AN EXPLANATION-CENTRIC APPROACH FOR PERSONALIZING INTELLIGENT AGENTS

Todd Kulesza

Thesis Proposal

Submitted 22 December 2011

ABSTRACT

Intelligent agents are becoming ubiquitous in the lives of everyday users, from simple recommenders like Google Suggest to the complex face recognition techniques used in modern photo albums. The research community, however, has only recently begun to study how people (1) establish an appropriate level of trust in, and (2) communicate with, such agents. I plan to design an explanation-centric approach to support end users in personalizing their intelligent agents and in assessing their strengths and weaknesses. My goal is to define an approach for helping people understand when they can rely on their intelligent agents' decisions, and allowing them to directly debug their agents' reasoning when it does not align with their own.

1 INTRODUCTION

Intelligent agents have moved beyond mundane tasks like filtering junk e-mail. Search engines now exploit pattern recognition to detect image content (e.g., clipart, photography, and faces); Facebook and image editors take this a step further, making educated guesses as to *who* is in a particular photo. Netflix and Amazon use collaborative filtering to recommend items of interest to their customers, while Pandora and Last.fm use similar techniques to create radio stations crafted to an individual's idiosyncratic tastes. Simple rule-based systems have evolved into agents employing complex algorithms. These *intelligent agents* are computer programs whose behavior only becomes fully specified *after* learning from an end user's training data. Advances in machine learning and pattern recognition continue to unlock new applications for intelligent agents, but two challenges prevent end users from fully benefiting from the automated decisions and recommendations from these tools.

First, end users of intelligent agents need to understand when they can rely on, or trust, their agent's work. Such trust is highly contextual—some of the agent's predictions may not matter at all to an end user, while others may matter a great deal. Further, the agent's reasoning is constantly changing as it learns from the user's behavior, so a system that was reliable yesterday may not be trustworthy today.

The second challenge is about personalization. When an intelligent agent's reasoning causes it to perform unexpectedly in the field, only the end user is in a position to personalize, or more accurately, *to debug,* the agent's flawed reasoning. Here, debugging refers to *mindfully and*

purposely adjusting the agent's reasoning (after its initial training) so it more closely matches the user's expectations. Recent research has made inroads into supporting this type of functionality [Amershi et al. 2010, Kapoor et al. 2010, Kulesza et al. 2010, Lim and Dey 2010], but debugging can be difficult for even trained software developers—helping end users, who have knowledge of neither software engineering nor machine learning, is no trivial task.

I believe these two challenges—establishing an appropriate level of trust in an agent, and aligning its reasoning with a specific end user's—are inherently linked. A sound understanding of an agent's reasoning seems a logical prerequisite for both assessing the agent's reliability and providing useful corrections to this reasoning. My thesis work aims to understand how people reason about intelligent agents, how people communicate with intelligent agents, and how intelligent agents can improve their reasoning using these communications. I hypothesize that explaining an agent's reasoning to end users will enable them to form better judgments of the agent's reliability, and help users to provide feedback that can substantially improve the agent's future predictions, as illustrated in Figure 1.

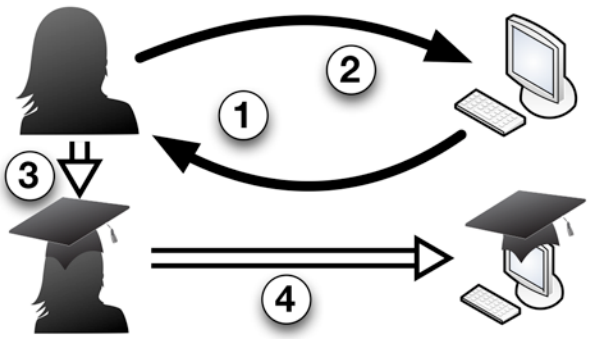

Figure 1: I envision a cyclic, explanation-based approach for users to learn about the agent's reasoning (1) and interactively adjust it (2). In the process, the user learns more about how to effectively steer the agent (3), with the eventual outcome of "more intelligent" intelligent agents (4).

1.1 Proposed Contributions

This thesis will make the following contributions:

- An understanding of how end users build mental models of intelligent agents, and how these models impact trust in and feedback toward intelligent agents.
- An understanding of how attributes of explanations (e.g., salience, visibility, and concreteness) impact mental models, and how they may prime end users to provide particular types of feedback.
- Methods for systematically identifying which of an intelligent agent's outputs matter to the user, and an exploration of adaptations to active learning techniques to elicit feedback that will best align the agent's reasoning with that of the end user.
- An approach for combining the above methods and an evaluation of this approach in the domain of music recommendation.

2 BACKGROUND AND RELATED WORK

2.1 Reasoning About Intelligent Agents

Because much of this thesis rests on end users' ability to understand and adjust an intelligent agent's reasoning, an understanding of how users build mental models of this reasoning is foundational. *Mental models* are internal representations that people generate based on their experiences in the real world. These models allow humans to understand, explain, and predict phenomena, with an end result of being able to then act accordingly [Johnson-Laird 1983]. Understanding how end users build mental models of intelligent agents, and how these models shift over time, may allow researchers to design more effective explanations of an agent's reasoning than an ad-hoc approach.

The contents of mental models can be concepts, relationships between concepts or events (e.g., causal, spatial, or temporal relationships), and associated procedures. For example, an end user's mental model of how a computer works may be as simple as a screen that displays everything typed on a keyboard, and the sense that it "remembers" these things somewhere inside the computer's casing. Mental models can vary in their richness—an IT professional, for instance, likely has a much richer mental model representing how a computer works than the above example.

The varying richness of mental models results in a useful distinction: *Functional* (shallow) models imply that the end user knows how to use the computer, but not how it works in detail, whereas *structural* (deep) models provide a detailed understanding of how and *why* it works. These models do not have to be entirely *complete* (i.e., encompass all relevant details) to be useful, but they must be *sound* (i.e., accurate) enough to support effective interactions. Many instances of inaccurate mental models (both functional and structural) guiding erroneous behavior have been observed [Jonassen and Henning 1996, Norman 1983]. In terms of assessing and personalizing intelligent agents, such erroneous behavior may manifest as an *inappropriate level of trust* in the agent, or as *harmful feedback* that causes the agent's reasoning to diverge even further from the end user's.

Mental models may develop "naturally" due to repeated use of or exposure to a system over time, or they may be induced more rapidly through instruction. Models that develop through use, however, may be very unsound. For example, Kempton estimates that as many as 50% of Americans erroneously believe their home furnace operates as a valve or faucet—that turning the thermostat higher actually raises the temperature of the air coming out of the furnace [Kempton 1986]. The source of this misunderstanding, he contends, is the abundance of devices that *do* operate using valves in our daily environments (e.g., water faucets, gas burners, automobile pedals), and the scarcity of familiar devices that behave similarly to thermostats. Providing explanations of how such unfamiliar or complex systems operate can avoid this problem, but may introduce a new challenge: how can a complex system convince an end user to pay attention to its explanations?

Theory suggests that this challenge can be overcome by communicating the benefits likely to be realized after learning more about an agent's reasoning (or conversely, the dangers—inappropriate levels of trust and harmful feedback—which can be mediated by such an understanding). Blackwell's Theory of Attention Investment hypothesizes that people determine the level of effort to expend on a given task based on their perceptions of cost, risk, and eventual benefit [Blackwell and Green 1999]. According to this theory, end users will be more likely to

invest time toward understanding an intelligent agent's reasoning when they believe the benefits will outweigh the expected costs (e.g., time and effort spent) and possible risks (e.g., failure to comprehend the agent's reasoning).

Intelligent agents may be able to naturally attract a user's attention by making a surprising decision or recommendation. Hastie investigated causal reasoning, finding that "when unexpected behaviors are attributed to a person, the perceiver is relatively likely to engage in causal reasoning to understand why these behaviors occurred." [Hastie 1984]. This finding appears to hold when the unexpected behavior is attributed to a machine, as well: Wilson et al. designed an approach—surprise-explain-reward—that successfully leveraged participants' curiosity about unexplained software behaviors to engage their attention [Wilson et al. 2003]. Participants thus engaged had the choice to view explanations describing an unfamiliar software testing tool (assertions) and the potential benefits of employing it; nearly all of their participants went on to make use of this tool [Wilson el al. 2003].

Once the agent has made a decision that was unexpected enough to arouse the user's curiosity, there is an opportunity to answer his or her "why?" question. Such an answer must be grounded in the agent's reasoning if it is to be sound, and so this explanation will necessarily affect the user's mental model of the agent. However, to date, very little work has explored how end users' build mental models of intelligent systems. One study found that prolonged use of such a system induced plausible models of its reasoning, and that these models were surprisingly hard to shift, even when people were aware of contradictory evidence [Tullio et al. 2007]. My preliminary work has found that users will change their mental models for an intelligent agent when the agent makes its reasoning transparent [Kulesza et al. 2010], however, some explanations by agents may lead to only shallow mental models [Stumpf et al. 2007]. The reasoning can also be made transparent via explicit instruction regarding new features of an intelligent agent, and this can help with the construction of mental models of how it operates [McNee et al. 2003]. None of these studies, however, investigated the effects of mental model construction and evolution on how end users personalized or assessed intelligent agents.

In addition to piquing curiosity, witnessing unexpected behavior by an intelligent agent is also likely to impact a user's trust in the agent's reliability. Prior work has investigated how end users establish trust in intelligent agents [Glass et al. 2008], identifying a number of themes (e.g., transparency, user expectations) that impact user trust in such systems; thus, the explanations offered as part of a surprise-explain-reward strategy may also alter user's trust in the agent's decisions. Other researchers have identified a positivity bias toward trusting intelligent agents, suggesting that most end users are willing to trust their agents without needing initial proof that they are actually trustworthy [Dzindolet et al. 2003] (this same bias is well-established among human interactions as well [Bruner and Tagiuri 1954]). However, Dzindolet et al. found that end users quickly lose trust in agents once even a handful of agent mistakes are observed—even in situations where the participant was aware that he or she was making more mistakes than the agent [Dzindolet et al. 2003].

One possible explanation for this swift erosion of trust comes from theories on expectations, or *schemas.* In schema theory, unexpected situations stand out as individual events in memory, while expected situations are lumped together as a single schema [Ashcraft 1994] (e.g., someone is unlikely to remember the specifics of brushing his or her teeth this morning, but everyone can likely recall a time they dropped their toothbrush someplace they did not want it to land). Thus, after only a few mistakes on the part of the intelligent agent, an end user may be unduly influenced by the memory of these aberrations, and so form a lesser degree of trust in the system than is warranted. My own preliminary work has also found evidence of this phenomenon—in a

10-minute assessment task, participants consistently reported that the intelligent agent they were assessing was less reliable than it actually was [Kulesza et al. 2011a].

Dzindolet et al. also explored explanations of why an intelligent agent might make mistakes, finding that when participants were aware of reasons why the agent *might* fail, they were inclined to *increase* their trust in its decisions [Dzindolet et al. 2003]. Such trust, however, was often unwarranted—explanations appeared to increase participants' trust in agents regardless of how well the agent performed, causing some concern that making an agent's reasoning transparent may inadvertently increase a user's reliance on the agent to the point where it is misused. More encouragingly, the same study found that continual feedback from the intelligent agent led to very accurate perceptions of its accuracy by participants, but *only* when participants were unable to directly observe its mistakes. Such a situation seems untenable in the real world, but provides a starting point for future research.

Even outside the realm of intelligent agents, it has been argued that users should be exposed to transparent and intuitive systems and appropriate instructions to build mental models [Sharp et al. 2007]. *Scaffolded instruction* is one method that has been shown to contribute positively to learning to use a new system [Rosson and Carrol 1990]; however, this approach may be more appropriate in high-criticality situations where users are initially motivated to systematically learn about the intelligent agent's reasoning. Making an agent's reasoning transparent can improve perceptions of satisfaction and reliability toward music recommendations [Sinha and Swearingen 2002], and may extend to other types of recommender systems as well [Herlocker and Konstan 2000, Tintarev and Masthoff 2007]. However, experienced users' satisfaction with such an agent may actually *decrease* [McNee et al. 2003]. Identifying the complex factors leading to end user trust in, and satisfaction towards, intelligent agents remains an open problem.

Externalizing mental models is a well-established research problem, and one without a simple solution. Precisely *how* researchers attempt to measure participants' mental models is likely to influence those models [Doyle and Radzicki 2008], reducing their validity. For example, asking participants to draw a flowchart of a system may encourage them to think of how information "flows" through a system more than they otherwise would. Norman discusses how verbal or written elicitations may be incongruous with participants' observed actions and will be necessarily incomplete [Norman 1983]. One method for remedying this incompleteness is to provide participants with the various components of a system and ask them to properly arrange or interconnect them, but this transforms the problem from having a bias toward recall to one with a bias toward recognition [Otter 2000]. Carley and Palmquist developed a partially automated technique for defining mental models based on textual analysis, but this approach is time consuming and may still suffer from problems of incompleteness [Carley and Palmquist 1992].

2.2 End-User Testing and Debugging

It is estimated that the vast majority of people engaged in computer programming are not doing so with the primary goal of writing programs [Scaffidi et al. 2005]. These *end user programmers* are people who, frequently lacking formal computer science training, program as a means to an end, rather than an end in itself. End-user programming (EUP) provides an informative lens through which to view end users' interactions with intelligent agents—the number of end users with training in machine learning is even less than those with training in computer science, yet end users informally *test* such agents to determine their suitability for a particular purpose, and *debug* the agent's reasoning by either providing examples of how it should behave, or via direct adjustment. Advancements that help end user programmers reason about problems in spreadsheet

formulas or web-based "mashups" may be adaptable and beneficial when end users reason about their intelligent agents.

Intelligent agents are formally tested prior to deployment by machine learning specialists using statistical methods [Hastie et al. 2009]. However, such methods do not substitute for *end users'* assessment of their assistants because pre-deployment evaluation cannot assess the suitability of after-deployment customizations to a particular user. Systematic testing for end users was pioneered by the *What You See Is What You Test* approach (WYSIWYT) for spreadsheet users [Rothermel et al. 2001]. To alleviate the need for users to conjure values for testing spreadsheet data, "Help Me Test" capabilities were added; these either dynamically generate suitable test values [Fisher et al. 2006] or back-propagate constraints on cell values [Abraham and Erwig 2006]. Statistical outlier finding has been used in end-user programming settings for assessment, such as detecting errors in text editing macros [Miller and Myers 2001], inferring formats from a set of unlabeled examples [Scaffidi 2007], and to monitor on-line data feeds in web-based applications for erroneous inputs [Raz et al. 2002]. These approaches use statistical analysis and interactive techniques to direct end-user programmers' attention to potentially problematic values, helping them find places in their programs to fix. My preliminary work has explored the utility of such approaches when testing intelligent agents, finding they helped participants discover more of an agent's failures and test more of the agent's logic than regular ad-hoc assessment alone [Kulesza et al. 2011a].

There are a number of debugging approaches that help end users leverage systematic testing to find and understand the causes of faulty behavior. For example, in the spreadsheet domain, WYSIWYT allows users to test spreadsheet formulas by placing checkmarks beside correct outputs and X-marks beside incorrect outputs [Rothermel et al. 2001]. A fault localization device then traces the data "flowing" into these cells, helping users locate cells whose formulas are likely to be faulty. Woodstein serves a similar purpose, but in the e-commerce domain: this approach helps users to debug problems by explaining events and transactions between e-commerce services [Wagner and Lieberman 2004]. These approaches exemplify successful attempts to help end users first identify, and then understand the cause of, program failures. To facilitate such understanding, they work to draw a user's attention to the faulty regions of a program's logic. The Whyline performs a similar function, and as it informs much of my preliminary research, will be discussed in some detail.

The Whyline [Ko and Myers 2008] pioneered a method to debug certain types of programs in an explanation-centric way. The Whyline was explored in three contexts, encompassing both end user programmers and professional developers: (1) event-based virtual worlds written in the Alice programming system [Ko 2006] , (2) Java programs [Ko and Myers 2008], and (3) the Crystal system for debugging unexpected behaviors in complex interfaces [Myers et al. 2006]. In each case, the tools help programmers understand the causes of program output by allowing them to select an element of the program and receive a list of *why* and *why not* questions and answers in response. These "Why?" questions and answers are extracted automatically from a program execution history, and "Why not" answers derive from a reachability analysis to identify decision points in the program that could have led to the desired output. In the Crystal prototype, rather than presenting answers as sequences of statement executions, answers are presented in terms of the user-modifiable input that influenced code execution. In all of these Whyline tools, the key design idea is that users select some output they want to understand, and the system explains the underlying program logic that caused it.

Part of my preliminary research involved translating the Whyline's design concept to the domain of intelligent agents [Kulesza et al. 2011b]. This built upon the work of Stumpf et al., who

explored end-user debugging techniques for text-classifying intelligent agents. Their research began by investigating different types of explanations, as well as user reactions to these explanations [Stumpf et al. 2007]; user studies later confirmed that even simple corrections from end users have the potential to increase the accuracy of the agent's predictions [Stumpf et al. 2008, Stumpf et al. 2009]. For some participants, however, the quality of the agent's predictions actually *decreased* as a result of their corrections—there were barriers preventing users from successfully debugging the agent's reasoning. I conducted a study [Kulesza et al. 2011b] that built upon this work, identifying and categorizing these barriers and the information participants requested to overcome them.

Other researchers have also focused on the barriers and information needs end users encounter while programming. For example, Ko et al. explored learning barriers that novice programmers encountered when learning how to solve problems in an unfamiliar programming environment [Ko et al. 2004]. In their study, the two most common barriers related to properly *using* programming interfaces, and *understanding* the source of compile-time and run-time errors. (My own research found that the barriers most critical to end users debugging a text-classifying intelligent agent differed; in this domain, they most often related to *selecting* features to adjust and *coordinating* the impact of such adjustments across multiple output labels.) Researchers from the WYSIWYT project categorized the information needs of end users debugging spreadsheets [Kissinger et al. 2006], enumerating the types of information that end users sought (e.g., help reasoning about spreadsheet formulas or information about debugging strategies).

Before users can test or debug an intelligent agent's reasoning, they must first be able to see it. Explanations of agents' reasoning have taken a variety of forms, such as relating user actions and the resulting predictions [Billsus and Hilbert 2005], or explaining how an outcome resulted from user actions [Tullio et al. 2007, Vig et al. 2009]. Lim et al. explored end user understanding of four types of explanations of a decision tree-based intelligent agent (*why*, *why not*, *how to*, and *what if*), finding that explanations about *why* an agent made a particular decision were most helpful to the participant's understanding of the agent and trust in the agent's reliability (participants reported the *why not* explanations to be similarly helpful, though comprehension tests suggested that participants did not actually understand these explanations as well as they believed they did) [Lim and Dey 2009].

Much of the work in explaining probabilistic machine learning algorithms has focused on the naïve Bayes classifier, often employing visualizations such as pie charts (where each pie slice describes the weight of evidence for a particular feature) [Becker et al. 2002] or bar charts (where each bar's position describes a feature's information gain) [Kulesza et al. 2011b]. Poulin et al. employed a stacked bar chart visualization to support the more general class of linear additive classifiers, and used a prototype to successfully help biologists identify errant training samples in their data [Poulin et al. 2006]. Lacave and Díez present several approaches, both textual and graphical, for describing general Bayesian networks, but these techniques are too computationally expensive to result in a real-time "dialog" with the end user [Lacave and Díez 2003]. Additionally, Lacave and Díez enumerate the types of information that intelligent agents can provide end users: explanations of evidence (e.g., "The size, shape, and color of this fruit suggest it is an apple"), explanations of the machine learning model (i.e., the static algorithm), and explanations of the agent's reasoning (e.g., precisely how an object's size, shape, and color contributed to its classification as *apple*).

The systems described above help end users *understand* an intelligent agent's reasoning, but they do not allow users to *debug* any faults the agent may have learned. Such two-way communication has traditionally be limited to supplying the agent with additional labeled training samples,

essentially trusting the agent to figure out the problem on its own. For example, some Programming by Demonstration (PBD) systems learn programs interactively, using machine learning techniques based on sequences of user actions (see [Lieberman 2001] for a collection of such systems). When debugging this type of program, end user corrections are often limited to the addition or removal of training data, such as Gamut's feature of "nudging" the system when it makes a mistake, leading to the addition or deletion of training examples [McDaniel and Myers 1997]; the only fallbacks for richer forms of debugging rely on a traditional programming language such as Lisp (e.g., Vander Zanden and Myers [1995]). Recent work with PBD systems supports more nuanced debugging of programs [Chen and Weld 2008], but Chen and Weld's technique only allows the user to retract actions in a demonstration, resulting in missing values being added to the training data rather than directly manipulating the classifier's reasoning. Amershi et al. bridge active learning [Settles 2009] with interactive concept learning, using a mixed-initiative approach that guides users toward identifying helpful training data for learning concepts not directly represented by a classifier's features [Amershi et al. 2009]. Still other systems allow users to patch up specific mistakes by an intelligent agent, but do not take these corrections into account when the agent makes future decisions. For example, if a CoScripter program misidentifies a web page object, the user can specify the correct object for the particular page; the fix, however, will not affect how the program identifies similar objects on different pages [Little et al. 2007].

Directly manipulating the reasoning of an intelligent agent has received only limited attention. This ability, however, is important for a number of reasons. Some problem domains lack sufficient training data for traditional labeling approaches to be effective; in these circumstances, allowing end users to directly specify the features an agent should use for classification can be significantly beneficial [Wong et al. 2011]. Instance labeling approaches are also at the mercy of external forces—they require appropriate training examples to be available when the user needs them, and suffer from class imbalance problems in "bursty" domains (e.g., e-mail classification). Taking a user-centric perspective, if an end user *wants* to tell the computer "Here, this is how you should behave", he or she should not have to first find real-world examples of that behavior—one should have the choice of explaining the desired behavior as concisely as one prefers. EnsembleMatrix [Talbot et al. 2009] is one system that supports such direct manipulations, providing users with both a visualization of an agent's accuracy and the means to adjust its reasoning; however, EnsembleMatrix is targeted at machine-learning experts *developing* complex ensemble classifiers, rather than end users *working with* the resulting classifiers. Another direct manipulation system, ManiMatrix [Kapoor et al. 2010], provides an interactive visualization of a classifier's accuracy and is intended to be used by end users with no machine learning background, but user interactions are restricted to the modification of a classifier's cost matrix.

3 STATEMENT OF THESIS

The purpose of my dissertation research is to improve peoples' experiences with intelligent agents in two specific ways: (1) by helping people establish appropriate levels of trust in an agent's work, and (2) by helping people fix their own personalized agents. My central thesis is that two-way communication between the user and the intelligent agent will be necessary for each of these tasks; thus, much of my research will investigate explanations (both from the agent to the user and from the user to the agent), their potential content, and the impact this content may have on users' mental models and the agent's reasoning.

4 RESEARCH GOALS AND METHODS

4.1 Balancing Benefits Against Costs

Because replicating experiments over the diverse range of intelligent agents is a prohibitively large undertaking, I propose to first focus on the domain of music recommendation systems. Music recommendation embodies two of the components of intelligent agents I care most about—they are widely used, and their work is *beneficial* without being *critical* to their end users. This is a more challenging problem than studying agents that perform critical tasks, because non-critical systems cannot assume that users are willing to give a high level of attention and effort to interact with the system: thus, an approach must be devised that not only provides users with clear benefits, but does so at a very low cost (effort). Investigating approaches with cost-benefit ratios attractive enough to engage even users of low-criticality systems may thus produce more general (i.e., widely applicable) solutions than focusing only on systems whose users are willing to tolerate relatively high costs (effort).

Research Objective 1: Exploring the cost/benefit tradeoff to assessing and personalizing intelligent agents.

RQ1.1 (cost/benefit): How much effort will end users expend to assess and provide feedback to intelligent agents when the perceived benefits are relatively low?

RQ1.2 (cost/benefit): How can interfaces lower the cost of assessing and providing feedback?

I am currently exploring RQ 1.1 via a longitudinal study (Study #1) of end users steering a customizable music recommendation system. The system is a website that supports richer forms of feedback than are available to state-of-the-art music recommendation platforms (e.g., Pandora or Last.fm); screenshots of the feedback choices available to participants are presented in Figure 2. For this study, participants were randomly divided into two groups—one received scaffolded instruction as to how the system selected its recommendations and how it would respond to user corrections, while the second group did not. Participants were given five days to personalize the recommender using their own computers and on their own time. After this period, participants returned to the lab where we tested their comprehension of the recommender system's reasoning and assessed their satisfaction with the resulting playlists.

The preliminary results from Study #1 suggest that end users are initially willing to spend time and effort debugging an agent's reasoning, and that the more sound their mental models become while working with an agent, the more beneficial they view these debugging efforts (Figure 3).

These results are extremely encouraging for two reasons. First, my hypothesis rests on the assumption that people will expend *some* effort to personalize intelligent agents, even if such effort is limited to extremely low-cost or infrequent interactions. While this might be more safely assumed for high-criticality intelligent agents, such as aging-in-place systems, most agents do not involve criticality; if we want our results to generalize to the wide range of deployed agents, we need to keep the perceived risk and effort of personalization lower than the perceived benefit.

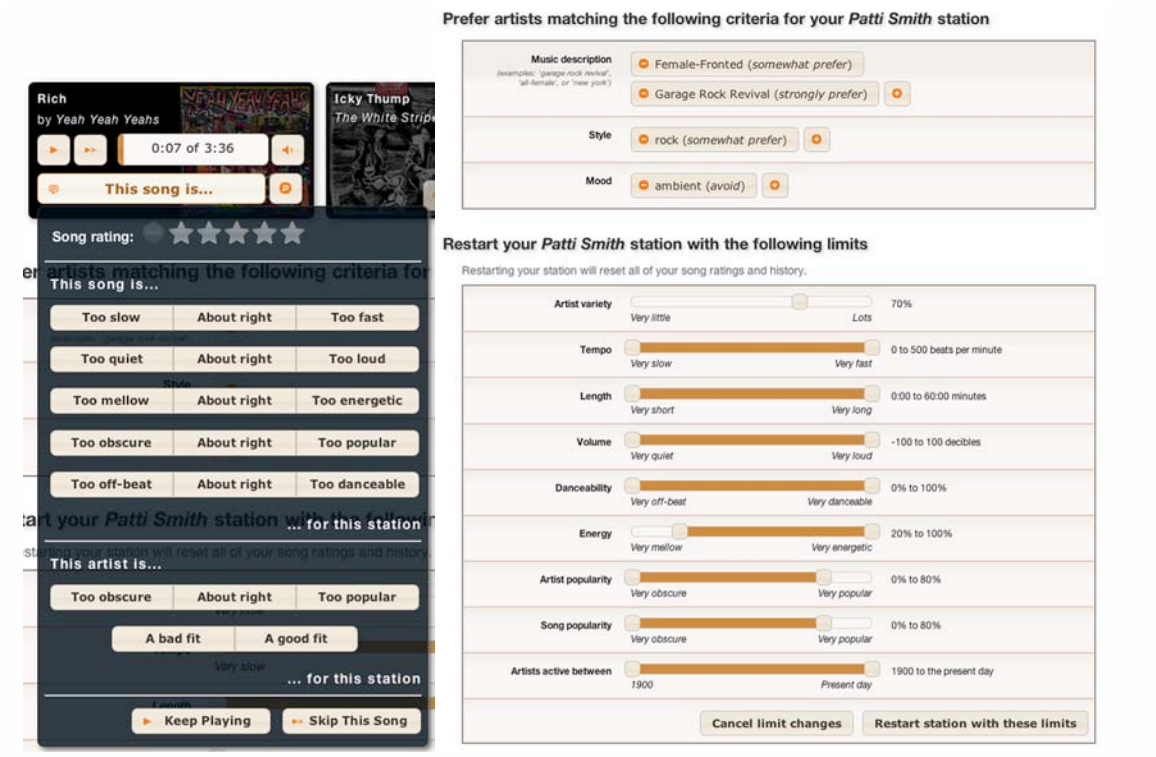

Figure 2: In Study #1, participants could steer the recommender by providing feedback relative to a particular song,
such as "This song is too slow" or "This artist is too obscure" (left),

The second reason follows naturally from the first—if end users view such personalization more favorably as they learn more about the agent's reasoning, then there is an opportunity to raise the perceived benefit of debugging. However, such perceived benefits would likely need to be realized by noticeable improvements in the agent's reasoning; without concrete evidence that the agent is improving, it seems unlikely that end users would continue to expend effort toward personalizing it. I hope to extend this analysis further, quantifying effort by examining the number of debugging attempts participants made.

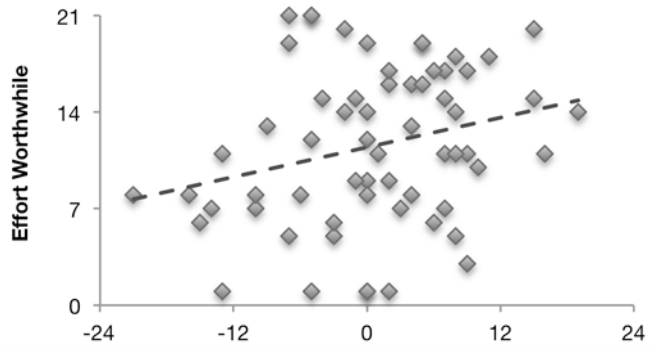

Figure 3: The extent of participants' mental model transformations (as measured via comprehension questions) was predictive of how worthwhile they viewed their debugging efforts (linear regression, *p*=.041, R2=.07, F(1,60)=4.37).

I next plan to explore RQ 1.2 via a two-part laboratory study (Study #2), using a progression of paper prototypes to investigate how end users expect to be able to personalize a music recommendation system. This study will be based upon the Natural Programming methodology [Pane et al. 2002], which stresses that programmable systems should be designed to reflect the users' goals and vocabulary as closely as possible. One notable drawback of Natural Programming, however, its lack of an evaluation methodology; Bogart et al. [Bogart 2012] are exploring an enhanced variant of Natural Programming (termed NP+) that adds a step to qualitatively evaluate the completeness of a language using a Wizard-of-Oz approach prior to a more risky summative evaluation.

My intention is to use the NP+ methodology to first identify situations where users (1) want the system to provide explanations of its actions and (2) want to correct the system's reasoning. Participants will also be asked to mark up the paper prototype with their corrections, allowing me to gather details on their preferred input methods and vocabulary. The second part of the study will present participants with a selection of explanations and feedback affordances, asking them to rank the affordances in order of preference across a range of scenarios. Participants will also be asked precisely what feedback they would give to their preferred affordance, as participant preferences may be influenced by the level of detail or type of feedback they intend to communicate to the system.

Focusing on low-criticality systems adds risk to this proposal. If my initial investigations fail to identify methods for end users to assess and personalize low-criticality intelligent agents, the remainder of my thesis would become untenable. Thus, I propose to use high-criticality agents as a fallback domain. I am well positioned to make such a transition if it becomes necessary; I have experience conducting research with a high-criticality agent (an "auto-coder" designed to assist social scientists conduct qualitative research [Kulesza et al. 2010]), and that same study also resulted in software prototype suitable for future investigations. Thus, regardless of the outcome of Research Objective 1, I will have an appropriate domain to conduct the remainder of my thesis research.

4.2 Explanations and End Users' Mental Models

My hypothesis is that sound structural mental models of the agent's reasoning will be critical to both helping users to establish appropriate levels of trust in an agent's work, and helping users to fix their own personalized agents. Because this work will inform the design of explanations of an agent's reasoning, I will also explore how different attributes of explanations (salience, visibility, concreteness, etc.) influence the feedback users provide intelligent agents. I plan to develop a taxonomy of these attributes and the various trade-offs associated with them.

Research Objective 2: Exploring the effects of mental models on assessment and personalization of intelligent agents.

RQ2.1 (mental models): Do sound structural mental models of an intelligent agent's reasoning lead to more appropriate perceptions of its reliability?

RQ2.2 (mental models): Do sound structural mental models of an intelligent agent's reasoning lead to improved end-user debugging of its reasoning?

RQ2.3 (explanation attributes): What attributes of machine-generated explanations are user-identifiable (i.e., how do end users describe them?), and how do these attributes influence the feedback provided by end users?

As discussed in Section 2.1, there is no agreed-upon formal methodology for eliciting mental models. My preliminary work has explored a scenario-based approach that involves asking participants to select from a choice of available inputs in order to elicit a particular output from the system. Reponses are weighted by correctness and confidence to arrive at a quantified score for mental model soundness. I plan to continue researching the validity of this methodology; it possesses the benefit of allowing researchers to focus their inquiries to specific aspects of mental models, supports simple comparisons between participants, and is grounded in the "runnable" nature of mental models [Johnson-Laird 1983] for external validity. However, because the possible inputs are all viewable, participants may be able to use recognition to arrive at an answer which, had they actually been "in the field", would have eluded them.

To explore RQs 2.1 and 2.2, I plan to conduct a study (Study #3) in which participants are presented with a series of scenarios of working with an intelligent agent. After each scenario, participants will be asked (1) what is wrong with the agent's current reasoning, (2) how they would expect to be able to fix these problems, and (3) how they expect the agent to perform for a selection of different inputs. The independent variable in this study will be the explanations provided to participants as part of each scenario. The results from this work will help to quantify the impact of sound models on end-user debugging and assessment of intelligent agents.

I have already conducted a study (Study #4) of RQ 2.3 using both qualitative and quantitative methods. First, formative work with a paper prototype identified a vocabulary end users wanted to employ when explaining corrections to an intelligent agent, along with the types of corrections they wanted to give the agent (Figure 4). The results of this work led to the design and development of a prototype with multiple explanation capabilities (Figure 5); an experiment conducted with four variants of this prototype found that presenting "run-time" debugging explanations to end users helped them debug the agent's reasoning more than "static" explanations of the agent's logic (Figure 6) [Kulesza et al. 2010]. Further analysis of the data from Study #2 will be used to complete the investigation of RQ 2.3. I plan to use a Grounded Theory approach to identify common attributes of participants' explanations about how the system should reason, resulting in a taxonomy of common explanation attributes. Because Study #2 will employ a natural programming methodology, the attributes I identify will reflect end users' own goals and vocabulary; thus, the taxonomy may not be complete, but it will be user-centric. A follow-up study will then explore how end users respond (in terms of feedback to the system) to explanations embodying these different attributes.

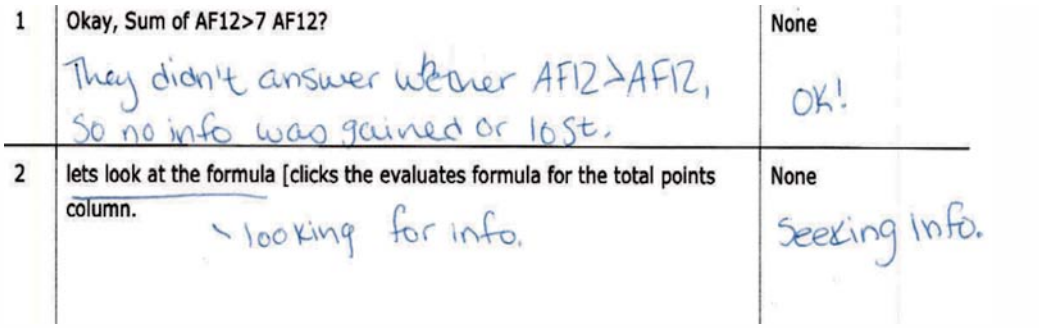

Figure 4: In Study #4, a paper prototype was used to gather "natural" explanations about participants' reasoning.

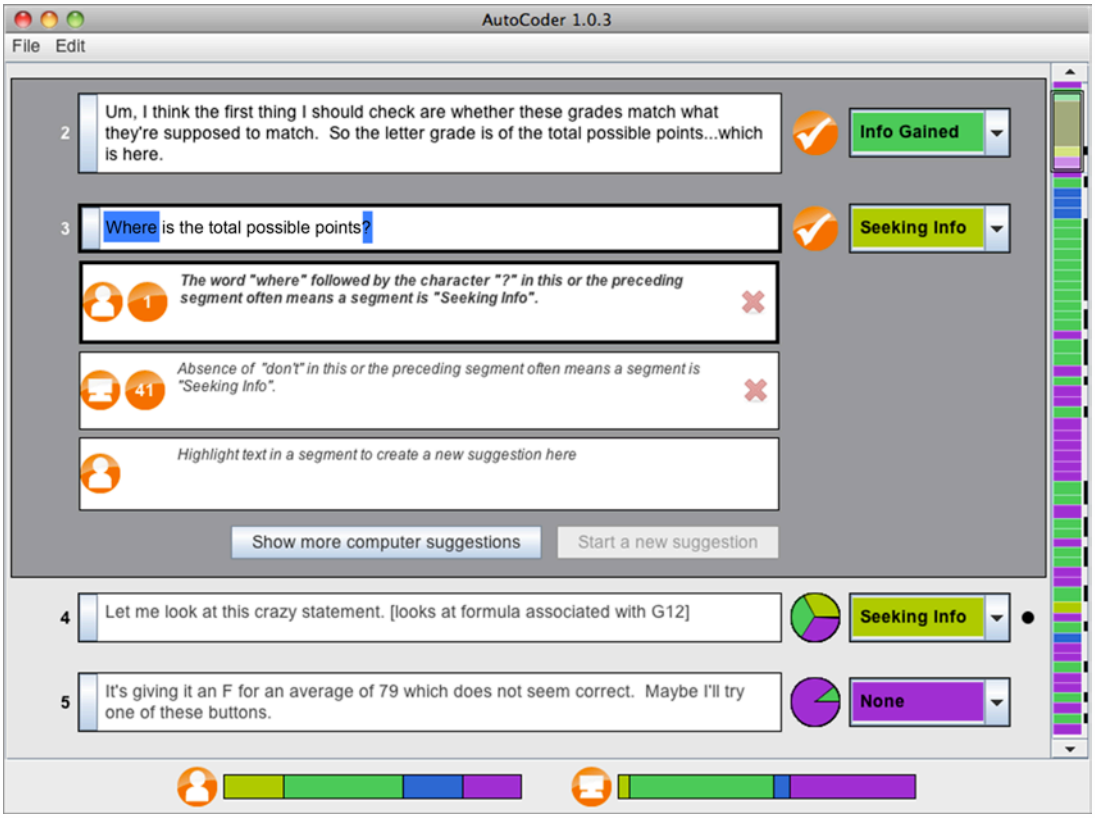

Figure 5: The high-fidelity prototype from Study #4. Participants were able to explain their reasoning by adding or removing features from the agent, and the agent explained its reasoning by describing its feature set, its confidence in each prediction, and the current class label balance.

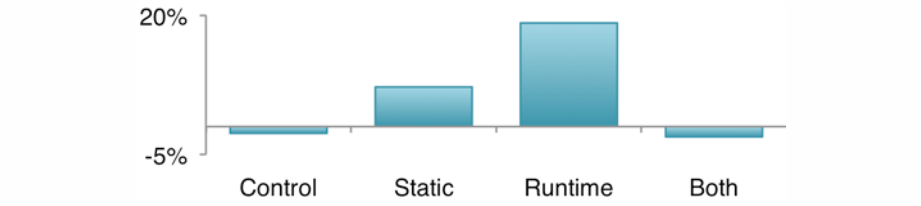

Figure 6: During Study #4, participants exposed to "runtime" details of the agent's reasoning significantly increased the agent's accuracy over the control group (Wilcoxon rank-sum test, Z=- 2.53, p<.02).

4.3 Assessing an Agent's Reasoning

How can end users efficiently tell an agent which of its decisions *need* to be correct, and how can agents help end users understand the reliability of its reasoning for these decisions? I plan to investigate methods allowing users to either implicitly or explicitly communicate this information to their intelligent agents, as well as what the agent can do with these details to help end users form an appropriate level of trust in its work.

Unlike many domains, even the very *act* of testing an intelligent agent provides additional data that the underlying algorithm can use to improve its reasoning (e.g., a positive or negative training instance). Thus, when end users assess intelligent agents, the agents are gaining a small amount of additional training data. Current work on active learning [Settles 2009] seeks to optimize the value of small amounts of training data, but in a manner which views end users as tireless oracles—the main goal of such techniques is to enable the *intelligent agent* to shore up "fuzzy" decision boundaries by eliciting more details from the end user. I hypothesize that the converse is attainable—enabling the *end users* to tell agents which of their decisions or decision boundaries need to be reliably accurate, so that any information the agents ask users to provide is directly related to what the user most needs the agent to get right.

Research Objective 3: Understanding how to effectively support end-user assessment of intelligent agents, and leveraging these assessments to support personalization of intelligent agents.

RQ3.1 (evaluation abstraction): At what level of abstraction do users prefer to assess an agent's reasoning (e.g., decision boundaries, individual predictions, etc.)?

RQ3.2 (evaluation prioritization): Can end users tell intelligent agents which predictions matter to them in a way that allows the agent to apply active learning techniques to focus on particular classes or decision boundaries?

I have begun exploring RQ3 with a statistical study exploring how end users respond to a systematic approach for assessing an agent's decisions, and testing a technique for leveraging user assessments to test similar agent predictions [Kulesza et al. 2011a]. To more formally explore appropriate levels of abstraction, I plan to conduct an additional laboratory study (Study #5) using multiple low-fidelity prototypes to present an agent's reasoning at different levels of abstraction. Participants will be asked a series of comprehension questions about each prototype to establish how much information end users gain from each level of abstraction, and will then be asked to indicate to the "computer" (by marking on the low-fidelity prototype) which of the

computer's decisions matter to them. Finally, participants will rank the prototypes based upon their own preference. Together, these results will answer RQ 3.1.

The data collected for this study will also be used to explore RQ 3.2. Using offline experiments, I will evaluate how well existing active learning techniques can leverage user-provided details about which of the computer's decisions matter to them, attempting to identify where additional user corrections or feedback would most help the intelligent agent.

4.4 Effectiveness

To conclude my dissertation research, I plan to use the results of Research Objectives 1-3 to inform an approach for end users to assess and debug the reasoning of intelligent agents. Research Objective 1 will contribute user-preferred input affordances and a "natural" vocabulary for interaction. Research Objective 2 will inform the design of explanations of the agent's reasoning, and provide suggestions for tailoring explanation content based on an application's expected audience and their goals. Research Objective 3 will also inform how the agent's reasoning is explained by suggesting appropriate levels of abstraction, as well as determine whether end user assessment can be leveraged by active learning techniques. I hypothesize that combining these elements together will help end users to more accurately assess and more efficiently personalize their intelligent agents than existing techniques.

My final study (Study #6) will involve two phases. First, the findings from my three research objectives will inform the development of a progression of low-fidelity prototypes designed to help end users assess and personalize a music-recommending intelligent agent. This prototype will be presented to users as part of a Wizard-of-Oz study, with the goal of evaluating the usability of the prototype's design. After making adjustments to fix any flaws discovered during this first phase, I will implement a high-fidelity version of this prototype. A summative user study will then be used to evaluate the overall success of (and individual strengths and weaknesses inherent in) the approach. In essence, I will be attempting to answer these final research questions:

Research Objective 4: Develop an explanation-centric approach for end-user assessment and personalization of intelligent agents.

RQ4.1 (effectiveness): Can end users reliably assess the accuracy of an intelligent agent via an explanation-centric approach?

RQ4.2 (effectiveness): Can end users reliably improve the accuracy of an intelligent agent via an explanation-centric approach?

5 TIMELINE

Table 1 outlines my research timeline. By using low-fidelity prototypes, I expect to run two formative studies in the first half of 2012, and a third before the year ends. This will then allow me to focus on a user-centric design involving iterative work with a low-fidelity prototype, culminating in a high-fidelity prototype for Study #6; I expect to complete this prototype during the first half of 2013, and plan to complete my dissertation later that year.

Table 1: Outline of when I expect to complete the six studies planned for this thesis and the conferences where I plan to publish the results.

Study	RQs	Completed	1st Half 2012	2nd Half 2012	1st Half 2013
Formative Study #1 (*hi-fi prototype*)	1.1	CHI 2012			
Formative Study #2 (*lo-fi prototype*)	1.2, 2.3		CHI 2013		
Formative Study #3 (*lo-fi prototype*)	2.1, 2.2		IUI 2013		
Formative Study #4 (*lo- and hi-fi prototypes*)	2.3	VL/HCC 2010			
Formative Study #5 (*lo-fi prototype*)	3.1, 3.2			IUI 2014	
Summative Study #6 (*lo- and hi-fi prototypes*)	4.1, 4.2				CHI 2014

6 CONCLUSION

Achieving the goals of this thesis will require investigations into several different core concepts—mental models, trust, and end-user programming. I have conducted preliminary work in each of these areas, often in the context of a laboratory study involving fully functional intelligent agents. While this work has given me a broad understanding of the problems faced by end users interacting with intelligent systems, as my research plan outlines, I will employ more formative studies for the remaining thesis research. This formative work will provide the necessary background to implement an explanation-centric approach to assessing and personalizing intelligent agents. I plan to evaluate this approach via a statistical study (Study #6) that explores how well participants are able to (1) assess and (2) personalize their agents, as compared to current techniques that provide no systematic assessment support and limit support for personalization to the labeling of training examples. Given the depth of the background knowledge informing this approach, even if it proves unsuccessful, I expect to gather enough information to identify and explain any remaining barriers. Thus, the outcome from Study #6 will not be a Boolean yes or no, but rather an analysis of which aspects of the explanation-centric approach worked as intended, which did not, and the factors which may have contributed to either result.

REFERENCES

Abraham, R., & Erwig, M. (2006). AutoTest: A Tool for Automatic Test Case Generation in Spreadsheets. In *Proc. VL/HCC* (pp. 43–50).

Amershi, S., Fogarty, J., Kapoor, A., & Tan, D. (2009). Overview based example selection in end user interactive concept learning. In *Proc. UIST* (pp. 247–256).

Amershi, S., Fogarty, J., Kapoor, A., & Tan, D. (2010). Examining multiple potential models in end-user interactive concept learning. In *Proc. CHI* (pp. 1357–1360).

Ashcraft, M. H. (1994). *Human memory and cognition*. HarperCollins College Div.

Becker, B., Kohavi, R., & Sommerfield, D. (2002). Visualizing the simple Bayesian classifier. In U. Fayyad, A. Wierse, & G. Grinstein (Eds.), *Information visualization in data mining and knowledge discovery*.

Billsus, D., & Hilbert, D. (2005). Improving proactive information systems. In *Proc. IUI* (pp. 159–166).

Blackwell, A., & Green, T. (1999). Investment of attention as an analytic approach to cognitive dimensions. *Collected Papers of the 11th Annual Workshop of the Psychology of Programming Interest Group* (pp. 24–35).

Bogart, C., Burnett, M., Douglass, S., Adams, H., and White, R. (2012). Designing a debugging language for cognitive modeling: an initial case study in Natural Programming Plus. In *Proc. CHI* (conditionally accepted).

Bruner, J. S., & Tagiuri, R. (1954). The Perception of People. In G. Lindzey (Ed.), *Handbook of Social Psychology*. Cambridge, MA: Addison-Wesley.

Carley, K., & Palmquist, M. (1992). Extracting, representing, and analyzing mental models. *Social Forces*, *70*(3), 601–636. Oxford University Press.

Chen, J., & Weld, D. (2008). Recovering from errors during programming by demonstration. In *Proc. IUI.*

Doyle, J., & Radzicki, M. (2008). Measuring change in mental models of complex dynamic systems. In Qudrat-Ullah, H., Spector, J.M., and Davidsen, P.I. (Eds.), *Complex Decision Making*. Springer Berlin (pp. 269–294).

Dzindolet, M. T., Peterson, S. A., Pomranky, R. A., & Pierce, L. G. (2003). The role of trust in automation reliance. *International Journal of Human-Computer Studies*, *58*(6), 697–718.

Fisher, M., II, Rothermel, G., Brown, D., Cao, M., Cook, C., & Burnett, M. (2006). Integrating automated test generation into the WYSIWYT spreadsheet testing methodology. *Transactions on Software Engineering and Methodology, 15*(2).

Glass, A., McGuinness, D., & Wolverton, M. (2008). Toward establishing trust in adaptive agents. In *Proc. IUI* (pp. 227–236).

Hastie, R. (1984). Causes and effects of causal attribution. *Journal of Personality and Social Psychology*, *46*(1), 44–56.

Hastie, T., Tibshirani, R., & H Friedman, J. (2009). The elements of statistical learning: data mining, inference, and prediction.

Herlocker, J., & Konstan, J. (2000). Explaining collaborative filtering recommendations. In *Proc. CSCW*.

Johnson-Laird, P. N. (1983). *Mental Models: Towards a Cognitive Science of Language, Inference, and Consciousness*. Cambridge University Press.

Jonassen, D. H., & Henning, P. (1996). *Mental models: knowledge in the head and knowledge in the world* (pp. 433–438). International Society of the Learning Sciences.

Kapoor, A., Lee, B., Tan, D., & Horvitz, E. (2010). Interactive optimization for steering machine classification. In *Proc. CHI* (pp. 1343–1352).

Kempton, W. (1986). Two theories of home heat control. *Cognitive science*, *10*(1), 75–90.

Kissinger, C., Burnett, M., Stumpf, S., Subrahmaniyan, N., Beckwith, L., Yang, S., & Rosson, M.B. (2006). Supporting end-user debugging: what do users want to know? In *Proc. AVI* (pp. 135–142).

Ko, A. (2006). Debugging by asking questions about program output. In *Proc. ICSE* (pp. 989–992).

Ko, A. & Myers, B. (2008). Debugging reinvented: asking and answering why and why not questions about program behavior. In *Proc. ICSE* (pp. 301–310).

Ko, A., Myers, B., & Aung, H. (2004). Six learning barriers in end-user programming systems. In *Proc. VL/HCC* (pp.199–206).

Kulesza, T., Burnett, M., Stumpf, S., Wong, W., Das, S., Groce, A., Shinsel, A., & Bice, F. (2011a). Where are my intelligent assistant's mistakes? A systematic testing approach. In

Proc. IS-EUD (pp. 171–186).
Kulesza, T., Stumpf, S., Burnett, M., Wong, W.-K., Riche, Y., Moore, T., & Oberst, I. (2010). Explanatory debugging: supporting end-user debugging of machine-learned programs. In *Proc. VL/HCC* (pp. 41–48).
Kulesza, T., Stumpf, S., Wong, W.-K., Burnett, M., Perona, S., Ko, A., & Obsert, I. (2011b). Why-oriented end-user debugging of naive bayes text classification. *ACM Transactions on Interactive Intelligent Systems*, *1*(1).
Lacave, C., and Díez, F. (2003). A review of explanation methods for Bayesian networks. *The Knowledge Engineering Review*, *17*(2), 107–127.
Lieberman, H. (2001). *Your wish is my command*: *programming by example*. Morgan Kaufmann.
Lim, B., & Dey, A. (2009). Assessing demand for intelligibility in context-aware applications. In *Proc. Ubicomp* (pp. 195–204).
Lim, B., & Dey, A. (2010). Toolkit to support intelligibility in context-aware applications. In *Proc. Ubicomp* (pp. 13–22).
Lim, B., Dey, A., & Avrahami, D. (2009). Why and why not explanations improve the intelligibility of context-aware intelligent systems. In *Proc. CHI* (pp. 2119–2128).
Little, G., Lau, T., Cypher, A., Lin, J., & Haber, E. (2007). Koala: capture, share, automate, personalize business processes on the web. In *Proc. CHI* (pp. 943–946).
McDaniel, R., & Myers, B. (1997). Gamut: demonstrating whole applications. In *Proc. UIST* (pp. 81–82).
McNee, S., Lam, S., & Guetzlaff, C. (2003). Confidence displays and training in recommender systems. In *Proc. INTERACT.*
Miller, R., & Myers, B. (2001). Outlier finding: focusing user attention on possible errors. In *Proc. UIST* (pp. 81–90).
Myers, B., Weitzman, D., Ko, A., & Chau, D. (2006). Answering why and why not questions in user interfaces. In *Proc. CHI* (pp. 397–406).
Norman, D. (1983). Some observations on mental models. In D. Gentner & A. Stevens (Eds.), *Mental Models*. Psychology Press.
Otter, M. (2000). Lost in hyperspace: metrics and mental models. *Interacting with Computers*, *13*(1), 1–40.
Pane, J., Myers, B., & Miller, L. (2002). Using HCI techniques to design a more usable programming system. In *Proc. Human Centric Computing Languages and Environments* (pp. 198–206).
Poulin, B., Eisner, R., Szafron, D., Lu, P., Greiner, R., Wishart, D., Fyshe, A., et al. (2006). Visual explanation of evidence in additive classifiers. In *Proc. AAAI* (pp. 1822–1829).
Raz, O., Koopman, P., & Shaw, M. (2002). Semantic anomaly detection in online data sources. In *Proc. ICSE* (pp. 302–312).
Rosson, M., & Carrol, J. (1990). Smalltalk scaffolding: a case study of minimalist instruction. In *Proc. CHI* (pp. 423–429).
Rothermel, G., Burnett, M., Li, L., Dupuis, C., & Sheretov, A. (2001). A methodology for testing spreadsheets. *Transactions on Software Engineering and Methodology 10*(1), 110–147.
Scaffidi, C. (2007). Unsupervised inference of data formats in human-readable notation. In *Proc. of 9th International Conference on Enterprise Integration Systems* (pp. 236–241).
Scaffidi, C., Shaw, M., & Myers, B. (2005). Estimating the numbers of end users and end user programmers. In *Proc. VL/HCC* (pp. 207–214).
Settles, B. (2009). *Active learning literature survey*. University of Wisconsin-Madison.
Sharp, H., Rogers, Y., & Preece, J. (2007). *Interaction design: beyond human-computer interaction*. John Wiley & Sons Inc.
Sinha, R. & Swearingen, K. (2002). The role of transparency in recommender systems. In *Proc. CHI* (pp. 830–831).
Stumpf, S., Rajaram, V., Li, L., Burnett, M., Dietterich, T., Sullivan, E., Drummond, R., &

Herlocker, J. (2007). Toward harnessing user feedback for machine learning. In *Proc. IUI* (pp. 82–91).
Stumpf, S., Rajaram, V., Li, L., Wong, W., Burnett, M., Dietterich, T., Sullivan, E., & Herlocker, J. (2009). Interacting meaningfully with machine learning systems: Three experiments. *International Journal of Human-Computer Studies*, *67*(8), 639–662.
Stumpf, S., Sullivan, E., Fitzhenry, E., Oberst, I., Wong, W.-K., & Burnett, M. (2008). Integrating rich user feedback into intelligent user interfaces. In *Proc. IUI* (pp. 50–59).
Talbot, J., Lee, B., Kapoor, A., & Tan, D. (2009). EnsembleMatrix: Interactive visualization to support machine learning with multiple classifiers. In *Proc. CHI* (pp. 1283–1292).
Tintarev, N., & Masthoff, J. (2007). Effective explanations of recommendations: user-centered design. In *Proc. RecSys* (pp. 153–156).
Tullio, J., Dey, A., Chalecki, J., & Fogarty, J. (2007). How it works: a field study of non-technical users interacting with an intelligent system. In *Proc. CHI* (pp. 31–40).
Vander Zanden, B. & Myers, B. (1995). Demonstrational and constraint-based techniques for pictorially specifying application objects and behaviors. *ACM Transactions on Computer-Human Interaction* 2*(4)*, 308–356.
Vig, J., Sen, S., & Riedl, J. (2009). Tagsplanations: Explaining recommendations using tags. In *Proc. IUI* (pp. 47–56).
Wagner, E., & Lieberman, H. (2004). Supporting user hypotheses in problem diagnosis. In *Proc. IUI* (pp. 30–37).
Wilson, A., Burnett, M., Beckwith, L., Granatir, O., Casburn, L., Cook, C., Durham, M., & Rothermel, G. (2003). Harnessing curiosity to increase correctness in end-user programming. In *Proc. CHI* (pp. 305–312).
Wong, W.-K., Oberst, I., Das, S., Moore, T., Stumpf, S., McIntosh, K., & Burnett, M. (2011). End-user feature labeling: A locally-weighted regression approach. In *Proc. IUI* (pp. 115–124).