Predicting Interruptibility for Manual Data Collection: A Cluster-Based User Model

**Aku Visuri[1], Niels van Berkel[2], Chu Luo[2],
Jorge Goncalves[2], Denzil Ferreira[1], Vassilis Kostakos[2]**
[1]Center for Ubiquitous Computing, University of Oulu, [2]The University of Melbourne
[1]{first.last}@oulu.fi, [2]{n.vanberkel;chul3;jorge.goncalves;vassilis.kostakos}@unimelb.edu.au

ABSTRACT

Previous work suggests that Quantified-Self applications can retain long-term usage with motivational methods. These methods often require intermittent attention requests with manual data input. This may cause unnecessary burden to the user, leading to annoyance, frustration and possible application abandonment. We designed a novel method that uses on-screen alert dialogs to transform recurrent smartphone usage sessions into moments of data contributions and evaluate how accurately machine learning can reduce unintended interruptions. We collected sensor data from 48 participants during a 4-week long deployment and analysed how personal device usage can be considered in scheduling data inputs. We show that up to 81.7% of user interactions with the alert dialogs can be accurately predicted using user clusters, and up to 75.5% of unintended interruptions can be prevented and rescheduled. Our approach can be leveraged by applications that require self-reports on a frequent basis and may provide a better longitudinal QS experience.

Author Keywords
Smartphones; Self-reports; Quantified-self; Interruptibility

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

INTRODUCTION

Quantified-Self (QS) applications are often criticised for lacking sustained long-term use [17, 24, 45]. While most users of QS applications start with an inherent motivation for self-monitoring, their engagement tends to decrease over time and the applications are subsequently abandoned. User motivation can be positively affected with user-driven insights [4], but collecting sufficient amount of data to generate such insights can introduce a reliance on *manual* data collection [3]. Choe *et al.* [8] recently highlighted the frequent requirement of self-reports in self-monitoring applications. Fritz *et al.* [15] report that the higher the effort to use an application, the shorter the lifetime of the application. As QS relies on *continuous* tracking [42], the issues presented here are core challenges within this application space. More critically, applications' abandonment often leads to the users returning to their old habits in the end [24].

Bentley *et al.* [3] use Android notifications to remind users to log data and increase the quantity of logged data. Our approach extends this work by presenting opportunistically displayed on-screen alert dialogs, shown during active smartphone use. These dialogs enable the user to log data directly, and do not require the user to launch a separate application, which is often the case with notification reminders. Notifications have some drawbacks as they can be presented to user even when the user is not present, and can exist on the background. Dialogs and notifications differ as notifications do not require immediate interaction from the user. The requirement for interaction with a dialog can, however, be experienced as interrupting. We attempt to mitigate this side-effect by pre-emptively predicting their likelihood of interruption. Specifically, we infer user interruptibility by reconstructing the *in-situ* context of the user using multiple sensors on their smartphone in real time. The field of inferring human interruptibility using sensors is widely explored under the umbrella of interruptibility [14]: recognising opportune moments to triggering notifications [22, 35, 39], effect of interruptions on quality of logged data [30], as well as the effect of interruptions on interactions and task completion [27].

The reliance on continuous data logging inherently increases users' compliance effort and may overwhelm them if performed over extended periods of time. We propose a method to reduce the data logging burden. We base our method on the frequency and brief nature of the large majority of smartphone usage sessions [11, 41, 46], as well as the frequent non-task oriented nature of smartphone usage [6, 28, 38]. By transforming these recurrent moments of brief smartphone usage (*e.g.*, 'killing time' [6]) into data contributions for QS applications, the quantity of logged data increases while the effort associated with data input is reduced. We analyse different smartphone user usage patterns - frequency and duration of use [46, 47], and interaction styles [10] - to form groups of users with similar traits, improving the prediction accuracy of presenting alert dialogs in an uninterrupting fashion.

We present the findings of a 4-week long study measuring alert dialogs' interruptibility and applicability in a QS context. Specifically, we deploy a mobile QS application which provides on-screen alert dialogs as an input mechanism for data logging. A total of 48 participants used our application for life-logging and we measured their interactions and attitudes towards the alert dialogs. We conduct a post-study survey to gather qualitative feedback. We evaluate several machine-learning classifiers' overall performance and power to minimise interruptions. We demonstrate the feasibility of how machine learning classifiers can decrease the frequency of interrupting prompts with the use of on-device sensors that measure the surrounding context, and the context of device usage. By clustering users into groups, we improved classifiers' accuracy without a required training period. The use of frequently occurring input prompts can benefit quantified-self applications in enabling opportunistic self-reported data input, crucial for longitudinal use.

RELATED WORK

The field of interruptibility research aims to determine a user's availability, readiness, and interest in a given content element [14]. Social and behavioural cues allow humans to assess a person's level of interruptibility [2, 19]. In 2005, Fogarty *et al.* [14] found that relatively simple external sensors can be used to successfully construct a model on a person's interruptibility. However, these sensors must be installed in the environment of the user pre-emptively (*e.g.*, to detect if a conversation is taking place).

Untimely interruptions affect the user in various forms, *e.g.* increased cognitive and information overload [34], delayed task completion [27], or reduced quality of logged information [30]. Okoshi *et al.* [34] showcase two methods to successfully address the problem of disruptive interruptions: 1) rescheduling interruptions to arrive at another time, and 2) mitigating the number or frequency of such interruptions. Successful timing of interruptions has a positive effect on both time management and task efficiency [22].

Using machine learning to detect interruptibility has proven to be a successful method in the past. Interruptibility has been assessed *e.g.*, through analysing both mobile phone sensor data (*e.g.*, device posture [39], acceleration sensor [20]), wearable devices with embedded sensors [25], and high-level data such as application usage patterns or location data [35, 38, 39]. From select features, predictions can be made regarding the user's interruptibility, *e.g.*, using activity breakpoints [33], or rules [29] to decide *when* and *what kind of* push notifications to present to the user. A variety of machine learning classifiers exist to create a general users' interruptibility model: a) trees, such as Random Forests [5] or C4.5; b) Bayesian classifiers, such as Naïve Bayes; or c) Support Vector Machines (SVM), such as LibSVM. For these models, researchers select similar classifier features, *e.g.*, Poppinga *et al.* [39] explored the correlation between different smartphone sensors and user interruptibility, where time of day and screen coverage are good predictors. Pielot [36] analysed the availability of mobile phone users to accept incoming calls using a variety of contextual data (*e.g.*, physical activity, screen status, day of the week) in addition to similar hardware sensors.

Users have distinguishable traits of device usage in terms of session frequencies and durations [46, 47], application selection [48, 49], and interactions [10]. Higher-level contextual features are able to construct the user's personal preferences in detail. Pielot *et al.* [37] model the user's attentiveness to messages from instant messaging applications using past behavioural usage data from these applications. When analysing the user's level of boredom [38], additional measures are analysed to obtain a higher degree of user context: audio jack state, airplane mode status, network activity, screen orientation, and current foreground application. On the other hand, recent work has put more focus on modelling user groups to achieve higher accuracies. Zhao *et al.* [49] and Welke *et al.* [48] analysed application usage and inferred several distinct user types with identifiable characteristics.

Previous work has shown the disruptive nature of smartphone information push methods, such as notifications [40]. The distractive nature is relevant for those applications or services that aim to use push methods to collect user data - for example applications in the area of QS or experience sampling. Mehrotra *et al.* [30, p.1] highlight this challenging balance for ESM studies: "*obtaining high quality data with ESM is challenging, as users may fail to respond honestly, or may even ignore the questionnaire prompts if they perceive the study as too burdensome*". To address the issue of burdensome notifications, researchers have explored various potential solutions. Context inference is used to assess the user's interruptibility in both the current and future contexts to decide the best time for interruption in terms of quality of the content of the response [30]. Hsieh *et al.* [21] display feedback to participants following an ESM questionnaire to make the submission of data more personally relevant. Leiva *et al.* [27] explore preventive methods, such as preparing the user to leave the current task before the interruption, and ways to guide the user back to the task by *e.g.* replaying the UI interactions prior to interruption.

Quoting Okoshi *et al.* [34, p.2]: "*Rather than forcing users to manually check whether new information is available, notifications instead push new information to users, resulting in faster and increased awareness."* The authors state that 'interruptive overload' can either be targeted through a) scheduling (deferring) notifications, or b) mitigation of the amount of notifications. As an end result, the Attelia middleware for notification management [34], reduced the cognitive load of the arriving notifications by 46%, and increased the response times (*i.e.*, users were quicker to interact with the notification) by 13%. Here, we aim to use machine learning to detect opportune moments for quantified-self data input (as alert dialogs), reducing the burden of data input requests and increase the response times.

OPPORTUNE INTERRUPTION USING ALERT DIALOGS

In an attempt to solve the issue of users being burdened by the requirement of continuous logging [3, 15, 42], we argue that by inferring situations in which the user is not actively performing a task, opportune moments for *data contributions* can be identified. Brown *et al.*'s [6] large scale study of mobile device use recognised three use types, where the user is not actively performing a task:

- **Occasioning use** is initiated by the device (*e.g.* an arriving notification from a messaging application) and creates a timeframe where the user can naturally attend to the device.
- **Filling time** enables otherwise 'dead time' to be put into use (*e.g.*, enjoying a video game or browsing social media via the smartphone).
- **Micro-breaks**, similar to *micro-usage of application use* [11], are brief sessions of device usage where the user shortly interrupts his main tasks or activity.

For all of these usage types, users do not necessarily have a clear task during the usage session - it is therefore less likely that the user experiences interruption when presented with an input prompt. Van Berkel *et al.* [46] analyse smartphone usage sessions based on sensor data tracked from participants' devices, and categorise usage sessions into *continuing* and *new* sessions, and these sessions indicate either chosen or forced breaks in smartphone use. By identifying otherwise unproductive usage sessions with *in-situ* context of the smartphone, we aim to partially transform these sessions into self-report data contributions (*i.e.*, effective quantified-self data).

One must keep the required interaction with a prompt short [43, 44]. Thus, a method needs to be developed that is a) quick to interact with, and b) when necessary, easy to dismiss. Approaches by previous work that leverage brief interactions are Slide to X [43] and Twitch crowdsourcing [44], both of which utilise the smartphone unlocking event to collect data. Our chosen interaction modality is an alert dialog - used predominantly on desktop environments and web browsers, but sparingly on smaller screens. When used for the presentation of information, alert dialogs are habitually ignored [7] and also found to be disruptive [31]. Akhawe *et al.* [1] performed a large-scale assessment of browser warning dialogs and conclude that the design of the dialog window has a tremendous impact on both the user experience and user's responsiveness to the dialog.

Two common issues with mobile information presentation methods are cognitive overload [27, 40] and interruption of the user's main task. We believe that, if presented and timed carefully, an alert dialog can minimise both issues. Alert dialogs are occasionally used in tandem with the Experience Sampling Method [26], and the response rates when used in this way are generally quite high. For example, Van Berkel *et al.* [46] report 83.78% response rate and average response time of less than 3 seconds. The interruptive nature of push notifications is also diminished when the source of the interruption is deemed as beneficial to the user [40]. We argue that this is also the case for QS applications, and the interruptive nature of prompts originating from these applications is reduced.

Our hypothesis is that, by leveraging naturally appearing periods of smartphone use where the user is not actively performing a task (*i.e.* aimlessly juggling applications while bored [6, 28]), we can transform such usage sessions into *data contributions*. This can increase the quantity and quality of the data gathered by QS applications and potentially motivate users to continue self-monitoring, if the user's interruptibility can be predicted before deciding whether or not to present the input prompts. Alert dialogs can potentially fill these conditions based on their inherent properties, described in Table 1.

Predicting Interruption In-Situ

To present alert dialogs at appropriate times (R2), we must be able to predict whether an alert dialog would be considered interrupting in the current *context*. Context is described by properties such as physical elements of the surrounding environment [9]. When considering a smartphone, context also includes the internal measured through the device, *e.g.*, the chosen application or battery status. Based on the understanding of mobile context, and the contextual information logged by previous work [35-39], we capture those contextual factors which have been shown to impact the user's response to interruptions – factors related to usage session, battery, application use, network status, physical activity, and time of the day. An important element of alert dialogs is the *requirement* for interaction, and as the alert dialog is presented on top of any other interface elements, it shifts the user's focus from a previous task. This can result in high probability of interruption.

		Description
Requirements	**R1**	Alert dialogs can be *interacted* with very *briefly*.
	R2	Alert dialogs can *be presented* to the user *at appropriate times*.
	R3	Alert dialogs are *easy to remove*.
	R4	Alert dialogs can be *quickly generated* (users prefer no loading screens [16, 32]).

Table 1. Conditional requirements for effortless self-reports.

Due to the brief interaction times associated with alert dialogs, we assume that the context of the event does not change during the interaction, according to conditions R1 (brief interactions) and R3 (quick dismissal). In our application, user can interact with the alert dialog by *accepting* or *dismissing* the dialog, and by either contributing data or opting not to contribute. The response interaction label is logged with the *in-situ* context. We then proceed to analyse the dataset in terms of whether the context of an instance could predict the interactions with the dialog. We argue that the cost of unwanted interruptions is higher to the user than the cost of potentially missed data contributions. Thus, our aim is to minimise the number of events where a dialog was generated but ultimately experienced as *unwanted*.

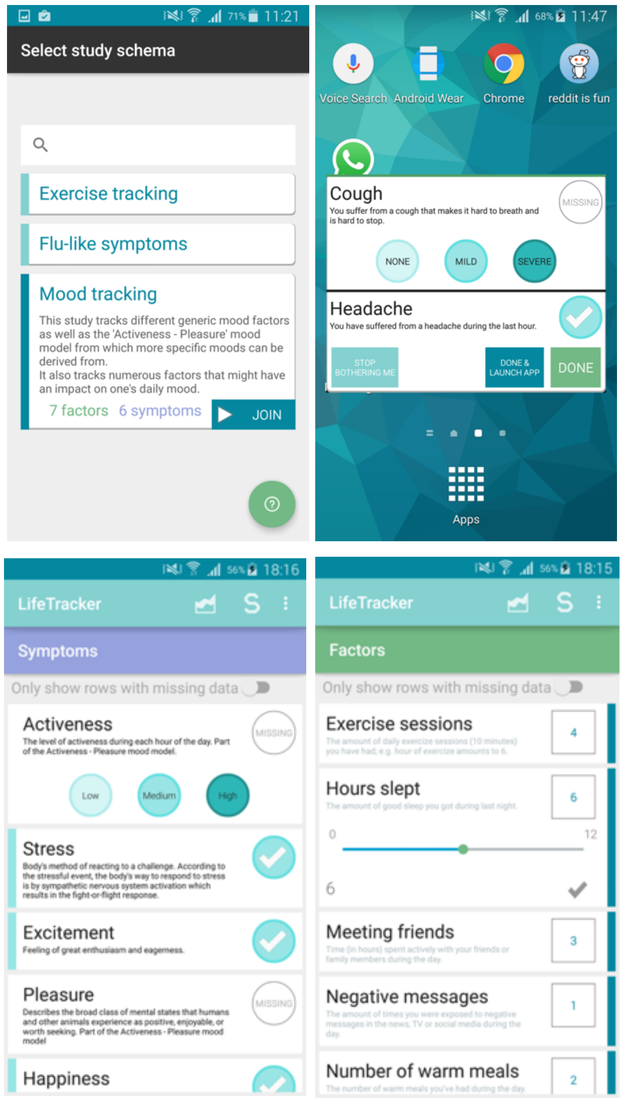
LIFETRACKER APPLICATION

Based on our four requirements (Table 1), we developed a QS application called **LifeTracker** to collect data using alert dialogs as the go-to method, and to collect user interactions with the alert dialogs. LifeTracker logs self-reported data, and sensor data logged on the background using the AWARE framework [12, 13] to collect the *in-situ* context of each presented dialog and securely synchronise the data to a remote server. To infer the context, we collect the sensor data motivated by previous work, data related to the usage session, and data related to the interactions with the LifeTracker application. We choose not to gather privacy sensitive information, such as application use or location data. Summary of tracked information is listed in Table 2.

The application has three different logging schemas (*i.e.*, set of tracked variables) custom-designed for different types of self-monitoring needs: *mood, physical exercise,* and *flu-like symptoms* (Figure 1 top left). Each schema has multiple variables that *can* be logged – not mandatory – once per hour or once per day. For example, the exercise schema allows participants to log their number of stretching sessions (daily) as well as their experienced level of dehydration (hourly). The data is captured using the alert dialogs (Figure 1 top right), or by explicitly launching the application (Figure 1 bottom).

Variable	*Description*
Data contribution	Data logged in the current usage session
Dialog delay	Delay (s) of dialog generation since device unlock
Session duration	Duration (s) of the usage session
Call	Call during the current usage session
No. of sessions	Number of usage sessions in the last five minutes
Last session	Time (s) since the last usage session
Session type	New or continuing session, using a 45 second threshold as described in [46]
Last interaction	Time (s) since last interaction with the LifeTracker application
Last contribution	Time (s) since last data contribution
Wi-Fi availability	Availability of Wi-Fi connection
Internet availability	Availability of internet connectivity
Network type	Cellular network connection type
Battery level	Battery level (%)
Battery charging	Charging state of the smartphone
Proximity	Smartphone screen covered or uncovered
Physical activity	Physical activity of the user (Google Activity Recognition API)
Hour	Hour of the day
Day	Day of the week

Table 2. Contextual variables collected by the LifeTracker application.



This study tracks different generic mood factors as well as the 'Activeness - Pleasure' mood model from which more specific moods can be derived from.
It also tracks numerous factors that might have an impact on one's daily mood.

You suffer from a cough that makes it hard to breath and is hard to stop.

You have suffered from a headache during the last hour.

The level of activeness during each hour of the day. Part of the Activeness - Pleasure mood model.

Body's method of reacting to a challenge. According to the stressful event, the body's way to respond to stress is by sympathetic nervous system activation which results in the fight-or-flight response.

Feeling of great enthusiasm and eagerness.

Describes the broad class of mental states that humans and other animals experience as positive, enjoyable, or worth seeking. Part of the Activeness - Pleasure mood model

The amount of daily exercize sessions (10 minutes) you have had; e.g. hour of exercize amounts to 6.

The amount of good sleep you got during last night.

Time (in hours) spent actively with your friends or family members during the day.

The amount of times you were exposed to negative messages in the news, TV or social media during the day.

The number of warm meals you've had during the day.

Figure 1. Views from LifeTracker application. Top: Schema selector (top left) and alert dialog (top right). Bottom: Application's main interface.

In addition to collecting data, the LifeTracker application also enables the user to view historical data using visualisations of weekly, daily, and hourly granularity. This feedback, in combination with the choice of a logging schema, ensures the inherent interest to interact with the application. This ensures the study does not simply collect 'clicks' – the data captured is also useful to the user. This becomes evident later as we show that users frequently interact with the application during the experiment.

User can interact with the alert dialog in three ways: 1) *dismiss* the dialog (left-most option in Figure 1 top right), 2) *accept* the dialog (right-most option), or 3) launch the LifeTracker application (second from right). The choice of interaction (*dismiss*, *accept*, or application launch) is separate from the data contributions, so it is possible for the user to *e.g.* contribute data, but simultaneously specify that the dialog was experienced as interrupting (by clicking dismiss) or choose not to contribute anything, but *accept* the dialog nonetheless. This allows us to collect context-rich data on participant interaction with the dialogs. It should also be noted that all interaction choices immediately remove the dialog from the view.

We designed the alert dialogs for brief interactions. Each variable requires minimal touch interactions. The variables are logged by three input modalities: three-tier scales, numeric ranges, or multiple choice - which can require more than one interaction. The application actively monitors the screen events of the smartphone. An alert dialog is only generated if the user is actively using the smartphone (screen is in *unlocked* state) and dialogs are at least five minutes apart. No alert dialogs are shown if a phone call is taking place and dialogs are automatically discarded after 60 seconds if left unattended. Upon unlocking the screen or after five minutes of continuous use, there is a probability for the application to generate an alert dialog. The probability of an alert dialog being presented starts at 50% for all participants. Depending on the user's interaction with the dialog, this chance will be adjusted for the next dialog presentation (+5% upon accept, -5% upon dismissing the dialog). This probably has a minimum of 5% and a maximum of 95%. This guarantees a lower burden on users who habitually select the dismiss option. After an alert dialog is removed, the combination of the collected context parameters (Table 2), and label (*accept*, *dismiss*, or application launch) of the response are stored.

Daily variables are only prompted after 6pm, as it makes little sense to ask a user to report incomplete information. More dialogs are generated later in the day (after 6pm) than during day-time (before 6pm), as users interact with their devices frequently and can quickly log all necessary prompts during each day-time hour. This is not necessarily the case after 6 pm, when the application requires significantly more information, *e.g.*, the amount of daily exercise or calorie intake.

EXPERIMENTAL SETUP

We recruited 48 participants (36 males, 12 females, aged 21-53 years old, M = 28.04, SD = 6.71) using mailing lists at our University on a first-come first-served basis. Each participant was required to own and use a mobile phone with Android 5.0 (Lollipop) or newer. The participants had varying academic backgrounds, ranging from Humanities and Law to Engineering and Natural Sciences. A total number of 16 participants had previously used a QS application. Previous usage of QS applications was not described as a requirement for the study. Understandably, the participant group is not a representation of general population, as the group consists mostly of young adults and university level students. However, as we show later in our results, the participant group is more diverse than initially thought, and they can provide useful insights to our research contributions. As the application could be considered as being disruptive to the user's everyday device usage, we compensated each user with 3 Euros for each day of participation.

We invite each participant to a study intake session in our lab. Here, we briefly explained the application's functionality, and mitigated users' privacy and security concerns: the data is anonymised, transferred and stored securely in a remote server through AWARE's enforced data privacy and security protocols (*e.g.*, encryption certificates, compression, and pseudonymous identifiers). Participants read and signed a consent form and we clarified any pending questions they might have. We demonstrated the functionality of the application, the different input methods (alert dialogs and the application), and how to interact with the alert dialogs.

We instructed them to use the LifeTracker application as they saw fit: we made clear they were not required to accept the alert dialogs, neither were they required to log all requested data (*i.e.*, all parameters of a set of tracked variables). We also asked them to fill a short survey with demographic information and answer the following set of questions:

- Q1) "Do you often read the arriving notifications immediately?", on a 4-point scale (Never, Sometimes, Usually, Always).
- Q2) "What kind of applications (categories) do you use on your personal smartphone?", according to a selection from Play Store categories, including the "Other" category, which allows the user to be more specific using free text.
- Q3) "Would you describe your smartphone use as active (frequent short periods of time), passive (only check when you are prompted by *e.g.* a notification), or mixed?"
- Q4) "Would you describe yourself as a technology enthusiast?", on a 4-point scale (Definitely not, Not really, Somewhat, Definitely).

After the four-week study period concluded, we invited each participant to fill an open-ended post-study survey to gather insights on the experienced interruptive nature and their experience with the application. We also asked the participants to describe "What influenced your decision to answer or reject a dialog?" (Q5).

Our experiment was designed to capture how users interact with the presented alert dialogs, rather than how they interact with the application itself. While the application offers benefits to the user via visualisations and potentially increased self-perception, as we did not validate the quality of the logged data in this experiment, we opted not to investigate our application's end-user benefits.

	Dialog accepted (non-interrupting)	*Dialog dismissed (interrupting)*
Data contributed	A. Non-interrupting, data contributed (N = 8,099)	C. Interrupting, data contributed (N = 374)
No data contributed	B. Non-interrupting, no data contributed (N = 7,490)	D. Interrupting, no data contributed (N = 3,277)

Table 3. Combinations of class labels based on interaction choice and existence of data contribution.

RESULTS

Participants interacted with a total of 19287 dialogs (daily mean = 13.63, SD = 10.11) and made 17,917 data contributions, of which 12,233 (68.3%) originated from the alert dialogs. From these 19287 dialogs, 15,434 (80.0%) were *accepted*, 3658 (19.0%) were dismissed, and 195 (1.0%) resulted in an application launch. A total number of 8,452 (43.9%) dialogs resulted in data contributions and an average of 10.25 daily contributions were made via dialogs, verified by a *t*-test (t(47) = 62.807, p < .05). We aggregate 395,344 screen events from the participants' smartphones into 82,332 (daily mean = 61.25, SD = 51.57) usage sessions - the period between screen becoming unlocked and either turning off or returning to locked state - with a mean duration of 4 minutes and 16 seconds. Based on the difference between the high dialog acceptance ratio (80.0%, N = 15434) and the number of data contributions from dialogs (N = 12,233), we observe that not all accepted dialogs resulted in data contributions.

We therefore created a separate classification scheme for our interruptibility analysis. We create a matrix of class labels in which we aggregated both the action (user contributed data or opted to not contribute) and interaction with the dialog (dialog accepted or dismissed), as shown in Table 3. Due to technical malfunctioning, 47 dialogs did not have a corresponding usage session. As some participants habitually accepted all dialogs, either mistakenly or to remove the dialog as fast as possible, this broader classification offers a more detailed understanding of the user's willingness to contribute data through the dialog.

This classification labels dialogs as 'interrupting' based on the user interaction (*accept* or *dismiss*), and on the actual contribution of data. For example, in class B the user selects the *accept* interaction (indicating it was non-interrupting) to hide the dialog but did not contribute any data from the dialog. Whether this class was considered as interrupting or not, and in what sense – did the user feel the data contribution cumbersome or was the dialog generated in an inopportune time? - does not have a clear generic answer. In class C the user contributed data, but defined the dialog as interrupting by hiding the dialog with the *"Do not bother me"* button. Class A and D provide the clearest desired dialogue interaction, dialogs that were accepted and data was contributed should always be shown (A), or dialogs that were always dismissed with no contributed should be always deferred (D). For classes B and C this decision is more ambiguous. In our analysis, we do not specify which action should be taken, but merely report whether each class is accurately predicted. An argument could be made for combining some of the classes (such as C and D as 'interrupting'), but we want to keep the different classification so that separate *actions* (*e.g.* defer, hide, show, and other possibilities) can be mapped separately to each class.

We begin by benchmarking a collection of machine learning classifiers and use the whole data set to build a general model, per classifier, using Weka [18]. The features are described in Table 2 and the class labels are described in Table 3. In Table 4 we highlight the best performing classifier, Random Forest, which we use in our analysis hereafter. Random Forest uses an internal unbiased validation measure called out-of-bag (OOB) accuracy [5], which removes the need for cross-validation. We use this out-of-bag accuracy as a way to measure and compare classifier accuracies. Random Forest functions by building N classifiers with M (max = 15) features each and classifies based on a voting mechanism from all N classifiers. To optimise the Random Forest size N (number of trees) and the number of features M used for each tree, we ran ten passes of the classifier for M = [5:10] and N = {250, 500, ..., 2000} and select N = 1500 and M = 5 as the optimal parameters.

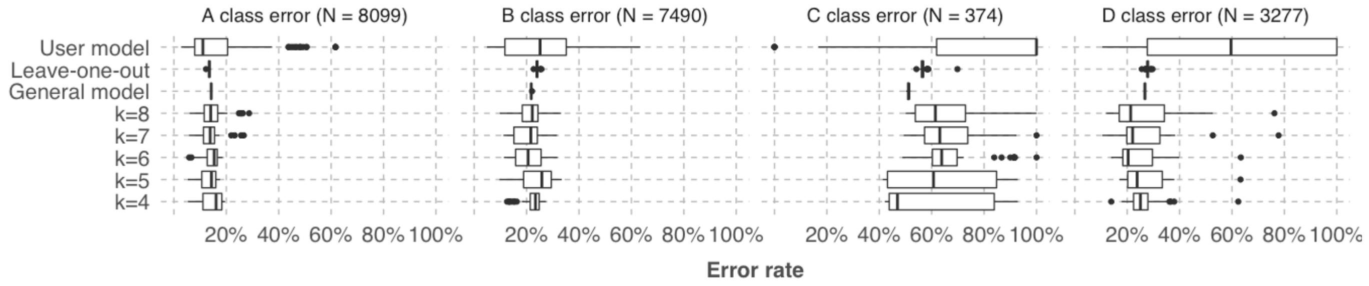


Figure 2. Measurement of machine learning accuracy using Out-of-bag error rates for different evaluation approaches and different classes (Table 3).

Classifier Type	*Classifier Name*	*Accuracy*	*ROC*
Bayesian	NaiveBayes	39.82%	0.7
Bayesian	BayesNet	72.58%	0.783
Tree	J48	74.29%	0.690
Tree	RF	**77.29%**	**0.830**
Rules	DT	68.98%	0.717
Meta	Bagging	73.37%	0.770
Meta	RandomSubSpace	75.43%	0.811
Lazy	LWL	51.76%	0.771
Functions	LibSVM	43.09%	0.510
Functions	SMO	61.60%	0.620

Table 4. Benchmarked machine learning classifiers.

As our dialog acceptance ratio was high (80.0%), we first analysed the distribution of the four classes in our dataset and observed that overfitting still exists for the A (42.0%) and B classes (38.9%, 81.9% combined). As Random Forests are shown to perform poorly for a biased dataset, we mitigate this by downsampling all overfitted classes from hereafter using random downsampling - half of all samples above the mean (of all four classes) are discarded at random. Due to the randomisation of discarded samples, we perform multiple passes of tests when necessary to minimise the bias due to any non-removed samples being overrepresented in the training data.

Predicting User Interactions

As our study setting aims to understand individual users, we do not attempt to simply generalise across our entire dataset. Instead, we take two separate approaches to classify user actions to understand how the classifier performs for a *previously unknown* user in predicting both whether the user would be interrupted by the dialog, and whether the user would be willing to contribute data.

We use two extremities as relative benchmarks. First, we trained and evaluated a general model with 20% of the data removed and used as testing data (to represent previously unknown interaction patterns), and iterated 20 times. Second, we evaluated a classifier for each user separately, built solely on their own data, and iterated 5 times per user. In this first approach, we leverage the leave-one-out method by extracting and using each user's own data as test data, and train the model with the user's own data excluded. This gives us insight about each separate user's fit into a more general model, without biasing the training data with instances of the user's own tracked data. This also offers us an assessment of how well a general classifier performs for a previously unknown user.

In our second approach, we experimented if a middle ground can exist between the two extremities of general and user models. The general model assumes we can use **all** existing data as training data for **all** (new) users, while the user model assumes that **each (new) user** first needs to collect personal training data. We leverage the concept of user types with identifiable traits and generated user clusters based on our pre-study questionnaire, combined with device usage patterns (hours of active smartphone usage). Each cluster contains users who share similar pre-determined traits - *e.g.*, inquired upon application installation (Q1-Q4, p.5) - and information regarding their device use (*e.g.*, *"Please specify during which period of the day you are most actively using your smartphone"*). We create k = {4,5,6,7,8} different clusters of users and use the leave-one-out mechanism for each of the users within these clusters (Table 5).Since we are using a separate testing set for each of our cases, we report both the internal out-of-bag accuracy for the training data (agreement *within* the users represented by the training data), as well as the out-of-bag accuracy of the test data (how well the test data, representing a previously unknown user, *fits into* the training data). We first perform the classification for general and user models. The internal mean out-of-bag accuracy is 79.0% (SD = 0.86%) for general model, and 80.4% (SD = 7.91) for the user models. The test out-of-bag accuracy is 72.6% (SD = 0.36%) for general and 79.1% (SD = 16.06%) for user. This showcases how the classifiers are in high internal agreement for both types, but personalised user models react more accurately to the testing data.

Classifying method	*Description of use*
General model	Classifier built from all available training data.
User model	Classifier built from each user's personal data.
Leave-one-out	Each user is extracted from the dataset, and used as test data while the remaining dataset is used as training data.
User clustering	Similar users are assigned into clusters, and leave-one-out is applied to each user in each cluster, with the remaining cluster data as training data.

Table 5. Different classifying methods applied to our dataset.

Next, we evaluate our first approach, the leave-one-out method. The out-of-bag accuracy is similar to the general model (M = 78.7%, SD = 0.27%, 48 trials). This is expected, as the training data is similar in both cases. However, the test out-of-bag accuracy is considerably lower (M = 68.1%, SD = 12.7%). This highlights the problem of fitting previously unknown users to a general model, and raises the need for a solution that 1) can be considered accurate, and 2) does not require the collection of training data before being useful.

Our second approach aims to mitigate these issues by constructing user profiles. We utilise demographic information (age, sex), the pre-study survey (Q1-Q4, p.5), and each user's device usage patterns (usage frequency during different hours of day) to form user groups with a total of 40 dimensions, using the k means clustering algorithm. The choice of clustering dimensions was purposely selected outside of machine learning features to reduce the bias of clustering choices on the actual classifier. According to these dimensions, from the best performing configuration (k = 7) we recognise identifiable characteristics of similarly acting groups, that share very similar traits to groups in [49], *e.g.*, users who frequently use their devices during night, or users who leverage smartphone capabilities in their work. Cluster 3 consists of users (N = 9) with very balanced characteristics, without any clear, extreme, or distinguishable extremities. Users in clusters 6 and 7 also share no clear distinguishable traits, and the clusters consist of less than five users and are thus discarded from now on. These users with no clear characteristics contribute 33% of all the users in our study setting. These characteristics could still be identified programmatically using black box methodology – *i.e.*, by creating clusters without clearly distinguishable characteristics of the clusters. Following user groups can be identified with distinguishable characteristics:

Cluster 1: Casual Users (10 users): Users in this group report the most passive device usage style, low tech enthusiasm, and slowest response rate to notifications. Their device use is more frequent during evening (8-10pm) than any other group. Also, this group has the highest (.50 > .25 for all participants) female ratio.

Cluster 2: Social Chatterers (7): The youngest age group (M = 20.85 < 28.04 for all, SD = .38), and all users are male. Users report frequent usage of communication apps, but do not use many other applications. Smartphone usage is balanced over the duration of the day (9am to 8pm), users quickly respond to notifications, and are likely to consider themselves tech enthusiasts.

Cluster 4: Night Owls (7): Most likely to use their smartphones between 10pm and 8am. This group likely consists of people with families abroad in different time zones. Users in this group are likely to use communication applications and otherwise displays a more mixed usage style, with low versatility in application usage.

Cluster 5: Work On-the-go (8): These users report an active usage style of their smartphones, respond to interruptions quickly, and report using e.g. finance applications frequently. Users in this group also have a daily rhythm that does not involve smartphone usage after 10pm. Demographically, this group is mostly male (.75 > .25 for all participants) and is older, on average, than the other groups (M = 35.6, SD = 2.69).

The leave-one-out method was then applied to each cluster and the results are visualised in Figure 3 along with the test results of previous methods (general model, user model, and the leave-one-out). The clustering methods average between 80.6% and 81.9% (SD = 2.8% to 3.9%) for out-of-bag accuracy and between 62.7% and 65.3% (SD = 13.0% to 16.5%) for test out-of-bag accuracy. The primary contributor

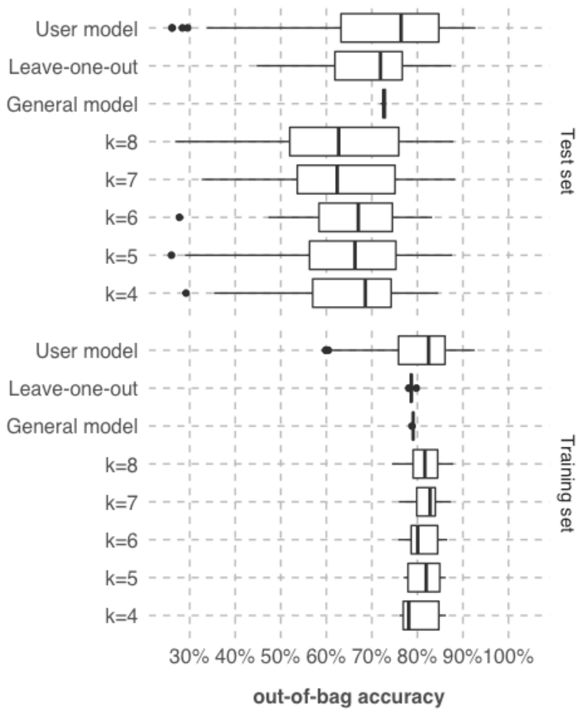
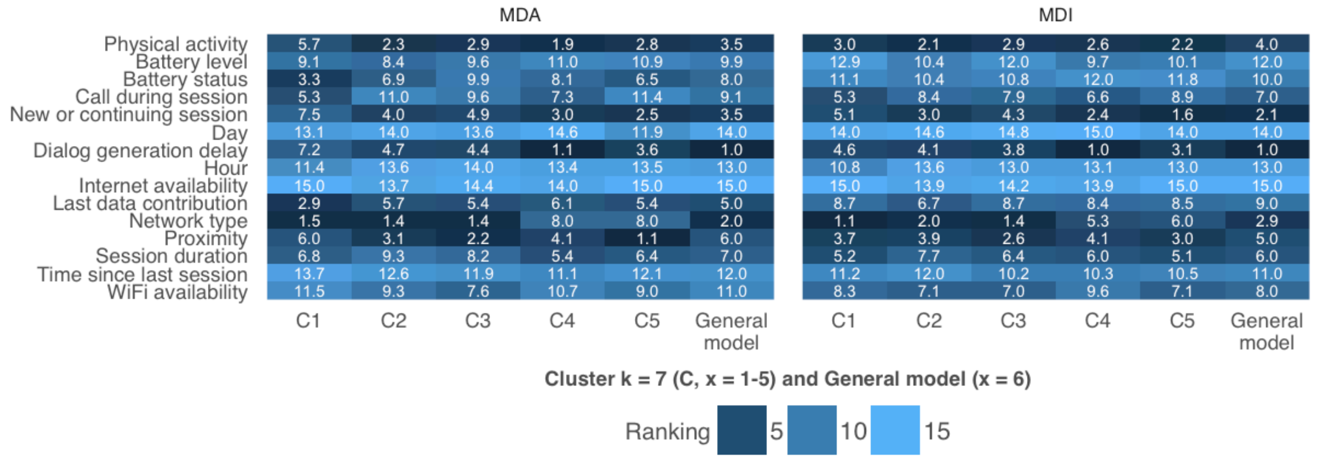


Figure 3. Random Forest accuracy for different machine learning evaluation approaches, for both the test set and the training set.

to reduced accuracy is likely the reduced amount of overall data (most clusters have between 8 and 15 users, which correlates to roughly 70-85% reduction in training data) for the classifier when using the clustering approach. The increased internal validation within the clusters is visible, as a *t*-test on cluster configuration k=7 shows significantly (p < .05, *t* = 3.20, df = 151.35,) higher mean accuracy (81.9%, SD = 2.8%) when compared to the user models (80.4%, SD = 7.9%). The mean is higher for all cluster configurations, but the results are not significant for other k values. Our main finding is that within all these cluster configurations, the groups of users selected using external factors performed as well as or better than the highly personalised user models.

Preventing Unwanted Interruptions

As our class labels indicate both user's willingness to contribute data and their preference of interacting with an alert dialog, we are also able to infer the likelihood of a classifier *preventing an unwanted dialog* from being generated. In addition to the overall accuracies of the classifiers, the error rates within different classes are also provided by the Random Forest classifier. The results for the same groups are visualised in Figure 2. All the approaches are accurate in predicting both the classes where users selected the *accept* interaction (classes A and B) (M = 14.9%, SD = 9.1% for A; M = 23.0%, SD = 9.6% for B), and manage to reduce 26.2% (SD = 8.8%, user model excluded) of the unwanted cases (D class indicating no data was contributed and the user *dismissed* the dialog). The user models are increasingly less likely (M = 62.0%, SD = 14.8%) to accurately predict the D class, which is likely due to several



	MDA						MDI					
	C1	C2	C3	C4	C5	General model	C1	C2	C3	C4	C5	General model
Physical activity	5.7	2.3	2.9	1.9	2.8	3.5	3.0	2.1	2.9	2.6	2.2	4.0
Battery level	9.1	8.4	9.6	11.0	10.9	9.9	12.9	10.4	12.0	9.7	10.1	12.0
Battery status	3.3	6.9	9.9	8.1	6.5	8.0	11.1	10.4	10.8	12.0	11.8	10.0
Call during session	5.3	11.0	9.6	7.3	11.4	9.1	5.3	8.4	7.9	6.6	8.9	7.0
New or continuing session	7.5	4.0	4.9	3.0	2.5	3.5	5.1	3.0	4.3	2.4	1.6	2.1
Day	13.1	14.0	13.6	14.6	11.9	14.0	14.0	14.6	14.8	15.0	14.0	14.0
Dialog generation delay	7.2	4.7	4.4	1.1	3.6	1.0	4.6	4.1	3.8	1.0	3.1	1.0
Hour	11.4	13.6	14.0	13.4	13.5	13.0	10.8	13.6	13.0	13.1	13.0	13.0
Internet availability	15.0	13.7	14.4	14.0	15.0	15.0	15.0	13.9	14.2	13.9	15.0	15.0
Last data contribution	2.9	5.7	5.4	6.1	5.4	5.0	8.7	6.7	8.7	8.4	8.5	9.0
Network type	1.5	1.4	1.4	8.0	8.0	2.0	1.1	2.0	1.4	5.3	6.0	2.9
Proximity	6.0	3.1	2.2	4.1	1.1	6.0	3.7	3.9	2.6	4.1	3.0	5.0
Session duration	6.8	9.3	8.2	5.4	6.4	7.0	5.2	7.7	6.4	6.0	5.1	6.0
Time since last session	13.7	12.6	11.9	11.1	12.1	12.0	11.2	12.0	10.2	10.3	10.5	11.0
WiFi availability	11.5	9.3	7.6	10.7	9.0	11.0	8.3	7.1	7.0	9.6	7.1	8.0

Figure 4. Feature rankings for MDA (Mean Decrease Accuracy) and MDI (Mean Decrease Impurity) for cluster k = 7.

users having the A and B classes overrepresented in their datasets (*accepting* all or a majority of presented dialogs). The C class is problematic to predict accurately, but also vastly underrepresented in the entire dataset (N = 374, 1.9%), and unlikely to reduce the overall accuracy of any approach. Similar to the overall accuracy, the user clusters are in higher agreement for the D class than other approaches (M = 24.5% < 26.2%, p < .05, *t* = 8.68, df = 180.33 after removing outliers) according to a *t*-test, which can also be considered the most important class for *preventing unwanted interruptions*.

Understanding Context

To further understand how and why our users interact with the dialogs in specific ways, we use feature extraction to gain insights on each factor. MDA (Mean Decrease Accuracy) shows the impact of each individual feature on the accuracy of the classifier if the feature is removed, and MDI (Mean Decrease Gini) is used as the impurity function. From the features of the k=7 cluster's classifier, we can also identify characteristics of the individual clusters that differ from the general model according to both tests (Mean Decrease Accuracy and Mean Decrease Impurity). We report alterations from the general model where both tests are in agreement and the features were ranked in the top half of the features.

Overall, the classifiers that used user clustering understood changes in physical activity and proximity in more detail, and individual clusters put weight on features such as network type and hour ("Casual Users", Cluster 1), Wi-Fi and internet availability ("Night Owls", Cluster 2), and session duration ("Work On-the-go", Cluster 5). Also, while the general model used the dialog delay as the most important feature, four out of five clusters found dialog delay to be less impactful. The same applies for session type (new or continuing session). The full averaged rankings for the Cluster (k = 7) and the general model are visualised in Figure 4.

In our post-study survey, we gathered insights from the participants regarding their experiences with the LifeTracker application. Q5 was an open-ended question, so we categorised the main reasons (one or more) for both accepting or dismissing a dialog. Ten answers were discarded from the categorisation due to low quality of the answer or the lack of proper focus. In eighteen (54.5%) answers the choice of interaction was due to a more important *priority task* being performed on the smartphone (P36: *"Most of the occasions when I rejected a popup were such that another application needed my concentration at that exact moment."*). One participant reported the *physical activity* (P12: *"I would reject the dialog if it came during work or exercise"),* four reported on the *time of the day*, three on the *surrounding* physical *context*, and two due to *bad mood*.

The majority of responses only listed reasons for dismissing a dialog. Three responses listed reasons specifically for accepting a dialog, two of these reported the *brief interactions* with the dialogs as reasons for accepting it (P17: *"If the question was just a button to select one option I mostly answered them because it was nearly as quick as rejecting a dialog."*) and one participant (P16) *lack of anything better to do*. The responses in the 'priority task' category can also be perceived as extensions of "lack of anything better to do". Some reasons correspond to the characteristics of the dialogs (brief interactions), while some correspond to the features used in the machine learning classifiers (*e.g.*, hour, physical activity, and *lack of anything better to do* are reflected in session type and duration).

DISCUSSION

We set out to study the interruptive nature of an input method designed to leverage user's spare time and reduce the burden required to make data contributions. These elements are key when motivating long term usage of quantified-self applications. Understanding of human interruptibility is crucial when using potentially disrupting data input methods.

We present results for an interaction method designed to fulfil specific requirements (Table 1) that are sculpted to reduce the cognitive load of the input method [27], decrease interaction time [43, 44, 46], and increase the quality of logged data [30]. We also attempt to understand both individual users and generic user *types* [48, 49] in order to enable applications to understand user's preferences without reliance on general models (often incorrectly assuming that all users behave similarly) or requiring an extensive training period to work correctly.

Contributing Data via Alert Dialogs

Overall, the number of dialogs (13.63, SD = 10.11) presented to our participants during each day of the study likely caused a burden on their smartphone usage, since the application only applied a very simple chance filter on the presentation of dialogs. However, it is necessary to prompt a user of a quantified-self application numerous times on a daily basis to ensure sufficient amount of data contributions, in order to generate meaningful insights. Considering the amount of general smartphone usage sessions each user participated in on a daily basis (M = 61.25, SD = 51.57), participants were not constantly interrupted but still frequently prompted.

Truong *et al.* [43] report an average of 49.83 daily contributions (compared to our 10.25) and [44] report a 37.4% task completion rate, but no number for daily contributions. However, neither approach considered a '*skip*' option so the users were always required to contribute in order to unlock their device. Results of the aforementioned study state that 44% of the study's participants felt that a skip option would have been necessary [44]. In [44], three of the ten participants also report that the forced data input on phone unlock significantly decreased the frequency of their phone unlocking. The requirement of repeated contribution and lack of a bypass method can significantly diminish the quality of the data [30], and can cause unnecessary burden to the user. The unlock screen interface also prohibits the user from using passwords or secret gestures to securely unlock his or her device - a limitation neither [30] or [44] addresses. Minimising the amount of daily interruptions while maximising the amount of data contributions is completed by deferring interruptive prompts, and minimising the cognitive load of the input method.

Dialogs and the use of lock screen as an input mechanism inherently reduce the cognitive load by design, so the remaining problem is to reduce the burden to the user, and the number of interruptions. Predicting the interruptive nature of an input method should be applied to all prompts, and the two methods (self-report within the unlocking interface, or prompting the user after unlocking the device) can be used in tandem, as long as the user is not overburdened and prompts are presented appropriately. Previous work has addressed this challenge via either general models; use of C4.5 machine learning classifier to predict acceptance of notifications attains up to 77% accuracy [39], and the InterruptMe library [35] offers up to 70% precision and recall. [23] uses decision-theoretic (DT) models to create user models that offer an increase in accuracy over a random probing mechanism, but at the cost of significantly reduced opportunities for self-reports. Additional take-away from our results is that the interrupting nature should not be derived directly from the interactions - as the 'non-disrupting' classes (A and B) were overrepresented within the interactions. When available, external measurements - like in our case looking at the existence of a data contribution from a dialog - can be more reliable as users can simply select *any* available interaction to remove a disrupting prompt [7]. Based on results in previous work and the accuracy of our benchmark general model, a more accurate method is clearly needed, that understands user's preferences.

Understanding Personal Preferences

To understand in more detail how different machine learning classifiers perform for previously unknown users, we leverage the leave-one-out cross validation method. Users are detached from the dataset one by one, the remaining data is used as the training set, and the detached user's data as a testing set when applicable. To our knowledge, this method has not been previously used as a method to understand users' personal preferences and users' fit into general models. As hypothesised, the leave-one-out method proves less accurate (68.1%) than the general model (72.6%) with the testing sets, and signifies how a general model is not always applicable to the individuals within a general population. This especially applies in situations where personal usage traits vastly differ, such as for smartphone usage [10, 46-49]. These personal differences were evident in our study, even though our participant group was quite homogenous by nature, in terms of sociological status, level of education, and age.

We use features (Table 2) that attempt to both understand the user's external context (physical activity, time), and smartphone usage (frequency of usage sessions and interactions). The appropriateness of the selected features in differentiating between users is apparent, as the personalised user models are more accurate (M = 80.4%, SD = 7.9%) than the general or leave-one-out, even with the reduced available training data. In addition, the responses from our final questionnaire following the four-week usage period correspond to the selected features. Users reported frequently dismissing the alert dialogs due to their physical activity, time of day, or type of device usage session - factors that directly correspond to the features selected for our machine learning classifiers and that are previously used as significant features [20, 35, 39]. This allows us to conclude that the choice of features accurately reflected the user's interaction behaviour.

However, both the use of general models and personalised user models have inherent problems. General models have poor fit for individual users with specific usage and behavior traits. Falaki *et al.* [10] report that smartphone users differ by at least one or more orders of magnitude in their device usage patterns. On the other hand, the use of personalised models requires significant amount of collected training data - which in turn requires time. We pick up on the recent trend [48, 49] of identifying user groups and use this concept to create *user clusters* based on external user reported features (*e.g.*, description of smartphone use, usage patterns throughout the day). We selected these dimensions for the clusters in order to showcase that applications can leverage user provided information to form preset configurations for *new users* of which the application or system has no prior information. The generated clusters had stronger inner agreement than the classifiers constructed from other training datasets (Figure 3), indicating the validity of this approach. The cluster-based classifiers were also most accurate in *preventing unwanted interruptions* (Figure 2) - a key measurement considering the interrupting nature of alert dialog as an input mechanism.

Feature ranking offers a glimpse inside the black box implementation of the Random Forest classifier, and we can observe how the different identifiable characteristics of clusters affect the predictions (Figure 4), *e.g.* the differences in physical activity patterns (MDI ranking in Figure 4) are more detailed than in the general model. As for the users within clusters, the "Work On-the-go" ranked session duration higher than in the general model - this archetype is involved in frequent short usage sessions, so interrupting this type of user during messaging or an important work-related call is likely unwanted. "Night Owls" ranked Wi-Fi and internet availability higher, compared to other groups. This group prioritises their longer device usage sessions - ones where they are less interrupted - to occasions where they have proper connectivity. And the "Casual Users" group put more weight on hour, indicating that as they use their device sparingly throughout the day, they prefer to be interrupted when it is most convenient to them (*e.g.*, during the evening hours). The effect of these fine-grained pieces of information become apparent through the way the classifiers function and take different features into consideration for different user types. These user clusters can also be generalised to an extent, considering extremely similar user clusters were generated in [49], consisting of "Night communicators", "Screen checkers", who are quick to respond to incoming prompts, and cluster of "Evening learners", similar in characteristics to our "Casual users". We do not claim that our findings are perfectly applicable to the general population. We did not analyse the characteristics of roughly third of our users (three clusters) as their usage traits were difficult to distinguish sufficiently, or the groups were considered too small. However, if the process of matching users to user groups is done programmatically, the automation process could also efficiently match these types of users to predetermined groups.

Our analysis is, to the best of our knowledge, the first attempt to leverage this type of user differentiation, based on a mixture of self-reported and sensor logged smartphone usage behaviour. The cluster-based approach resulted in the highest accuracy (Figure 3), and was most likely to reduce unwanted interruptions (Figure 2), and had the highest prediction accuracy within the training data set (Figure 3). The prediction accuracies also show improvements over previous work [35, 39]. These results are encouraging and we argue this research can pave the way for intelligent applications that can be personalised more effortlessly. Most importantly, these applications no longer require extensive learning periods or manually applied configurations. By inquiring about the usage habits of new users, their requirements for the application, and their preferences, it becomes possible to match this information with an existing user base. Applications are thus able to extract pre-generated group models for each new user.

Limitations and Future Work

Although our application is not designed to merely collect data, we focus our work on the data collection process, and do not consider the quality of the logged data or what the end-user benefits of using our application were. Additionally, while our participant group was homogenous in terms of demographics, they showed diversity in how they use their smartphones, and our main aim was to validate our proposed approach.

The machine learning models we use were not evaluated *in-the-wild*, but our use of leave-one-out method offers us insight in how accurately the classifier would react to unforeseen events. Our aim is to replicate our approach in a longer field study in the future. This would also verify the impact of our approach in long-term application use.

CONCLUSION

QS applications habitually suffer from abandonment of use and often the motivational methods aimed to increase the longitude of use suffer from lack of data. We conducted a four-week long user study with 48 users and analyse the use of potentially intrusive on-screen alert dialogs as self-reporting mechanisms. We identify five distinct user groups, based on features external to their interactions, and showcase how the Random Forest classifier can accurately predict user interruptibility within these groups.

Personal applications should not rely on generalised models, as differences in smartphone use between users have been brought up repeatedly in literature, and also in the results we have presented in this work. Different user types are more active during different times of day, have different usage styles in terms of usage session frequency and duration [46, 47], prefer different types of applications [48, 49], and interact with their devices differently [10]. This leads to the inclination to model users either individually, or within specified user type groups. Applications can leverage our approach to use historical data from their user base as training data for new users, by matching characteristics of new users to existing user groups. However, users should not be overburdened by constantly requiring data contributions, especially if multiple applications leverage the same method and compete for user's attention.

ACKNOWLEDGEMENTS
This work is partially funded by the Academy of Finland (Grants 276786-AWARE, 286386-CPDSS, 285459-iSCIENCE, 304925-CARE), the European Commission (Grant 6AIKA-A71143-AKAI), and Marie Skłodowska-Curie Actions (645706-GRAGE)

REFERENCES

1. Devdatta Akhawe and Adrienne Porter Felt. Year. Alice in Warningland: A Large-Scale Field Study of Browser Security Warning Effectiveness. In *22nd USENIX Security Symposium*, Washington, D.C., USENIX Association.
2. RG Barker. 1968. *Ecological psychology : concepts and methods for studying the environment of human behavior*. Stanford University Press, Stanford, California.
3. Frank Bentley and Konrad Tollmar. 2013. The Power of Mobile Notifications to Increase Wellbeing Logging Behavior. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM, 1095-1098. 10.1145/2470654.2466140
4. Frank Bentley, Konrad Tollmar, Peter Stephenson, Laura Levy, Brian Jones, Scott Robertson, Ed Price, Richard Catrambone and Jeff Wilson. 2013. Health Mashups: Presenting Statistical Patterns between Wellbeing Data and Context in Natural Language to Promote Behavior Change. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 20 (5). 30. 10.1145/2503823
5. Leo Breiman. 2015. Random Forests. *Machine Learning October 2001*, Volume 45 (1). 5-32. 10.1023/A:1010933404324
6. Barry Brown, Moira McGregor and Donald McMillan. 2014. 100 Days of iPhone Use: Understanding the Details of Mobile Device Use. In Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services, ACM, 223-232. 10.1145/2628363.2628377
7. Raymond Chen. 2003. The default answer to every dialog box is "Cancel", Microsoft.
8. Eun Kyoung Choe, Saeed Abdullah, Mashfiqui Rabbi, Edison Thomaz, Daniel A Epstein, Felicia Cordeiro, Matthew Kay, Gregory D Abowd, Tanzeem Choudhury and James Fogarty. 2017. Semi-Automated Tracking: A Balanced Approach for Self-Monitoring Applications. *IEEE Pervasive Computing*, 16 (1). 74-84.
9. Anind K. Dey. 2001. Understanding and Using Context. *Personal Ubiquitous Comput.*, 5 (1). 4-7. 10.1007/s007790170019
10. Hossein Falaki, Ratul Mahajan, Srikanth Kandula, Dimitrios Lymberopoulos, Ramesh Govindan and Deborah Estrin. 2010. Diversity in Smartphone Usage. In Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services, ACM, 179-194. 10.1145/1814433.1814453
11. D. Ferreira, J. Goncalves, V. Kostakos, L. Barkhuus and A. K. Dey. 2014. Contextual Experience Sampling of Mobile Application Micro-Usage. In International Conference on Human-Computer Interaction with Mobile Devices and Services, 91-100. 10.1145/2628363.2628367
12. D. Ferreira, V. Kostakos and A. K. Dey. 2015. AWARE: mobile context instrumentation framework. *Frontiers in ICT*, 2 (6). 1-9. 10.3389/fict.2015.00006
13. Denzil Ferreira. 2013. Aware: A mobile context instrumentation middleware to collaboratively understand human behavior, Department of Computer Science & Engineering University of Oulu Acta Univ. Oul. C 458.
14. James Fogarty, Scott E. Hudson, Christopher G. Atkeson, Daniel Avrahami, Jodi Forlizzi, Sara Kiesler, Johnny C. Lee and Jie Yang. 2005. Predicting Human Interruptibility with Sensors. *ACM Trans. Comput.-Hum. Interact.*, 12 (1). 119-146. 10.1145/1057237.1057243
15. Thomas Fritz, Elaine M. Huang, Gail C. Murphy and Thomas Zimmermann. 2014. Persuasive Technology in the Real World: A Study of Long-term Use of Activity Sensing Devices for Fitness. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM, 487-496. 10.1145/2556288.2557383
16. D; Turban Gehrke, E. Year. Determinants of successful Website design: relative importance and recommendations for effectiveness. In *Hawaii International Conference on Systems Sciences*, Hawaii, USA. 10.1109/HICSS.1999.772943
17. Rúben Gouveia, Evangelos Karapanos and Marc Hassenzahl. 2015. How Do We Engage with Activity Trackers?: A Longitudinal Study of Habito. In Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing, ACM, 1305-1316. 10.1145/2750858.2804290
18. Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann and Ian H. Witten. 2009. The WEKA Data Mining Software: An Update. *SIGKDD Explor. Newsl.*, 11 (1). 10-18. 10.1145/1656274.1656278
19. Mary Jo Hatch. 1987. Physical Barriers, Task Characteristics, and Interaction Activity in Research and Development Firms. *Administrative Science Quarterly*, 32 (3). 387-399. 10.2307/2392911
20. Joyce Ho and Stephen S. Intille. Year. Using context-aware computing to reduce the perceived burden of interruptions from mobile devices. In *Proceedings of*

the SIGCHI Conference on Human Factors in Computing Systems, ACM, 909-918. 10.1145/1054972.1055100

21. Gary Hsieh, Ian Li, Anind Dey, Jodi Forlizzi and Scott E. Hudson. 2008. Using Visualizations to Increase Compliance in Experience Sampling. In Proceedings of the 10th International Conference on Ubiquitous Computing, ACM, 164-167. 10.1145/1409635.1409657

22. 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, ACM, 93-102. 10.1145/1357054.1357070

23. Ashish Kapoor and Eric Horvitz. Year. Experience sampling for building predictive user models: a comparative study. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 657-666. 10.1145/1357054.1357159

24. Evangelos Karapanos. 2015. Sustaining User Engagement with Behavior-change Tools. *interactions*, 22 (4). 48-52. 10.1145/2775388

25. Nicky Kern, Bernt Schiele and Albrecht Schmidt. 2007. Recognizing context for annotating a live life recording. *Personal and Ubiquitous Computing*, 11 (4). 251-263. 10.1007/s00779-006-0086-3

26. Reed Larson and Mihaly Csikszentmihalyi. 1983. The Experience Sampling Method. In *Flow and the Foundations of Positive Psychology*, Wiley Jossey-Bass, 41-56.

27. Luis A. Leiva, Matthias Böhmer, Sven Gehring and Antonio Krüger. 2012. Back to the app: the costs of mobile application interruptions. In MobileHCI'12, 291-294. 10.1145/2371574.2371617

28. Aleksandar Matic, Martin Pielot and Nuria Oliver. Year. Boredom-computer interaction: boredom proneness and the use of smartphone. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ACM, 837-841. 10.1145/2750858.2807530

29. Abhinav Mehrotra, Robert Hendley and Mirco Musolesi. 2016. PrefMiner: Mining User's Preferences for Intelligent Mobile Notification Management. In ACM International Joint Conference on Pervasive and Ubiquitous Computing, ?

30. Abhinav Mehrotra, Jo Vermeulen, Veljko Pejovic and Mirco Musolesi. 2015. Ask, But Don't Interrupt: The Case for Interruptibility-Aware Mobile Experience Sampling. In Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers, ACM, 723-732. 10.1145/2800835.2804397

31. Motamedi S., M. Hasheminejad and Choe P. Year. Driving Safety Considered User Interface of a Smartphone: An Experimental Comparison | SpringerLink. In *International Conference on Cross-Cultural Design*, Springer. 10.1007/978-3-319-20934-0_15

32. F. F Nah. 2003. A Study on Tolerable Waiting Time: How Long Are Web Users Willing to Wait? In 9th Americas Conference on Information Systems, Tampa, FL, USA. 10.1080/01449290410001669914

33. Mikio Obuchi, Wataru Sasaki, Tadashi Okoshi, Jin Nakazawa and Hideyuki Tokuda. Year. Investigating interruptibility at activity breakpoints using smartphone activity recognition API. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, ACM, 1602-1607. 10.1145/2968219.2968556

34. Tadashi Okoshi. Year. Attelia: Reducing user's cognitive load due to interruptive notifications on smart phones. In *Pervasive Computing and Communications (PerCom), 2015 IEEE International Conference on*, IEEE. 10.1109/PERCOM.2015.7146515

35. 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, ACM, 897-908. 10.1145/2632048.2632062

36. Martin Pielot. 2014. Large-scale Evaluation of Call-availability Prediction. In Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing, ACM, 933-937. 10.1145/2632048.2632060

37. Martin Pielot, Rodrigo de Oliveira, Haewoon Kwak and Nuria Oliver. 2014. Didn't You See My Message?: Predicting Attentiveness to Mobile Instant Messages. In Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems, ACM, 3319-3328. 10.1145/2556288.2556973

38. Martin Pielot, Tilman Dingler, Jose San Pedro and Nuria Oliver. 2015. When Attention is Not Scarce - Detecting Boredom from Mobile Phone Usage. In Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing, ACM, 825-836. 10.1145/2750858.2804252

39. Benjamin Poppinga, Wilko Heuten and Susanne Boll. 2014. Sensor-Based Identification of Opportune Moments for Triggering Notifications. *Pervasive Computing, IEEE*, 13 (1). 22-29. 10.1109/MPRV.2014.15

40. Alireza Sahami Shirazi, Niels Henze, Tilman Dingler, Martin Pielot, Dominik Weber, Albrecht Schmidt, Alireza Sahami Shirazi, Niels Henze, Tilman Dingler, Martin Pielot, Dominik Weber and Albrecht Schmidt.

Year. Large-scale assessment of mobile notifications. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 3055-3064. 10.1145/2556288.2557189

41. Tapio Soikkeli, Juuso Karikoski and Heikki Hämmäinen. 2011. Diversity and End User Context in Smartphone Usage Sessions. In International Conference on Next Generation Mobile Applications, Services and Technologies, IEEE, 7-12. 10.1109/NGMAST.2011.12

42. Melanie Swan. 2013. The Quantified Self: Fundamental Disruption in Big Data Science and Biological Discovery. *Big Data*, 1 (2). 85-99. 10.1089/big.2012.0002

43. Khai N. Truong, Thariq Shihipar and Daniel J. Wigdor. 2014. Slide to X: unlocking the potential of smartphone unlocking. In Proceedings of the 32nd annual ACM conference on Human factors in computing systems, 3635-3644.

44. Rajan Vaish, Keith Wyngarden, Jingshu Chen, Brandon Cheung, Michael S. Bernstein, Rajan Vaish, Keith Wyngarden, Jingshu Chen, Brandon Cheung and Michael S. Bernstein. Year. Twitch crowdsourcing: crowd contributions in short bursts of time. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 3645-3654. 10.1145/2556288.2556996

45. N. van Berkel, C. Luo, D. Ferreira, J. Goncalves and V. Kostakos. 2015. The Curse of Quantified-Self: An Endless Quest for Answers. In International Joint Conference on Pervasive and Ubiquitous Computing Adjunct, 973-978. 10.1145/2800835.2800946

46. Niels van Berkel, Chu Luo, Theodoros Anagnostopoulos, Denzil Ferreira, Jorge Goncalves, Simo Hosio and Vassilis Kostakos. 2016. A Systematic Assessment of Smartphone Usage Gaps. In Conference on Human Factors in Computing Systems, 4711-4721. 10.1145/2858036.2858348

47. A. Visuri, Z. Sarsenbayeva, N. van Berkel, J. Goncalves, R. Rawassizadeh, V. Kostakos and D. Ferreira. 2017. Quantifying Sources and Types of Smartwatch Usage Sessions. In Conference on Human Factors in Computing Systems. 10.1145/3025453.3025817

48. Pascal Welke, Ionut Andone, Konrad Blaszkiewicz and Alexander Markowetz. Year. Differentiating smartphone users by app usage. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ACM, 519-523. 10.1145/2971648.2971707

49. Sha Zhao, Julian Ramos, Jianrong Tao, Ziwen Jiang, Shijian Li, Zhaohui Wu, Gang Pan and Anind K. Dey. Year. Discovering different kinds of smartphone users through their application usage behaviors. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ACM, 498-509. 10.1145/2971648.2971696