Asking Questions and Developing Trust

Stephanie Rosenthal and **Manuela Veloso** and **Anind Dey**
Carnegie Mellon University
Pittsburgh, PA
{srosenth, veloso, anind}@cs.cmu.edu

Abstract

In a number of domains, researchers instrument an interface or the environment with software and hardware sensors to collect observational data, but it is often quite hard to label that data accurately. We are interested in how an agent can ask many humans at once for help to label its data and we present two studies towards this goal. First, we investigate how a computer can automatically elicit labels from users as they interact with different technologies. Then, we present a study comparing different algorithms for how an agent decides which users to trust when a lot of people answer the agent's questions and the answers are conflicting. We discuss the implications of the results in each of these studies and present some ideas for future work towards agents asking humans questions.

Introduction

Suppose that a robot is navigating an environment to perform a task. As it is performing the task, it comes across an object it has not seen before or cannot recognize. Although the object may not be directly useful to the robot's task, if the robot learns what it is, it may help the robot achieve goals faster in the future. The engineer in charge of the robot's task is currently busy and cannot provide help at this time. The robot takes a picture of the object and can save the picture to ask the engineer later or can email the picture now to many people in the building to ask what the object is. Whenever the robot asks, it may want to include context about what was happening when the picture was taken like which room the robot is in and which direction it is facing, what time of day it is, and who else is around. The robot may also include possibilities for what it thinks the object is. All of this information may help both the engineer and the email list reply to the robot more accurately so the robot does not learn the wrong thing.

Additionally, if the robot decides to email the entire building, it may receive several replies from different people in the building. Their answers may be different and they may have different experience with objects in the room. The robot uses previous knowledge of who has answered correctly in the past and which current answer is repeated most often to judge which answer is correct. If the engineer corrects the answer later, the robot can update its knowledge about the people who answered to include whether they were correct for the new object. In this way, the robot can learn new information by asking for help and learn which responders to trust by analyzing their accuracy over time.

We believe it is inevitable that agents will need to interact with humans as they complete their tasks as the agents may encounter new scenarios they were programmed for originally. As part of that interaction, humans will need to give agents help when needed. There has been extensive research in the types of help humans can give robots (*e.g.,* (Rybski *et al.* 2007) and (Saunders, Nehaniv, & Dautenhahn 2006)). In collaborative tasks specifically, robots may need to ask for advice or assistance from humans. The *collaborative control* model (Fong 2001) assumes humans and robots to be *partners* or *collaborators* in tasks and that they will help each other to accurately and effectively achieve their goals. We're interested in how an agent, robot or otherwise, can ask humans questions to label its data, as is often done with experience sampling (Larson & Csikszentmihalyi 1983). More specifically, we want to maximize the proportion of correct labels users provide so that data is as *accurate* as possible by either asking few people very good questions or many people to learn who to trust.

We present two studies conducted to better understand how to help agents ask humans to help in labeling data accurately. The focus of our first study is to understand how varying an agent's questions affects the accuracy of users' responses. Additionally, we investigate how the optimal combination of different question features will change based on the domain they are provided in. We vary the agent's questions across five dimensions taken from the HCI literature, including dimensions for usability and contextual information. We present guidelines for wording questions based on the results of the study.

In our second study, we assume that it is cheap to ask many users the questions developed from the first study (*i.e.,* asking a crowd, asking people online) and evaluate how agents should calculate trust in users when they receive multiple conflicting responses to their questions. Specifically, because agents' questions may fall into different categories (*e.g.,* easy *vs.* hard questions), an agent may trust different users for different types of questions (*e.g.,* only experts for hard questions). We use recommender systems as an example of agents receiving data that falls into different (product) categories. We compare the resulting agent actions (*e.g.,* recommendations) and user trust distributions between algorithms that include data from all product categories (*combined* systems) and algorithms that include only data from one product category (*doman-specific* systems). We discuss the results and implications of the work for agents that may have different users respond to different categories of questions. In the next section, we discuss how to determine which data that agents should ask about and how they should ask about it.

Related Work

Researchers often instrument an interface, robot, or the environment with sensors to collect observational data but it can be difficult or costly for a human expert to label that data accurately. Machine learning algorithms such as semi-supervised learning and unsupervised learning try to infer the labels of the unlabeled data points without any human intervention (Mitchell 1997). However, it is not always possible for these algorithms to classify all data accurately. A learner needs some other way of asking users for accurate labels. Active learning algorithms try to infer the labels for the unlabeled points but can request labels for some data, if necessary (Cohn, Atlas, & Ladner 1994). We are interested in how different applications can elicit accurate labels for data from users of those applications.

Requesting Labels From Users

Because people may understand one part of the state-space more another, some data may be easier or less costly for people to label than other data points. *Cost-sensitive active learning* algorithms maximize the expected accuracy of the resulting classifier while minimizing the cost of requesting labels (*e.g.,* (Margineantu 2005)). It may be more cost-effective to ask users to label many "easy" points that each may be less informative for the classifier instead of a few more difficult-to-label points that are very informative, because the combination of the many points results in a more accurate classifier. Similarly, when multiple users available to ask, it may be easier or less costly to ask some users rather than others. It may be more expensive to ask experts to label data than novices. However, the novices may be more likely to give inaccurate labels. *Proactive learning* algorithms maximize the expected accuracy of the resulting classifier while minimizing the cost of requesting labels from users (Donmez & Carbonell 2008). Even if a novice user is very inexpensive to query, if they provide very inaccurate labels, it may not be cost effective to ask that user a question.

However, both of these algorithms assume that the cost distribution of data or users is known ahead of time. Additionally, proactive learning assumes some prior knowledge about the expertise of users ahead of time. While this prior knowledge about expertise may be available in some applications, it may be cheaper to ask many new or unknown users a question instead of waiting for a few experts. For example, posting a question to an online forum is a cheap way to elicit a fast (and reasonably accurate) response from many unknown users instead of waiting for a known user to respond. In these situations, we will show it is possible to develop an online trust model with the responses of the users using experts algorithms.

Eliciting Accurate Labels from All Users

If an agent does not know who to trust before it asks a question, it must ask questions that are understandable by all users. Recently, researchers demonstrated that a person could accurately help an email sorter to sort emails when the computer included contextual information about the email scenario as it prompted the user for a classification of that email (Stumpf *et al.* 2007). In this study, subjects were provided different *levels of contextual information*: either keywords from the email, rules, or emails the computer identified as being similar to the one being looked at, and were then asked to sort that email into a folder. The study found that although subjects were not always able to correctly label the emails, they were very accurate. However, the study did not relate the *amount of contextual information* they provided nor did they discuss how varying the amount of contextual information might affect the accuracy of the subjects feedback. Additionally, subjects in the study were able to provide supplemental feedback about why they gave emails a particular label in order to teach the learner which features were most important in making the classification. However, the study did not relate the contextual information the agent provided to the amount or quality of *supplemental information* the subjects supplied.

A follow-up study found that the additional information subjects provided was largely driven by their understanding of how the system worked and how the system explained contextual information to the user (rule-based, keyword-based, *etc.*). This finding is supported by work in retrieval feedback (Salton & Buckley 1990) in information retrieval applications that identifies the differences between the sensor-level context that computers use and collect and the high level activities that users think about. If an agent provides contextual information to elicit information at the wrong level, a person may not be able to understand the data point or situation they are asked to classify. For example, in image retrieval, a computer may use color and pixel colors to determine a label for the image while people may look at high level shapes and textures (Rui *et al.* 1998). The previous email study found that subjects were able to provide more and better supplemental information when given the rule-based context instead of keyword-based because subjects reported they understood it better. Because it is often difficult for a computer to communicate at a high level with a user or for a person to understand the pixel level representation of a picture, it is necessary to balance the improvement in accuracy and increase in supplemental information that comes with providing high-level context with the costs of calculating that context. This balance depends on how the users of each particular application understand the *low-level context* in the domain, the task, and the content of the questions the agent asks.

Usable Elicitation

In addition to providing context while asking questions, other work has focused on making the classification task easier for people. Some user interfaces provide the user with a prediction or *suggestion for the answer* to a classification task, to cut down on the amount of work the user has to do: confirm an answer vs. generate an answer (*e.g.,* (Stumpf *et al.* 2005)). An interface may automatically fill in fields in a form or provide a prediction for which folder to sort a piece of email into (*e.g.,* (Faulring *et al.* 2008)).

Studies on context-aware, expert, and recommender systems all show that providing users with the level of *uncertainty* in the systems' predictions improves their overall usability (e.g., (Banbury *et al.* 1998),(Mcnee *et al.* 2003),(Antifakos, Schwaninger, & Schiele 2004)) In one task where users had to remember a set of numbers, an imperfect memory aid that displayed uncertainty information was shown to increase the percentage of correct numbers the users recalled and decrease the percentage of incorrect numbers they reported, when compared to the same display without the uncertainty information (Antifakos, Schwaninger, & Schiele 2004). Regardless of the exact confidence values, users showed improved performance. However, it is not clear whether usability affects the accuracy of the labels obtained when asking questions.

We will first describe a study in which we evaluate how agents can elicit accurate answers from all types of users by varying the wording of its questions. Although, we may not expect to get the same accuracy from novices as from experts, we aim to maximize accuracy over all users. Then, we assume that asking questions to many people is cheap (*i.e.,* in a crowd or over the internet) and that an agent can expect responses from possibly unknown novices and experts. We discuss how an agent can use the answers it receives to determine which users to trust in future questions. Finally, we discuss possible future work in asking questions and subsequently learning from users' responses.

Asking Questions

We are first interested in how an agent can elicit greater proportions of correct answers and generalizable information from people by varying the wording of its questions. Specifically, we examine the impact of varying the content of the questions in a knowledge elicitation task along five dimensions previously described in the related work, namely explaining uncertainty, amount of contextual information, low-level vs. high-level context, suggesting an answer, and requesting supplemental information, to understand how each impacts how people label data. We measure the proportion of correct answers people provide as well as the quality of supplemental information provided and users' opinions of the questions. Additionally, because the wording of questions depends on the type of data that needs a label and the presentation of the agent's question depends on the interface, we present this approach in three different domains: a desktop application on a large screen computer, activity recognition on a mobile device, and learning by demonstration on a robot to understand what combinations of dimensions will be more effective in the three domains and why. We provide results for the impact of the dimensions in both domains, validate those results against community advice from each domain about how to ask questions in each, and finally discuss the differences in the results between the domains.

Method

Subjects in our studies were told that they were testing new technologies (an email sorter, a physical activity coach, and a robot) we had developed that learn by asking questions. Because people are typically good at classifying data from each of these domains, the application would ask them for help if it could not correctly label the data itself.

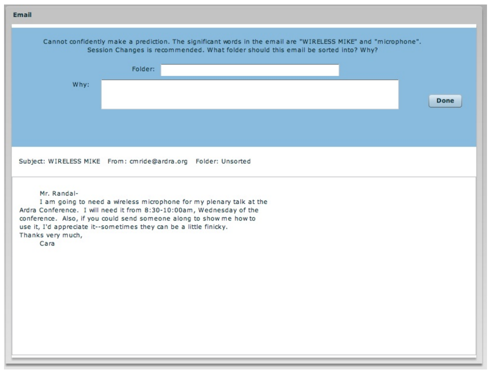
The participants were given a primary task to perform that was indirectly related to the learning application, and were told they had a limited time to complete it. Participants worked on their own task while the application observed them and collected data. They were informed that they could help answer the application's questions when it interrupted their task if they had time and that the application would continue to try and learn whether or not they answered questions or skipped them. They were also told that they would be given a second similar performance task that the application could help them complete more quickly if they helped it learn during this first task. However, participants were also reminded that answering questions was not their primary task and that doing so may slow them down in completing that task.

All three study applications are modeled based on the idea that there is some underlying active learner that identifies data points to classify and uses an agent to ask users for the labels it needs. The data that the agent asks users about in the tasks all have single labels that are easily confirmable, in order to make the study feasible. The agent must interrupt the user's primary task in order to ask the participant a question but the answer to the question will ultimately improve the learning application's performance. We model the tradeoff between future incentive of increased application accuracy and present delays of answering questions to understand of how people answer questions under pressure (*i.e.,* ignoring the application questions altogether or rushing to give fast, instead of well thought out, responses).

To understand how an agent can ask questions in different domains to optimize the correctness of the responses it receives and the quality of supplemental information it requests, we vary the wording of the questions it asks about based on the five dimensions presented above and measure both the correctness and quality of the responses and the user's opinions of the questions. Subjects were given 12 minutes to complete their primary task. After either completing the task or after time had expired, participants were given a survey about their experiences with the questions. After completing the survey, they were told there was not enough time to conduct the second "performance" task and were dismissed after payment.

Tasks

We will now describe the three tasks that we had users perform: email sorting, physical activity recognition, and teach-



Email

Cannot confidently make a prediction. The significant words in the email are "WIRELESS MIKE" and "microphone". Session Changes is recommended. What folder should this email be sorted into? Why?

Folder:

Why:

Subject: WIRELESS MIKE From: cmride@ardra.org Folder: Unsorted

Mr. Randal-
I am going to need a wireless microphone for my plenary talk at the Ardra Conference. I will need it from 8:30-10:00am, Wednesday of the conference. Also, if you could send someone along to show me how to use it, I'd appreciate it--sometimes they can be a little finicky.
Thanks very much,
Cara

Figure 1: The email agent displayed a question when it could not determine which folder the email belonged.

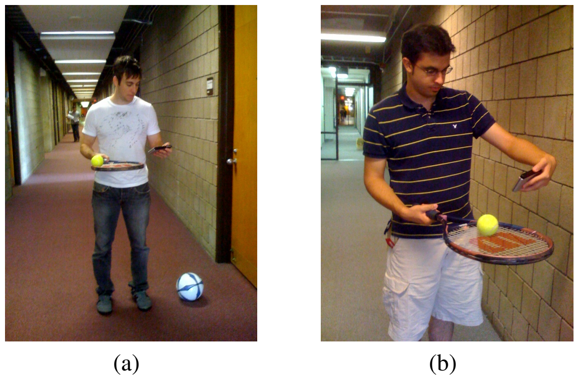


Figure 2: Participants were interrupted to ask which activity they were performing.

ing a robot to recognize block shapes. The agents and active learning mechanisms in all three tasks were Wizard-of-Oz'ed to control which data points the users were asked about and when they were asked about them.

Email Sorting The participants' primary email task was to go through some provided email about an uncoming academic conference, and consolidate all the changes that needed to be made to the conference schedule and website. Subjects were given a Microsoft Excel spreadsheet of information about conference speakers, sessions, and talks and asked to make changes to the spreadsheet based on changes conference participants had emailed about (Steinfeld *et al.* 2006). They were also given an email application filled with conference emails and were told that the learner had already sorted most emails into folders based on the type of changes that needed to be made to the Excel spreadsheet. Subjects were told that the application classifies each email in the inbox into a folder and if it cannot do so, it sorts the email into the Unsorted folder and will ask them to classify it into the appropriate folder (*i.e.,* provide the appropriate label). Participants only had to make changes to the spreadsheet based on emails from two folders plus the "Unsorted" emails that could be sorted into one of those two folders. The application would pop up a question when the subjects clicked on the "Unsorted" emails (See Figure 1)

Activity Recognition For the second part of the experiment, the subjects' primary task was to perform physical activities from a list provided. They were told they were testing a new physical activity coach on a handheld device that could detect the different activities they performed. They were given a list of 12 activities and the materials to perform them and told they had 12 minutes to complete as many as possible. Activities included walking around the room, jumping up and down, putting with a golf club, and using a hula-hoop. If the activity recognizer did not understand what the participant was doing, the participants were told it would interrupt the activity and display the questions on the mobile device. More concretely, the agent assigns a label or exercise name to sensor data as a user is performing an exercise, and if it is uncertain about the exercise it interrupts the user. An application like this one may record users' activities for their doctors to analyze their physical activity levels, and thus users have an interest in answering its questions to ensure it correctly identifies when they are performing a physical activity. Subjects were told to respond to the message on the device to tell the coach what activity they were performing (See Figure 2).

Teaching a Robot For the third part of the experiment, the participants' primary task is building structures out of blocks in a timed setting. They sat in front of the robot and were given a set of 50 wooden blocks, containing 5 different colors and 6 different block shapes (Figure 4(a)). The subjects were given 4 pictures of block structures, each composed of 20-35 blocks, to build in 12 minutes (Figure 4(c)). The subject was told the robot was performing a block shape recognition task. Humans are better at recognizing objects and robots can leverage this by interrupting the human to ask questions about those objects. As subjects built the structures, the robot interrupted with questions claiming it could not determine the shape. The robot asked about 2 blocks for each structure the participants built, for a total of 8 questions, and the participants were to respond verbally to the

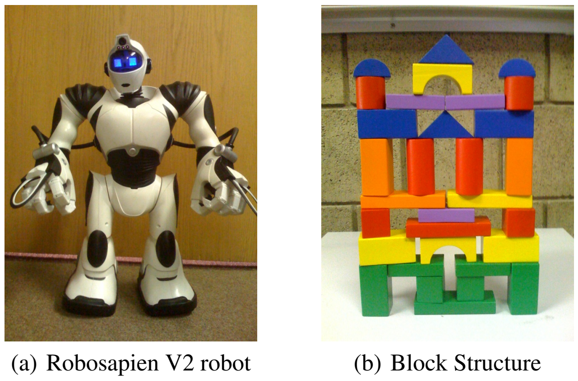


Figure 3: The Robosapien V2 robot watches participants and asks questions about the blocks it cannot recognize.

robot with their answers. If participants had time while performing their primary task, they could help teach a robot to recognize block shapes by answering its questions.

Measures

We collected the subjects' responses to each question for analysis. Because an agent would benefit more from correct answers to questions rather than incorrect ones, we assessed the user responses to the questions primarily based on correctness, but also on the quality of supplemental information when available. We also gave subjects surveys about their opinions of the applications asking questions including whether they found them annoying.

Correctness Users' responses were classified as correct answers if their last answer (some users changed their minds) was correct and incorrect otherwise. For example, if a subject disagreed with the suggestion, but gave an equally correct reference, it was classified as correct.

Qualitative After completing the task, participants were given questionnaires on their experiences with each technology. They were asked whether they thought the applications questions were annoying and whether they found each of the five dimensions particularly useful. Answers were coded as either "Yes" or "No" to each of the six questions. Additionally, participants were asked whether it was easy or hard to answer the questions on a Likert scale from 1 (very easy) to 5 (very hard).

Results and Discussion

We evaluated both our quantitative and qualitative results for each study. Chi-Square tests were used to analyze the significance of the categorical response (correctness) against the categorical independent variables (our five dimensions). T-tests and One-way ANOVAs were used to analyze the significance of the secondary continuous response (quality of supplemental information) against the independent variables (our five dimensions). We find, for example, that participants on the email task were more accurate when sufficient context was provided compared to extra and no context (Figure 4(a)). Participants also were more accurate when they were provided suggestions in the email domain (Figure 4(b)). Participants provided less informative supplemental information to the agents when the agents provided them extra context in the question (Figure 4(c)). Based on the set of results for each domain, we define a set of guidelines for each domain (presented below) that an agent should use when planning the wording of questions to present to users.

1. Email Sorting - *explain uncertainty, provide sufficient low-level context, suggest an answer, and request supplemental information*
2. Activity Recognizer - *do not explain uncertainty, provide sufficient low-level context, provide suggestions, and request supplemental information*
3. Teaching a Robot - *explain uncertainty, give extra contextual information, give a suggestion, and ask users for additional information*

Note that in the robot task, we did not vary the questions along the high vs. low context dimension because the robot did not have enough sensors to vary the context in that way. We validated those guidelines against advice we received from domain experts on how to present and request information from people specifically for each domain. Through our evaluation, we found that the agents that use our guidelines elicited more correct answers and/or better quality supplemental information and/or higher usability ratings than the community advice. We now discuss the agent assumptions we made for each domain and the implications of the similarities and differences in the guidelines for the domains.

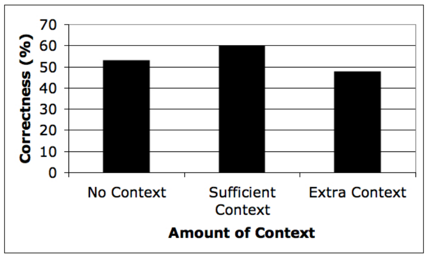
Domain Guidelines Now that we have identified the most appropriate way to phrase a question to improve labeling, we will now assume that, we hypothesized that the agents' questions would vary widely between domains. However, we found that our results led us to nearly the same guidelines for the email and mobile device domains and very different guidelines for the robot domain. First, although there were differences in the email and mobile domains, the subjects' prior knowledge about the tasks was a more important influence on the outcome. This result implies that it is not necessarily the domain or the task that drives differences in guidelines, and that the nearly identical email/mobile guidelines are applicable in a more general set of tasks, namely those where users have domain knowledge that they can apply in helping the knowledge elicitor.

However, we find that the robot guidelines are very different from the email and mobile guidelines. The robot's questions were given verbally about spatial locations for the user. Although subjects had prior knowledge about block shapes, they needed to perform a verbal-visual translation to answer the robot's questions which takes longer time and more concentration than the email and mobile tasks. Additionally, the ambiguity in spatial language requires additional context for the subjects to confirm they found what the robot was asking about. Because both of these classes of problems are common and we have validated them, other researchers can use these guidelines *today* to increase the accuracy of their labels without using our approach to find new guidelines and without additional validation. This saves time and effort that could be dedicated to using the accurately labeled data in new and interesting ways.

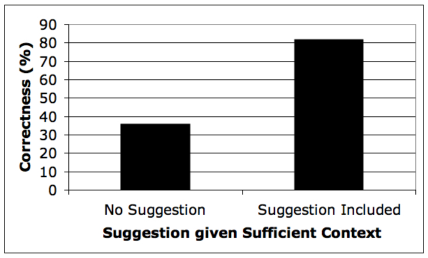
Next, we assume that an agent can use our guidelines to ask questions to a general audience and receive many responses at once. The agent must then determine which users and responses to trust to use the information effectively.

Developing Trust

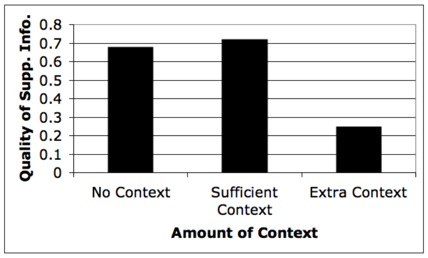
Given that it is relatively inexpensive for an agent to ask questions to a large number of people at once (*e.g.,* online forums, *etc.*) and determine different categories of data to ask about (*e.g.,* easy or hard questions), we are then interested in how an agent can determine which responses to trust when it receives many conflicting answers to its questions. Specifically, we examine how an agent's actions differ as



(a) Subjects' Correctness by Amount of Context (Email)



(b) Subjects' Correctness by Suggestion (Email)



(c) Subjects' Quality of Supplemental Info. (Activity Recognition)

Figure 4: As the amount of context and suggestions varied, the participants' accuracy changed as well as the amount of supplemental information they provided.

we vary whether the agent determines which reviewers to trust and actions to take for each separate category of situation versus one trust determination based on all categories combined. As an agent asks questions, receives answers, and takes actions using the responses (*online*), it recalculates who it trusts based on the success of the action using an *experts algorithm* (*e.g.,* (Auer *et al.* 1995) and (Littlestone & Warmuth 1994)).

For this work, we use a recommender system application where an agent's action is to predict (on a scale from 1 to 5) whether a user will like a particular product based on numeric recommendations from reviewers in the system. The recommender system is either comprised of a single category of products (one *domain specific* system for each of DVDs, Books, Houswares, etc) or comprised of all products together (one *combined* system of all recommendations in all categories). The recommender system "asks" all the reviewers for reviews of products and provides a prediction to the user based on the reviews. The agent decides which reviewers to trust for each individual user with a probability distribution over the reviewers. We define the most trustworthy reviewers (for a particular user) as those who provide reviews closest to the user's actual reviews given in hindsight after the user receives the prediction. The probabilities of the reviewers are updated using the $\ell 1$ distance from the agent's prediction to the actual score users gave the products (on the same 1 to 5 scale) (loss function).

We expect that the accuracies of the different systems are largely based on the convergence of the probability distribution to a steady state. Each domain-specific recommender takes the same amount of time to converge to find the most trustworthy reviewers as the single combined recommender, so it takes the domain-specific recommender N times as long to find the most trustworthy in *every* one of the N categories. By combining categories and assuming that the trustworthy reviewers for each category are the same, the combined system eliminates much of the convergence time of all the domain-specific systems. However, intuitively, it seems that a user might trust some reviewers about their DVD selection but may not trust them at all for books or housewares. Although it may take longer, the domain-specific recommender provides more accurate predictions when the trustworthy reviewers are different in each category. We analyze how often each recommender system is more accurate to understand whether it is more important to find the trustworthy reviewers faster or to find more accurate weights for each reviewer. We will use mean squared error to measure the accuracy of each of the two types of recommender systems (domain-specific and combined) for each user.

Method

To test whether combining domain-specific recommendation systems will decrease the accuracy compared to their separate counterparts, we built recommendation systems from three real datasets and analyzed the performance on a test set of users. We split each of the recommendation datasets into subsets by categories of products. Each subset was used for a domain- or category-specific recommendation system and the full set was used for the combined system. We selected a random subset of the users in the dataset as test users of the systems. We implemented and tested the two recommendation system agents (domain-specific and combined) for each test user with two different experts algorithms. Each agent for a new test user started with a uniform trust distribution over all current users. As the test user looked at and reviewed products, the experts algorithm agent would use the predictions from reviewers it trusts to give the user a prediction. Based on the test users' actual satisfaction with the product, the agent then updates the trust of all the reviewers by comparing the users' satisfaction to each reviewer's prediction. The agent trusts the reviewers who gave more similar recommendations as its test user. Our goal is to maximize the number of correct recommendations the agent makes to the test users by determining and using the most similar reviewers to make the prediction.

Product Recommendation Data

We built our experimental framework using data from three very popular online recommendation systems. Each recommendation system uses collaborative filtering with reviews on a scale from 1 to 5 that users could choose for various products they are familiar with. Open-ended reviews were not included in this data or for our framework.

General Products Our first recommendation dataset was initially designed to examine the impact of viral marketing on a popular shopping website (Leskovec, Adamic, & Huberman 2006). Over two years of data collection, 3,943,084 reviewers made 15,646,121 recommendations for 548,523 products. 5813 products were discontinued and were thus removed from the set. The dataset provides features per product and features per recommendation. The product features provided are ten product categories (*i.e.,* Book, DVD, Electronics). For each recommendation, the dataset provides the ID of the reviewer, the numerical recommendation as defined above, the date that the recommendation was given, the number of people who rated the review, and the number of people who rated the review as helpful. On average, reviewers made recommendations for about 100 products, with one recommending over 12000, and each product had between 3 and 2500 recommendations.

Movies The Netflix dataset contains "over 100 million movie ratings for over 480 thousand Netflix customers from over 17000 movie titles" (net 2008). We use movie genres for the categories of the domain-specific recommendation system. Genres are not included in the Netflix data, so we cross-referenced movie titles with movie genres and obtained Science Fiction, Fantasy, Special Interest, Kids/Family, Comedy, Western, Action/Adventure, Musical/Performing Arts, Documentary, Drama, and Art/Foreign. Because of the necessity to have mutually exclusive genres, only the movies with a single genre were used. This resulted in a smaller dataset of 400 movies and over 4 million ratings. 50 users were randomly chosen from the set to test.

Music The Yahoo! Research Webscope Music review dataset contains over 717 million reviews by 1.8 million Yahoo! Music reviewers for 136 thousand songs collected from 2002 to 2006 (Yahoo! Research 2008). Each reviewer was guaranteed to have rated at least 30 songs and each song was rated by at least 20 reviewers. There are 20 main genres of music in the data which were used as our categories for the domain-specific systems. For each new user, there were an average of 100,000 reviewers giving advice on all their songs.

Results

We built the recommendation systems described above and used them to make predictions about users' preferences. We calculate error as the *mean squared* distance from the recommendation system's prediction to the user's actual opinion (MSE). Our hypothesis is that the combined and domain-specific algorithms yield the different error rates for each user. Specifically, we expect that users with reviewers that are highly trusted in one category and minimally trusted in others would have higher error from the combined system which does not distinguish categories. We tested that hypothesis using an ANOVA (Analysis of Variance) on the MSEs of both systems.

We analyzed each user separately to understand the possible differences between the datasets as well as the differences between similarity algorithms and recommendation systems. We found that approximately one half of our test users received higher accuracy predictions from the domain-specific recommender system (ΔMSE = .5, std. dev = .4) ($F = 7.14, df = 1, p < 0.01$) and half received higher accuracy predictions from the combined system (ΔMSE = .25, std. dev. = .2) although this was not statistically significant. As expected, the domain-specific recommenders perform better on users with different trustworthy reviewers in different categories. Combining these different probability distributions can cause large changes to the relative trustworthiness some reviewers and result in different (and worse) predictions. However, when the trustworthy users were the same across categories, there was no reason to keep them separate and wait for each category to converge. The combined recommender system provided more accurate predictions when the most trustworthy reviewers were most trustworthy in all categories.

Because our results show that both recommender systems provide accurate predictions for some users, we suggest that a user-dependent selection algorithm is needed to use the best system for each user. It could test both systems for each new user and then when the probability distributions have converged, it can pick the system that has been most accurate thus far. Additionally, a different selection algorithm could combine only the categories for which the trustworthy reviewers are the same and leave the other categories apart.

Conclusions and Future Work

In this work, we assume that an agent, like a robot, can act autonomously in the world but can seek human help when it is uncertain of its environment. We're interested in the language of how an agent can ask humans questions to label its data most accurately when many (possibly novice) people can give answers. More specifically, we want to maximize the proportion of correct labels users provide so that data is as *accurate* as possible by asking many people for answers and learning who to trust.

We have approached the problem of agents asking many humans for help from two directions - the user side to understand what wording of questions are correlated with higher proportions of correct answers and the agent side to determine which users to trust once it has responses. We presented a methodology to explore how users' responses change based on the wording of questions and design guidelines to allow other researchers to develop agents that receive accurate responses from humans without having to repeat our methodology. Additionally, we show that developing trust in human responders largely depends on which are trustworthy in each category of questions.

There are still many open questions on both the user and agent sides of asking questions. It may be useful for an agent to ask many users in different locations for answers, to ask in different interfaces or venues depending on where a user is currently located, and to decide which set of users to ask based on a particular metric. For each of these problems, the questions that an agent asks may vary in order to get the most accurate answers possible from users. Additionally, when an agent receives responses, it may need to decide what to do when waiting for a response, decide which labels belong to which sensor data if there is a time delay to receive answers, and possibly relate longer term activities to a single label. The challenge will be in creating agents that can learn from humans but that also take into account the uncertainty in response rates and responses of their human helpers.

References

Antifakos, S.; Schwaninger, A.; and Schiele, B. 2004. Evaluating the effects of displaying uncertainty in context-aware applications. *UbiComp 2004: Ubiquitous Computing* 3205:54–69.

Auer, P.; Cesa-Bianchi, N.; Freund, Y.; and Schapire, R. 1995. Gambling in a rigged casino: The adversarial multi-armed bandit problem". *FOCS* 322–331.

Banbury, S.; Selcon, S.; Endsley, M.; Gorton, T.; and Tatlock, K. 1998. Being certain about uncertainty: How the representation of system reliability affects pilot decision making. *Human Factors and Ergonomics Society Annual Meeting Proceedings, Aerospace Systems* 36–39(4).

Cohn, D.; Atlas, L.; and Ladner, R. 1994. Improving generalization with active learning. *Machine Learning* 0885-6125:15(2):201–221.

Donmez, P., and Carbonell, J. 2008. Proactive learning: Cost-sensitive active learning with multiple imperfect oracles. *CIKM* 613–617.

Faulring, A.; Mohnkern, K.; Steinfeld, A.; and Myers, B. 2008. Successful user interfaces for radar. *CHI 2008 Workshop on Usable Artificial Intelligence*.

Fong, T. 2001. Collaborative control: A robot-centric model for vehicle teleoperation. *infoscience.epfl.ch*.

Larson, R., and Csikszentmihalyi, M. 1983. The experience sampling method. *New Directions for Methodology of Social and Behavioral Science* 15:41–56.

Leskovec, J.; Adamic, L.; and Huberman, B. 2006. The dynamics of viral marketing. *ACM Conf. on Electronic Commerce* 228–237.

Littlestone, N., and Warmuth, M. 1994. The weighted majority algorithm. *Information and Computation* 212–261.

Margineantu, D. 2005. Active cost-sensitive learning. *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence* 525–526.

Mcnee, S. M.; Lam, S. K.; Guetzlaff, C.; Konstan, J. A.; and Riedl, J. 2003. Confidence displays and training in recommender systems. In *Proceedings of the 9th IFIP TC13 International Conference on HumanComputer Interaction*, 176–183.

Mitchell, T. 1997. *Machine Learning*. McGraw Hill.

2008. The Netflix Dataset and Prize, www.netflixprize.com.

Rui, Y.; Huang, T.; Ortega, M.; and Mehrotra, S. 1998. Relevance feedback: a power tool for interactive content-based image retrieval. *Circuits and Systems for Video Technology, IEEE Transactions on* 8(5):644–655.

Rybski, P.; Yoon, K.; Stolarz, J.; and Veloso, M. 2007. Interactive robot task training through dialog and demonstration. *ACM SIGCHI/SIGART Human-Robot Interaction*.

Salton, G., and Buckley, C. 1990. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science* 41(4):288–297.

Saunders, J.; Nehaniv, C.; and Dautenhahn, K. 2006. Teaching robots by moulding behavior and scaffolding the environment. *ACM SIGCHI/SIGART Human-Robot Interaction*.

Steinfeld, A.; Bennett, R.; Cunningham, K.; Lahut, M.; Quinones, P.-A.; Wexler, D.; Siewiorek, D.; Cohen, P.; Fitzgerald, J.; Hansson, O.; Hayes, J.; Pool, M.; and Drummond, M. 2006. The radar test methodology: Evaluating a multi-task machine learning system with humans in the loop. Technical Report CMU-CS-06-125, CMU-HCII-06-102, Carnegie Mellon University, School of Computer Science, Pittsburgh, PA.

Stumpf, S.; Bao, X.; Dragunov, A.; and Dietterich, T. 2005. Predicting user tasks: I know what you're doing. *20th AAAI*.

Stumpf, S.; Rajaram, V.; Li, L.; Burnett, M.; and Dietterich, T. 2007. Toward harnessing user feedback for machine learning. *Proceedings of the 12th international conference on Intelligent User Interfaces*.

2008. Music user ratings of songs with song attributes v1.0. Yahoo! Research Webscope, www.research.yahoo.com.