Experiments in UNIX Command Prediction

Brian D. Davison and Haym Hirsh
Department of Computer Science
Rutgers, the State University of New Jersey
Piscataway, NJ 08855, USA
{davison,hirsh}@cs.rutgers.edu

Most users demonstrate regularities in their work with a computer system. Even when a user's activities are unique, those interactions often exhibit systematic patterns. Accordingly, there has been a range of work developing systems that recognize regularities in computer usage (Cypher 1993; Mitchell *et al.* 1994; Schlimmer & Hermens 1993). Our current effort is to consider different methods that would enable us to build similar pattern-recognition into the UNIX command shell (Hirsh & Davison 1997; Davison & Hirsh 1997), although the concept of command prediction is similarly applicable to other user interfaces.

The prediction of a user's next command by the shell requires some mechanism that considers the user's current session and from that information generates a prediction for the next command. We gather data by recording the sequence of commands executed as well as other pertinent information about the state of the shell. This forms the basis for applying an off-the-shelf inductive-learning method.

To determine the potential for such methods, we collected command histories from over 70 users and computed online predictive accuracies for five methods on their data (Davison & Hirsh 1997). Online evaluation (train on the previous n data points; test on the current one) was used instead of the traditional training and test set partitioning or cross-validation because of the sequential nature of the data.

We found that relatively straightforward, knowledge-free methods were able to correctly predict the next command (without arguments) that the user would execute up to 45% of the time. The best performance was achieved using C4.5 (Quinlan 1993) on two features (the two previous commands), and a training window of 1000 data points. Each subject contributed histories with, on average, 2184 commands, each with an average length of 3.77 characters. Assuming that a correct prediction could be inserted with a single character, this method would have saved just over 31% of the keystrokes typed, which is very close to the 33% expected if predictability were independent of command length.

Other combinations of different algorithms, window sizes, and number of features were also considered. Additionally, we tried incorporating two attributes that represented state information (the terminal type and machine name), but found that on average they were not useful.

Also notable was that the simple methods of predicting whatever command was most recently used (MRC) and most frequent command (MFC) had average online predictive accuracies of 30% and 18%, respectively.

We have demonstrated the applicability of a 'weak' method to the task of command prediction. It is our belief that this work can form the basis upon which domain-specific knowledge can be added to achieve even better performance.

References

Cypher, A., ed. 1993. *Watch What I Do: Programming by Demonstration*. Cambridge, MA: MIT Press.

Davison, B. D., and Hirsh, H. 1997. Toward an adaptive command line interface. In *Proceedings of the Seventh International Conference on Human-Computer Interaction*. San Francisco, CA: Elsevier Science Publishers.

Hirsh, H., and Davison, B. D. 1997. An adaptive UNIX command-line assistant. In *Proceedings of the First International Conference on Autonomous Agents*. Marina del Rey, CA: ACM Press.

Mitchell, T.; Caruana, R.; Freitag, D.; McDermott, J.; and Zabowski, D. 1994. Experience with a learning personal assistant. *Communications of the ACM* 37(7):81–91.

Quinlan, J. R. 1993. *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Schlimmer, J. C., and Hermens, L. A. 1993. Software agents: Completing patterns and constructing user interfaces. *Journal of Artificial Intelligence Research* 1:61–89.