Exploring User Attitudes Towards Different Approaches to Command Recommendation in Feature-Rich Software

Michelle Wiebe
University of Manitoba
Winnipeg, Canada
wiebem14@myumanitoba.ca

Denise Y. Geiskkovitch
University of Manitoba
Winnipeg, Canada
denise.geiskkovitch@gmail.com

Andrea Bunt
University of Manitoba
Winnipeg, Canada
bunt@cs.umanitoba.ca

ABSTRACT
Feature-rich software applications offer users hundreds of commands, yet most people use only a very small fraction of the available command set. Command recommenders aim to increase awareness of an application's capabilities by generating personalized recommendations for new commands. A primary distinguishing characteristic of these recommenders concerns the manner in which they determine command relevance. *Social* approaches do so by analyzing community usage logs, whereas, *task-based* approaches mine web documentation for logical command clusters. Through a qualitative study with sixteen participants, in this work we explored user attitudes towards these different approaches and the supplemental information they enable.

Author Keywords
Software learnability; Recommender systems; Explanations

ACM Classification Keywords
H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCTION
Feature-rich software, such as Photoshop and the GNU Image Manipulation Program (GIMP), can contain hundreds or even thousands of commands. Most users, however, take advantage of very little of an application's power. For example, prior work indicates that people tend to use only a very small percentage (e.g., 5%) of their software's command set (e.g., [9]).

One approach to helping users become more aware of their software's capabilities has been through command recommender systems. These systems monitor the commands that a user is using, and then return new and ideally relevant commands. In determining command relevance, such recommender systems have applied two primary strategies. *Social* command recommender systems apply collaborative filtering techniques on community usage logs (e.g.,[11,12]), whereas *task-based* recommenders mine web documents (describing how to perform particular tasks) for logical command groupings [8].

Prior work indicates that the manner in which an intelligent system communicates and justifies its advice can substantially impact user attitudes towards, and acceptance of the system (e.g., [3,5,7,13]). Prior work also suggests that the nature of such explanations will depend heavily on the system's underlying reasoning techniques [16]. This is particularly true when considering the fundamental differences between social and task-based approaches to command recommendation. For example, in explaining its recommendations, a social recommender could provide statistics on community and "peer" usage, whereas a task-based recommender could list related tasks.

This research seeks to understand how users might respond to social vs. task-based command recommenders from the perspective of the explanations that they facilitate. In particular, we conducted a qualitative laboratory study with sixteen participants to examine how users respond to social vs. task-based command recommendation approaches and how the manner in which the systems justify their recommendations impacts users' attitudes. Our findings point to potential individual differences with respect to preferences and attitudes towards the utility of social vs. task-based recommender systems, and highlight a number of important considerations moving forward.

RELATED WORK
Our coverage of related work focuses on the two areas central to this research: command recommender systems for feature-rich software and intelligent systems that augment their advice with supplementary explanations.

Command Recommenders
Prior work on command recommender systems for feature-rich software has primarily focused on the algorithmic components of providing tailored recommendations. As described above, one primary approach to recommending new commands has been to leverage usage logs collected from an application's community of users [10–12]. An example is CommunityCommands [10,12], which uses collaborative filtering techniques to determine relationships between commands. A second approach to command recommendation has been to use command clusters mined

IUI'16, March 07 - 10, 2016, Sonoma, CA, USA

ACM 978-1-4503-4137-0/16/03$15.00
DOI: http://dx.doi.org/10.1145/2856767.2856814

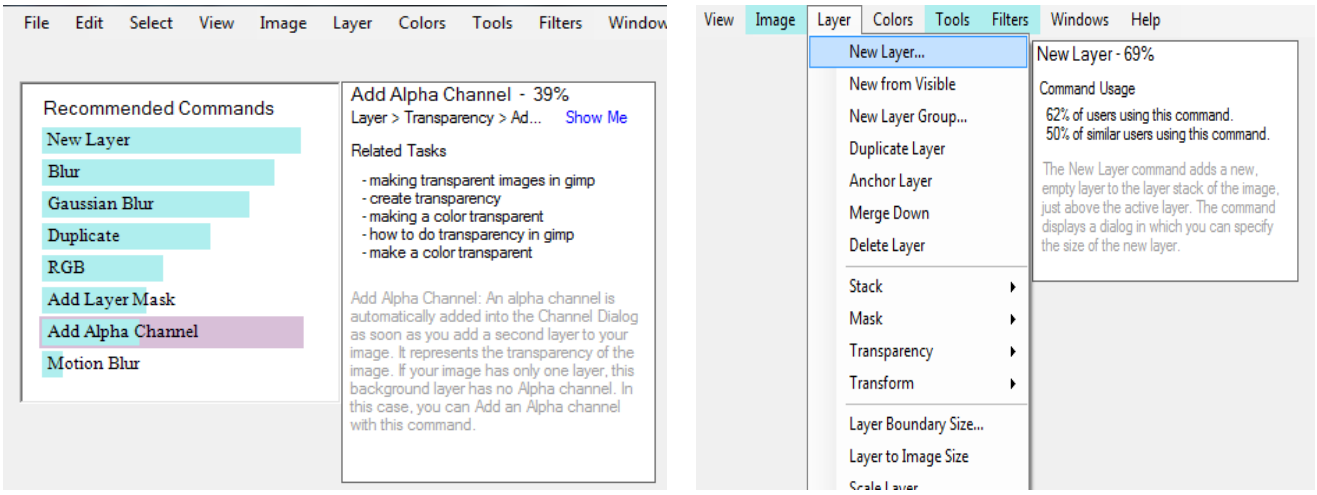

Figure 1: The supplementary information for each recommender system type: Task-Based (Left) and Social (Right). The left shows what the recommendations looked like when displayed in a palette. The right is an example of integrated recommendations.

from web documentation. For example, the QFRecs system [8] leverages Query-Feature Graphs [4] (command-to-task mappings mined from web documentation) as an automatically generated plan library. Using this plan library the system then estimates the user's most likely set of current tasks based on their recent command use. Based on these task estimates, the system recommends additional commands that have been associated with those tasks in online documentation (see [8] for a more detailed description).

In evaluating the above approaches, researchers have focused on assessing the relevance of the system's recommendations to a user's workflow (e.g., [8,12]), as well as the impact of the recommendations on task performance (e.g., [8]) and recommendation uptake (e.g., [10]). Little attention, however, has been provided to the manner in which the systems communicate their recommendations to their users, which is the focus of our work.

Explanations in Intelligent Systems

There is a rich history of exploring why and how intelligent systems should explain their behavior to their users (e.g., [1–3,5,6,13,14]). While a great deal of this work supports the inclusion of such insight [5,13], prior work also suggests that the content of the explanations (or supplemental information) can impact user perceptions of the utility of those explanations and of the system's behavior (e.g., [2,3,6,14]). We extend this work by comparing two types of explanation interfaces for command recommender systems.

THE RECOMMENDER SYSTEMS AND THEIR EXPLANATION INTERFACES

In this section, we describe two explanation interfaces that we implemented and studied as part of this work: one for a social recommender based on the CommunityCommands approach [12], and one for a task-based recommender based on the QFRecs system [8]. In both cases, the systems' explanations provide insight into why the commands have been recommended as opposed to explaining the technical details of the underlying algorithms. We created both recommender systems in a mock GIMP interface.

Guided by previous work on explaining collaborative-filtering based movie recommendations [6], our *social* recommender supplements each command recommendation with the percentage of all users who have used the command, along with the percentage of similar users who have used it (see Fig. 1, Right).

The task-based recommender system, on the other hand, displays a list of tasks associated with each command (see Fig. 1, Left). This list of tasks is taken from the recommender system's underlying Query-Feature Graph, which maps each command in the interface to a set of high-level descriptions of potential tasks involving that command [4].

Both explanation interfaces include the system's confidence in its recommendation (displayed next to the command name) and a tool-tip-like description of the command.

We implemented two presentations techniques for each recommender type, which differ according to *where* the recommendations are located in the interface. With the *integrated* technique, recommendations are presented within the menus (see Fig. 1 Right), whereas with the *palette* technique, recommendations are removed from the menus, found instead in a separate interface component (see Fig. 1 Left). We implemented these two techniques in the event that user response towards the different forms of supplemental information depends on where the recommendations were located in the interface.

STUDY

The goal of our qualitative study was to solicit user attitudes and opinions towards the social and task-based approaches to command recommendation and the two forms of supplemental information that they afford (i.e., information on community usage vs. information on related tasks).

Design

Our primary factor of interest was *system type (social vs. task-based*), which was a within-subjects factor in our study (i.e., all participants experienced both systems). We also included *recommendation location (integrated vs. palette)* as a between-subjects factor.

Participants

Sixteen participants (7 male, 9 female) took part in the study. Participants were 18-51 years old (mean age 29.5) and were recruited from the university community through the use of public bulletins. Three participants were GIMP novices, 8 reported using the software at least once per year, 4 used it at least once per month and 1 used it at least once per week Participants were provided with a $15 honorarium.

Tasks and Procedure

While our focus was on qualitative data (i.e., attitudes towards the recommender systems and their supplemental information), we wanted participants to be able to experience working with the recommender systems. To do so, we created four isomorphic tasks modeled after online tutorials for the GIMP software. Each task involved a list of steps for the participant to complete. The names of commands were removed from the steps, and only information about what the participant needed to accomplish in that step was included (e.g., "remove all color from image" for the desaturate command).

Using fully implemented versions of the recommender systems as guides, we tailored the actual recommendations generated in the study to ensure that both recommender systems were equally helpful to the tasks at hand. Specifically, we ensured that the needed command was presented in the recommended set of commands for each step. For the social recommender system's, we used GIMP usage data collected through the Ingimp project as the community data source [15]. For the task-based recommender system, we used the Query-Feature Graph as described in [8].

The remainder of the procedure for the study was as follows: Participants completed two tasks, with a maximum of 15 minutes each, with each recommender system (social or task-based). Upon finishing these two tasks, they completed a short system-specific questionnaire. After interacting with both systems, they completed an additional comparative questionnaire, and we conducted a semi-structured exit interview. The order of the recommender systems was counterbalanced, while task order was randomized. The entire study session lasted 60-90 minutes.

Results

Our analysis did not find any interactions between participants' attitudes towards the recommender systems and the location of the recommendations (integrated vs. palette). Thus our description of results considers only recommender system type. We begin by presenting the questionnaire data, and follow this with insight from the interviews as to why participants preferred one system over another.

Questionnaire Item	Task-Based	Social
How satisfied are you with the recommendation application?	5 (1.2)	5.1 (1.3)
This recommendation application would motivate you to learn new commands in the future.	5.19 (0.96)	5 (2.4)
How trustworthy were the recommendations?	4.86 (1.85)	4.81 (1.10)

Table 1: Mean (stdev) responses to questionnaire items administered after each system (1 == low, 7 == high).

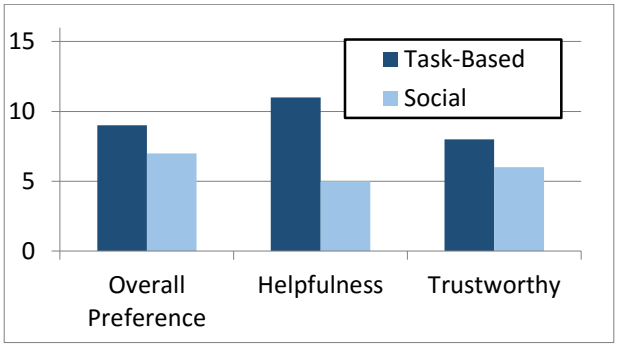

Figure 2: Users' preference overall and in terms of the systems' helpfulness and trustworthiness.

Ratings and Overall Preference Data

The questionnaires administered after each condition revealed limited impact of system type on user attitude toward each system directly after use (see Table 1). As would be expected, an RM ANOVA confirmed that there was no significant impact of system type on participants' mean responses towards their satisfaction with the system, the degree to which they would be motivated to learn new commands, and the trustworthiness of the recommender system ($p >= 0.5$ in all cases). We did, however, see a great deal of variability in participants' responses, which we elaborate on in the next section.

Responses on a comparative questionnaire revealed that users' overall preference between the two systems was almost evenly divided (see Figure 2). There was a larger difference, on the other hand, in terms of which system participants found more helpful (see "Helpfulness" in Fig. 2). Along this dimension, 11 participants favored the task-based system and 5 participants favored the social recommender.

Interviews

In the interviews, we asked participants to explain why they preferred one recommender system over another. Their responses point to a number of key issues for these types of command recommendations and their supplemental information.

Valuing Context of Use: Given the focus on new commands, a number of participants appreciated the additional context that the task-based recommender provides. The following two quotes illustrate this type of feedback:

I found them both helpful, but the list of related tasks gave me more context for what other sorts of things this might do, or might be related to. (P8)

I prefer when the recommendations give me some idea about which task each command can be used rather than what other people use. (P3)

The quotes above suggest that when recommending new commands, an indication of how the command might be relevant to the user's work is appreciated.

Identifying Rare Commands: While not a commonly expressed sentiment, one participant felt that a task-based recommender system had the potential to recommend more nuanced commands that might not be as popular:

The task specific ones were better for lesser known commands that a beginner user may not know about. (P10)

Mixed Views on Command Popularity: As for the social recommender, participants expressed mixed views on the value of "popularity" based recommendations. A couple of participants felt that knowing the popularity of a command could help them decide whether or not it was worth learning:

If a lot of people are using a particular tool, it means that it must be useful, but if it's something that maybe not a lot of people are using, then maybe it's not that useful. (P12)

Whereas many others felt skeptical of the "wisdom on the masses":

I felt that the percentages of users just meant that they might be making the same mistakes that I might be making. So just because everybody jumps off a cliff, doesn't mean it's the right thing to do. (P2)

I want to do it the fastest way. Just because someone recommended it, doesn't mean that it's better. (P5)

It did say the percentage of people use this, but then I think 'what if they could be wrong' (P9)

The above quotes suggest that with social recommender systems, some users are skeptical as to whether others are doing things the "right" or most efficient ways.

Desire for Additional Information: A few participants wanted more insight into where the usage statistics were coming from. Without this information, they lacked confidence that the "similar users" (see Fig.1, Right) had been properly identified:

I trust my coworkers, I know what they're capable of. I don't necessarily trust other people. (P5)

Some other participants felt that sheer usage statistics alone were not sufficient without knowing whether or not the command has been helpful:

Success rate! So if this command was recommended to someone else, and they were successful at their task, that would be interesting to see. Because it's not just recommendation, it's like 'oh yeah they did it, and this is how it turned out.' So it was successful or it wasn't to what extent. (P2)

While doubts concerning data source were most commonly expressed in regards to the social recommender, one participant wanted to be reassured that both data sources had been carefully selected:

I think if I knew exactly where these recommendations came from, for example, the task specific coming straight from a useful website, and the social input coming from experienced users, then I think I'd be more likely to consider it reliable. (P11)

DISCUSSION AND FUTURE WORK

Our results indicate that the type of command recommender system that users' value might to be highly dependent on the individual. Our results also suggest a number of ways that these systems and their supplemental information can potentially be improved in the future. We note, however, that our findings are based on a relatively small sample, which consisted primarily of novice or intermediate GIMP users. Participants in our study also interacted with the recommenders in a task-focused manner, as opposed to, for example, a more exploratory learning context. Future evaluations should explore the generalizability of our findings to larger sample sizes, as well the impact of software expertise and context of use on attitudes and preferences.

Given our participants' split opinions, one interesting avenue for future work would be to explore a hybrid approach for either the recommender itself or for the supplementary information it provides. For example, if both approaches were implemented simultaneously, a user could potentially filter or inspect the recommendations according to the data source that they find most reliable or motivating.

Our results also suggest an opportunity to allow users to tune the systems by letting them refine and restrict the data sources that each algorithm uses. For the social approach, however, this could potentially introduce privacy concerns, as a tightly restricted peer group could also permit detailed monitoring of command usages and workflows.

Finally, in moving beyond qualitative impressions, future field evaluations should examine the impact of the different approaches on recommendation uptake. For example, it could be that the social vs. task-based recommender systems differ in their abilities to ultimately encourage users to experiment with new commands.

CONCLUSIONS

In this paper we explored two different approaches to command recommendation for feature-rich software (social vs. task-based) from the perspective how they explain their recommendations. The results of our qualitative study revealed strong individual differences in users' attitudes towards the systems and their command recommendations. Our results also point to a number of promising avenues for future investigation, including exploring hybrid approaches to command recommendation and creating interfaces that allow users to tailor the systems' data sources.

ACKNOWLEDGMENTS

This work was supported by the National Sciences and Engineering Research Council (NSERC).

REFERENCES

1. Andrea Bunt, Matthew Lount, and Catherine Lauzon. 2012. Are explanations always important? A study of deployed, low-cost intelligent interactive systems. *Proceedings of the ACM Conference on Intelligent User Interfaces*, 169–178. http://doi.org/10.1145/2166966.2166996
2. Andrea Bunt, Joanna Mcgrenere, and Cristina Conati. 2007. Understanding the utility of rationale in a mixed-initiative system for GUI customization. *Proceedings of the International Conference on User Modeling*, 147–156. http://doi.org/10.1007/978-3-540-73078-1_18
3. Kate Ehrlich, Susanna E. Kirk, John Patterson, Jamie C. Rasmussen, Steven I. Ross, and Daniel M. Gruen. 2011. Taking advice from intelligent agents: The double-edged sword of explanations. *Proceedings of the ACM Conference on Intelligent User Interfaces*, 227–134. http://doi.org/10.1145/1943403.1943424
4. Adam Fourney, Richard Mann, and Michael Terry. 2011. Query-feature graphs: Bridging user vocabulary and system functionality. *Proceedings of the ACM Symposium on User Interface Software and Technology*, 207–216. http://doi.org/10.1145/2047196.2047224
5. Shirley Gregor and Izak Benbasat. 1999. Explanations from intelligent systems: theoretical foundations and implications for practice. *MIS Quarterly* 23, 4: 497–530. http://doi.org/10.2307/249487
6. Jonathan L Herlocker, Joseph a Konstan, and John Riedl. 2000. Explaining collaborative filtering recommendations. *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, 241–250. http://doi.org/10.1145/358916.358995
7. Anthony Jameson. 2009. Understanding and dealing with usability side effects of intelligent processing. *AI Magazine* 30, 4: 23–40.
8. Adnan Alam Khan, Volodymyr Dziubak, and Andrea Bunt. 2015. Exploring personalized command recommendations based on information found in Web documentation. *Proceedings of the ACM Conference on Intelligent User Interfaces*, 225–235. http://doi.org/10.1145/2678025.2701387
9. Benjamin Lafreniere, John S Whissell, Charles L. A. Clarke, and Michael Terry. 2010. Characterizing large-scale use of a direct manipulation application in the wild. *Proceedings of Graphics Interface*, 11–18.
10. Wei Li, Justin Matejka, Tovi Grossman, Joseph A. Konstan, and George Fitzmaurice. 2011. Design and evaluation of a command recommendation system for software applications. *ACM Transactions on Computer-Human Interaction* 18, 2: 1–35. http://doi.org/10.1145/1970378.1970380
11. Frank Linton and Hans-Peter Schaefer. 2000. Recommender systems for learning: Building user and expert models through long-term observation of application use. *User Modelling and User-Adaptive Interaction* 10, 2-3: 181–208. http://doi.org/10.1023/A:1026521931194
12. Justin Matejka, Wei Li, Tovi Grossman, and George Fitzmaurice. 2009. CommunityCommands : Command recommendations for software applications. *Proceedings of the ACM Symposium on User Interface Software and Technology*, 193–202. http://doi.org/10.1145/1622176.1622214
13. Pearl Pu and Li Chen. 2006. Trust building with explanation interfaces. *Proceedings of the ACM Conference on Intelligent User Interfaces*, 93–100. http://doi.org/10.1145/1111449.1111475
14. Simone Stumpf, Vidya Rajaram, Lida Li, Margaret Burnett, Tomash Dietterich, Erin Sullivan, Russell Drummond, and Jonathan Herlocker. 2007. Toward harnessing user feedback for machine learning. *Proceedings of the ACM Conference on Intelligent User Interfaces*: 82–91. http://doi.org/10.1145/1216295.1216316
15. Michael Terry, Matthew Kay, Brad Van Vugt, Brandon Slack, and Terry Park. 2008. Ingimp: introducing instrumentation to an end-user open source application. *Proceedings of the ACM Conference on Human Factors in Computing Systems*, 607–616. http://doi.org/10.1145/1357054.1357152
16. Nava Tintarev and Judith Masthoff. 2011. Designing and evaluating explanations for recommender systems. In *Recommender Systems Handbook*. 479–510. http://doi.org/10.1007/978-0-387-85820-3