On Predicting User Intent

Nadya Belov
Department of Computer Science
Drexel University
Philadelphia, PA 19104
belov@drexel.edu

Introduction

Recent advances in computing capability and approaches have underlined the need for user support in task execution. This is especially evident in dynamic, real-time systems where decision making is critical and errors are costly. Environments such as air traffic control centers leave little time for human operators to make critical decisions. Operators must contend with a significant amount of information to be parsed and considered before a decision is made. The main outcome of working in such an environment is cognitive overload, which leads to a significant error rate in decision making. Although intelligent and fully automated systems can be built to execute simple tasks, such complex tasks as monitoring air traffic are still mostly designated for human users. How then can we alleviate cognitive overload of human operators in time-critical and demanding systems?

Many in the human-computer interaction community have concentrated on looking for a solution to relieving cognitive overload through the application of user interface engineering conventions. These methods are expressed as rules which, when followed, are said to result in user interfaces that have high usability. However, regardless of the user interface, the operator must still make decisions and perform all of the tasks himself. This paper presents an outline for creating an intelligent agent to assist the user. The role of the agent is to attempt to predict the user's intent. The agent can then act on its perceptions with the goal of accelerating the task's completion and easing the cognitive load of the user by manipulating the user interface.

Background

Previous work in predicting user intent has concentrated on several human-computer interaction aspects. For example, a method for predicting user intent in order to resolve mouse pointing ambiguity has been created (Noy 2001). This research resulted in an approach that answers the question: *On what did the user intend to click?* The approach provides three heuristic functions that evaluate the possible item on which the user meant to click. A voting system is then employed to resolve the conflicting results from the heuristics. Prediction of user actions in appliance control by isolating individual tasks as series of basic appliance commands using k-clustering has also been studied (Isbell Jr., Omojokun, & Pierce 2004). Markov chains are used to predict user intent from the isolated tasks. Lastly, research in prediction of user actions in command-line shell environments by using trees and Markov chains also exists (Hirsh & Davison 1997).

A significant amount of investigation has been conducted in predicting user intentions. Many methods, both online and offline, have been discovered. However, to the author's knowledge, none have been applied to time-critical, online systems such as airport traffic control.

Theory

User tendencies and reactions remain similar in alike situations. For example, most people prefer to define a set of preferences for their web browser, as well as other applications. Let us consider an air traffic control operator; his daily tasks are well defined. Furthermore, the operator has previously been trained to react to a variety of situations, from normal operations, such as conflict resolution, to emergency situations. The operator knows that if, for example, an anomalous situation arises, then he will need to perform a specific series of tasks. By providing the operator with an intelligent assistant able to predict his intent, the assistant can carry out basic user interface navigation tasks and allow the operator to spend more time on the decision making process or information review. If the operator is notified of an alert, his time should not be spent selecting to view the specifics of an alert and notifying others whose help he may require. Instead, his time should be spent reviewing the available information and making a decision based on the reviewed information. This could be accomplished by delegating the notification and information retrieval tasks to the intelligent agent, whose sole role is to predict the user's intent; in this case to acquire information about the task at hand.

Operator tasks are tightly defined in time-critical environments. Furthermore, operator actions tend not to oscillate severely in similar situations. Due to this, the author believes that on-line learning, which is time consuming, is unnecessary in such environments. Instead, the author proposes employing a mechanism to identify basic user tasks by evaluating typical user behavior. The behavioral data is

recorded while the users are operating the system. Complex tasks are derived from the observed data. From this information, using cross-validation machine learning methods, basic and complex tasks are isolated and models of behavior are developed. A hidden Markov decision model is built from the developed complex task models. This model can then be implemented as the logic component of the intelligent agent. The model is the policy by which the agent operates. However, because users' operational policies for different situations tend to differ according to personal judgment, dynamic policy adjustments are performed.

Application

In order to demonstrate the application and advantages of the approach described above, we present the same approach applied to an online bookstore. Let us imagine that the bookstore wishes to improve its user interface to make account registration and maintenance easier for the user. If this task is presented to a human-computer interaction specialist, he would first establish the users' persona. Specifically, he will develop a model for the classifications of users likely to use the application. He would then need to manually define all possible tasks that can be achieved by the application through examining the application's requirements document. Following the formal outlining of the possible tasks, the human-computer specialist will outline each action that can be taken at each state of a particular task. This process is extremely time consuming for any user interface of intermediate complexity. Furthermore, gathering user data and suggestions in order to improve the user interface is costly. The task of creating an intelligent agent from user suggestions is likely to produce a less capable agent than using a machine learning approach for extracting the same information from application logs. Presented below is an approach that is much less time consuming.

The usefulness and applicability of this approach from the user's perspective is best demonstrated in the following scenario. A potential bookstore user decides to purchase several books that contain useful information about the subject matter they are researching. First, they need to create an account. After they have registered with the site and created an account, they are faced with a plethora of options on the bookstore site. They can change preferences and information on their account, browse the bookstore inventory or make purchases. Additionally, they may look through their purchase logs and edit their wish-list. The machine learning approach described below can help the user automatically update his or her wish-list and suggest other relevant books.

Data is continuously collected and is in the form of HTTP requests. Users are identified by the IP address and user sessions can be identified as continuous login sessions from an IP address. Once the log files are parsed and individual user sessions are created, one is able to construct *replays* of the sessions. That is, each user session is assigned a unique id and each sequence of states is assigned a probability of occurrence. *N-grams* are employed in order to evaluate the probability that given that a user is at a particular section of the login process, he or she will perform a specific action. Let us present an example of a session: $C = \{S_0, S_1, S_2, S_3, S_4, ..., S_{10}\}$ where S_i represents a state in the login process. A *3-gram* is represented as $\{S_i, S_{i+1}, S_{i+2}\}$ and a *2-gram* is represented as $\{S_i, S_{i+1}\}$. The purpose of the *N-grams* is to predict, given the previous states, the probability that the next state the user will proceed to is the n^{th} in the gram. *N-grams* can be expensive to compute, depending on the size of the logs and the richness of the data within them. Furthermore, it is necessary to establish the best N necessary for the *N-grams*. That is, if the N chosen is too low, then the accuracy of prediction of the next state will be too low, if N is chosen to be too high, then the probabilities will be learned directly from the logs which will also produce inaccurate results. Therefore, choosing the most appropriate N is paramount. This process is accomplished by computing the entropy of the series of *N-grams* computed form the user sessions. The lower the entropy of a model, the richer the information produced by that *N-gram* model. By computing the entropy of a selected set of *N-grams*, we isolate the best N. However, the *N-gram* with every N value may still be valuable depending on a specific situation. In order to maximize the performance of our state prediction algorithm, a linear interpolation I presented in the following equation:

$$\begin{aligned} I \leftarrow 0.0134P(S_1) \quad &+ \quad 0.222 \cdot P(S_1|S_2) \\ &+ \quad 0.765 \cdot P(S_1|S_2, S_3). \end{aligned}$$

We create a hidden Markov model using the interpolation provided in the above equation and the session models for all user sessions.

Remarks and Future Work

This paper has outlined a problem inherent in current human-computer interaction design methodology. An innovative approach to this problem has been discussed, integrating intelligent agents and machine learning with human-computer interface user support. User reactions to their application environment are a valuable resource. Future directions of this work will lead to the investigation of online self-augmenting algorithms for modifying agent behavior based on user reactions. The reactions, whether approving or otherwise (i.e. the user undoes the agent's action), can be used to modify the agent's logic.

References

Hirsh, H., and Davison, B. D. 1997. An adaptive UNIX command-line assistant. In *Proceedings of the first international conference on Autonomous agents*, 542–543.

Isbell Jr., C. L.; Omojokun, O.; and Pierce, J. S. 2004. From devices to tasks: automatic task prediction for personalized appliance control. *Personal Ubiquitous Computing* 8:146–153.

Noy, D. 2001. Predicting user intentions in graphical user interfaces using implicit disambiguation. In *Proceedings of Human Factors in Computing Systems*.