

Task Prediction for Desktop Activities

TaskTracer [8] is a system that allows users to organize their information by labeling them with high-level units that make sense to them (which we call tasks), such as "teach CS534", "prepare NSF proposal B", "make travel arrangements for conference C". We assume that the behavior of the user at the desktop is an interleaved timeline of different tasks and each task is associated with a set of resources (e.g. documents, folders, web pages, emails, contacts, etc.) relevant to that task. Once tasks have been defined, the user can indicate to the system the name of the current task and switch to new tasks. TaskTracer records all resources that are accessed by the user and automatically associates them with the current declared task. This data can be used by machine learning components[1] to provide intelligent assistance to the user to make finding and refinding information easier [6, 13].

These benefits assume that the data is not too noisy and therefore the user must let the system know when they have changed tasks. To reduce the cost of task switches, we have been working to supplement the user's manual task declaration with a TaskPredictor (Figure 1) that attempts to automatically predict the current task of the user [14, 15]. Our experience with TaskPredictor tells us that there still remain two main challenges to increasing acceptance by users: 1) reducing the number of interruptions to the user, and 2) improving the accuracy of predictions.

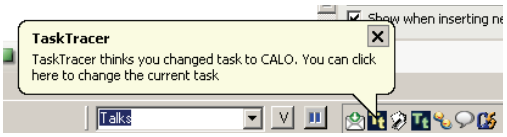

Figure 1. The TaskPredictor icon makes a prediction and informs the user through a balloon notification. Also shown are the TaskSelector to indicate tasks (on left) and the main TaskTracer icon (on right).

[1] The types of tasks that we typically encounter have no or only loosely fixed structure and are highly distinctive of the individual knowledge worker. A *plan* often has flexibility but the user is executing a specific structured activity. We are not suggesting that these two approaches are mutually exclusive.

Reducing the number of interruptions

In order to reduce the number of interruptions to the user, a cost-sensitive approach to task prediction and notification in the user interface is needed [10]. Currently, we already reduce the number of possible notifications made by the TaskPredictor through two main ways. First, we assume that in an unbroken segment of time in which a particular window has focus and the name of the document in that window does not change, the user is still working on the same task and hence we do not have to make a prediction. Second, to keep user costs within reason for TaskPredictor, we do not make a prediction for every window/document. We are willing to accept lower coverage in order to maintain high precision, because we would like to minimize the number of interruptions, particularly if these interruptions may provide wrong predictions. Thus, the TaskPredictor employs a probabilistic threshold for predicting tasks for windows/documents; if the threshold is not reached then notification is not made. We are currently exploring further options to better address user cost issues by exploiting task boundaries [1, 11].

Improving the accuracy of predictions

One way of increasing the accuracy of predictions is by incorporating user feedback. TaskPredictor at the moment only allows the user to indicate that a prediction was wrong and to specify what the correct prediction should have been. This is just a glimpse of the rich knowledge users have about the correct prediction, and we would like to better harness the user's knowledge to improve learning.

To investigate the feasibility of rich user feedback, we conducted a formative think-aloud study with 13 email users using 122 messages from the publicly available Enron dataset (user farmer-d). In this study [17], machine learning algorithms (Ripper and Naïve Bayes) sorted email messages into folders and explained their reasoning using three different simplified but faithful explanation paradigms: Rule-based, Keyword-based, and Similarity-based. The participants were asked to provide feedback to improve the pre-dictions. No restrictions were placed upon the form or con-tent of participants' feedback. We observed and video-taped their activities and comments throughout the experiment, as well as collecting their work products, and questionnaires investigating their preferences and behavior. From a user perspective, we assessed the participants' willingness to provide feedback, accuracy in doing so, and ability to understand the different explanations. From an algorithm perspective, we analyzed the participants' feedback to determine how easily its types of advice could be understood and assimilated by machine learning algorithms.

The system explains to the user

We provided simplified but faithful explanations of the learning algorithms, which were concrete in terms of specific features that were visible in the current email message. Using these design principles, we found that Rule-based explanations were the most understandable and preferred but that a substantial number of the participants chose one of the other explanation paradigms as their favorite. This implies that machine learning systems may need to support multiple explanation paradigms in order to effectively reach all of their users.

The factors that contributed to participants' overall preference of an explanation were approval of reasoning soundness, clear communication of reasoning, perceived accuracy, and a less technical style of expression. Similarity-based explanations had serious understandability problems in our experiment.

	KB-English	KB-common sense	KB-domain	KB-other	Total	%
1. Adjust weight	11	11	4	13	39	12%
2. Select different features (words)	70	64	25	16	175	53%
3. Parse or extract in a different way	7	17	10	0	34	10%
4. Employ feature combinations	9	5	2	1	17	5%
5. Relational features	0	9	5	0	14	4%
6. Other	3	12	4	33	52	16%
Total	100	118	50	63	331	
%	30%	36%	15%	19%		

Table 1: Types of participants' changes (in rows) that required various background knowledge (in columns).

The user gives rich feedback to the system

The participant corrections brought the accuracy rates for all paradigms to almost identical levels (71-72%). When the participants disagreed with the machine, participants were usually right, but not always (introducing some noise into the data).

Table 1 shows the results of our analysis in terms of what type of feedback participants made and the knowledge upon which it was based. Among the feedback were requests for reweighting features, feature combinations, relational features, and even wholesale changes to the algorithms. Almost a third of the participations' suggestions relied on knowledge of English, or on some knowledge that could be encoded once and then reused. Roughly half of the suggestions for improvement appear to be amenable to automated assimilation with existing methods.

The results open new questions for research on methods for assimilating complex user suggestions for feature extraction, relational features, and incorporating

constraints on solutions found by learning algorithms. They provide evidence that machine learning systems can explain their reasoning and behavior to users, and that users in turn can provide rich, informative feedback to the learning system. This suggests rich user-machine collaboration as a promising direction for intelligent user interfaces to learn more effectively, by better harnessing of the intelligence of users.

References

[1] Adamczyk, P. and B. Bailey. 2004. If Not Now, When? The Effects of Interruption at Different Moments Within Task Execution. *Proc. CHI*, Vienna, Austria.

[2] Bannon, L., Cypher, A., Greenspan, S., Monty, M. 1983. Evaluation and analysis of users' activity organization. *Proc. CHI*, 54–57.

[3] Bardram, J., Bunde-Pedersen, J., Soegaard, M. 2006. Support for activity-based computing in a personal computing operating system. *Proc. CHI*, 211-220.

[4] Bellotti, V., Ducheneaut, N., Howard, M., Smith, I. 2003. Taking email to task: The design and evaluation of a task management centered email tool. *Proc. CHI*, 345–352.

[5] Bellotti, V., Dalal, B., Good, N., Flynn, P., Bobrow, D., Ducheneaut, N. 2004. What a to-do: Studies of task management towards the design of a personal task list manager. *Proc. CHI*, 735-742.

[6] Bao, X., Herlocker, J.L., Dietterich, T.G. 2006. Fewer Clicks and Less Frustration: Reducing the Cost of Reaching the Right Folder. *Proc. IUI*, Sydney, Australia, 2006.

[7] Czerwinski, M., Horvitz, E., Wilhite, S. 2004. A Diary Study of Task Switching and Interruptions. *Proc. CHI,* 175-182.

[8] Dragunov, A., Dietterich, T. G., Johnsrude, K., McLaughin, M., Li, L. and Herlocker, J. L. 2005. Tasktracer: A Desktop Environment to Support Multi-Tasking Knowledge Workers. *Proc. IUI*, San Diego, CA.

[9] Gonzalez, V., Mark, G. 2004. "Constant, constant, multi-tasking craziness": managing multiple working spheres. *Proc. CHI*, 13–120.

[10] Horvitz, E., Kadie, C., Paek, T., Hovel, D. 2003. Models of attention in computing and communication: from principles to applications. *Comms ACM 46(3)*, 52-59.

[11] Iqbal, S., P. Adamczyk, Zheng, X.S., Bailey, B. 2005. Towards an Index of Opportunity: Understanding Changes in Mental Workload During Task Execution. *Proc. CHI*, Portland, OR.

[12] Kaptelinin, V. 2003. UMEA: Translating interaction histories into project contexts. *Proc. CHI*, 353–360.

[13] Lettkeman, T., Stumpf, S., Irvine, J., Herlocker, J. 2006. Predicting Task-Specific Webpages for Revisiting. *Proc. AAAI-06*, Boston, MA.

[14] Shen, J., Li, L., Dietterich, T.G., Herlocker, J. L. 2006. A hybrid learning system for recognizing user tasks from desk activities and email messages. *Proc. IUI*, Sydney, Australia.

[15] Shen, J., Li, L., Dietterich, T.G. 2007. Real-Time Detection of Task Switches of Desktop Users. *Proc. IJCAI-07*, Hyderabad, India.

[16] Smith, G., Baudisch, P., Robertson, G. , Czerwinski, M. , Meyers, B. , Robbins, D. , Andrews, D. 2003. GroupBar: The TaskBar Evolved. *Proc. OZCHI* , Brisbane, Australia,34-43.

[17] Stumpf, S., Rajaram, V., Li, L., Burnett, M., Dietterich, T.G., Sullivan, E. Drummond, R., Herlocker, J. 2007. Toward Harnessing User Feedback For Machine Learning. *Proc. IUI'07,* Honolulu, HI.