Investigation of Principles of Context-Sensitive Reminding

Ece Kamar, Eric Horvitz
{eckamar,horvitz}@microsoft.com

October 2010

Technical Report
MSR-TR-2010-174

We describe research on principles of context-sensitive reminding that show promise for serving in systems that work to jog peoples' memories about information that they may forget. The methods center on the construction and use of a set of distinct probabilistic models that predict (1) items that may be forgotten, (2) the expected relevance of the items, and (3) the cost of interruption associated with alerting about a reminder. We describe the use of this set of models in the Jogger reminder prototype that employs predictions and decision-theoretic optimization to compute the value of reminders about meetings.

Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
http://www.research.microsoft.com

1 Introduction

In the course of daily life, people often forget information that would be valuable to them if they had remembered it at the right time. We present a study of principles for context-sensitive reminding that hold promise for effective personal reminder systems. The approach employs a set of probabilistic models learned from labeled data that predict outcomes required for effective reminding. These outcomes include (1) the probability that information will not be remembered, (2) the relevance of the forgotten information in a current or forthcoming setting, and (3) the cost of transmitting the reminder to a user within a current context. The work is inspired by prior work on principles of alerting aimed at creating personal agents that can mediate if and when notifications should be delivered to users in the course of daily life. We shall review the set of models and describe how we combine them into a working prototype named Jogger. We highlight key ideas in the context of reminders about meetings.

Within the Jogger prototype, forthcoming meetings are drawn from user's online calendar stored in an Exchange server. The system relies for reminder actions on guidance from ongoing decision-theoretic analyses that consider inferences from probabilistic models that are learned from data. We describe how Jogger calls in symphony predictive models for relevance, memory, and interruption.

We focus in a study on the sample use of Jogger for reminding users about forthcoming meetings—or about meeting details. We evaluate the approach via simulations on real-world calendar data. The results show that the context-sensitive reasoning methods of the prototype can successfully balance the benefit of a reminder with its cost. We find that under varying costs of interruption, Jogger performs significantly better than a traditional, non-adaptive reminder system that does not have access to predictive models.

2 Related Work

Several reminder systems are described in previous work. The ComMotion [13] and Memory Glasses [4] systems deliver reminders based on the location of users as sensed by GPS sensors. Cyreminder is a context-sensitive system that manages the timing of a reminder based on a user's context [5]. The Forget-Me-Not system organizes users' documents based on contextual cues, such as task orderings and the locations at which they are generated, and provides users with tools for retrieving documents based on contextual cues [12]. The Towel system proposes digital to-do lists as reminders [3]. A personal assistive agent designed as a part of the CALO project, initiates reminders for future tasks proactively by applying BDI reasoning [14]. None of these reminder systems employ a principled methodology for identifying the relevance, value, and timing of a reminder.

Decision-theoretic approaches have been applied to manage and filter notifications in different application domains. Boger et al. apply POMDPs to

provide task assistance to patients with memory problems [1]. Horvitz et al. propose learning from user input and building decision-theoretic models that reason about the cost of interruption to classify and selectively deliver notifications [8]. Jogger follows this line of research and demonstrates the applicability of these ideas to reminder systems.

The Coordinate [10] system employs probabilistic inference models learned from data to predict meeting attendance, meeting priority, and dynamically computed costs of interruption for meeting contexts. The BusyBody [9] system predicts the cost of interruption in desktop computing contexts. Jogger harnesses predictive models developed in the Coordinate and BusyBody research efforts, and integrates them with probabilistic models of recall to capture the expected value of reminders based on users' situations.

Previous work on building adaptive reminder systems focused on changing visual presentations of reminders based on user preferences, reminder urgency and user attention [15]. Jogger follows a complementary approach and provides formal methods for computing the expected value of reminders and for distinguishing useful reminders from others based on multiple inferences and contextual information.

3 Expected Value of a Reminder

Reminder systems seek to assist users by jogging their memories about upcoming events. Reminders are useful in helping users to recall tasks that need to be accomplished or providing users with other enabling information (e.g., names of people met before in a social setting). The information can lead to enhanced task efficiencies and outcomes. As an example, beyond reminding a user about a forgotten meeting, appointment reminders can provide users with information that crispens fuzzy memories about locations, people, and topic information.

An ideal reminder system should consider both the benefit that a reminder may generate for the user and the cost of interruption associated with transmitting the reminder. We introduce a reminder system which follows a decision-theoretic approach to distinguish reminders that are beneficial for a user's performance from the ones that are not.

We shall now discuss how we compute the cost and benefits of a reminder. We shall introduce several key probabilities in the analysis. In Section 4, we shall focus on the construction of predictive models from data to capture the value of a reminder in Jogger with these probabilities.

A reminder for task m is beneficial for a user if the expected utility for receiving a reminder about m is higher than the cost of interruption given the current state of the user. The utility of a reminder depends on the cognitive state of a user: has the user forgotten all or some information that might be included in a reminder.

Jogger considers three mental states with respect to recall of information useful for completing tasks under consideration: (1) F^m represents the state in which a user has forgotten all about m, (2) D^m represents the state in which

the user has forgotten or is unsure about a subset of details regarding the task, such as its location, start time (or deadline), and other participants, and (3) R^m represents the state in which the user remembers that task m exists and also remembers all of the details regarding the task.

Given evidence E that comprises observations about a user's state, $p(F^m|E)$, $p(D^m|E)$, $p(R^m|E)$ are the probabilities of the user being in states F^m, D^m, R^m respectively given evidence E. F^m, D^m and R^m are mutually exclusive and collectively exhaustive. Thus, $p(R^m|E) = 1 - p(F^m|E) - p(D^m|E)$.

The enhancement of the performance of a user with regard to a task or event changes depending on the timing of the remembering of relevant information. As an example, if a user completely forgets about a meeting, she will not be able to participate nor contribute to a task. If a user forgets some details about a forthcoming meeting (e.g., the location of a meeting), her utility may decrease because of tardy arrival. $U(F^m, E)$, $U(D^m, E)$ and $U(R^m, E)$ represent user's utilities for being in states F^m, D^m, and R^m respectively. These utilities are modeled to be sensitive to the context of a user and the attributes of task m.

A user may maintain a long list of tasks or events (e.g., on an electronic calendar) but are interested in contributing or participating in only a subset of the larger list of events. It is important for a reminder system to distinguish relevant tasks and events from the others within a context. The benefit of a reminder about task m to a user depends on whether and how much m is relevant to the user's plans. $p(A^m|E)$ is the likelihood that the user would engage in task m if she remembers about m. $COI(m, E)$ represents the cost of interrupting the user by delivering a reminder about m, given evidence E about the user's state.

Given the different states that a user may be in and utility values associated with them, we compute the *expected value of reminding* (EVR) by considering the likelihood of each state, the relevance of the reminder and utilities associated with being in each state. EVR is calculated as given below:

$$\begin{aligned} EVR(m) =& p(F^m|E)\ p(A^m|E) \\ & (U(R^m, E) - U(F^m, E)) \\ & + p(D^m|E)\ p(A^m|E) \\ & (U(R^m, E) - U(D^m, E)) \\ & - COI(m, E) \end{aligned} \tag{1}$$

It is beneficial to send a reminder to a user if the reminder is associated with a positive EVR value. If so, the expected benefit of interrupting and sending a reminder is greater than the cost of interruption associated with it.

3.1 Meetings and Memory

We now review the use of the utility and relevance models proposed in Equation 1 for the sample case of recall and reminding about meetings. A user is in state F^m if she has forgotten that meeting m exists. She is in state D^m if she has

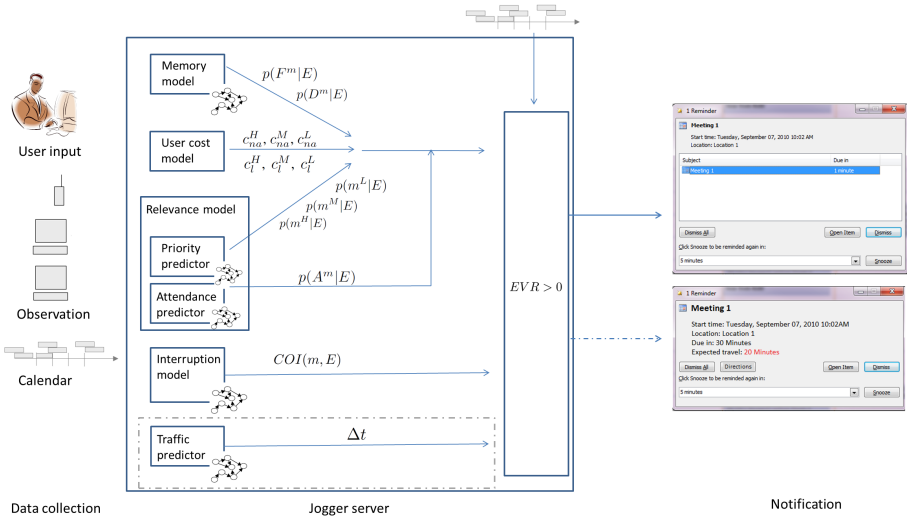

Figure 1: Jogger system components. Two interfaces of the system are presented on the right, one for the original system architecture (marked with straight arrow) and one that includes consideration of traffic-sensitive travel to a meeting (dashed arrow).

forgotten or is unsure about the details of m such as its location or time. The user is in state R^m if she remembers all the details of the meeting. Bayesian models for predicting the likelihood of user being any of these states based on evidence E are presented in Section 4.1.

We assume that a reminder about meeting m is only beneficial, and thus relevant to a user if the user intends to attend the meeting. In the meeting reminder context, the relevance likelihood of m is realized by $p(A^m|E)$, the probability of attending meeting m given E, evidence about the meeting. A probabilistic model for predicting probability of attendance is presented in Section 4.2.

Next, we formalize utility values for being in any of the three possible memory states. We make the assumption that if a user is in state F^m, the user fails to attend meeting m; if the user is in state D^m, she misses the first t minutes of the meeting because of problems with recalling the details about the meeting; and if the user is in state R^m, the user is on time for the start of a meeting.

Meetings are not equally prioritized in users' schedules. If the expected cost of interruption (COI) is high for the current state, the user may not prefer to be reminded for a low priority meeting although she would be willing to be reminded for a high priority meeting. Jogger system has the priority predictor for inferring priority probabilities for meetings. For any given meeting m, the system predicts the probability that m has high priority $p(m^H|E)$, medium priority $p(m^M|E)$ and low priority $p(m^L|E)$.

We ask the user to evaluate the value of time for three possible cases; the minute cost for being late, c_l^H for high, c_l^M for medium, c_l^L for low priority meetings; the total cost for not attending to a meetings, c_{na}^H for a high, c_{na}^M for a medium, c_{na}^L for a low priority meeting, and the minute cost for being early, c. These values represent the user's willingness to pay in dollars for preventing to be in any one of these situations. Quantifying users' willingness to avoid undesirable outcomes with dollar values has been used in decision analyses in several fields (e.g., medical decision analyses).

$COI(m, E)$ represents the dollar value a user is willing to pay for not to receive an interruption about m given evidence E. A model for capturing COI is presented in Section 4.3.

The expected value of a meeting reminder is calculated as below by combining user costs with associated memory states:

$$\begin{aligned} EVR(m) =& p(A^m|E)\ (p(F^m|E)\ c_{na} + p(D^m|E)\ c_l\ t) \\ & - COI(m, E) \end{aligned} \tag{2}$$

where c_{na} and c_l values are calculated as below given the priority model:

$$\begin{aligned} c_{na} =& (p(m^H|E)\ c_{na}^H) + (p(m^M|E)\ c_{na}^M) \\ & + (p(m^L|E)\ c_{na}^L) \end{aligned} \tag{3}$$

$$\begin{aligned} c_l =& (p(m^H|E)\ c_l^H) + (p(m^M|E)\ c_l^M) \\ & + (p(m^L|E)\ c_l^L) \end{aligned} \tag{4}$$

4 Predictive Models

As described above, we require probabilistic inferences, $p(A^m|E)$, $p(F^m|E)$, $p(D^m|E)$ and the probability distribution over meeting priority to compute the expected value of reminding. We now review the construction of set of predictive models used in Jogger to infer these probabilities from calendar data and context.

Jogger has access to appointments drawn from Exchange, along with a constellation of atomic and derived meeting properties that serve as evidential features about the meetings. A set of appointments drawn from several months of the online calendar are composed into a case library of training and testing sets of meeting instances. We asked participants to tag meetings with several labels via a tagging tool. One label encodes a user's assessments about whether they will attend a meeting, and that the meeting is thus relevant. A second set of labels is used to assess meeting priority, used to compute the cost of missing or being late for a meeting. A third label represents whether users would forget about the meeting or about important meeting details. In particular, users are asked to tag meetings to indicate whether they would likely (or did) recall versus forget the meeting's existence and further, whether they would need to

be reminded about important details of a meeting, such as its location and attendees.

A set of predictive models required for inferences about ideal reminders is constructed from this tagged training set. The *relevance model*, built from cases tagged by attendance, is used to infer the probability that a future meeting will be attended. A second *priority model* predicts the probability distribution over the priority of a meeting. The *memory models*, built from the labels on recall, provide inferences about the likelihoods that future meeting would be completely forgotten, or whether the meeting would be remembered coarsely but that meeting details would be needed. A third class of predictive model, *interruption models*, are borrowed from prior efforts on the Coordinate and BusyBody systems (as described below). These models provide the cost of transmitting an alert to users in meeting and desktop settings, respectively.

4.1 Memory Models

We constructed memory models for predicting that a user has forgotten the existence of a meeting, and for predicting that a user has forgotten the details of a meeting, including its location, date and time, subject and attendee list.

Each user of Jogger invests effort in personalizing two memory models by applying supervised learning as described above. To collect training data, we provided users a form that displays a sorted list of past meetings for tagging. For each meeting, the form has a row providing information about the meeting (e.g., its subject, organizer, attendees, location) and about the context the user was in at the time of the meeting (e.g., number of meetings during the day, previous meeting, next meeting, number of meetings that week, location of the previous meeting, end time of the previous meeting, etc.) so that the user is able to recall her cognitive load for that day and the state of her memory regarding that meeting when she is providing training instances. Each row has two Boolean fields for the user to fill out to indicate whether the user has completely forgotten about the meeting and whether the user has forgotten some details regarding the meeting. Given that states F^m, D^m and R^m are mutually exclusive, a user is allowed to fill at most a single one of the Boolean fields. If a user does not fill any of the fields, that instance is labeled to belong to state R^m.

The system generates a training set by combining each meeting instance tagged by a user with a set of attributes acquired from the user's personal Outlook profile. These attributes include the day and time of the meeting, its duration, subject, location and organizer, the number and nature of attendees, the response status of the user, whether the meeting is recurring and whether the time was marked as busy in the user's calendar. By accessing the Microsoft Active Directory service, the organizational relationship between the user, the organizer and the attendees are also included as meeting attributes. Additionally, we identify meetings with atypical locations, attendees and organizers by checking a personalized database of meetings and by marking the ones that appear less than a small, predetermined percentage.

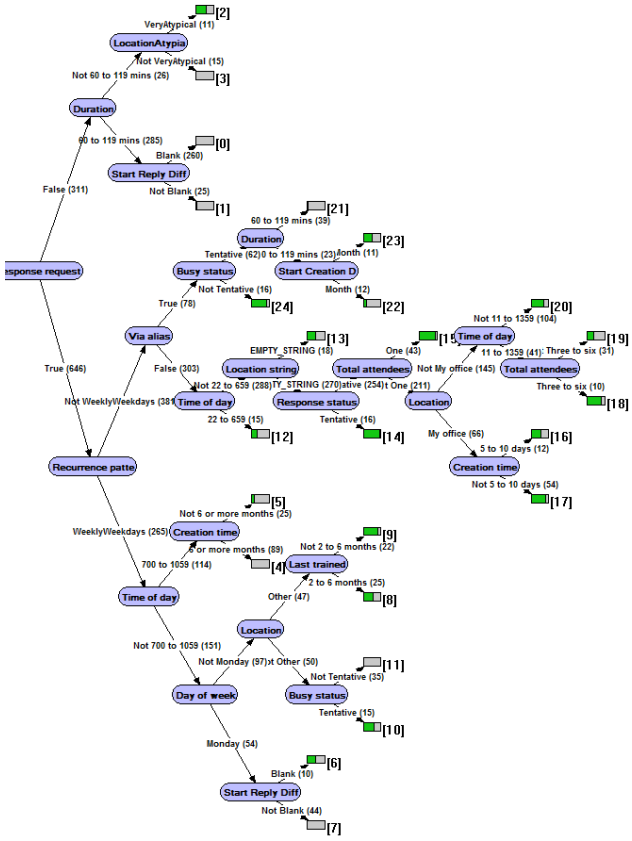

Figure 2: A memory model for Subject 1 for predicting the probability of forgetting about the occurrence of a meeting, based on the selected attributes of a meeting.

We perform Bayesian structure learning to build probabilistic models that predict whether a user has forgotten that a meeting exists, and whether a user has forgotten about some details of a meeting. The system selects a model by performing heuristic search over feasible probabilistic dependency models guided by a Bayesian scoring rule [1] [2, 6]. The procedure generates decision trees for predicting the probability of forgetting a meeting and forgetting the details of a meeting based on the attributes represented in the models.

Two subjects shared their calendar data to evaluate the performance of the memory models. The subjects tagged a subset of their own appointments. We individually trained memory models for them by performing cross validation. We used 85% of the tagged data for training and used the remaining 15% for

[1] A similar approach has been followed in previous work for modeling context-sensitive costs and cost of interruption [11, 10]

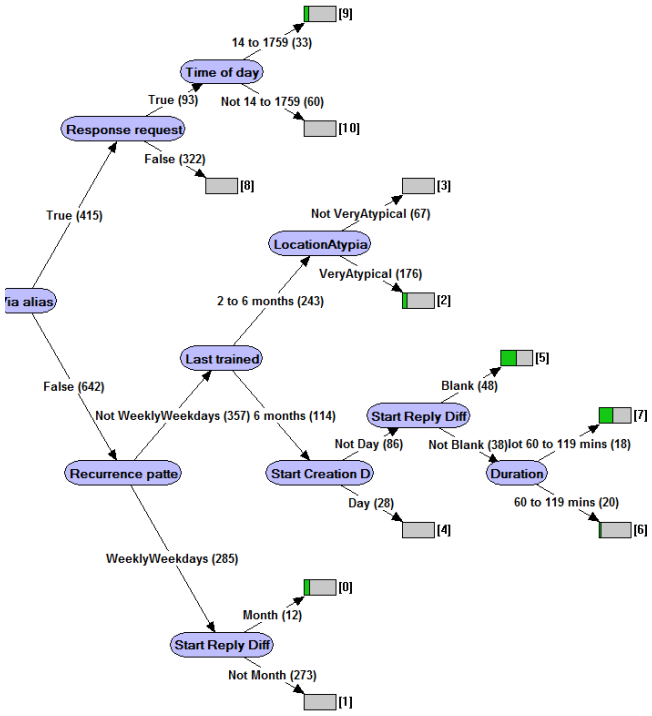

Figure 3: A memory model for Subject 1 for predicting the probability of forgetting details of a meeting, based on the selected attributes of a meeting.

testing. Two sample predictive memory models are presented in Figures 2 and 3.

Table 1 presents the prediction accuracies of the memory models generated for Subjects 1 and 2. The accuracies of the generated models are compared with a marginal model that assigns each instance to the most frequent class label. The results show that the generated models reach above 85% accuracy for both memory models and both subjects. They also indicate significant improvements over the marginal models for all instances except for the forgot details model for Subject 1.

The graphical representations of the memory models displayed in Figures 2 and 3 highlight the similarities and differences in predicting whether a user has forgotten about a meeting, and whether a user has forgotten about the details of a meeting. In both memory models, whether a response was requested for a meeting request, the recurrence of a meeting, whether the user is invited via alias or message group and whether the location of a meeting is atypical are selected as prominent attributes. However, the day of the week, the number of attendees and whether the user is busy do matter for predicting whether the user has forgotten about a meeting, whereas these attributes do not seem

	Forgot All		Forgot Details	
	Subject 1	Subject 2	Subject 1	Subject 2
Learned Model	85%	96%	90%	94%
Marginal Model	68%	54%	89%	78%

Table 1: Classification accuracies of learned memory models in comparison with the marginal models.

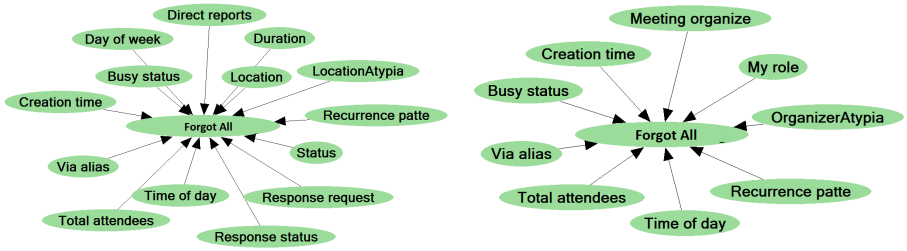

Figure 4: Memory models learned for Subject 1 (left) and Subject 2 (right) that infers the likelihood of their forgetting that a meeting exists, based on the attributes of the meeting and surrounding context.

to matter for predicting whether the user has forgotten about the details of a meeting.

Figure 4 compares two Bayesian networks learned individually for Subjects 1 and 2 for predicting their likelihood of forgetting that a meeting exists. Although the models for predicting the likelihood of forgetting a meeting share some prominent common attributes such as the recurrence pattern of a meeting, whether a response was requested, and whether the invitation was sent to a distribution list, many of the attributes appear in two models are distinct. The duration, location and day of a meeting emerge as important attributes of Subject 1's model, whereas the organizer of a meeting, the number of attendees and user's role in the meeting are chosen to appear in Subject 2's model. This comparison highlight the fact that the way people forget may differ, and thus building predictive models that can learn about these personal characteristics may be crucial for the success of an context-sensitive reminder system.

4.2 Attendance and Priority Models

Jogger borrows predictive components from the Coordinate system [10] for predicting the relevance and the importance of a meeting. A list of past meetings of a user is converted into a draft training set by applying a set of heuristics about attendance based on observations including user's desktop activity. The

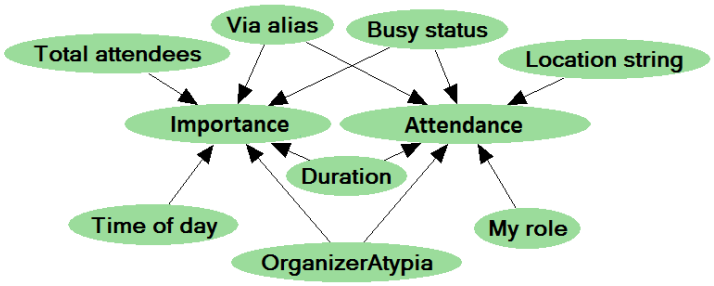

Figure 5: Relevance model learned for Subject 1 for predicting meeting attendance and meeting priority based on the selected attributes of a meeting.

draft is provided to users for them to manually correct attendance fields and to specify whether a meeting is low, medium or high priority.

Based on the training data, a model is constructed for predicting the likelihood of a user attending a meeting and the priority of a meeting based on the attributes of a meeting. These selected attributes include the day, time, location, organizer and attendees of a meeting, the role of the user, whether the user was invited by name or a distribution group and the user's response to the invitation. A Bayesian network generated for Subject 1 for predicting meeting attendance and meeting priority is presented in Figure 5.

4.3 Interruptibility Models

Jogger uses a two-layer approach to estimate the cost of interrupting a user. It brings together the activity-based predictions of the cost of interruption inferred by BusyBody and the meeting-based interruptability prediction model of the Coordinate system. By doing so, we are able to infer the cost of interrupting a user when the user is in her office by observing office activity, and when the user is in a meeting based on the importance of the meeting.

Jogger accesses BusyBody to find out whether a user is in her office at the time of inference. If so, it makes a query to BusyBody to infer the cost of interrupting her based on her office activity. The BusyBody system collects evaluations from users about their costs of disruption as dollar amounts they are willing to pay to avoid an interruption in different states of interruptability. The system monitors a user by observing whether the user is typing on the computer, using the mouse, the applications used by the user, whether the user is engaged in a conversation. The system performs supervised learning of the cost of interruption by using a value of information analysis to guide probes of the users state via a pop up query. Given user assessments and contextual evidence collected, the system learns a personalized Bayesian network model with a Bayesian learning procedure. Given $p(I_j|E)$ is the probability of being

in interruptability state I_j given evidence E, and $C(I_j)$ is the associated cost of interrupting in state I_j, COI, the cost of interruption, is computed as below,

$$COI = \sum_j p(I_j|E)\ C(I_j) \tag{5}$$

When a user is not in her office, Jogger accesses Coordinate's meeting priority prediction model to estimate the cost of interruption based on the meeting the user may be attending. Given that c_i^H, c_i^M, c_i^L are user-assessed costs for being interrupted in high, medium and low priority meetings, COI is calculated as,

$$COI = p(m^H|E)\ c_i^H + p(m^M|E)\ c_i^M + p(m^L|E)\ c_i^L \tag{6}$$

If there exists more than one meeting appointment in a user's calendar for the moment of inference, COI is calculated with respect to the meeting with the highest attendance probability. If there are no meetings associated with the time of inference, the cost of interruption is assigned to a predetermined small cost value.

5 Jogger Architecture

The Jogger prototype computes the expected value of a reminder and weights this with the cost of interruption. Jogger accesses inferences from its relevance, memory, priority, and interruption models to determine whether to deliver a reminder to a user. A systematized overview of the system is presented in Figure 1. The system includes a data collection component, a collection of predictive models, an interface to users' calendars, a utility-based decision-making model and a notification layer that interrupts users when needed and delivers reminders.

The data collection component is designed to gather relevant information about the context that a user is in. The component accesses a user's calendar, monitors computer activity, and detects video and audio signals and acquires user input. The information collected from the data collection component is used for inferences needed to compute the net expected value of reminders. The models are used to predict whether a meeting is important, whether a user intends to attend the meeting, whether the user has forgotten about the meeting, and the cost of interrupting the user based on her current context. These inferences of these models are combined with user assessed costs as described by Equation 2 to form the expected value of a reminder.

The Jogger system actively checks users' calendars to identify reminder opportunities. For each reminder opportunity, the system accesses the data collection component and gathers information about the current state of the user. Based on the data collected, the system infers the expected value of reminding the user and interrupts and reminds the user only if the associated value is positive.

6 Locations, Travel, and Traffic

Reminder systems are traditionally designed to refresh users' memory about a task or a meeting. Increasing connectivity of computing devices enables to design reminders as information delivery tools. More valuable reminder systems can be designed when cues about future tasks are accompanied with information that will make it easier to accomplish the tasks. For instance, a reminder for a meeting can also deliver up-to-date information about the traffic conditions on the route to the meeting, if the location of the meeting is available, so that users may have higher chance of being at the meeting on time. We now discuss an extension of Jogger to reason about the value of delivering real-time travel information and travel directions via reminder notifications.

Several research projects [7] and real-world systems (e.g., Clearflow traffic-sensitive directions on Bing Maps) provide estimates of time-dependent travel times to destinations. We use t_{live} to represent the estimate for the duration of travel to a meeting. We shall use $t_{expected}$ as the expected travel duration under regular traffic conditions. We consider Δt, the difference between t_{live} and $t_{expected}$, as the estimated error in user's prediction of the duration of travel to the meeting location.

The error in a user's prediction of traffic conditions affect whether the user will be on time for a meeting. For instance, if the road heading to a meeting location is congested due to an accident and the live travel estimate is 10 minutes more than expected, the estimated utility of delivering a reminder with live travel estimates should integrate avoiding the extra 10 minute cost for being late. Consequently, the EVR calculations are updated as given below to consider this additional cost.

$$\begin{aligned} EVR(m) =& p(A^m|E) \\ & (p(F^m|E)\ c_{na} + p(D^m|E)\ C(D^m, E) \\ & + (1 - p(F^m|E) - p(D^m|E))\ C(R^m, E)) \\ & - COI(m, E) \end{aligned} \tag{7}$$

The expected cost for being in state D^m is represented as $C(D^m, E)$, in which c is the minute cost for arriving at the meeting location early, and t and c_l are defined as given in Section 3.1.

$$C(D^m, E) = \begin{cases} (t + \Delta t)\ c_l & if\ (t + \Delta t) > 0 \\ (-t - \Delta t)\ c & if\ (t + \Delta t) \leq 0 \end{cases} \tag{8}$$

The expected cost for being in state R^m is represented as $C(R^m, E)$. A user in state R^m may incur costs for not knowing about live traffic predictions.

$$C(R^m, E) = \begin{cases} \Delta t\ \ c_l & if\ \Delta t > 0 \\ (-\Delta t)\ c & otherwise \end{cases} \tag{9}$$

6.1 Timing of Reminders

The timing of reminders is important to maximize the benefit that users receive from reminder systems. The expected utility of a reminder changes in time as expected cost of interruption changes. For example, a user talking on phone may not be available to receive a notification but may benefit from receiving the reminder when she finishes talking a few minutes later.

The challenge of timing reminders ideally based on changing traffic conditions and cost of interruption is a complex problem on its own; we do not focus on that challenge here. Jogger employs a set of heuristic rules to time reminders. The prototype searches through time to find the moments that satisfy three conditions: (1) the expected utility of reminder is positive, (2) user has sufficient time to get to a meeting location after receiving the reminder, (3) the time of the reminder is as close as possible to expected trip initiation time so that the user's memory about the meeting is fresh at the departure time.

Jogger uses the predictions of traffic services to find the optimal time for reminding the user, with the assumption that the cost of interruption is constant through time. If the user's expected travel duration is higher than the live estimate, the user should be reminded before $t_m - t_{expected}$ so that the user can postpone her take off according to the live estimate. If the live estimate of travel duration higher than the user's expectation, the reminder needs to be scheduled before $t_m - t_{live}$, so that the user can move the departure time earlier to be on time for the meeting. Given that t_m is the start time of meeting m, t_{offset} is the preferred interval between receiving a reminder for m and starting a trip to m, the optimal reminder time, t^*, is calculated as below,

$$t^* = \begin{cases} t_m - t_{expected} - t_{offset} & t_{expected} \geqslant t_{live} \\ t_m - t_{live} - t_{offset} & otherwise \end{cases} \tag{10}$$

Jogger reminds a user about meeting m at time t^*, if $EVR(m)$ value computed at time t^* is positive. If that's not the case, Jogger searches the interval t^*, $t^* + t_{offset}$ to find a time with positive $EVR(m)$ value. If no such moment exists, Jogger does not send a reminder about meeting m.

7 Empirical Evaluation

Empirical evaluation of the Jogger system with real-world users requires long-term deployment of the system in offices that are equipped with special software and hardware components for sensing the state of a user correctly (e.g., BusyBody uses a microphone system to detect conversations). This section presents a proof of concept analysis of the system based on real-world data. This study analyses the value offered to users with the Jogger system under varying domain conditions.

We activated Jogger for a user in our organization (referred as Subject 1 in Section 4.1) that sits in an office equipped with the necessary infrastructure. Jogger generated memory, attendance and priority models based on training

data collected from the user. It accessed to the BusyBody component that performs real-time inference about the cost of interrupting the user. We asked the user to assess costs for being late to low-medium-high priority meetings, costs for missing low-medium-high priority meetings, and to estimate the number of minutes she would be late to a meeting if she forgot the details. Jogger stayed active for a 2 week period on the user's machine and performed EVR calculations for 103 distinct reminder opportunities in real-time. The system kept a log file that reports the EVR value for each reminder opportunity and the output of each predictive model. Jogger delivered a reminder to the user for every reminder opportunity with positive EVR. Since location information was not available for all meetings in the user's calendar, the evaluation used the simple version of the system that does not reason about about live traffic predictions.

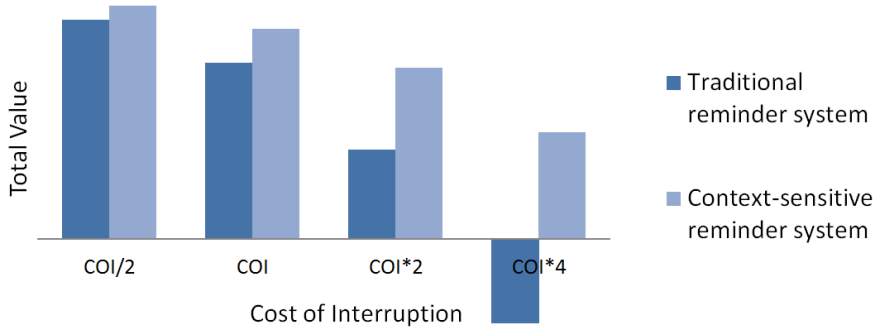

Figure 6: Comparison of the effectiveness of the context-sensitive reminder system and the traditional reminder system for increasing costs of interruption.

Our first set of evaluations analyze the value generated by reminder systems under the assumption that the EVR values computed by the system are accurate [2]. The analysis compares the performance of two distinct reminder systems to the baseline of not having any reminders: A traditional reminder system that sends reminders for every meeting on a calendar without considering the value of a reminder, and our context-sensitive reminder system that delivers a reminder to a user if the expected value of a reminder is positive. The total value generated by the traditional reminder system is the sum of EVR values for all reminder opportunities. The total value generated by Jogger is calculated as the sum of EVR values for all reminders sent to the user (for all reminder opportunities with positive EVR).

The analysis shows that even the traditional reminder system generates positive value for the user. The user is better off having a traditional reminder system instead of not having any reminders. On the other hand, having Jogger,

[2] We are currently working on a follow-up study that evaluates the accuracy of EVR values based on user feedback.

a context-sensitive reminder system, instead of a traditional reminder system increases the total value generated by reminder systems by 19.5% while decreasing the total number of interruptions by 46.6%.

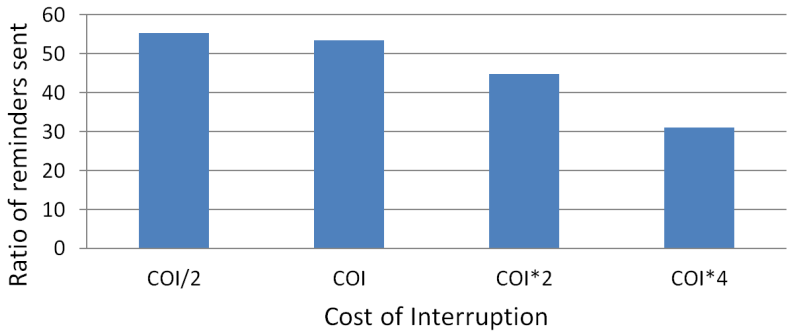

Figure 7: Ratio of the reminders sent by the context-sensitive system to all reminder opportunities for increasing costs of interruption.

To further illustrate the adaptability of the context-sensitive reminder system, we performed simulations on data collected from our study. These simulations evaluated the effect of cost of interruption on the effectiveness of the traditional and context-sensitive reminder systems. We varied the cost of interruption from the half of the original value to the four times of the value. Figure 6 reports the values generated with traditional and context-sensitive reminders systems for varying costs of interruption when compared with the baseline of not delivering any reminders. The value generated with the traditional system drops fast as the cost of interruption increases. The cost of the traditional system becomes more than its benefit when the cost of interruption is high. However, Jogger can successfully trade off the cost of a reminder with its benefit and thus adapt itself to increasing costs of interruption. The context-sensitive reminder system generates positive value for the user for all costs of interruption values. It manages to do so by dynamically reducing the number of reminders sent to the user as the cost of interruption increases as shown in Figure 7.

The decisions made by Jogger are based on a set of predictions performed by probabilistic models. Although these models are reported to have relatively high accuracies [3], errors in these predictions may occasionally result in incorrect decisions, such as delivering a reminder that is not valuable, or not delivering a valuable reminder. We performed a set of simulations on the collected data to analyze the sensitivity of the effectiveness of the context-sensitive reminder system to imperfections in the prediction models. We added some noise to the predictions made by each probabilistic model, and compared the decisions made by the system under this noise to the ground truth - the decisions of the orig-

[3]The prediction accuracies of the probabilistic models are reported in Section 4.1 and in previous work [10, 9].

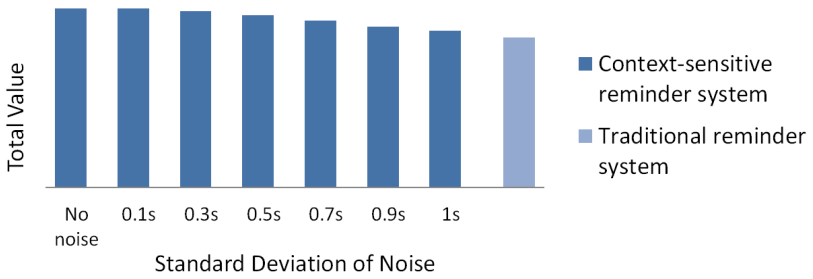

Figure 8: Sensitivity of the effectiveness of the context-sensitive reminder system to imperfections in the prediction models.

inal system with no noise added. The added noise is randomly sampled from a Gaussian distribution for each reminder opportunity. The distribution used for generating noise for each probabilistic model has zero mean and a standard deviation proportional to the sample standard deviation calculated from the predictions performed by the model for a subset of reminder opportunities. The added noise represents the possible errors made by the probabilistic models. Figure 8 reports the total value generated by the reminder system as distributions with larger standard deviations are used to sample noise to be added to predictions of the models. The reported values are averaged over 1000 trials. As predicted, the analyses indicate that the effectiveness of the system drops as the model predictions become less accurate. However, the effectiveness of the context-sensitive reminder system is still higher than the traditional system even when the predictions of the probabilistic models are very noisy - when noise added to each prediction model is sampled from a distribution with high standard deviation (as high as the standard deviation of predictions obtained from the model). These results highlight the fact that having multiple prediction models contributing to the reminder system enables the system to be robust to prediction errors, and to generate value for users even when prediction models are noisy.

8 Conclusion and Future Work

We have focused on identifying opportunities and challenges for developing a context-sensitive reminder system that distinguishes valuable reminders from the ones that are not, and thus reduces the disruptive effects of reminder systems. We presented methods and models in Jogger, a prototype focusing on meeting reminders. Jogger employs multiple predictive models of users' memory and interruptability, and meeting relevance, and performs decision-theoretic reasoning to capture the value of a reminder. We analyzed the value of context-sensitive alerting in changing interruptibility conditions in real-world data. Although the Jogger prototype focuses on meeting reminders, the proposed meth-

ods and models are general.

For future work, we are exploring several extensions, which include (1) deploying Jogger in the real-world with multiple users and evaluating system performance, (2) improving the predictive models employed in Jogger with real-time user feedback and active learning, (3) enriching the library of predictive models used in Jogger by modeling factors that may contribute to the expected value of a reminder, (4) developing more sophisticated decision-making models for better timing context-sensitive reminders, and (5) applying the models and methods presented in this work to building reminder systems for complex task domains. We believe that machine learning and reasoning hold great promise for the development of personalized reminder systems that come to understand the nuances of users' memories, situations, and needs for memory jogging, and that such systems may one day provide great value to people in the course of daily life.

References

[1] J. Boger, P. Poupart, J. Hoey, C. Boutilier, G. Fernie, and A. Mihailidis. A decision-theoretic approach to task assistance for persons with dementia. In *International Joint Conference on Artificial Intelligence*, volume 19, page 1293. Citeseer, 2005.

[2] D. Chickering, D. Heckerman, and C. Meek. A Bayesian approach to learning Bayesian networks with local structure. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 80–89. Citeseer, 1997.

[3] K. Conley and J. Carpenter. Towel: Towards an intelligent to-do list. In *Proceedings of the AAAI Spring Symposium on Interaction Challenges for Artificial Assistants*, 2007.

[4] R. DeVaul, B. Clarkson, and A. Pentland. The Memory Glasses: Towards a wearable context aware, situation-appropriate reminder system. In *CHI 2000 Workshop on Situated Interaction in Ubiquitous Computing*, 2000.

[5] A. Dey and G. Abowd. Cybreminder: A context-aware system for supporting reminders. In *Handheld and Ubiquitous Computing*, pages 201–207. Springer, 2000.

[6] N. Friedman and M. Goldszmidt. Learning Bayesian networks with local structure. *Learning in graphical models*, pages 421–460, 1998.

[7] E. Horvitz, J. Apacible, R. Sarin, and L. Liao. Prediction, expectation, and surprise: Methods, designs, and study of a deployed traffic forecasting service. In *UAI*, 2005.

[8] E. Horvitz, C. Kadie, T. Paek, and D. Hovel. Models of attention in computing and communication: from principles to applications. *Communications of the ACM*, 46(3):52–59, 2003.

[9] E. Horvitz, P. Koch, and J. Apacible. BusyBody: creating and fielding personalized models of the cost of interruption. In *Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 507–510. ACM, 2004.

[10] E. Horvitz, P. Koch, C. Kadie, and A. Jacobs. Coordinate: Probabilistic forecasting of presence and availability. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI 2002)*, pages 224–233. Citeseer, 2002.

[11] E. Horvitz, P. Koch, and M. Subramani. Mobile opportunistic planning: methods and models. *User Modeling 2007*, pages 228–237, 2009.

[12] M. Lamming and M. Flynn. Forget-me-not: Intimate computing in support of human memory. In *Proceedings of FRIEND21*, volume 94, pages 2–4. Citeseer, 1994.

[13] N. Marmasse and C. Schmandt. Location-aware information delivery with commotion. In *Handheld and Ubiquitous Computing*, pages 361–370. Springer, 2000.

[14] K. Myers and N. Yorke-Smith. Proactive behavior of a personal assistive agent. In *Proceedings of Autonomous Agents and Multiagent Systems*, volume 7. Citeseer, 2008.

[15] J. Weber and M. Pollack. Evaluating user preferences for adaptive reminding. In *CHI'08 extended abstracts on Human factors in computing systems*, pages 2949–2954. ACM, 2008.