An Implicit Dialogue Injection System for Interruption Management

Tomoki Shibata
Tufts University
Medford, MA, USA
tshibata@cs.tufts.edu

Alena Borisenko
Tufts University
Medford, MA, USA
alena.borisenko@tufts.edu

Anzu Hakone
Tufts University
Medford, MA, USA
anzu.hakone@tufts.edu

Tal August
University of Washington
Seattle, WA, USA
taugust@cs.washington.edu

Leonidas Deligiannidis
Wentworth Institute of Technology
Boston, MA, USA
deligiannidisl@wit.edu

Chen-Hsiang Yu
Wentworth Institute of Technology
Boston, MA, USA
yuj6@wit.edu

Matthew Russell
Tufts University
Medford, MA, USA
mrussell@cs.tufts.edu

Alex Olwal
Google Inc.
Mountain View, CA, USA
olwal@google.com

Robert J.K. Jacob
Tufts University
Medford, MA, USA
jacob@cs.tufts.edu

ABSTRACT

This paper presents our efforts in redesigning the conventional on/off interruption management tactic (a.k.a. "Do Not Disturb Mode") for situations where interruptions are inevitable. We introduce an *implicit dialogue injection* system, in which the computer implicitly observes the user's state of busyness from passive measurement of the prefrontal cortex to determine *how* to interrupt the user. We use functional Near-Infrared Spectroscopy (fNIRS), a non-invasive brain-sensing technique. In this paper, we describe our system architecture and report results of our proof-of-concept study, in which we compared two contrasting interruption strategies; the computer either forcibly interrupts the user with a secondary task or requests the user's participation before presenting it. The latter yielded improved user experience (e.g. lower reported annoyance), in addition to showing a potential improvement in task performance (i.e. retaining context information) when the user was busier. We conclude that tailoring the presentation of interruptions based on real-time user state provides a step toward making computers more considerate of their users.

CCS CONCEPTS

• **Human-centered computing → Interaction techniques**.

KEYWORDS

Interruption, functional Near-Infrared Spectroscopy (fNIRS), Implicit User Interfaces, implicit interactions, implicit dialogue injection, HumanSketch

AH2019, March 11–12, 2019, Reims, France

ACM ISBN 978-1-4503-6547-5/19/03…$15.00
https://doi.org/10.1145/3311823.3311875

ACM Reference Format:
Tomoki Shibata, Alena Borisenko, Anzu Hakone, Tal August, Leonidas Deligiannidis, Chen-Hsiang Yu, Matthew Russell, Alex Olwal, and Robert J.K. Jacob. 2019. An Implicit Dialogue Injection System for Interruption Management. In *Augmented Human International Conference 2019 (AH2019), March 11–12, 2019, Reims, France.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3311823.3311875

1 INTRODUCTION

Improvements in smart computer technologies seem also to be forcing users to live with constant interruptions. Emails, texts, and even software update notifications interrupt and draw users' attention away from their tasks.

Some systems already provide a Do Not Disturb Mode feature to fight against indiscriminate computer notifications. Yet, the feature may not be appropriate when some notifications must be received and acted upon in a timely manner. The "Do Not Disturb Challenge" points out that "notifications may affect people negatively, but they are essential" [31].

Some current smartphones and other devices could automatically toggle the feature by taking environmental context, such as time and location, into account. However, most such options still only control *when* the computer can interrupt the user. In this paper, we propose an alternative that modifies *how* the computer should interrupt the user by taking the user's internal, physiological context into account in cases when the interruptions are unavoidable.

Consider a simple scenario where you want to ask a question to your team member at work. She is in a room next door and the door is half open. Before speaking to her, you almost unconsciously leverage your perception to determine how busy she is. If you figure she is in the middle of doing something, you could knock on the door first to check her availability; otherwise, you might speak to her immediately.

While this scenario is feasible for humans, it has heretofore been impossible for a computer due to its inability to recognize a user's state of busyness in an implicit way. To address this, we introduce an *implicit dialogue injection* system that permits the computer to

exploit measurement of the user's momentary state to modify its behavior. The system introduces two key concepts: (1) *HumanSketch*, a conceptual human model exposing the user's "busyness" in a form that the computer can understand, and (2) *implicit dialogue*, the method for the computer to perceive information exposed by HumanSketch. We deployed the system as an interruption management technique, in which the computer implicitly obtains the user's momentary busyness right before triggering an interruption; it can thus select its interruption strategy accordingly.

In order to determine busyness, our design calls for using functional Near-Infrared Spectroscopy (fNIRS) to measure changes of tissue oxygenation in the prefrontal cortex (PFC) and Support Vector Machine (SVM) to encode these changes into degrees of busyness.

As a proof-of-concept, our user study simulated a collaborative environment, in which the user works on an on-screen, memory intensive primary task, while the computer works in parallel on a different task. During the user's primary task, the computer, presumably in need of assistance, interrupts the user to introduce a secondary (interruption) task. When triggering an interruption, the computer selects one of two interruption strategies: one forces the user to immediately perform the interruption task, and the other allows the user to start handling of the interruption at the time of their choosing.

To estimate the maximum potential of our approach within a single user study, we simulated and used a perfect human model, in which the difficulty level of the user's assigned task at the moment was treated as a proxy measure of the user's busyness at that moment. At the same time, our system encoded its fNIRS measurements of the user's brain into degrees of busyness in real time and logged them as the performance of the actual HumanSketch model, which we used for later analyses.

We measured user performance of both the primary memory task and the interruption task, including interruption and resumption lags. We also collected subjective ratings of annoyance and respect [1] in terms of the computer's strategies, in addition to, task workloads assessed with NASA-TLX [12].

The main contributions of this paper are: (1) creation of a prototype system that leverages the user's momentary busyness to select one of two interruption strategies; (2) a user study measuring its best case performance with perfect measurement of user state; and (3) an experimental evaluation of our ability to approximate such measurements on actual tasks in real time.

2 RELATED WORK

Past work has explored the effects that interruptions have on a user's ability to complete tasks, and developed context-aware interruption systems to deliver interruptions at opportune times.

Effects of Interruptions. McFarlane and Latorella [25] identified the importance of interfaces handling interruptions effectively as a key challenge in HCI, arguing that as human-computer interfaces shift towards delegation and supervision, designing system to intelligently interrupt a user will become increasingly important. Brumby et al. [7] showed that interruptions can have a detrimental effect on a user's performance; after being interrupted, users tended to make more errors on their resumed task. However, they showed that this effect was mitigated by a forced time lag after the interruption; users resumed the task slowly and therefore were less error-prone post-interruption. Users also tend to be slower to complete interrupted tasks, especially cognitively complex ones [6, 8].

Kushlev et al. [22] showed that high levels of smartphone notifications correlated with self-reported hyperactivity and attention deficit, suggesting that interruptions can spread a user's attention too thin. Iqbal and Horvitz [19] found in a field-study with a system having only on/off interruptions that users with no interruptions (off) self-interrupted in order to seek out new emails or view notifications.

Past work suggests that interruptions can be damaging to a user's work flow, yet are also important for the user to feel aware of what is going on around them. We expand this work by presenting a redesigned interruption management system that is aware of a user's busyness level, allowing interruptions in appropriate manners.

Context-Aware Notification System. Some researchers have developed interruption systems that intelligently use context information to balance notifications with other tasks. InterruptMe [30] is a mobile application that leverages a user's context, such as location, time of day, and activity to deliver notifications at the least disruptive time. Users reported higher satisfaction and faster response times compared to a context unaware system. Interruption systems that interrupt the user at identified breakpoints in a workflow are also more effective at reducing user frustration [1, 18].

Bounded deferrals [13] provide another example of a context-aware interruption system, where users can defer incoming notifications for a specified amount of time, allowing for minimal distraction to handle notifications and specify it for a less busy time. Another system, PRIORITIES [14], balances contextual cost of interruption and criticality of email. The system determines these costs via analysis of user activity and content of messages. Automatic email delivery mediation systems [21] aimed at notifying users at times of lower workload, resulting in email being less disruptive to users when using the system. BusyBody [15] is another example of a system that can predict a user's interruptibility during a certain task.

Yuan et al. [37] argued that interruption systems should have more gradations than just interruptible or not and introduced an interruption model for smart phone users using personality traits in their user model (but not measured user state) to predict how busy–and therefore how interruptible–a user was.

Many of the above systems rely on information not intrinsic to the user's cognitive load and attention, but instead focus on task information. Systems could be more aware of the user using sensing technologies. For example, PAUI [10] uses physiological sensors (EEG, HRV) to measure user's mental load to modulate interruptions. Our work extends these context-aware interruption systems by showing the viability of brain sensing technology as an input modality for determining how a computer interrupts its user.

Brain sensing technology in HCI research. Brain sensing technology has improved to the point where it can be used in managing interruptions. While several techniques, particularly EEG can be used, we focus on functional Near-Infrared Spectroscopy (fNIRS) in this paper. For example, Solovey et al. [33] built an interface that detected and adapted to a user's mental state of multi-tasking using fNIRS, showing that these mental states could not be measured

by conventional means such as response time or keystrokes. CARSON [28] used both brain measurement and the importance of a message to find the best timing for its delivery. Additionally, Phylter [2, 32] demonstrates the use of fNIRS to deliver interruptions in an auxiliary display.

Brain sensing has been demonstrated successfully in a variety of other experimental real-time HCI settings as well, such as dynamically adjusting task difficulty [3], on-screen target expansion [4], anger detection [5], and learning [38]. fNIRS has also been used for evaluating a visualization interface [29] and for studying the effects of web form layout [23].

We expand this work by focusing on a user's busyness level inferred from measurements of fNIRS, making it a potentially effective input modality for intelligent interruption systems.

3 IMPLICIT DIALOGUE INJECTION SYSTEM

We learn from anecdotal experiences that people often use the five senses to coordinate their actions when communicating with others. Therefore, we consider a key to making a computer well-behaved is to give the computer the ability to sense its user. It will become increasingly crucial to find appropriate behaviors of computers as we approach an era in which humans work with AI and more intelligent computers.

3.1 System Architecture

To begin working towards coordinating computers' behaviors, we developed the *implicit dialogue injection* system, in which the computer is granted access to measurements of the user's momentary state; therefore, it now can select its behavior accordingly.

Figure 1 illustrates the workflow of the implicit dialogue injection system. The system consists of two key concepts: (1) *HumanSketch*, a conceptual human model that captures a user's momentary state inferred from physiological measurements of the user, and (2) *implicit dialogue*, a form of query-response message exchanges between the frontend computer, serving a human-facing program, and the HumanSketch. The system architecture follows the Command-Query Separation principle (CQS) [26], in which a command updates a HumanSketch and a query, issued to the HumanSketch via an implicit dialogue, allows the frontend computer to perceive the state of the user, used to coordinate its behavior.

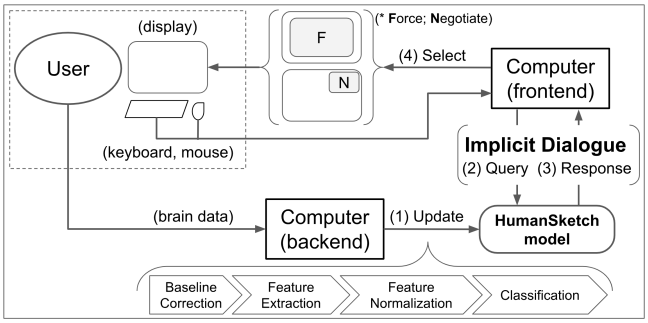

Figure 1: The workflow of the implicit dialogue injection system: The backend computer keeps taking in brain signals and (1) updating HumanSketch. When the frontend computer needs to interrupt the user, the computer injects an implicit dialogue which (2) queries and (3) obtains a response from HumanSketch. Lastly, considering the response (busyness), the frontend computer (4) selects its interruption strategy (Force or Negotiate; See also Apparatus).

Unlike conventional explicit dialogue conversations (e.g. a computer responds to user's mouse clicks), an injection of the implicit dialogue conversation can take place at any moment without the user explicitly getting involved in the process. We thus claim that our system is a realization of "implicit interactions" [20] specialized for the case of a computer communicating with a human user.

3.2 Implementation

We use our implicit dialogue injection system as the foundation upon which to implement an interruption management technique. Our implementation involved two separate software applications, frontend and backend, each of which corresponds to one of the two computers shown in Figure 1. The frontend application, using HTML and JavaScript, served as the human-facing program (See Apparatus). The backend application, written in Java, contained a HumanSketch model and was responsible for updating it with measurements.

We used functional Near-Infrared Spectroscopy (fNIRS), measuring changes of tissue oxygenation in the prefrontal cortex (PFC), and combined with a Support Vector Machine (SVM) to encode the changes into the degree of busyness. The encoding was our attempt to create a mapping from a state of probed region relating to cerebral activities [11, 16] to busyness (busy or not).

3.2.1 Updating HumanSketch. The bottom part of Figure 1 shows the signal processing pipeline to update the HumanSketch. The fNIRS device (Imagent from ISS Inc., Champaign, IL) and its accompanying software (Boxy) measured changes in oxygenated and deoxygenated hemoglobin concentration, HbO and Hb (in μM) respectively, at the probed region (PFC), and sent them to our backend application.

We used two custom forehead probes, each of which arranged the four light sources and one detector. The source-detector distances were 0.8 (cm) for one pair and 3.0 for the other three. At each source, two light wavelengths, 690 and 830 (nm), were emitted. A low-pass filter of 0.5 (Hz) was applied to remove high frequency noises mainly due to motion artifacts. The total 16 data channels (i.e. 8 per HbO and Hb) were sampled at 17.36 (Hz); we used the 12 channels corresponding to the 3 cm source-distance pairs.

A baseline correction was applied to each channel to find relative changes from the baseline mean. Using a 30-sec moving window frame, chosen to accommodate the relatively slow changes in fNIRS measurement, the feature extraction process was performed approximately twice per second. Within the process, the mean and linear regression slope of the second half of the window frame per channel were calculated, and were treated as the features of an instance for a machine learning classifier. This process was our attempt to capture trend changes in the time series data. Feature normalization, using the Min-Max scaler, was applied to each feature, whereas the scale range was determined at the calibration phase (see Procedure). The normalized instances were fed to a model (LIBSVM, version 3.21 [9]

with the linear kernel), and the classification result, busy or not busy, was used to update the HumanSketch.

In training the SVM model at the calibration phase, the grid search [17] was performed to find the hyperparameter (C) for the SVM. For each search, 5-fold cross validation was performed 5 times and its average was used to mitigate effects on randomly selecting training and validation dataset within this process. The SVM model was trained per participant; the computation to train a model took a couple of minutes beyond the actual training data collection.

3.2.2 Querying HumanSketch. We followed the server-client model. Our frontend application initiated a connection and queried our backend. Upon receiving a query, the backend application responded with a string message "busy" or "not busy" in the JSON format. The message exchanges were performed over the TCP/IP protocol.

4 METHOD

The main objective of the user study was twofold: (1) find effects of modulating a computer's strategy in making interruptions based on a user's busyness; and (2) evaluate our ability to infer the busyness in real time from brain measurements. To understand the maximum potential of our approach, we used a simulation of a perfect human model in our deployed system, in which the 'known' difficulty of the primary task was considered as the user's busyness. Meanwhile, the deployed system kept updating the actual HumanSketch model in real time and logged the inferred user's busyness every time the computer needed to interrupt the user.

As a proof-of-concept, we created a stripped down scenario where a user and computer collaborate to achieve a shared goal. The computer sometimes needed assistance from the user to complete its task, which resulted in an interruption on the user-assigned task. We used an on-screen, memory-intensive card matching task as the user's primary task. The task is a proxy for a range of common computer usage in a real world setting, but provides us sufficient control for the user study. We introduced two task difficulty levels, Easy and Hard, to modulate the user's busyness, and chose to examine two types of interruption strategies, Force and Negotiate.

In our experiment, each session consisted of two phases: the calibration phase for training the supervised machine learning model, followed by the experiment phase in which the main scenario described above took place.

4.1 Participants

We ran our experiment on 10 participants (4 female) aged between 18 and 35 (mean 23.2, SD 5.0). All participants were recruited via flyers posted on a university campus, were right-handed, had normal or corrected-to-normal vision and spoke English at a native level. All were paid US $10 per hour of the experiment duration in addition to a potential performance bonus of US $5. All the experiments were conducted in a quiet laboratory space, and each took approximately 90 minutes.

4.2 Apparatus

A stationary desktop computer and 20-inch widescreen monitor were used, and the participants performed tasks using a regular mouse and keyboard.

4.2.1 Memory Task. Figure 2 shows the interface of the memory task, which is an engaging card matching game, similar to Tasse et al. [34]'s work. The initial board state was a 4×5 grid of cards placed face down. On card click, the selected card was flipped to reveal an English alphabet letter. On click of another card, that card was also flipped to reveal its letter. If the two letters were the same, the two flipped cards were removed from the board, indicating success. If not, the cards were flipped back over after a brief constant delay. The task window also contained a timer showing remaining time for a given trial above the interactive game board.

Difficulty: We designed two versions of the memory task to vary difficulty. In Easy version (Figure 2a), there were guides to indicate a matching pair (no alphabet letter was shown), so the user simply had to flip the highlighted cards to obtain a guaranteed match. In Hard version (Figure 2b), there were no guides, so the user had to memorize and recall previously revealed letters to find a match.

The interruption task, presented in the middle of each memory task trial, consisted of a request to enter the solution to a distorted math problem in CAPTCHA [35] form (Figure 2c). This emulated a type of problem which is much more efficiently tackled by humans than computers. The difficulty level of math problems (i.e. adding two 3-digit numbers) was kept the same across all conditions. This interruption task is of type IRC 110 [24], because it highly interrupts the memory task, requires a rapid and accurate response, and is completely separated from the memory task.

Interruption Strategy: We implemented two types of interruption strategies. In Force strategy (Figure 2c), the interruption task window popped up and completely occluded the entire memory task window, so the user was forced to complete the interruption task before returning to the memory task. Conversely, in Negotiate strategy (Figure 2d), a small pop-up window appeared next to the game board to notify the user that the computer required assistance; the user could thus still work on the memory task and start the interruption task at their convenience by accepting the notification prompt.

4.2.2 N-Back Task. In the calibration phase, the user performed the n-back task, and measurements of the user's brain (fNIRS data) were used to train the SVM model. The n-back task, well-studied in psychology [27], consists of identifying whether a given stimulus matches the n-th stimulus back; increases in n correlate with increases in workload. The task has been successfully used in previous studies to classify workload using fNIRS data [3, 29].

We used a visuospatial 2-back and 0-back in the present study. The task window frame presented a 3×3 square grid with 'yes' and 'no' buttons in the center grid. Every couple of seconds, one of the 8 grids flashed to black, and in 2-back, the user clicked 'yes' if the location of the black grid matched the black grid from 2 flashes before and 'no' otherwise. In 0-back, the user simply observed the grids flashing black and clicked 'yes' each time. The point was to let the computer learn whether the user is in a busy state (with 2-back) or in a not busy state (with 0-back).

4.3 Procedure

Upon arriving, each participant filled out a consent form and a set of demographic questions. Participants were provided with detailed task instructions and practiced using tutorial versions of both the

Figure 2: Memory task board states. (a) Easy difficulty with guides highlighting a guaranteed pair match. (b) Hard difficulty, no guides, the user has just flipped over the "I". (c) Force strategy obscures the board until CAPTCHA problem is answered. (d) Negotiate strategy presents a pop-up next to the board asking the user for permission to display the CAPTCHA problem.

n-back and memory tasks to ensure there was no confusion over the tasks' objectives or interface. After a short break, the fNIRS probes were placed on the participant's forehead using cotton headbands, and the room lights were turned off, with only ambient lighting left on to reduce interference with the probe detectors. During the experiment, participants were instructed to limit head movements to avoid disrupting the data recording.

In the calibration phase, the n-back task began with a 3-minute baseline in which participants were instructed to close their eyes and clear their mind. After the baseline, a total of 30 trials (15 per 2- and 0-back) was presented in randomized order. Each trial lasted 30 seconds, and there was a 30-second break before each trial.

In the experiment phase, the memory task also began with a 3-minute baseline, followed by a total of 4 trials (one per condition; See Design). Each trial was preceded by a 30-second break and succeeded by the NASA-TLX post-questionnaire. Each trial lasted 60 seconds in total. The duration from the beginning of the memory task trial to the moment when the computer triggers an interruption was designed to be identical to the duration of the n-back trial, 30 sec. The timer paused during the interruption task. Participants were instructed to find as many matching card pairs, as quickly and accurately as possible within the allotted time. To mitigate learning effects across conditions, the trial orders were predetermined, and each order was assigned to at least one and at most two participants.

4.4 Design

The study used a within-subjects design with two independent factors, each of which had two levels: difficulty (Easy and Hard) and strategy (Force and Negotiate), resulting in 4 (2×2) conditions.

The dependent variables were: **(a)** Participant responses to six NASA-TLX categories, evaluating overall experience of the task. **(b)** Participant ratings of whether they felt annoyed at or respected by the computer's strategy in making the interruption. The terms annoyance and respect were modeled after measurements by Adamczyk and Bailey [1] and referred to the question, "how (annoyed | respected) were you by the interruption in accomplishing what you were asked to do?" **(c)** Performance measurements from the interruption task, including time completion, interruption lag and resumption lag (measured in milliseconds). The interruption lag was defined as the time it took the participant to start the interruption task after it was introduced. The resumption lag was defined as time from completing the interruption task to the next card selection of the memory task. **(d)** Performance measurements from the memory task, including the number of correct matching card pairs and time taken to find the first correct pair after interruption (ms).

In addition, we recorded the response from the actual HumanSketch model, estimating the participant's busyness in real time, each time when an interruption was triggered.

5 RESULTS

We report analysis on 40 data points collected (i.e., 10 participants × 2 difficulties × 2 strategies). We applied the Aligned Rank Transform (ART) [36] to nonparametric data to test statistical significances using ANOVA. We used the Wilcoxon signed rank test for post-hoc cross-factor pairwise comparisons, unless otherwise specified. For space the condition names are sometimes abbreviated: E = Easy, H = Hard, F = Force, and N = Negotiate.

5.1 (a) Task Workload

Table 1 shows participant responses to the six NASA-TLX [12] categories. The grand mean for the mental demand was 7.35. Noticeably, Hard difficulty (mean 11.6) was more mentally demanding than Easy (mean 3.15). On the other hand, there was a modest difference between the two strategies within each difficulty. Using an ANOVA on the aligned rank transformed data, the main effect of difficulty was statistically significant ($F_{1,9} = 58.718, p < .001$). Post-hoc tests using Holm correction showed significant differences between Easy-Force and Hard-Force as well as between Easy-Negotiate and Hard-Negotiate. As expected, no significant difference was found between Easy-Force and Easy-Negotiate as well as between Hard-Force and Hard-Negotiate.

The grand mean for the physical demand was 2.85. As Table 1 shows, there were modest differences among the four conditions. An ANOVA with ART reported that the main effect of difficulty on physical demand was not statistically significant.

Table 1 also shows statistical analysis for the other categories. In short, no significant difference on strategy or interaction was found for the categories. In addition, no significant difference between the two strategies within each difficulty in the other categories was found.

Thus, our analysis of the participants' overall experiences suggest that our experiment control succeeded in creating an environment in which mental demand varied based on the difficulty, while physical demand was kept consistent across all conditions.

Table 1: Mean responses of task workload index

	Easy		Hard		Effect F-val. (p)	
Category	**F**	**N**	**F**	**N**	**Difficulty**	**Strategy**
Mental Demand	3.4	2.9	11.8	11.3	58.7 **(<.001)**	1.12 (.32)
Physical Demand	2.5	2.3	3.1	3.5	2.93 (.12)	0.11 (.75)
Temporal Demand	7.0	5.4	12.1	11.1	14.4 **(.004)**	5.04 (.051)
Performance	17.5	19.0	14.2	13.5	15.7 **(.003)**	0.13 (.72)
Effort	6.7	6.8	11.9	11.6	6.04 **(.036)**	0.07 (.79)
Frustration	5.2	3.7	9.3	8.9	7.31 **(.024)**	0.95 (.36)

Note: F = Force, N = Negotiate, Response range: 1-21, lower is less.

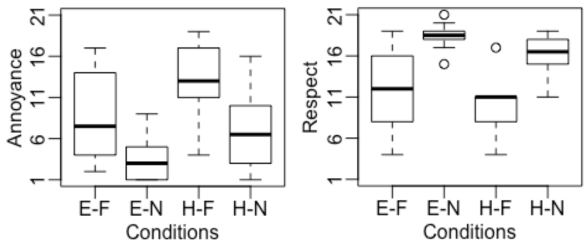

Figure 3: Ratings of annoyance and respect. The thick line in each boxplot represents the median value.

5.2 (b) Annoyance and Respect

Figure 3 shows the ratings of whether the participants felt annoyed at or respected by the interruption. Recall that the two items were chosen to specifically understand effects of the computer's strategy in making an interruption. A 21-point scale of range 1(very low) - 21(very high) was used for annoyance and 1(failure) - 21(perfect) for respect. We consider lower values for annoyance and higher values for respect to indicate that the computer's strategy was more considerate of the participants.

The grand mean for annoyance was 8.08. Hard-Force (12.8) was the most annoying, followed by Easy-Force (8.7), Hard-Negotiate (7.2) and Easy-Negotiate (3.6). In fact, 19 out of the 20 total pair data points showed Force strategy was more annoying than Negotiate. An ANOVA with ART revealed a significant effect of the strategy on annoyance data ($F_{1,9} = 20.697, p < .01$). Post hoc tests using Holm correction reported significant differences between Easy-Force and Easy-Negotiate as well as between Hard-Force and Hard-Negotiate. The effect of the difficulty ($F_{1,9} = 4.3837, p > .05$) and interaction ($F_{1,9} = 0.42509, ns$) were not significant.

The grand mean for respect was 14.08. As Figure 3 depicts, Negotiate strategy (mean 17.2) was rated as more respectful than Force (mean 11.0). An ANOVA with ART showed that the main effects of difficulty ($F_{1,9} = 12.703, p < .01$) and of strategy ($F_{1,9} = 45.312, p < .001$) were statistically significant, while the interaction ($F_{1,9} = 0.0027, ns$) was not. Post-hoc tests using Holm correction reported significant differences between Easy-Force and Easy-Negotiate as well as between Hard-Force and Hard-Negotiate.

When designing our system, we hypothesized that the interruption could be less annoying while the user is being less mentally taxed. However, the participant ratings seem to highlight a fact that Negotiate strategy tends to be less annoying and appear more respectful than Force strategy in both difficulties.

Table 2: Average time of the interruption task

	Easy		Hard		Effect F-val. (p)	
Metrics (sec)	**F**	**N**	**F**	**N**	**Difficulty**	**Strategy**
Time completion	19.2	17.4	17.9	14.6	0.46 (.51)	3.35 (.10)
Interruption lag	0.01	2.20	0.01	3.84	3.67 (.09)	28.2 **(<.001)**
Resumption lag	0.98	0.92	1.12	1.10	9.45 **(.01)**	0.78 (.40)

Note: F = Force, N = Negotiate

5.3 (c) Measurements from Interruption Task

Table 2 summarizes the three measurements from the interruption task, used to understand impacts of interruption. Note that the participants correctly solved all the math problems presented as the interruption task.

The completion time in Hard-Negotiate condition seems to be shorter than the others; however, using an ANOVA, the main effect of difficulty and of strategy were not statistically significant. The average interruption lag of Negotiate condition was longer than of Force condition in each difficulty, reflecting a part of our experimental controls. Unsurprisingly, an ANOVA reported that the effect of the strategy was statistically significant, as shown in Table 2. Post-hoc tests, cross-factor pairwise comparisons using paired t-test with Holm correction, showed that the differences between Easy-Force and Easy-Negotiate as well as between Hard-Force and Hard-Negotiate are significant. The average resumption lag time of each condition, shown in Table 2, was in a narrow range. An ANOVA reported that the main effect of the difficulty was statistically significant; however, post-hoc comparisons did not find any significant difference, likely because of the small sample size (see Discussion). Finally, none of the interaction effects were found to be significant.

5.4 (d) Measurements from Memory Task

For space we focus on reporting post-interruption measurements and effects of the strategy (Force vs. Negotiate) within each difficulty (Easy or Hard).

Figure 4 (Left) represents the average time taken to find the first card match pair after the interruption. Contrary to the resumption lag, there exists a measurable difference between Easy (mean 1.46 sec) and Hard (mean 6.22 sec) difficulty. Using an ANOVA, we found that the main effect of difficulty ($F_{1,9} = 42.97, p < .001$) was statistically significant. Post-hoc tests, cross-factor pairwise comparisons using paired t-test with Holm correction, showed that differences between Easy-Force and Hard-Force as well as between

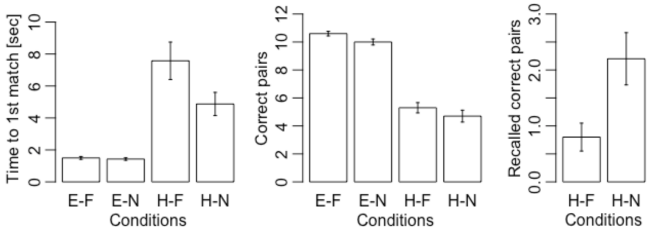

Figure 4: Measurements of the memory task after the interruption. The error bars represent standard errors.

Easy-Negotiate and Hard-Negotiate were statistically significant. The main effect of strategy ($F_{1,9} = 4.33, p > .05$) (although $p = .0672$) and interaction ($F_{1,9} = 3.779, p > .05$) ($p = .0838$) was not statistically significant at $\alpha = .05$ level. However, on average within Hard difficulty the participants took longer time to find the first correct matching card pair after interruption with Force strategy (mean 7.58 sec) than with Negotiate strategy (mean 4.87 sec).

As such, we further investigated our logs by introducing an additional pair count category - recalled correct pairs, which corresponded to the number of correct pairs that were selected by recalling cards that had been revealed before the interruption. As a result, in Hard-Negotiate condition, 9 out of the 10 participants recalled a card previously revealed before the interruption in their first correct card matching pair since the interruption. On the other hand, in Hard-Force condition, only 3 out of the 10 did the same. We think this explains why the participants were able to find the first card matching pair faster in Hard-Negotiate condition than in Hard-Force condition.

Figure 4 (Middle and Right) shows the number of correct pairs and recalled correct pairs out of the total correct pairs found after the interruption. Note that with our task design, the concept of recall is only applicable to the conditions involving Hard difficulty. On average, the participants were able to recall more in Hard-Negotiate condition (2.2 card pairs) than in Hard-Force condition (0.8 card pairs). With Shapiro-Wilk test, the normality of the data was not assumed. A Wilcoxon signed rank test showed that there is a significant effect of strategy ($W = 0, Z = -2.563, p < .05, r = .573$).

Altogether, it appears that Negotiate strategy affected the participants' tactics on finding matching pairs. The participants utilized their memory from before the interruption in order to find matching pairs after the interruption. The results indicate that Negotiate strategy increased recall memory performance.

5.5 Performance of HumanSketch Model

We lastly report measurements of our ability to infer the user's state of busyness in real time with brain measurements. Our system used machine learning (See System Architecture); thus, we report its classification accuracy as the performance of HumanSketch, i.e., how well our system measured the user's busyness. Note that a machine learning model was trained per participant.

Figure 5 (Left) represents the average cross validation accuracy (%) of the n-back (calibration) task and the average real time classification accuracy (%) of the memory task using the trained model. The true class labels of the calibration task were 2-back (more memory intensive) and 0-back (less memory intensive). Thus, we assumed that the trained model is applicable to classify the instances of the memory task, in which the difficulty levels, Hard and Easy, serve as true class label.

As noticed, the accuracy for the calibration task (mean 83.3%, median 83.3%) was higher than that for the memory task (mean 62.5%, median 75.0%). We expect this is due to levels of control we were able to give to each task. The calibration task was a standard experimental task known to modulate memory usage, while the memory task was a stripped down version of a potential real world task. Nevertheless, Figure 5 (Right) shows the breakdown, and an encouraging fact is that accuracy for the memory task for 6 of the 10 subjects was equal to or higher than 75%. (See also Discussion).

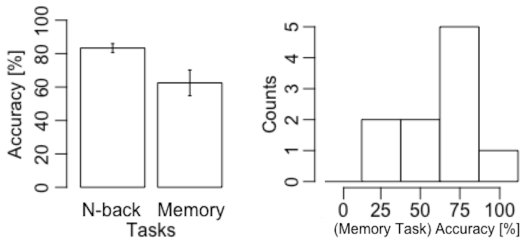

Figure 5: (Left) Classification accuracy of the n-back and memory task. The error bar represents standard error. (Right) Histogram showing the distribution of the classification accuracy of the memory task.

6 DISCUSSION AND FUTURE WORK

Design Guideline. Using our interpolation of the data, we discuss design choices for a computer's behavior in interrupting its user. Our observations were as follows. Unsurprisingly, if the context of the interruption task requires the user's immediate attention, then the computer had better choose Force interruption strategy than Negotiate. (Table 2, Interruption lag) On the other hand, Force strategy appeared to be more annoying and less respectful than Negotiate. Therefore, if we aim to maximize user experience, then the computer had better choose Negotiate over Force. (Figure 3) In addition, within Hard difficulty, Negotiate strategy resulted in more recalled correct pairs than Force, so Force affected the user's memory more than Negotiate. Therefore, if we aim to maximize for user's performance while the user is busier, then the computer had better choose Negotiate over Force. (Figure 4, Right)

Altogether, we suggest that when the system recognizes that a user is busier, it should choose a negotiated interruption, which provides an appropriate compromise between user experience and performance. On the other hand, when it recognizes that a user is not busy, it may choose a forced interruption to avoid the user actions for switching tasks (e.g. one less mouse click per interruption), though at a cost of increased annoyance, giving designers a trade-off for efficiency.

Limitations. We list limitations in our user study, which provide directions for future work. (1) Our sample size is relatively small ($N = 10$); increasing it using an expanded participant pool will help to increase the internal validity of our study. (2) Our experiment was conducted in a controlled laboratory environment and tested only the on-screen visual interruption in the memory task. Exploring other interruption modalities (e.g. audio, haptic, or even a combination of them) in real world scenarios will help to provide more extensive design guidelines. (3) It would also be helpful to measure user acceptance of the fNIRS technology in order to assess the ecological validity of the study. (4) Our approach used supervised machine learning. While it still remains challenging to collect a sufficient number of quality training data within a reasonable experiment duration, a larger training dataset is expected to increase robustness and to help to understand user's states more precisely. (5) Although we focused on using our human model to design the computer's behaviors, integrating the human model and with a task model based on the user's explicit interactions would yield a larger pool of possible computer behaviors.

Implications of Implicit Dialogue Injection System. This paper focused only on a state of busyness and interruption strategy. We now point to the extensibility of our system architecture, which incorporates a human model into interaction designs by providing a clear separation between how to understand a user's state and how to leverage such understandings. In concept, a 'perfect' HumanSketch model is an oracle that has perfect awareness of any potential facet of the user. Although a creation of such an oracle may be not yet possible in practice, introducing and combining different kinds of physiological sensors could help us to create more robust, sophisticated HumanSketch models, exposing a variety of complex human states in a form that computers can perceive. To close, we bring up an open-ended question to interaction designers: If you have access to such a perfect HumanSketch model (oracle), what kinds of "implicit interactions" [20] would you design?

7 CONCLUSION

It is unlikely to be possible to return to an era of fewer interruptions as computer technologies keep growing. To accommodate the inevitability of interruptions that computers generate, we have introduced the implicit dialogue injection system. It envisions a way to enable a computer to recognize its user so it can act in a more human-like fashion. Our proof-of-concept study, testing the two types of interruption strategies with a consideration of the user's busyness level, demonstrated that the presentation of interruptions affects both the user's performance and experience. From this, we formed a design guideline for computer behavior in making interruptions based on the user's state. Finally, we discussed the state-of-the-art of our system on estimating user's state as well as extensions for future work. We hope the architecture and concepts in our proposed system can contribute to improving future human-computer collaborations by making the computer more considerate of its user without sacrificing performance.

ACKNOWLEDGMENTS

We would like to thank Kristen Tgavalekos, Sam Hincks, Thao Pham, Angelo Sassaroli, Sergio Fantini, and Remco Chang from Tufts University for their considerable help on configuring the equipment and designing the experiment. We thank Google Inc. for support of this research.

REFERENCES

[1] Piotr D. Adamczyk and Brian P. Bailey. 2004. If Not Now, when?: The Effects of Interruption at Different Moments Within Task Execution. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '04)*. ACM, New York, NY, USA, 271–278. https://doi.org/10.1145/985692.985727

[2] Daniel Afergan, Samuel W. Hincks, Tomoki Shibata, and Robert J. K. Jacob. 2015. Phylter: A System for Modulating Notifications in Wearables Using Physiological Sensing. In *Foundations of Augmented Cognition*. Springer International Publishing, Cham, 167–177.

[3] Daniel Afergan, Evan M. Peck, Erin T. Solovey, Andrew Jenkins, Samuel W. Hincks, Eli T. Brown, Remco Chang, and Robert J.K. Jacob. 2014. Dynamic Difficulty Using Brain Metrics of Workload. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 3797–3806. https://doi.org/10.1145/2556288.2557230

[4] Daniel Afergan, Tomoki Shibata, Samuel W. Hincks, Evan M. Peck, Beste F. Yuksel, Remco Chang, and Robert J.K. Jacob. 2014. Brain-based Target Expansion. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. ACM, New York, NY, USA, 583–593. https://doi.org/10.1145/2642918.2647414

[5] Gabor Aranyi, Fred Charles, and Marc Cavazza. 2015. Anger-based BCI Using fNIRS Neurofeedback. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. ACM, New York, NY, USA, 511–521. https://doi.org/10.1145/2807442.2807447

[6] Brian P Bailey, Joseph A Konstan, and John V Carlis. 2000. Measuring the effects of interruptions on task performance in the user interface. In *Systems, Man, and Cybernetics, 2000 IEEE International Conference on*, Vol. 2. IEEE, 757–762.

[7] Duncan P Brumby, Anna L Cox, Jonathan Back, and Sandy JJ Gould. 2013. Recovering from an interruption: Investigating speed- accuracy trade-offs in task resumption behavior. *Journal of Experimental Psychology: Applied* 19, 2 (2013), 95.

[8] Ivan Burmistrov and Anna Leonova. 2003. Do interrupted users work faster or slower? The microanalysis of computerized text editing task. *Human-Computer Interaction: Theory and Practice (Part I)–Proceedings of HCI International* 1 (2003), 621–625.

[9] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A Library for Support Vector Machines. *ACM Trans. Intell. Syst. Technol.* 2, 3, Article 27 (May 2011), 27 pages. https://doi.org/10.1145/1961189.1961199

[10] Daniel Chen and Roel Vertegaal. 2004. Using Mental Load for Managing Interruptions in Physiologically Attentive User Interfaces. In *CHI '04 Extended Abstracts on Human Factors in Computing Systems (CHI EA '04)*. ACM, New York, NY, USA, 1513–1516. https://doi.org/10.1145/985921.986103

[11] Peter T Fox, Marcus E Raichle, Mark A Mintun, and Carmen Dence. 1988. Nonoxidative glucose consumption during focal physiologic neural activity. *Science* 241, 4864 (1988), 462–464.

[12] Sandra G. Hart. 2006. Nasa-Task Load Index (NASA-TLX); 20 Years Later. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 50, 9 (2006), 904–908. https://doi.org/10.1177/154193120605000909

[13] Eric Horvitz, Johnson Apacible, and Muru Subramani. 2005. Balancing Awareness and Interruption: Investigation of Notification Deferral Policies. In *User Modeling 2005*. Springer Berlin Heidelberg, Berlin, Heidelberg, 433–437.

[14] Eric Horvitz, Andy Jacobs, and David Hovel. 1999. Attention-sensitive Alerting. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI'99)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 305–313. http://dl.acm.org/citation.cfm?id=2073796.2073831

[15] Eric Horvitz, Paul Koch, and Johnson Apacible. 2004. BusyBody: Creating and Fielding Personalized Models of the Cost of Interruption. In *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work (CSCW '04)*. ACM, New York, NY, USA, 507–510. https://doi.org/10.1145/1031607.1031690

[16] Yoko Hoshi and Mamoru Tamura. 1993. Detection of dynamic changes in cerebral oxygenation coupled to neuronal function during mental work in man. *Neuroscience Letters* 150, 1 (1993), 5 – 8. https://doi.org/10.1016/0304-3940(93)90094-2

[17] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. 2003. A practical guide to support vector classification. (2003).

[18] Shamsi T. Iqbal and Brian P. Bailey. 2008. Effects of Intelligent Notification Management on Users and Their Tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. ACM, New York, NY, USA, 93–102. https://doi.org/10.1145/1357054.1357070

[19] Shamsi T. Iqbal and Eric Horvitz. 2010. Notifications and Awareness: A Field Study of Alert Usage and Preferences. In *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work (CSCW '10)*. ACM, New York, NY, USA, 27–30. https://doi.org/10.1145/1718918.1718926

[20] Wendy Ju. 2015. The design of implicit interactions. *Synthesis Lectures on Human-Centered Informatics* 8, 2 (2015), 1–93.

[21] Yasumasa Kobayashi, Takahiro Tanaka, Kazuaki Aoki, and Kinya Fujita. 2015. Automatic Delivery Timing Control of Incoming Email Based on User Interruptibility. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '15)*. ACM, New York, NY, USA, 1779–1784. https://doi.org/10.1145/2702613.2732825

[22] Kostadin Kushlev, Jason Proulx, and Elizabeth W. Dunn. 2016. "Silence Your Phones": Smartphone Notifications Increase Inattention and Hyperactivity Symptoms. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 1011–1020. https://doi.org/10.1145/2858036.2858359

[23] Kristiyan Lukanov, Horia A. Maior, and Max L. Wilson. 2016. Using fNIRS in Usability Testing: Understanding the Effect of Web Form Layout on Mental Workload. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 4011–4016. https://doi.org/10.1145/2858036.2858236

[24] D. Scott McCrickard, C. M. Chewar, Jacob P. Somervell, and Ali Ndiwalana. 2003. A Model for Notification Systems Evaluation&Mdash;Assessing User Goals for Multitasking Activity. *ACM Trans. Comput.-Hum. Interact.* 10, 4 (Dec. 2003), 312–338. https://doi.org/10.1145/966930.966933

[25] Daniel C McFarlane and Kara A Latorella. 2002. The scope and importance of human interruption in human-computer interaction design. *Human-Computer Interaction* 17, 1 (2002), 1–61.

[26] Bertrand Meyer. 1988. *Object-oriented software construction*. Vol. 2. Prentice hall New York.

[27] Adrian M Owen, Kathryn M McMillan, Angela R Laird, and Ed Bullmore. 2005. N-back working memory paradigm: A meta-analysis of normative functional neuroimaging studies. *Human brain mapping* 25, 1 (2005), 46–59.

[28] Evan M. Peck, Emily Carlin, and Robert J. K. Jacob. 2015. Designing Brain-Computer Interfaces for Attention-Aware Systems. *Computer* 48, 10 (Oct 2015), 34–42. https://doi.org/10.1109/MC.2015.315

[29] Evan M. Peck, Beste F. Yuksel, Alvitta Ottley, Robert J.K. Jacob, and Remco Chang. 2013. Using fNIRS Brain Sensing to Evaluate Information Visualization Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 473–482. https://doi.org/10.1145/2470654.2470723

[30] Veljko Pejovic and Mirco Musolesi. 2014. InterruptMe: Designing Intelligent Prompting Mechanisms for Pervasive Applications. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '14)*. ACM, New York, NY, USA, 897–908. https://doi.org/10.1145/2632048.2632062

[31] Martin Pielot and Luz Rello. 2015. The Do Not Disturb Challenge: A Day Without Notifications. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '15)*. ACM, New York, NY, USA, 1761–1766. https://doi.org/10.1145/2702613.2732704

[32] Tomoki Shibata, Evan M. Peck, Daniel Afergan, Samuel W. Hincks, Beste F. Yuksel, and Robert J.K. Jacob. 2014. Building Implicit Interfaces for Wearable Computers with Physiological Inputs: Zero Shutter Camera and Phylter. In *Proceedings of the Adjunct Publication of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST'14 Adjunct)*. ACM, New York, NY, USA, 89–90. https://doi.org/10.1145/2658779.2658790

[33] Erin Treacy Solovey, Francine Lalooses, Krysta Chauncey, Douglas Weaver, Margarita Parasi, Matthias Scheutz, Angelo Sassaroli, Sergio Fantini, Paul Schermerhorn, Audrey Girouard, and Robert J.K. Jacob. 2011. Sensing Cognitive Multitasking for a Brain-based Adaptive User Interface. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 383–392. https://doi.org/10.1145/1978942.1978997

[34] Dan Tasse, Anupriya Ankolekar, and Joshua Hailpern. 2016. Getting Users' Attention in Web Apps in Likable, Minimally Annoying Ways. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 3324–3334. https://doi.org/10.1145/2858036.2858174

[35] Luis von Ahn, Manuel Blum, Nicholas J. Hopper, and John Langford. 2003. CAPTCHA: Using Hard AI Problems for Security. In *Advances in Cryptology — EUROCRYPT 2003*, Eli Biham (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 294–311.

[36] Jacob O. Wobbrock, Leah Findlater, Darren Gergle, and James J. Higgins. 2011. The Aligned Rank Transform for Nonparametric Factorial Analyses Using Only Anova Procedures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 143–146. https://doi.org/10.1145/1978942.1978963

[37] Fengpeng Yuan, Xianyi Gao, and Janne Lindqvist. 2017. How Busy Are You?: Predicting the Interruptibility Intensity of Mobile Users. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 5346–5360. https://doi.org/10.1145/3025453.3025946

[38] Beste F. Yuksel, Kurt B. Oleson, Lane Harrison, Evan M. Peck, Daniel Afergan, Remco Chang, and Robert JK Jacob. 2016. Learn Piano with BACh: An Adaptive Learning Interface That Adjusts Task Difficulty Based on Brain State. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 5372–5384. https://doi.org/10.1145/2858036.2858388