

HAYM HIRSH, CHUMKI BASU, AND BRIAN D. DAVISON

LEARNING to Personalize

Recognizing patterns of behavior helps systems predict your next move.

A typical user exhibits many patterns when interacting with a computer system. Machine-learning algorithms are being used to recognize such regularities and integrate them into the system, to personalize the system's interactions with its user. Systems that achieve such automatic personalization have been called "self-customizing software," in that the system's responses to a user are automatically customized to the personal characteristics of the user [9]. The key

question in the design of such self-customizing software is what kind of patterns can be recognized by the learning algorithms. At one end, the system may do little more than recognize superficial patterns in a single user's interactions. At the other, the system may exploit deeper knowledge about the user, what tasks the user is performing, as well as information about what other users have previously done. The challenge becomes one of identifying what information is available for the given "learning to personalize"

task and what methods are best suited to the available information.

When I used the email program on my PC to forward the file of this article to the editor of this magazine I executed a series of actions that are mostly the same ones I would take to forward any file to another user. I typically click on an item on a menu that pops up a window for the composition of an email message. A fairly routine sequence of actions then follows—I compose the message, select a menu item that creates a pop-up window into which I enter the name of the desired file to be forwarded, finally completing the

file-forwarding task by clicking on the send icon of my mail system. Readers will likely be able to think of stereotypical sequences of actions that they, too, perform in their everyday interactions with their computer system—there is nothing particularly distinctive about this example of forwarding a file via email.

We would like computer systems to recognize such stereotypical sequences of actions. Moreover, since every user is different, we should not need to anticipate all possible patterns ahead of time, but rather would hope our computers could learn them online, as each user performs his or her computer-based interactions. In the absence of any detailed knowledge about the task the user is trying to achieve and the actions the user performs, what can the computer system do? The most simple-minded—and surprisingly effective—approach for predicting what a user will do next is based on the simple fact that users often repeat themselves. If you've previously seen the last few user actions in a sequence, then the action that followed them in the past is a better guess than most for what action the user will take next.

Should the system look for previous occasions where the user executed the single action the user just performed? Or should the system consider the sequence of, say, 19 previous actions the user just performed? More generally, how many preceding actions should the system match against the user's history to predict the user's next action? The answer to this question involves a trade-off. On one hand, finding a match to a long sequence of preceding actions means the match is more specific. However, doing this may require going much farther back in time to find a match (assuming a match can even be found) than would be the case if you only considered a smaller number of preceding actions. Going farther back in time can mean going to a time period during which the user's pattern of actions may have been different. Also, the smaller the number of preceding actions considered, the more often the computer system will find matches in the user's past history, providing more data and, thus, a more confident prediction about the user's likely next action.

A number of researchers have developed methods that predict user's actions based solely on the user's history of past interactions with a computer system. Interestingly, methods that look solely at the user's single immediately preceding action, comparing it to past situations where the user previously took that action, often give rise to surprisingly good predictions. Versions of this idea have been explored in tasks ranging

from Unix command-line prediction [4, 7], to prediction of Web retrieval requests [8], to keystroke prediction for calculators [3].

Our own work on the Incremental Probabilistic Action Modeling (IPAM) prediction method [4], learns to predict the actions of a user interacting with the Unix command-line shell. IPAM simply records the commands observed to date, and for each command maintains a probability distribution over all commands that may follow. Each time a command is observed, the command that next followed it is recorded. The probabilities of all the commands that were not selected are scaled down by a constant factor, and the probability of the selected command is increased by a corresponding amount so that the relevant numbers once again sum to one. On studies involving 77 Unix users we found this method can predict a user's next command with 40% accuracy. Furthermore, if the system is able to present a menu of, say, five best guesses, almost 75% of the time the user's next command is in that list of five guesses. IPAM was also used to perform command completion—if the first character of the next command is observed and only those possible next commands that begin with that letter are considered, almost 80% of the time the predicted top command is actually executed, and almost 90% of the time it is one of the top-three predictions.

The punch line to all this is that IPAM merely looks at the user's last action, paying more attention to the recent past than the distant past through its scaling down of the probabilities of those actions not taken at the current point in time. But can we do much better than this fairly simple approach? Although some modest gains have been effected through the use of more complex prediction methods [7], the experimental evidence suggests that the high-level answer is no if the system is constrained to pure pattern matching of recently executed actions to the user's past history of actions. This leads to two possibilities. The first is to accept the accuracy levels that these shallow methods provide, paying attention instead to how the (sometimes incorrect) predictions can be effectively integrated into the computer system. Approaches for doing this have included placing the system's predictions into the command-line prompt [5], binding the top-rated commands to shortcut keys on the user's keyboard [7], providing the user with a check-box list of possible text completions in a note-taking interface [9], and prefetching pages into a Web-browser cache [8]. However, if the desired

personalization goal envisioned by a computer-system designer requires higher levels of predictive accuracy, or if the system must be able to make predictions in situations it has never experienced, a second course of action must be taken: developing methods for better prediction by exploiting other sources of information.

Content-based Prediction

Consider a person reading online news stories. What we would like is a system that observed what stories the user has read—and, more importantly, has not read—and learns to present the user with new articles the user will want to read. Although there is clearly systematicity in the user's actions, here the patterns require deeper analysis. Rather than mimicking user actions taken in the past, a system that effectively personalizes itself to a user's news-story interests must look inside the stories to understand how to distin-

guish those stories that interest the user from those that do not. Systems that personalize in such a fashion are often said to be "content-based," in that they base their predictions on the contents of the artifacts about which they are concerned. being so close that it is redundant with a previous story—the article is said to be of interest to the user. If not, a profile that represents the user's longer-term interests is consulted. This profile represents a simple probabilistic classifier that assigns a probability of interest to a story based on how frequently its words occur in stories the user has previously found interesting compared to the word frequencies in articles that were deemed to be of no interest. Evaluations of News Dude on a collection of real user data showed this two-model approach performed better than either model in isolation, yielding over 75% accuracy in predicting user news story preferences.

In summary, content-based approaches typically offer us a means for describing items of user interest and a means for comparing item descriptions to locate close matches. However, when using these approaches, we also usually find ourselves considering the prefer-

SINCE EVERY USER IS DIFFERENT, WE SHOULD NOT
NEED TO ANTICIPATE ALL POSSIBLE PATTERNS
AHEAD OF TIME, BUT RATHER WOULD HOPE OUR COMPUTERS
COULD LEARN THEM ONLINE, AS EACH USER PERFORMS HIS
OR HER COMPUTER-BASED INTERACTIONS.

guish those stories that interest the user from those that do not. Systems that personalize in such a fashion are often said to be "content-based," in that they base their predictions on the contents of the artifacts about which they are concerned.

News Dude [2] performs such content-based prediction to learn its users' news-story preferences. News Dude downloads news stories on such topics as politics, business, and sports. These stories are presented to the user, who is able to provide four forms of feedback to the system: the article is interesting; not interesting; redundant with what the user already knows (typically as a result of having received the information through some other recently read stories); and more articles on the given topic are desired (which is presumably stronger positive feedback than the merely "interesting" feedback option). To predict whether its user will be interested in a new story, News Dude forms two content-based user profiles. The first is a short-term profile that judges the *interestingness* of a story by how close it is to other stories the user recently read, where similarity is based on co-occurrence of words appearing in the stories. If a new story is sufficiently close—without

ences of a single user. Here, too, we can ask whether it is possible to do better. The News Dude approach learns news story preferences for each user in isolation. Even if given multiple users, it is not able to analogize across users so the reading habits of other, similar users can inform the prediction process. This is in contrast to our expectations, that we should be able to exploit such additional information in learning to predict a user's interests. But how?

Collaborative Methods

To answer this question, consider how you decide whether to go see a particular movie. Sometimes we base our decisions on published reviews, but often we use "word of mouth," basing our decision on feedback from others whose opinions we value and share. This is the basic idea underlying the collaborative methods adopted by systems such as Firefly [10], where profiles are based on user-assigned ratings of items, and predictions are based on finding users who assigned similar ratings to items the user has previously experienced. Collaborative methods recommend new items also rated highly by the collection of similar users.

The Firefly system uses such collaborative methods to recommend music to its users. It begins by obtaining ratings from its user for an assortment of musicians and music recordings. It then finds a collection of other users whose ratings of common items are similar. Finally, it recommends back to its user items that had high ratings across this set of similar users. Their experiments showed that even though their system pays no heed to the content of the items, simple matching between user ratings profiles is enough to perform surprisingly well at predicting a user's future music ratings.

So if content-based methods exploit one kind of information (about the contents of each item the user accesses) and collaborative methods exploit a second kind of information (about what others thought of each item), then combining both sources of information should do even better, right? The answer is "yes," but how to do so is not immediately clear. Content-based methods don't provide obvious ways to exploit information about other users, and collaborative methods don't provide obvious ways to exploit information about the contents of the items under consideration.

WebWatcher, a "tour guide for the World Wide Web," provides one approach to combining both content-based and collaborative information to learn its users' Web-page browsing habits [6]. WebWatcher learns to prioritize the links on each page a user visits based on the similarity of the user's goals to those of other users who have previously visited this page. At the core of WebWatcher is content-based prediction, basing its decisions on the text contents of each Web page and the anchor text of each link on the page. Information about other users is utilized by converting the task into one in which the existing content-based methods can be directly applied. Each user is required to enter a text description of the goal of the browsing session. This description is then associated with each link the user traverses. This makes it possible to assess user similarity (the core of any collaborative approach) through measures of the similarity of these text descriptions (the core of any content-based approach). Their studies show their combination of content-based and collaborative information yields results nearly as good as those achieved by humans assigned the same task, albeit at the cost of requiring each user to provide additional information about the goal of each browsing session.

An alternative approach—one that truly combines both sources of information, and paves the way for the incorporation of other sources of information—was developed in our work on movie recommendation to avoid imposing the requirement that the user provide some description of the information need [1]. A pure

content-based approach would consider the preferences of a single user, focusing on features that describe the content of the movie (such as its plot summary, its genre, or the list of actors in the film), and learn which features are best suited to predicting the user's evaluation of a movie. In contrast, a purely collaborative approach would ignore the fact that the items of interest are movies, simply finding a set of users whose ratings are similar to those of the given user, and recommending new movies whose ratings were high among these similar users.

The key idea to combining these two approaches is to recognize that each rating we obtain concerns some user/movie pair. What we would like to do is extrapolate a procedure from a collection of past user/movie pairs that will take new user/movie pairs and predict what the user's rating will be of that movie. But what do we know about a user/movie pair on which such learning and prediction can be based? The answer is we can know a fair bit about the movie itself (information about its content and about other users' opinions of it) and also about the user (information about the given user's opinions of other movies). Both content-based and collaborative information become simple pieces of information about a given user/movie pair. Given that we can represent the content-based and collaborative information for the observed user/movie pairs as a set of discrete features, we then want to learn a function that predicts the user's future opinion when given comparable information about a new user/movie pair. Our results on over 45,000 movie ratings (that is, user/movie pairs) showed this approach, if suitably executed, can surpass the performance of both pure content-based and collaborative prediction methods.

Using the Past to Predict the Future

Personalizing computer systems to the needs of their users requires recognizing patterns in users' behavior. Central to this task of learning to personalize is the task of building a model of the user. In the absence of any additional information, simply assuming the past repeats itself does a credible job of predicting a user's future interactions with a computer system, finding in the past actions similar to those taken in the present. Prediction based solely on the user's history of past actions constructs a user profile based on what the user has previously seen or done. However, even when this data is abundant, our predictions are limited by the fact this captures only one dimension of a user's behavior. As a result, we are compelled to search for other sources of data that can complement a model based on the shallow history of a user's past actions.

Each user action takes place in the context of a spe-

cific task, and the main question is what information is available so that the process of predicting the future can be maximally informed by the past. Our ultimate goal is to predict a user's actions while exploiting as much information as can be obtained. Every method for predicting a user's future actions is based on some form of user profile or model that links information about the user or the task to expectations about the user's behavior. Content-based methods build models that link information about the contents of items a user manipulates to the user's preferences concerning those items. Collaborative methods build models that link information about other users' preferences to those of a given user. These approaches for exploiting and building models from individual information sources have now given rise to work that attempts to integrate information from multiple sources. The expectation is the more cues we have linking our individual as well as collective experiences with the present conditions, the better chance we have of predicting the user's action. Developing techniques capable of using such a range of information sources is the next challenge to be met in building computer systems that effectively learn from and about their users. **C**

REFERENCES

1. Basu, C., Haym, H., and Cohen, W.W. Recommendation as classification: Using social and content-based information in recommendation. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*. 1998.
2. Billsus, D. and Pazzani, M. A hybrid user model for news story classification. In *Proceedings of the Seventh International Conference on User Modeling*. June 1999.
3. Darragh, J.J., Witten, I.H., and James, M.L. The reactive keyboard: A predictive typing aid. *IEEE Computer* 23, 11 (Nov. 1990), 41-49.
4. Davison, B.D. and Hirsh, H. Predicting sequences of user actions. *Predicting the Future: AI Approaches to Time-Series Problems*. Tech. Report WS-98-07, AAAI Press.
5. Hirsh, H., and Davison, B.D. An adaptive UNIX command-line assistant. In *Proceedings of the First International Conference on Autonomous Agents*. 1997.
6. Joachims, T., Freitag, D., and Mitchell, T. WebWatcher: A tour guide for the World Wide Web. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence*. (Aug 1997); 770-775.
7. Korvemaker, B. and Greiner, R. Predicting Unix command lines: Adjusting to user patterns. In *Proceedings of the 17th National Conference on Artificial Intelligence*. (Aug. 2000).
8. Padmanabhan, V.N. and Mogul, J.C. Using predictive prefetching to improve World Wide Web latency. *Comput. Commun. Rev.* 26, 3 (July 1996), 22-36.
9. Schlimmer, J.C. and Hermens, L.A. Software agents: Completing patterns and constructing user interfaces. *J. Artif. Intell. Research* 1 (1993), 61-89.
10. Shardanand, U. and Maes, P. Social information filtering: Algorithms for automating 'word of mouth.' In *Proceedings of CHI'95 Conference on Human Factors in Computing Systems*. 1995. ACM Press, New York, N.Y.

HAYM HIRSH (hirsh@cs.rutgers.edu) is an associate professor; **CHUMKI BASU** (cbasu@cs.rutgers.edu) and **BRIAN D. DAVISON** (davison@cs.rutgers.edu) are Ph.D. candidates in the computer science department at Rutgers University, Piscataway, NJ.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

©2000 ACM 0002-0782/00/0800 \$5.00

Communications of the ACM

October 2000

Special Bonus Distribution at OOPSLA 2000
(Minneapolis, Minnesota - October 15-19, 2000)

This issue includes a special section on Component-based Enterprise Frameworks.
Articles will focus on:

Major elements and characteristics comprising components • Current and future UML capabilities for modeling components and frameworks • Major characteristics of object-oriented enterprise frameworks • Frameworks applied to e-business solutions • Solving real-world parallel architecture and performance issues • Using XML to build and use frameworks

For more information contact:
ACM Advertising 212-626-0687
acm-advertising@acm.org