Elsevier Editorial System(tm) for International Journal of Human-Computer Studies
Manuscript Draft

Manuscript Number:

Title: Understanding Smartphone Notifications' User Interactions and Content Importance

Article Type: Original Article

Keywords: Smartphone; Notifications; Interactions; Semantic analysis; Machine learning

Corresponding Author: Mr. Aku Visuri, MsC

Corresponding Author's Institution: University of Oulu

First Author: Aku Visuri, MsC

Order of Authors: Aku Visuri, MsC; Niels van Berkel, MSc.; Tadashi Okoshi, PhD; Jorge Goncalves, PhD; Vassilis Kostakos, PhD

Abstract: We present the results of our experiment aimed to comprehensively understand the combination of 1) how smartphone users interact with their notifications, 2) what notification content is considered important, 3) the complex relationship between the interaction choices and content importance, and lastly 4) establish an intelligent method to predict user's preference to seeing an incoming notification. We use a dataset of notifications received by 40 anonymous users in-the-wild, which consists of 1) qualitative user-labelled information about their preferences on notification's contents, 2) notification source, and 3) the context in which the notification was received. We assess the effectivity of personalised predictions models generated using a combination of self-reported content importance and contextual information. We uncover four distinct user types, based on the number of daily notifications and interaction choices. We showcase how usage traits of these groups highlight the requirement for notification filtering approaches, e.g., when specific users neglect to manually filter out unimportant notifications. Our machine learning-based predictor, based on both contextual sensing and notification contents can predict the user's preference for successfully acknowledge an incoming notification with 91.1% mean accuracy, crucial for time critical user engagement and interventions.

Suggested Reviewers: Martin Pielot PhD
Researcher, Telefonica Research
martin.pielot@telefonica.com
Martin is a leading researcher in smartphone notitifiations, interruptibility, and machine learning.

Veljko Pejovi
Assistant Professor, University of Llubljana
Veljko.Pejovic@fri.uni-lj.si
Veljko is an expert in mobile sensing and analysis of mobile data tracers and would provide important insights on the area of our paper.

MIkio Obuchi
Keio University
fabius@ht.sfc.keio.ac.jp
Mikio is a newer researcher who specialises in mobile interruptibility research.

Abhina Mehrotra
University College London
a.mehrotra@ucl.ac.uk
Abhina is one of the leading researchers in smartphone notifications and his work on the PrefMiner application was one of the early motivations for our work.

Opposed Reviewers: Denzil Ferreira
University of Oulu
denzil.ferreira@oulu.fi
Denzil Ferreira is the supervisor of the corresponding author, and thus should not be considered as a reviewer for this work.

***For Editor only: Cover Letter**

Corresponding author: Aku Visuri, University of Oulu, Finland
Aku.visuri@oulu.fi

Niels van Berkel, The University of Melbourne, Australia
n.vanberkel@student.unimelb.edu.au

Tadashi Okoshi, Keio University, Japan
slash@ht.sfc.keio.ac.jp

Jorge Goncalves, The University of Melbourne, Australia
jorge.goncalves@unimelb.edu.au

Vassilis Kostakos, The University of Melbourne, Australia
vassilis.kostakos@unimelb.edu.au

We hereby declare that all aforementioned authors have all made substantial contributions to our work titled: Understanding Smartphone Notifications' User Interactions and Content Importance

This work has not been published previously nor is it currently under review in any other medium.

Our submission consists of total of 11.5K words and is included in this document starting from page four. All figures should be colorized, when applicable, and are included within the submission.

Declaration of Interests: none

We hereby declare that we, the authors, nor any corresponding financial or governing body affiliated with this submission have no competing or other interests that might influence our work.

Abstract

We present the results of our experiment aimed to comprehensively understand the combination of 1) *how smartphone users interact with their notifications*, 2) *what notification content is considered important*, 3) *the complex relationship between the interaction choices and content importance*, and lastly 4) *establish an intelligent method to predict user's preference to seeing an incoming notification*. We use a dataset of notifications received by 40 anonymous users *in-the-wild*, which consists of 1) qualitative user-labelled information about their preferences on notification's contents, 2) notification source, and 3) the context in which the notification was received. We assess the effectivity of personalised predictions models generated using a combination of self-reported content importance and contextual information. We uncover four distinct user types, based on the number of daily notifications and interaction choices. We showcase how usage traits of these groups highlight the requirement for notification filtering approaches, *e.g.*, when specific users neglect to manually filter out unimportant notifications. Our machine learning-based predictor, based on both contextual sensing and notification contents can predict the user's preference for successfully acknowledge an incoming notification with 91.1% mean accuracy, crucial for time critical user engagement and interventions.

Figure 1
Click here to download high resolution image

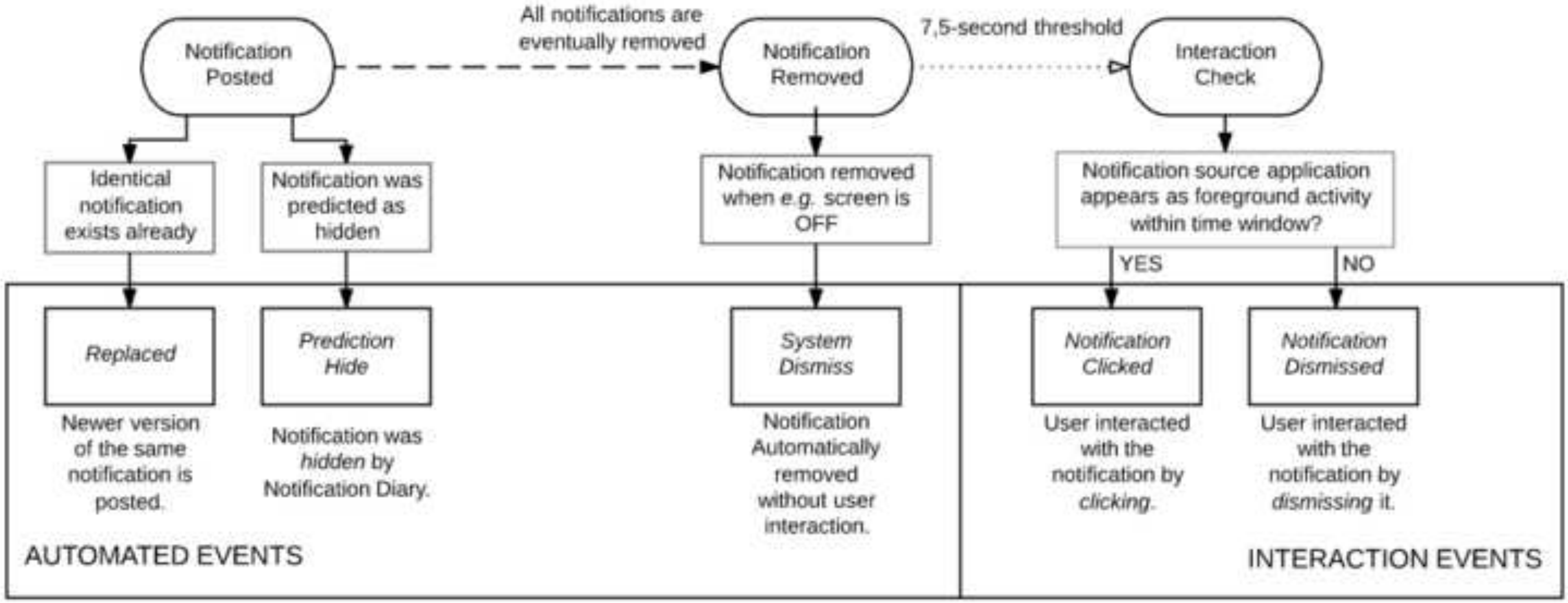


Figure 2
Click here to download high resolution image

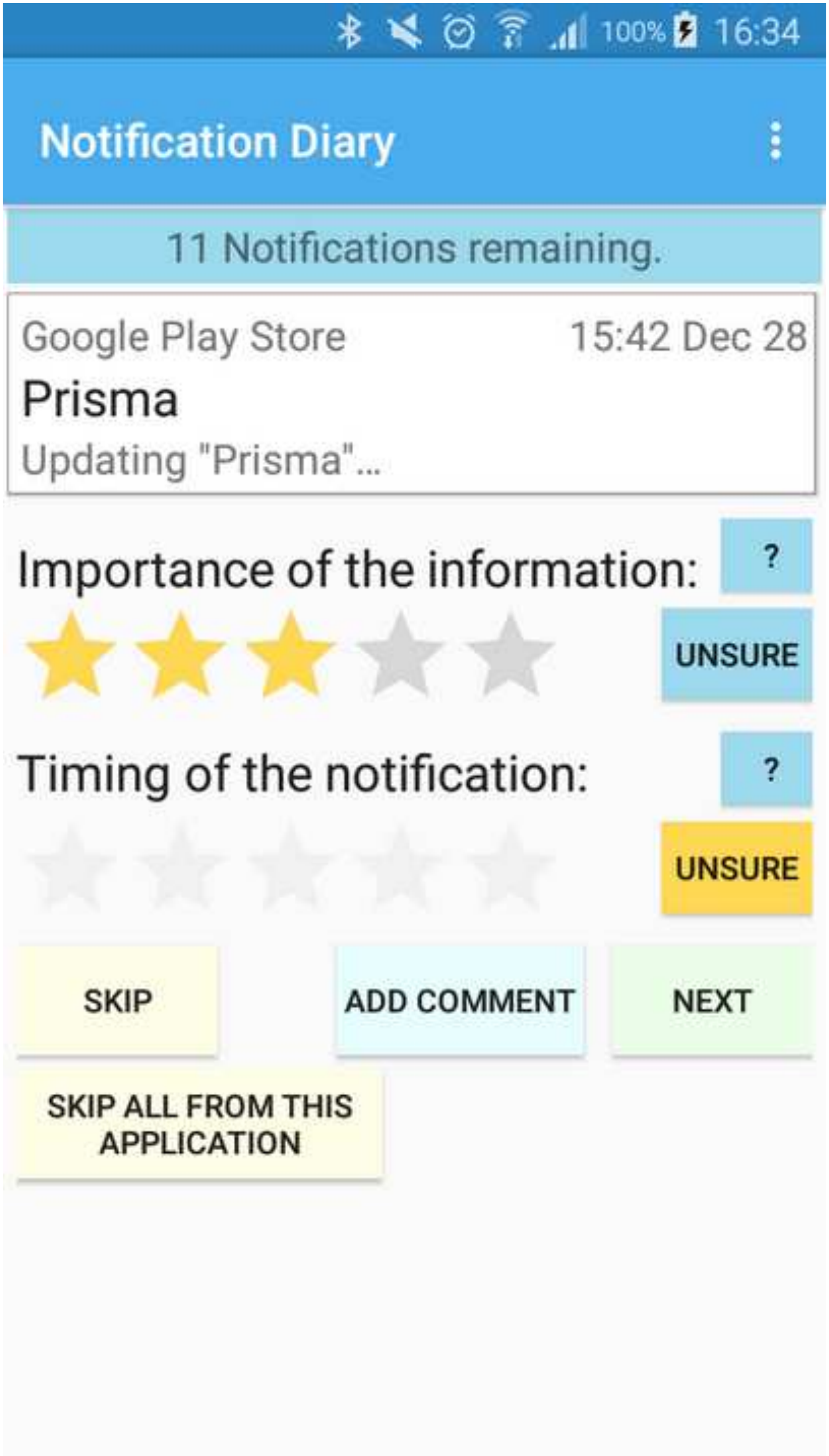


Figure 3
Click here to download high resolution image

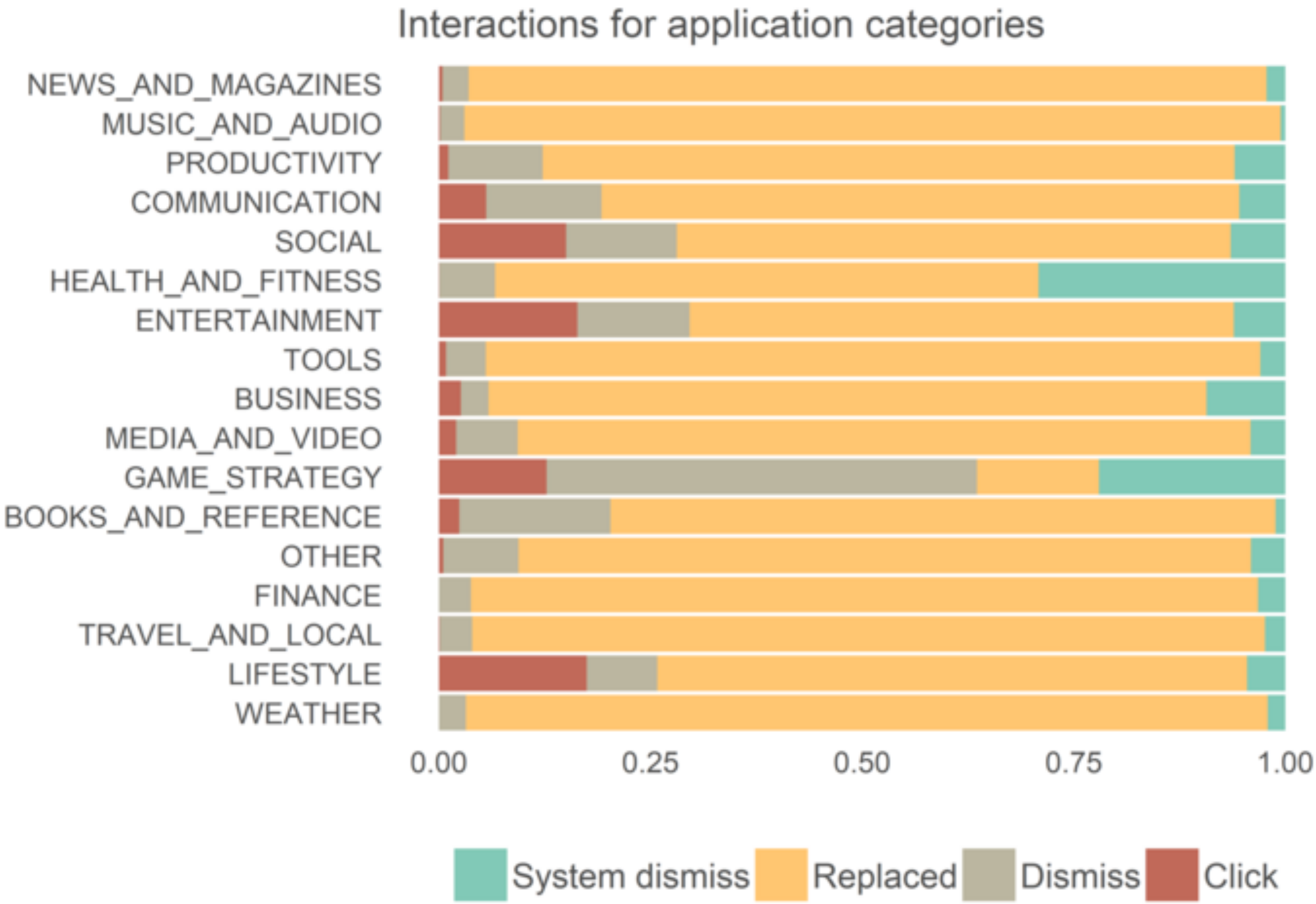


Figure 4 upper
Click here to download high resolution image

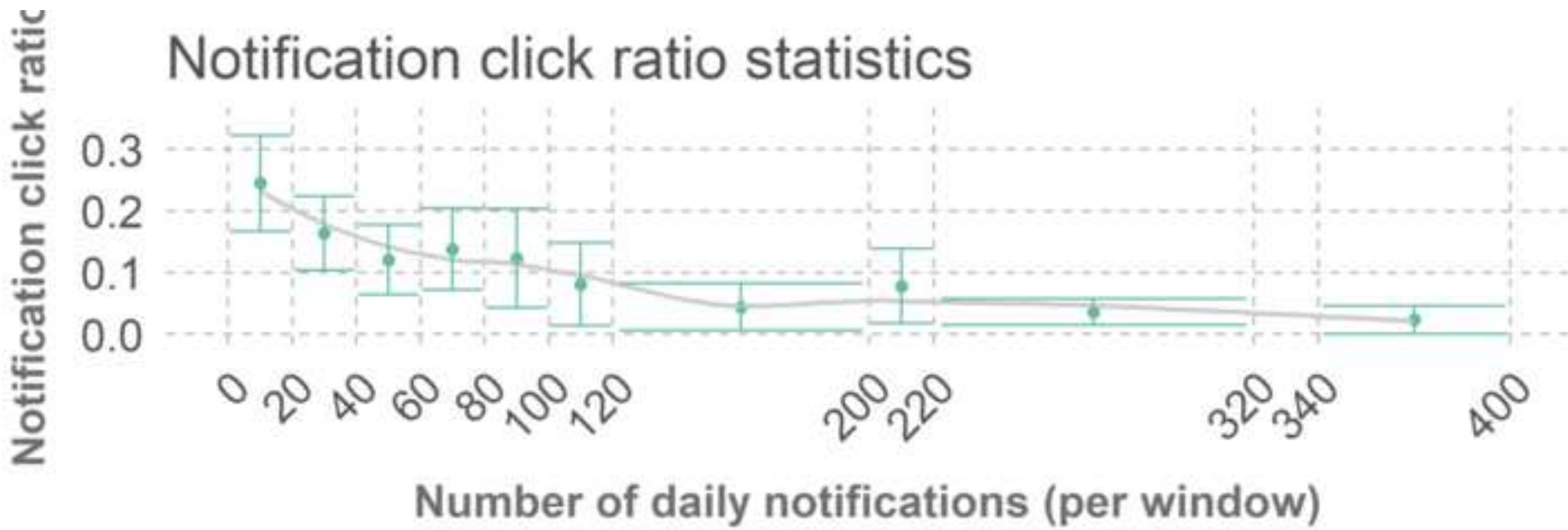


Figure 4 lower

Click here to download high resolution image

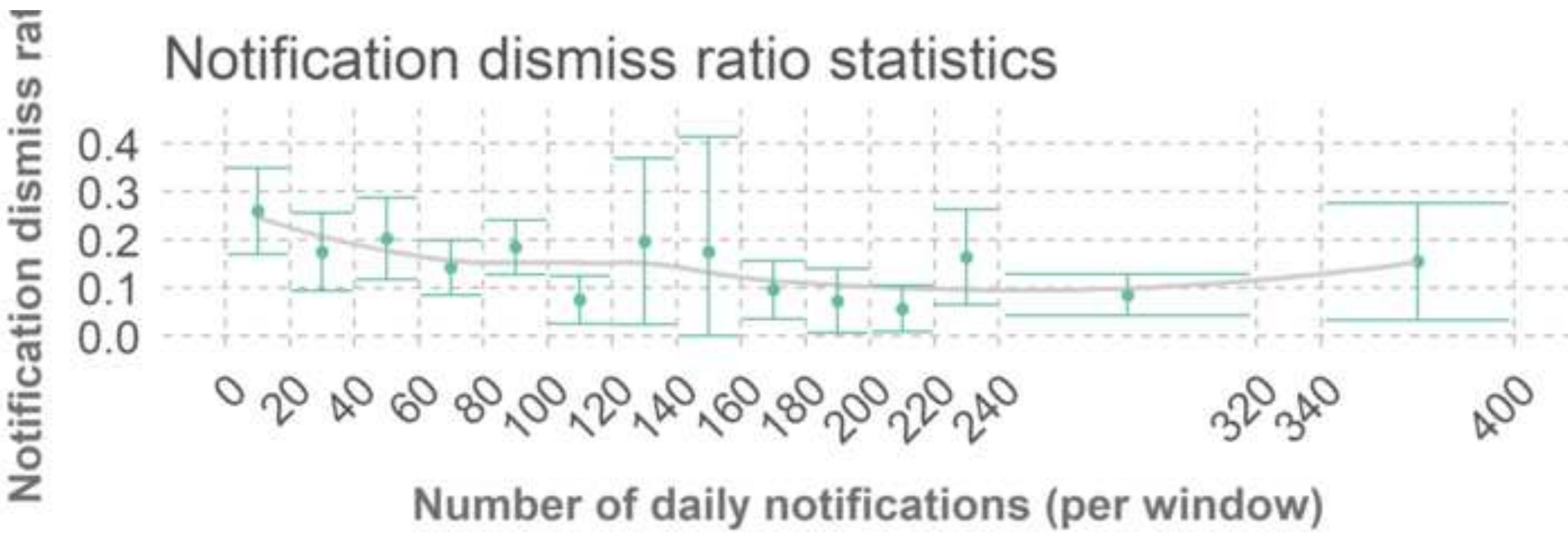


Figure 5
Click here to download high resolution image

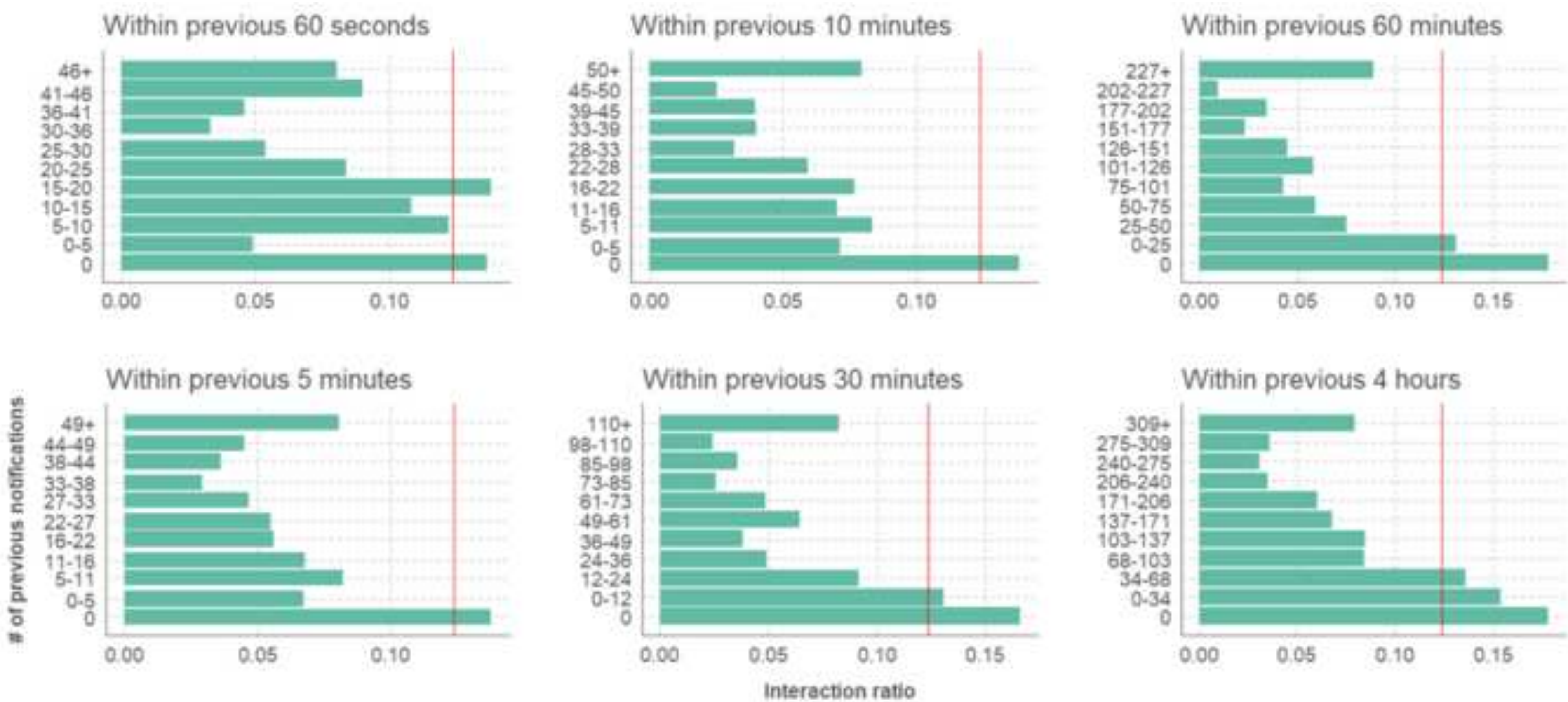


Figure 6
Click here to download high resolution image

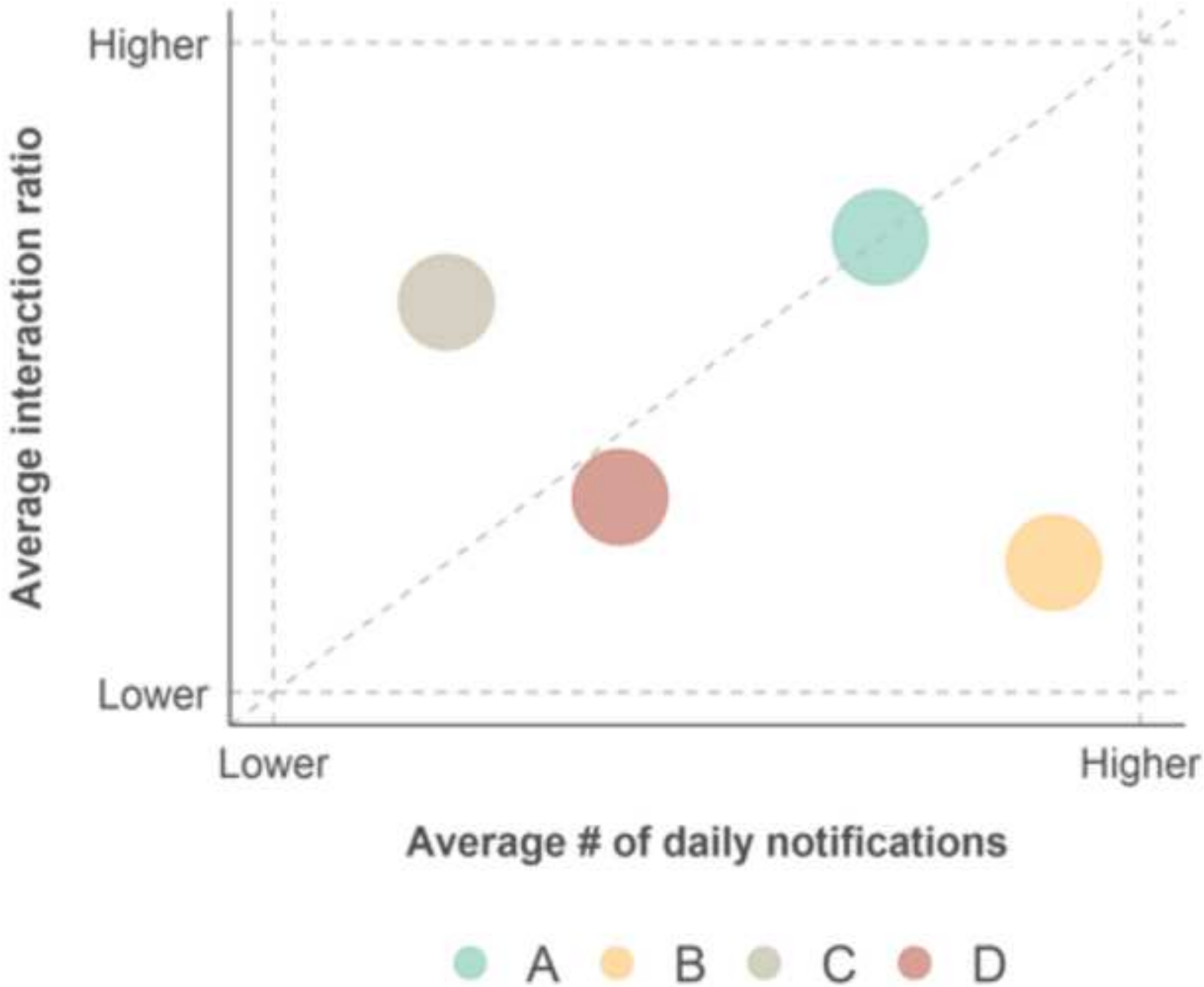
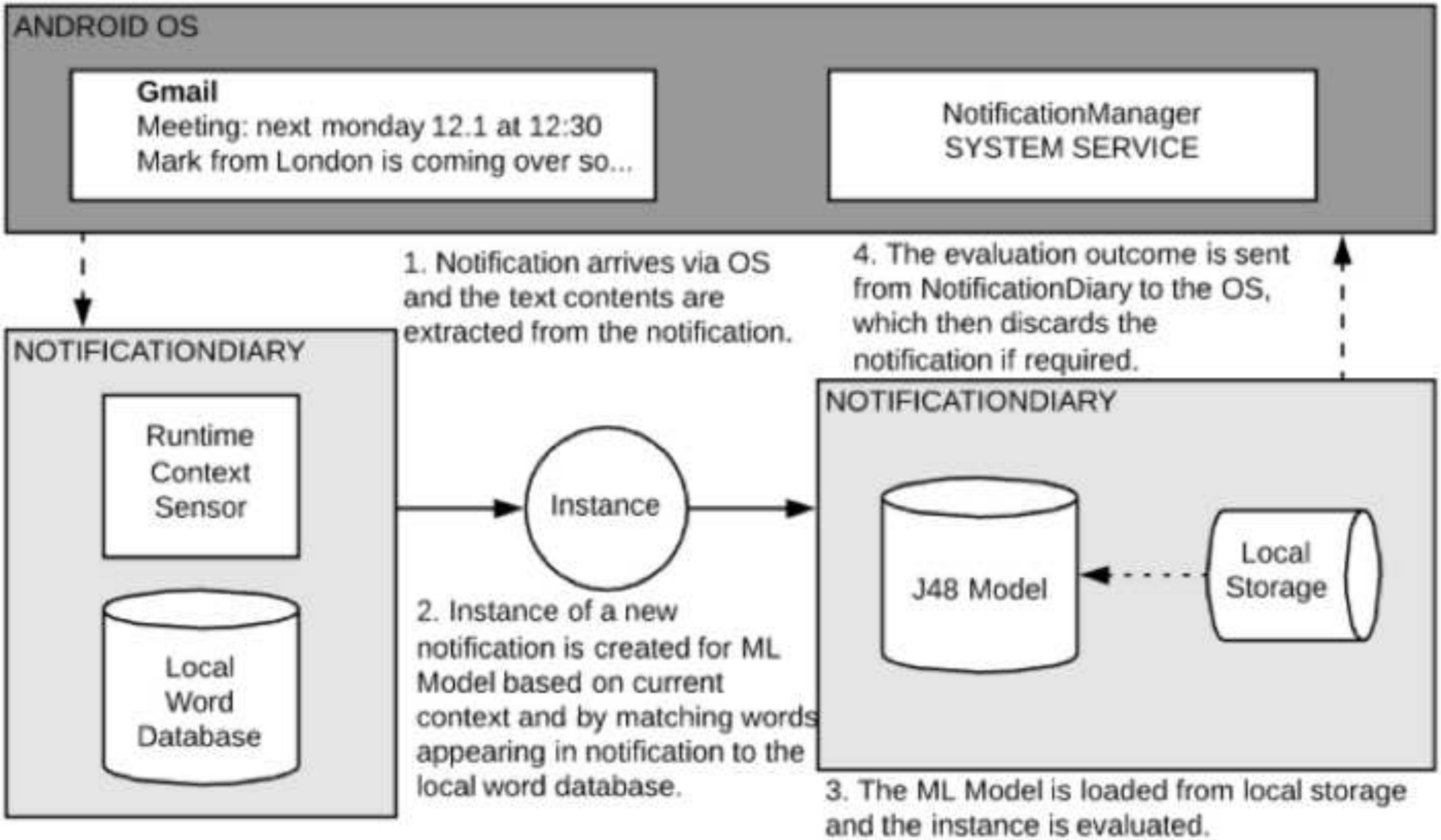


Figure 7
Click here to download high resolution image



Corresponding author: Aku Visuri, University of Oulu, Finland
Aku.visuri@oulu.fi

Niels van Berkel, The University of Melbourne, Australia
n.vanberkel@student.unimelb.edu.au

Tadashi Okoshi, Keio University, Japan
slash@ht.sfc.keio.ac.jp

Jorge Goncalves, The University of Melbourne, Australia
jorge.goncalves@unimelb.edu.au

Vassilis Kostakos, The University of Melbourne, Australia
vassilis.kostakos@unimelb.edu.au

We hereby declare that all aforementioned authors have all made substantial contributions to our work titled: Understanding Smartphone Notifications' User Interactions and Content Importance

This work has not been published previously nor is it currently under review in any other medium.

Our submission consists of total of 11.5K words and is included in this document starting from page four. All figures should be colorized, when applicable, and are included within the submission.

Declaration of Interests: none

We hereby declare that we, the authors, nor any corresponding financial or governing body affiliated with this submission have no competing or other interests that might influence our work.

Author Biography

Aku Visuri

Aku Visuri is a human-computer interaction PhD student at the Center for Ubiquitous Computing located at the University of Oulu, Finland. His background is in designing and implementing solutions for companies in the healthcare business. He received his MSc in Computer Science and Information Networks from University of Oulu in 2016. In his PhD work, the focus is on ubiquitous computing and quantified-self (QS). Specifically understanding users of QS applications and technologies, such as wearable devices, and designing new methods to enable users with more efficiency.

Niels van Berkel

Niels van Berkel is a PhD student at the University of Melbourne, where he is part of the Interaction Design Lab. In his PhD research, the focus is on active human sensing through ubiquitous devices, most prominently smartphones. A large portion of his work focuses on the methodological aspects of active data collection (*e.g.*, Experience Sampling Method. He has a background in interaction design and computer science and has been involved in the design and development of a wide variety of projects.

Tadashi Okoshi

Tadashi Okoshi is a Project Assistant Professor of Graduate School of Media and Governance, Keio University. He is a computer scientist focusing on distributed systems, mobile and ubiquitous computing, context-aware computing, and "attention-aware" computing. His current research topic is human-attention-awareness and its management in ubiquitous computing and cyber physical systems. He holds B.A. in Environmental Information (1998), Master of Media and Governance (2000) from Keio University, M.S. in Computer Science (2006) from Carnegie Mellon University, and Ph.D. in Media and Governance (2015) from Keio University, respectively. He has 7 years of experience of entrepreneurship, software architecting, product management, and project management in IT industries.

Jorge Goncalves

Jorge Goncalves is a Lecturer in Human-Computer Interaction at the University of Melbourne where he is part of the Interaction Design Lab. Previously, he worked as a postdoctoral researcher at the University of Oulu in the Center for Ubiquitous Computing. He received a PhD with distinction (2015) in Computer Science and

Engineering from the University of Oulu, and a BSc (2009) / MSc (2011) in Computer Science and Engineering from the Madeira Interactive Technologies Institute, University of Madeira (Portugal) under the Carnegie Mellon University | Portugal partnership. His research interests include ubiquitous computing, HCI, crowdsourcing, civic engagement, social computing, and mobile sensing.

Vassilis Kostakos

Vassilis Kostakos is a Professor in Human-Computer Interaction at the University of Melbourne School of Computing and Information Systems. He is a Marie Curie Fellow, a Fellow in the Academy of Finland Distinguished Professor Program, and a Founding Editor of the PACM IMWUT journal. He holds a PhD in Computer Science from the University of Bath. His research interests include ubiquitous computing (Ubicomp), human-computer interaction (HCI), social computing, and Internet of Things.

Understanding Smartphone Notifications' User Interactions and Content Importance

Abstract

We present the results of our experiment aimed to comprehensively understand the combination of 1) *how smartphone users interact with their notifications*, 2) *what notification content is considered important*, 3) *the complex relationship between the interaction choices and content importance*, and lastly 4) *establish an intelligent method to predict user's preference to seeing an incoming notification*. We use a dataset of notifications received by 40 anonymous users *in-the-wild*, which consists of 1) qualitative user-labelled information about their preferences on notification's contents, 2) notification source, and 3) the context in which the notification was received. We assess the effectivity of personalised predictions models generated using a combination of self-reported content importance and contextual information. We uncover four distinct user types, based on the number of daily notifications and interaction choices. We showcase how usage traits of these groups highlight the requirement for notification filtering approaches, *e.g.*, when specific users neglect to manually filter out unimportant notifications. Our machine learning-based predictor, based on both contextual sensing and notification contents can predict the user's preference for successfully acknowledge an incoming notification with 91.1% mean accuracy, crucial for time critical user engagement and interventions.

Keywords

Smartphone, Notifications, Interactions, Semantic analysis, Machine learning

1 Introduction

Mobile notifications allow applications to inform users of incoming messages, new system events, and reminders, without requiring explicit interaction. Users receive upwards from 60 daily notifications [30,35], of which many are considered unimportant by the recipient. In response, researchers aim to reduce the interruptive nature of unwanted notifications [18,25,31,33] via sensing technologies or by understanding the qualitative nature of notifications. While a large body of work exists on predicting notification-driven interruptibility through situational context, these methods fail to capture the other side of the challenge - is a notification important to the user. Thus, there is a need to better understand the relationship between *interacting* with notifications – how users choose to interact – and the *perceived importance* of the notification contents.

Here, we aim to understand the underlying importance of individual notifications, how users interact with them, and which factors influence their interaction choices. To investigate the motivation for *interacting* with notifications, we use self-reported information about the importance of notification contents, notification source, as well as the context of presentation. The motivation is captured in terms of notifications the user would *prefer* to see regardless of the interaction, *e.g.*, notifications that should be presented even if habitually ignored or dismissed, and the notifications that the user might consider *irrelevant* or disrupting. Our findings highlight the varying nature of users' strategy for manually filtering out notifications in terms of *how often* *users opt to interact with notifications* and the interaction *choices*, and the ever-present need for a notification management system, aiming to *prevent information overload* - especially considering how frequently users neglect to manually filter out excess notifications.

We also evaluate a notification management system based on these principles. Our system predicts notification importance based on semantic analysis of the similarity of arriving notification and previous notifications. The system also passively collects information about the user's context and combines the aforementioned importance with user context to create a detailed prediction model used to assess whether the user wishes to see the new notification or not. This combined approach shows vast improvements over previous similar systems, highlighting how understanding notification contents can further increase prediction accuracy in filtering out unwanted notifications.

2 Related Work

The role of smartphones has moved away from simple messaging and news-reading to an extended tool aiming to help the user in other aspects of life, *e.g.,* personal health, work, or keeping up with larger social circles, noteworthy when presenting notifications from different, but equally important sources [9]. The notification content [8] and the identification of opportune moments for presenting notifications [7,13,26,33] both play a vital role in notifications' receptivity. Additional factors also impact the pursuant interactions, such as social relationships in case of messaging applications [21].

Mehrotra *et al.*'s PrefMiner [18], a tool for mining user preferences and to generate intelligent and easily understandable rules (*"Stop notifications from Facebook that contain 'candy' and 'crush' words in the title."*) to hide or show selected notifications, confirming the notion of *reminder notifications*, notifications that contain important information from the calendar or alarm events, and that such notifications habitually *dismissed*. Other work uses context-awareness [12,31] and breakpoints in phone activities [7] to predict user interruptibility. Clark [2] finds that the user's response to an interruption can be: a) acknowledgment and an agreement to handle notifications later (*i.e.*, **defer**); b) a decline to handle the interruption (*i.e.*, **dismiss**). The previous work on interruptibility focus on three methods – defer [4,13], dismiss [18], or identifying opportune moments [29] – to mitigate the interruptive effects of notifications.

Applications rely on mobile notifications to present information to the user, to request their attention, or to elicit phone use. As more applications trigger notifications, the amount of daily notifications is drastically larger [30,35]. Users select which notifications to interact with (*i.e.*, click) and which to dismiss (*i.e.*, swipe away). Such choice can depend on a multitude of factors associated with either notification contents or presentation context [8,18,21]. Notifications are inherently disruptive and distractive [35]. To address this, Leiva *et al.'*s work [14] tried to overcome disruptions by either preparing the user to be interrupted or guiding the user when returning to the task. Alas, users do place value on receiving notifications, as long as the sources are of importance to them [35]. For example, Samahi Shirazi *et al.'s* large-scale assessment of mobile notifications validates that users value notifications differently depending on the notifications' source. Some notifications are *expected to be swiped* away – triggered by user-initiated actions (*e.g.*, download completed) or from system events (*e.g.*, battery running low) – accomplishing a simple goal of informing the user. While a portion of notifications is deemed as unimportant or unwanted, only a fraction of mobile phone users consciously manages their notification settings [40].

Notifications from messaging applications and updates on people or events, such as the news, are deemed important [8,35]. Meanwhile, notifications not associated with communication applications are often received less favorably [15,19,22]. Whether this reduced attention is due to being overwhelmed with other (mainly communication) notifications, or because of the actual content is perceived as less useful, is yet to be explored. The use of computer-mediated communication is also shown to be an indicator of user availability and openness to further cues [16,28,32], which begs to question whether the user's current state of mind is influenced by communication applications to be more receptacle to interruptions, or whether the use of communication applications showcases breaks in concentrations and other tasks. Identifying such breakpoints in smartphone usage has been shown to be a valuable tool in recognising opportune moments for notification delivery [6].

From the viewpoint of context influencing attentiveness, previous literature has taken either the approach of evaluating the influence of single variables, or comprehensive systems considering a combination of contextual factors. The effect of time of day is often explored, but it alone has been shown to not be a sufficient variable [41]. Another consideration was the influence of the user's physical location on notification attentiveness, but while a user was shown to be more available while at work [36], the response times to notifications to do not vary depending on location [19]. Physical activity, namely the breaks between activities, often indicate attentiveness [12]. Similarly, any task or activity requiring concentration is shown to be a poor moment for interruptions [27]. Other smartphone-based sensors can also extend this understanding, e.g., the ringer mode and vibration settings are shown to influence the speed at which people attend to new messages and notifications [30,31]. Pielot *et al.* [31] show that simple features extracted from the phone can predict attentiveness to mobile instant messages and reduce the interruptive nature of such generated notifications. The user's attentiveness to presented notifications and engaging with notifications can be measured in more detail via machine learning models [29]. Fischer *et al.* [8] analyse the impact of mobile notifications' content and their timing on user receptivity and conclude that *content* is a more important factor than timing when considering the interruptive nature of notifications. Okoshi *et al*. [25] deployed a large-scale interruptibility estimation logic and demonstrated that by deferring notifications to a more appropriate time of the day the response time can be significantly reduced. Previously, they investigated ways to reduce user's cognitive load due to interrupting notifications [24]. Lastly, De Russis and Roffarello considered ways to include user preferences, in addition to context, in notification delivery [3].

While a larger portion of previous work is aimed at identifying opportune moments for delivering notifications similar to the *opt-in* concept (*i.e.*, when should a notification be shown), another approach is aimed at comprehensively manage notifications through *opting out* of unwanted notifications. Mehrotra *et al.* [18] suggest that *usable* interruptibility and notification management systems should attempt to achieve the goal of reducing interruptions *without compromising* the reception of any useful and important information. Useful measurements for notifications' acceptance include *response time* [7,31], and *click rates* [18,35]. Dismissed notifications are considered either rejected or unwanted by the user. However, the content of such notifications can still be of value to the user - we argue that dismissed notifications may contain valuable content and should be considered as acknowledged and having fulfilled their purpose. Assuming these notifications are always unwanted will undeniably lead to reducing the amount of useful and important information to the user, thus compromising the main goal of notification management systems.

2.1 Contribution

Previous literature has analysed smartphone notifications from the stance of i) notifications as a source of distraction, or ii) methods to mitigate notifications as distractions with the use of notification management techniques. The end outcomes of notifications in terms of interactions - *"what happens to notifications and why?"* - and which factors influence this decision, remain underexplored. The main contribution of our work is to develop a systematic understanding of notifications – *which types of notifications* are considered important, *how* users interact with notifications, and *why*. Finally, our contributions include a deeper understanding of the underlying reasons for interaction choices via combining contextual and qualitative information and showcase how to improve the intelligence of notification management systems by merging these two information sources.

The paper is structured as follow: first, we start by describing the data collection methodology and analysis used to determine the relationship between content importance and user's interaction choices with notifications. Second, we uncover distinct manual notification filtering mechanisms identified from within our study participants. Third, we describe our implemented combined notification management system and its effectiveness. While each section briefly discusses the significance of these results, a full-fledged discussion is included at the end of the paper.

3 Notification Diary

We developed an application called Notification Diary to collect contextual and user-originated qualitative information about notifications - the user-perceived importance of the notifications - and how users interacted with those notifications. We deployed Notification Diary on Google's Play Store and made intermittent advertisement campaigns using social media, and on our university campus. The data collection occurred during the first quarter of 2017 (*i.e.*, January - March). The application contains a consent form and information about the purpose of the application, *i.e.*, data collection for research purposes, and includes both a short tutorial and guidelines on how to appropriately use the application. This ensures all participants are equally informed of the experiment, and the capabilities of the application. A total of 40 anonymous users installed and used the application for an average of 12.2 days (SD = 14.41). We collected the demographics information available in Google's Play Store application analytics.

3.1 Data Collection

Notification Diary collects data from notifications on the user's smartphone passively (using background processes and sensor readings) and actively (using retroactive user-reported information). We collect four different types of information: 1) quantitative information logged from the smartphone, 2) contextual information of the situation when the notification arrived, 3) notification information and content (only stored locally on the phone to ensure privacy) and 4) qualitative annotations about the notification content and timing of its presentation, provided by the user. We also collect the end outcome for each notification, *i.e.,* how was it eventually removed – whether the user interacted with the notification via *clicking* or *dismissing* it. The summary of the collected sensor and the user-reported information is presented in Table 1.

Contextual information	
Location	User's physical location using geofences that annotate encrypted location identifiers (for *e.g.*, work, home)
Physical activity	Physical activity of the user, using Google Awareness API (walking, still, running, in a vehicle, *etc.*)
Headphone jack	Boolean state of the device headphone jack (whether earphones are plugged in or not)
Ringer mode	State of the device ringer mode (silent, vibration, normal)
Screen state	State of the device screen mode (off, on, locked, unlocked)
Battery information	The battery level (%) of the device, and the charging state (whether the charging cable is plugged in or not)
Network information	Boolean state of Internet connectivity and Wi-Fi availability
Foreground application	The current foreground application on the smartphone, stored as the unique application package name

Notification information	
Source application	The package name of the application emitting the notification
Contents	The title and message text extracted from the notification contents, configured by the application that emitted the notification
Notification outcome	How the notification was eventually removed from the notification tray; due to user clicking or swiping away the notification, being automatically discarded by the system, being replaced, or being hidden by Notification Diary's predictions (refer to Figure 1)

User labels (UL)	
UL1: importance	The user-perceived importance and/or relevance of the notification contents on scale of 0-5
UL2: timing of notification	The user-perceived interruptive nature of the notification on a scale of 0-5

Table 1. List of the relevant sensor and user-reported information collected by Notification Diary application.

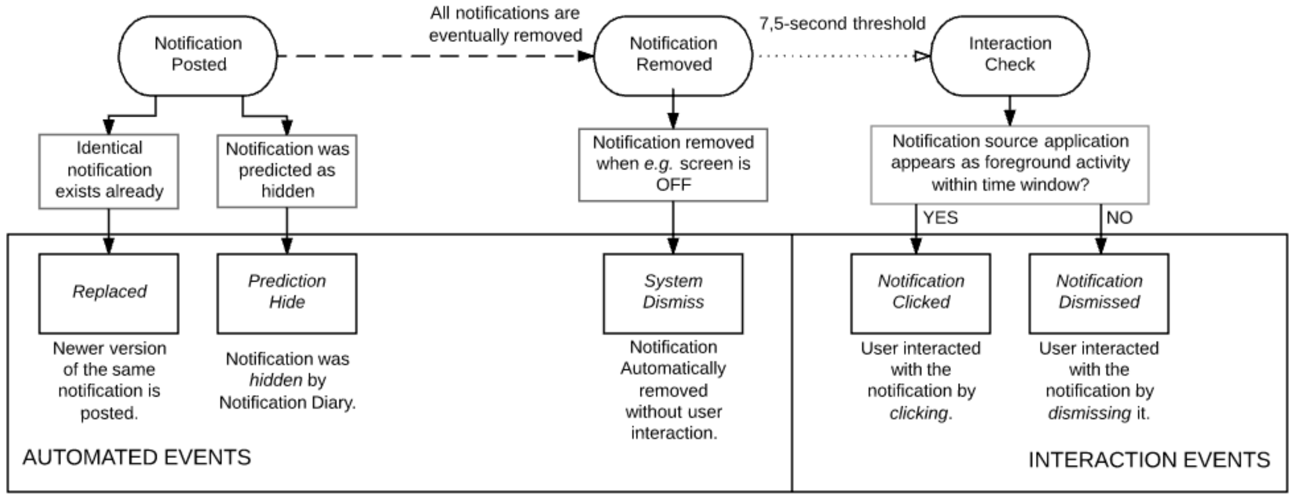


Figure 1. Overview of notifications' interaction states

3.2 Extracting Notification Interactions

On Android, access to notifications' state is limited across applications. We implement a method that indirectly infers user interaction via the foreground applications on the smartphone by means of an Accessibility Service. Most notifications allow only simple interactions, *i.e.*, swipe to dismiss, or click to launch the application. Based on this assessment, notification interactions can be extracted by collecting data on the active foreground application after a notification is removed from the notification tray, as shown in **Error! Reference source not found.**.

When a notification is removed from the notification tray (upper part of Android's main interface), it is either removed programmatically or by user interaction. When a notification is removed, we analyse potential foreground activities taking place within the subsequent 7.5 seconds - during which the majority of Android applications can cold start[1] - and if the notification's source application package exists as one of the foreground activities within this threshold the interaction is labeled as a *click*. Some edge-cases exist, such as if the foreground application is already the same package as the notification source (*e.g.,* you receive a WhatsApp notification from another group discussion while already actively using the application) - in which case when the notification is removed, and the user remains in the same application we are unable to verify whether the notification was *clicked* or *dismissed*. The interaction for this notification is marked, but not included in either click or dismiss class. We acknowledge that not all notifications are interacted with (*i.e.,* manually filtered out or clicked), as some are automatically removed or *replaced*. These events are sub-categorised as automated events. Automatically removed notifications are labeled as *system dismissed*, and can be removed for various reasons, *e.g.* the notification timing out, or the information being received on another device.

Replaced notifications are sent by applications that leverage notification stacks[2] (*e.g.* 'You have 4 new messages') to combine multiple notifications. The notifications included in the stack are posted repeatedly, and also update the same 'title' notification repeatedly. Each individual notification within a stack is also posted repeatedly, causing the amount of arriving notifications to

[1]http://blog.nimbledroid.com/2016/02/17/cold-start-times-of-top-apps.html

[2]https://developer.android.com/guide/topics/ui/notifiers/notifications.html

quickly balloon if this behaviour is unaccounted for. Creating a stack of four messages requires an initial message (**1 notification arrived**), the second message (the stack title – You have 2 new messages – arrived, and the two message notifications get repeated resulting in **3 new arriving notifications**), the third (**4 new messages** including the stack title), and the fourth (**5 new messages**). Thus, instead of just receiving four individual notifications, the stack mechanism results in 13 notifications logged. With our approach of identifying the replaced notifications, 12 out of these 13 (and beyond) notifications are marked as *replaced* and will not interfere with the overall notification count appearing in the user's notification tray.

Since not all applications send their notifications according to the aforementioned standard theme, *e.g.* the Facebook Messenger (the main UI does not create new foreground activities) or Play Store downloads and updates (clicking a notification does not launch the application that generated the notification), we disregard notifications from the following applications: clock, Android system, Play Store downloads, and Facebook Messenger. Some notifications also allow interactions within the notification - *e.g.* Spotify ('Next song), Chromecast ('Play'), and WhatsApp ('Reply') - without removing the notification from the tray. The context and notification contents of each notification are still stored.

Notification Diary can also optionally automatically hide arriving notifications; thus, these notifications are labeled as *hidden*. The process of hiding notifications is based on machine learning predictions, using contextual features, semantic analysis, and information given by the user on content and timing to categorise arriving notification as either *shown* or *hidden*. This process and the associated results are presented later in this paper.

3.3 Labeling Notification Information

The application stores locally the information from each notification, and retroactively asks the user to label each *dismissed* and *clicked* notification in terms of how important it was (UL1, Table 1), and whether or not the notification was presented at an appropriate time (UL2) in a diary view.

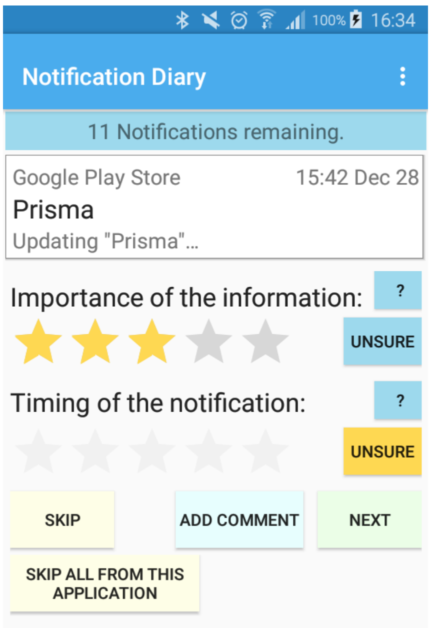


Figure 2. Interface from Notification Diary application highlighting the user-report process. User's current choices (3 out of 5, and 'unsure') are highlighted in yellow.

Diary view can be accessed by launching the application, or clicking a notification sent occasionally by Notification Diary informing the user of notifications with missing labels. When labeling, the user is given information about the contents of the notification, source application, and the time when the notification was interacted with. An example of the labeling interface - the main screen of the application - is shown in Figure 2. The user can also ignore labeling a certain notification if uncertain ('Unsure' option) of either timing or notification contents, or simply wishes not to give labels for any particular notification ('Skip' option). Lastly, the application includes the option to add comments on each labeled notification.

4 Furthering the Knowledge in Notification Interactions

Total of 40 individuals contributed during our data collection period. 113,197 notifications were generated in the dataset, of which majority are *replaced* notifications – indicating that user did not have the opportunity to react to the notification or chose not to - or notifications dismissed by the OS (*'system dismiss'*). Summary of the logged notifications and their interactions are displayed in Table 2. On average, users interacted with 12.3% of all notifications (results of rows E and F in Table 2), and the majority of the interactions (78.9%) are swipes.

A	**Number of study participants:**	40	
B	**Total number of logged notifications:**	113,197	
C	**Average number of daily notifications:**	313.4 (SD = 803.2)	
D	**Average number of daily interactions:**	46.0 (SD = 84.5)	14.7% of C
	Total number of:		
E	***Clicked* notifications**	2,968	2.6% of B
F	***Dismissed* notifications**	11,019	9.7%
G	***Replaced* notifications**	93,563	82.7%
H	***Automatically removed* notifications**	5,614	5.0%
I	**User-labelled content importance**	4,520	

Table 2. Summary of logged notifications, their interactions, and user-labelled information.

4.1 Content Importance

Notifications arriving from different sources are perceived and preferred differently by users [35]. It is considered that user interactions with notifications are directly indicative of the perceived importance or usefulness of the notification [18,25]. We first investigate if the relationship between notifications and user interactions is more complex than we previously assumed - *i.e.* is the choice of interaction directly indicative of the notification's perceived importance.

Using the Chi-Squared test, we verify that the distribution of reported content importance (on a scale from 0.5 to 5) is significantly different for the clicked and dismissed notifications ($x^2 = 207.9$, df = 10, $p < .05$). The average importance for clicked notifications is 3.91, and 3.22 for the dismissed. However, while 49.2% of the clicked notifications are ranked as high importance (5), 44.2% of the dismissed notifications are similarly ranked as high importance. The significant difference lies in the other end, as 12.6% of the clicked notifications and 28.6% of the dismissed are ranked as low importance (1 or below). As not every notification was labeled with user-provided information, we verify the relationship between reported content importance and interaction choice by investigating the labeled notifications specifically. Using the Chi-Squared test we can verify significance between the two variables (x^2 = 211.07, df = 9, p < .05) and measure an acceptable effect size using Cramer's V (= .216).

As it is also reported that the source application of the notification plays a role in its importance, we also wanted to explore whether the interaction choice can apply to determining the importance of a notification, based solely on its source category. We apply an application categorisation of each notification, according to its source application package, resulting in a generic application category (*e.g.,* "Social and Internet", "Productivity", "Games", *etc.*). The application category is retrieved from the Google Play Store and then a generic category is applied according to the original category. Using the user-given labels of content importance, we measure the effect using Pearson's Chi-Squared test between the reported content importance values, and the application categories. We can verify that the category has an impact on the *content importance* (x^2 = 1517, p < .05). However, as seen in Figure 3 where the categories are ranked according to their mean content importance ('News' highest, 'Weather' lowest), the interactions with notifications (or the user's neglect to interact) from different sources differ drastically, and the reported content importance does not correlate with the interaction selections.

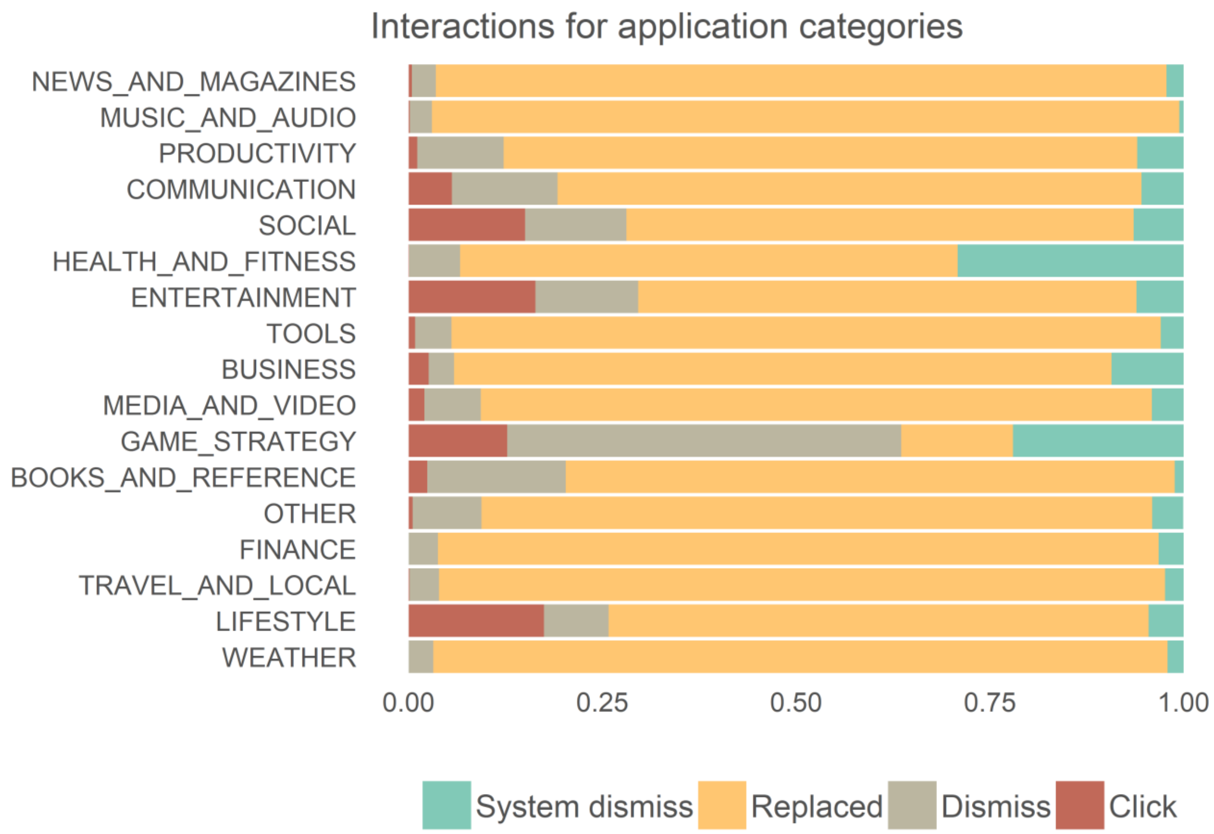


Figure 3. Interaction choices for different application categories, ordered based on mean content importance. From most important (top) to least important (bottom).

The interaction decision is clearly made separately on a notification-to-notification basis and driven by a combination of factors. Previous work suggests that users selectively prefer notifications from different sources [8,35], and explicit interaction with the notification is indicative of user preference on ultimately seeing particular notifications [18]. Here, we show that the interaction alone does not describe these user preferences comprehensively, and larger factors impact the interaction decisions. Consider the following combination of results:

- A high number of daily notifications (row C in Table 2), low interaction ratio with notifications (row D), and high number of ignored or missed notifications (row G).
- Strong likelihood of swiping notifications considered important.
- Discrepancy between source application importance and interaction choice (Figure 3).

With these results we can reasonably say that the binary classification of desired notifications using merely the interactions (click or dismiss) as measurements is not only inadequate (based on how infrequently users interact), but also likely incorrect, as surely users place value on more than just the 21.1% of notifications they opt to click, as showcased by the high frequency of dismissed yet important notifications.

4.2 Understanding Interaction Choices

Aside from delivery context in terms of device usage and notification contents, a number of other factors can influence the interaction choice. Time of day can play a role in user availability and activeness to respond [1,5,38] , as can fatigue due to information overload [11,37].

According to the Chi-Squared test there are differences in number of notifications across different hours of the day (x^2 = 33,189, df = 23, p < .05) and using Pearson's correlation, we can observe a reasonable effect size (r = .38, p < .05) between the time of day (hour using a 24-hour clock) and the number of hourly notifications. Majority of the notifications arrive after work hours (44.2% of all notification arrive between 5 pm and 12 midnight). While there is a significant effect on the hour of the day on user's option to interact with a notification (Chi-Squared, x^2 = 5920.3, df = 92, p < .05), the only noticeable difference in ratios between the interaction choices is from 4pm to 6pm, when users are more likely to *click* notifications. This behaviour is likely associated with daily work hours, and due to users *e.g.*, actively responding to messages received earlier during the day.

We hypothesise that one other factor influencing user's interaction choice is being overloaded with information, *i.e.*, becoming fatigued and neglecting to interact when presented with larger quantities of notifications. Initially, we use Pearson's product-moment correlation to observe a weak correlation (*r* = .192, p < .05) between the daily number of arriving notifications and the daily number of clicked notifications, and a strong correlation (*r* = .833, p < .05) on the daily number of dismissed notifications. Investigating this behaviour further, both interaction types show increases in interactions, when the number of daily notifications is relatively low (below hundred daily notifications). However, user's attention span seems to diminish as the number of daily notifications increase, as the frequency at which notifications are interacted strongly dips beyond this threshold.

To verify this impact on interaction frequencies, we apply a window size of 20 on the number of arriving daily notifications and merge data within each window (*i.e.*, 0-20, 20-40, …). We combine windows with an insufficient number of samples together to produce a significant mean ratio within each window of size 20 or larger. We then measure the interaction ratios using Pearson's correlation and can reveal a negative correlation for both ratio of clicked notifications (*r* = -.85, p < .05) and dismissed notifications (*r* = -.82, p < .05). Analysing the different windows and the window size's impact on the ratio, we can see that the overall willingness to interact diminishes beyond the aforementioned 100 daily notifications, although the effect is not as drastic for willingness to dismiss. The difference in interaction ratio beyond and after the 100 notifications received threshold is -.106 for clicking, and -.073 for dismissing, and the dismiss ratio levels higher (at .119) than the click ratio (at .052). The different window sizes and corresponding interaction ratios are visualised in Figure 4. This significantly implies that when users receive a higher number of notifications, they more frequently neglect to interact with notifications.

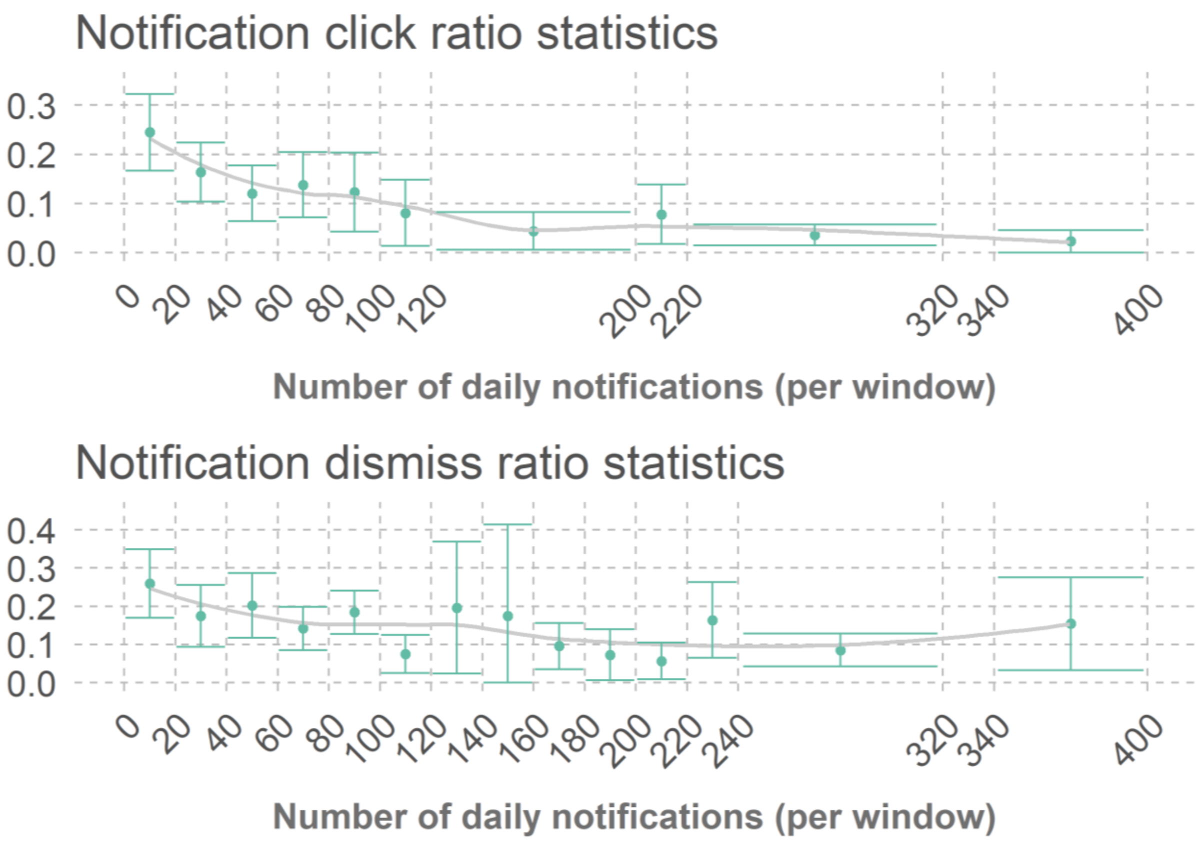


Figure 4. Interaction ratios for clicking and dismissing notifications, according to the total number of received daily notifications.

Proving our hypothesis is still incomplete, as the previous results merely generalises the behaviour, but the longitudinal effect of information overload is yet to explore – how does the number of notifications received affect the interaction choice for an individual notification?

For each notification, we crawl the dataset for the number of notifications that arrived at that specific user within six distinct time windows – during the previous 60 seconds, 5 minutes, 10 minutes, 30 minutes, 60 minutes, and 4 hours. We then combine the information from all users within each time window and calculate an interaction ratio: the number of notifications that the user interacted with within that time window vs. the total number of notifications that arrived within that time window – and a click ratio, according to the number of notifications within the time window. For example, when two notifications were received within the previous 60 seconds, users interacted with the new notification 70 times and neglected to interact 1,239 times, thus resulting in an interaction ratio of .053 for a 60-second time window and two notifications received. This process is replicated for each time window and each number of previous notifications. In each window, we observe the effect of more notifications arriving resulting in less interactions by first verifying the existence of the difference with Chi-Squared (x^2 = [750 … 6115] and $p < .05$ for all windows) and measuring the size of the effect with Cramer's V (value ranging from 0.081 to 0.232).

Next, we analyse where (for which time window) the effect of the higher number of notifications resulting in fewer interactions is potentially strongest, using Pearson's correlation. The effect is smallest (r = -.37) within a 60-second time window, increases up to 30 minutes (r = -.66), and then gradually diminishes again until it reaches similar value than for the first window (r = -.36, p < .05 for all window sizes). As this indicates a significant effect, we then calculate the interaction ratio in ten quantiles within each window. The results are visualised in Figure 5 with the red line annotating the overall mean interaction ratio (12.3%).

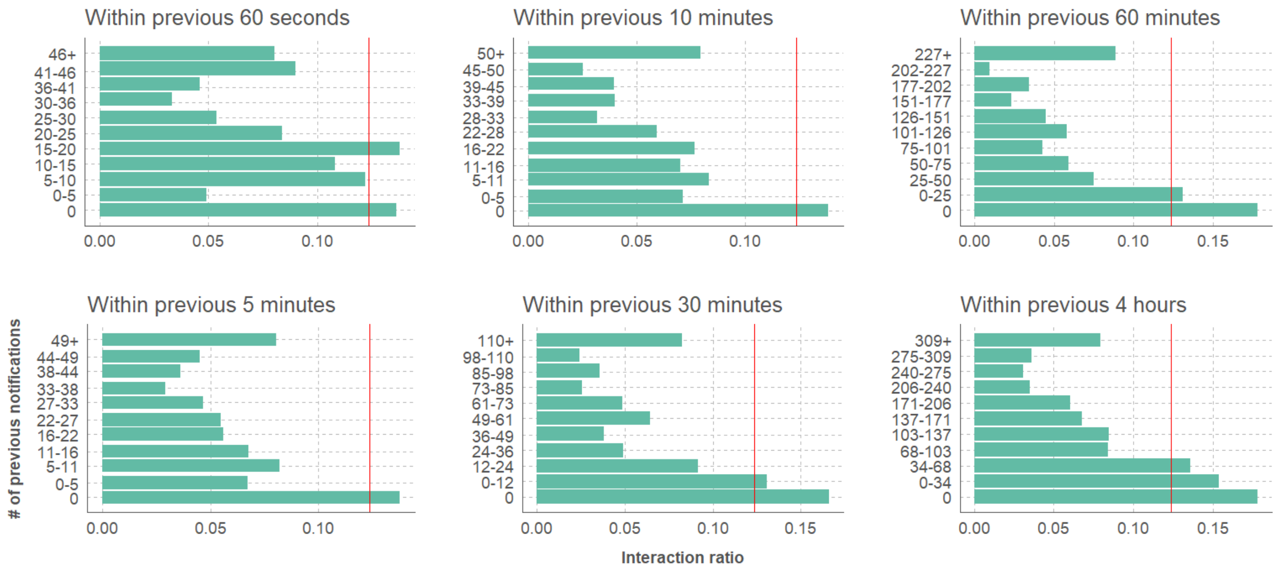


Figure 5. Likelihood of a user interacting with a new notification according to number of previously arrived notifications within specific time windows. The red line annotates the overall mean interaction ratio (12.3%).

Within all of these time windows, users are significantly more likely to interact with a new notification *if no previous notifications had arrived during the time window*. The interaction ratio in these cases surpasses the overall mean. The user's willingness to repeatedly interact also diminishes beyond 60 seconds, only to return at 30 minutes or more, and even there it is only apparent if only a small number of notifications had previously arrived. For the first 60-second window, we see that the users remain active, even if numerous (*e.g.*, 15-20) notifications just arrived. This behaviour often revolves around the use of communication applications and related correspondence. Overall, the results in Figure 5 indicate that:

a) Users exhibit **interaction fatigue** quickly, i.e., users are reluctant to repeatedly interact with arriving notifications.
b) The important **information** in the notification tray **gets lost** when numerous notifications arrive simultaneously - or within a brief time window, *e.g.*, 5-30 seconds - and the users are unable to locate and interact with specific notifications they otherwise would interact with.
c) The effects of b) **can compound** when and **if users neglect to interact** (manually filter) away unnecessary information from the notification tray.

In the next chapter, we aim to identify distinct usage styles and their associated problems related to lack of interacting, and how prominent those problems are. Lack of interaction with notifications quickly leads to users' devices being overwhelmed with notifications, which reduces the usefulness of the medium and the amount of information provided by notifications in general. Lack of interaction also highlights the need for an automatic management system – as humans seemingly often neglect or opt not to do so.

5 Distinct Notification Filtering Styles and Habits

As the interaction choices differ for each individual user, with each user having personal preferences and configurations – such as installed applications and generic smartphone usage traits, we next aim to differentiate between different types of users. Identifying *groups* of users as opposed to generalising, or treating each user individually, can be an effective mean of identifying similarities in users [23], and in developing accurate and autonomous intelligent systems [39]. Clustering methods allow us to differentiate between different usage styles according to, in our case, the *interaction frequencies* and the *number of arriving notifications*.

With our dataset of 40 users, we apply a *k*-means clustering algorithm and iterate with varying number of clusters with *k* = [2:10] to represent a varying number of *different user types*. Since the centroids generated by k-means can have slight internal variance (*i.e.,* the results for the same dataset for *k* value x can produce slightly different results), we also iterate through each value of *k* ten times. We then measure an *evaluation score* for each cluster configuration using a scoring mechanism using both Davies-Bouldin and Dunn indices. Both indices measure the level of internal agreement of the clusters (intra-cluster similarity) and the separation between clusters (distances between generated clusters). Both indices have the same (50%) weight for the evaluation score. As the Davies-Bouldin index is minimised, the score is inverted for the calculation of the evaluation score. Thus, the variables help us identify a cluster configuration (which user belong to which group) where each cluster contains users with similar usage styles, each cluster contains a similar number of users, and the overall configuration is not needlessly fragmented, *i.e.*, some clusters only containing one or two users. We identify the best configuration to be the following four different user types, with their differences highlighted in Figure 6.

- **Group A (N = 5)**: Active users, who interact with notifications most frequently (16.87% of all notifications), while receiving the second highest number of daily notifications (M = 242.5, SD = 86.8).
- **Group B (5)**: Show the highest negligence towards notifications (87.6% replaced ratio) while also receiving the highest number of daily notifications (M = 329.9, SD = 159.8).
- **Group C (17)**: Receive the least notifications (M = 60.2, SD = 49.9), and interact reasonable frequently (13.9%)
- **Group D (13)**: Least active in *clicking* notifications (M = 6.91 daily clicks, < 20.3, 11.4, 9.8 for groups A, B, C, respectively), and neglect notifications often (82.5%), with a reasonable number of daily notifications (M = 151.2, SD = 142.6).

These groups show not only differences in the number of notifications, but significant differences in ways to respond to and manually filter out notifications. The results seem to indicate similarities to the previous chapter, as a high number of notifications seems to lead to less frequent interactions, but similar low interaction frequency does not exist for usage styles where only a handful of notifications were received during the day. Personal differences are also showcased in more detail, as the Group A members retain their interaction activity even with a high number of daily notifications.

The most notable recognisable result is still the high frequency of replaced notifications for all usage styles. Dividing the usage styles further into passive (groups B and D) and active (A and C) according to their interaction frequency, all groups still maintain retain relatively high negligence to notifications (A with 77.6% and C with 83.2% replaced ratio). The problem with neglecting to filter away notifications is not necessarily with information being directly lost, as surely the notifications are at some point still *seen* by the user. The problem relies more in the notification tray becoming overly cluttered, at which point the information available in the notification tray significantly diminishes. Imagine the notification tray containing more than a few items, at which point some notifications shift away from the initial view and become hidden in the bottom of the list. Optionally, the information in *bundled* notifications is also limited, as the user has no direct access to the notification contents (*e.g.*, single messages), only to the top-level notifications (*e.g.*, '*You have 12 new messages*'). This behaviour, recognised quantitatively for the first time in this paper, signifies the importance of either active manual (via interaction), or intelligently autonomous notification filter mechanisms. In the next chapter, we explore this notion of intelligent filtering based on notification contents, in addition to the more traditional, purely contextual, filtering.

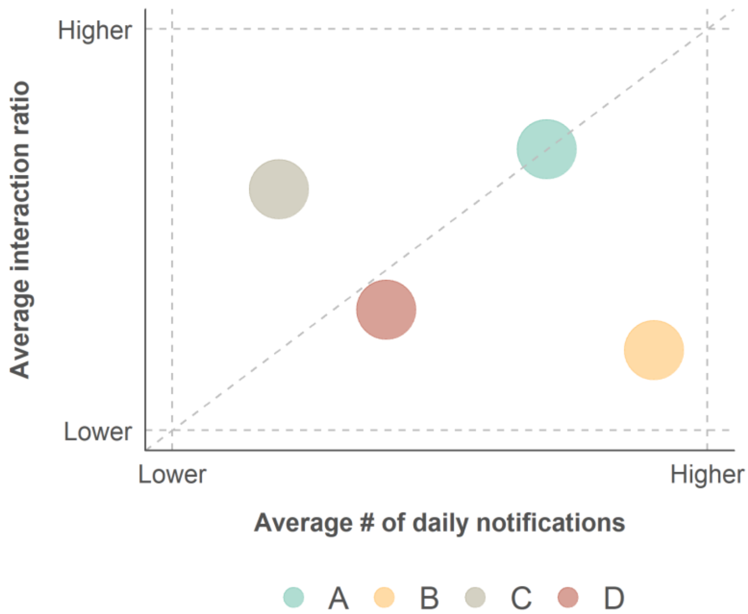


Figure 6. Usage styles of different user groups according to the number of daily notifications and the frequency of user interacting with notifications.

6 Combined (Content-Contextual) Intelligent Notification Filtering

In our application, the user can enable two options within the prediction mode – an intelligent method for hiding unwanted notifications – which can be enabled once sufficient amount of training data is collected by the application. The first option (prediction mode) enables the application to generate machine learning models, and the second option (notification hiding) hides incoming notifications according to the insight provided by the generated models. We use both clicked notifications, and user-given labels as training data for the models, and the prediction mode can be enabled once a minimum of 50 training data points is acquired.

Based on the user reported labeling of context and timing, and the user interaction with the notification, we can map the preferred action for each notification based on previous work: show, hide, or defer [2,20] (Table 3).

		Was the notification appropriately timed?	
		Yes	**No**
Are the contents important or relevant?	**Yes**	Always show the notification	Defer until next use
	No	Preferably hide*	Always hide the notification

Table 3. Proposed actions for notification filtering based on user-reported information.

It is not intuitively clear what should be done about notifications that are non-interrupting but contain no important information (third result column marked with a *). To solve this ambiguity, we use a 70% (contents) - 30% (timing) weighted average, based on the effect of *content* and *time of delivery* of mobile interruptions, originally presented in [8]. We choose the action (*show* or *hide*) according to the value with a threshold of three (mean on a five-point scale used for evaluation both variables) - values less than three indicate *hide*, and higher than three indicate *show*. Clicked notifications are assumed to be both of importance to the user and appropriately timed and are categorised as *show*. The action for *deferring* (*i.e.,* delaying) notifications to a later time was omitted from our application. We opted to use a binary classification to *show* and *hide* to simplify the application, and our experiment - adding the defer option would make both the prediction mode functionality, as well as post-experiment analysis, overly complex, as there would exist a subset of notifications functioning differently (deferred) than other notifications. To create a machine learning model to predict whether a specific notification should be shown or hidden to the user, in addition to the context we wish to also understand the semantic characteristics of the arriving notifications and process the data of existing notifications.

6.1 Text Pre-processing

Our analysis identifies clusters (or bins) of related keywords that appear in notifications, and then characterises each notification based on which clusters of keywords it uses. This analysis is conducted for each user independently and locally on their phone, and therefore the keyword clusters vary between users.

All notification text is first pre-processed by transforming it to lowercase, removing all non-alphabetical or numerical symbols and stop-words (commonly used words) such as 'and', 'to', or 'be'. We then create a graph of related words, where *nodes* denote **words**, and **words that appear together** in the same notification are connected by *edges*. The *weight* of an edge is the frequency of those two words appearing together. Each node also includes the frequency of a given word being used within the dataset (*'size'*). The nodes with the largest *size* are then selected as k = {10,15,20,25,30} *centroids* with a minimum *distance* of at least two nodes apart from each other. Words that never appear with other words (*i.e.* "islands") are then discarded. The range of k values is based on evaluating the prediction accuracy from data collected in our pilot study, which showed that with k-values outside (above or below) of this range the prediction accuracy rapidly deteriorates.

We then shuffle the nodes and create k *clusters* by assigning each non-centroid node to a centroid (*cluster*) within distance d = {1,2,3…} and removing it from the next round of iteration, until no more nodes remain. Shuffling ensures no bias based on *e.g.*, first character of a node (word). In case of a tie, nodes are placed in clusters based on their weight (frequency of appearing together) to the node they share their edge with. This operation creates k-*word bins* containing words that appear together in the notification contents. Each word bin is then used as an individual factor for the machine learning model, and the value of the bin is the number of words in the notification contents that match the contents of the bin.

6.2 Combined Prediction Analysis

After the user has labeled 50 notifications – the minimum amount of training data we assume to create a somewhat accurate classifier – the user has the option to enable predictions, resulting in the application creating the first prediction model, and then automatically updating this model periodically, every 48 hours. Since the computations are performed on the client we opted to evaluate and rely on lightweight classifiers, which we evaluated using data collected during our pilot testing. We use the C4.5 classifier, using the WEKA java library [10], due to its efficiency in previous work using similar factor types [17], and perform all calculations during run-time in the application, as a background process. The choice of more complex classifiers, e*.g.,* Random Forest or SVM, was due to mobile run-time analysis, *e.g.*, battery over increased computation time.

Using the combination of contextual variables and the notification content analysis we built a machine learning classifier using the dismissed and clicked notifications as training data. The classifier uses two classes (*show* and *hide*) to determine the outcome of each notification. For dismissed notifications, the class label is based on the weighted average [8] of the importance (.7 weight) and timing (.3 weight) provided by the user, or the value of an individual entry (importance or timing) if the user was unsure for either value. For clicked notifications, the class is directly set as *show*.

The model is trained and created on the client, as we wanted to ensure the user's privacy and withhold them from having to share notification contents (*e.g.*, private messages, emails) with the researchers. The calculations for creating new models are automatically performed every 48 hours and updated if the new model is considered more accurate than the previously used model. The process is performed as a background activity and is only performed when the device is charging.

The application creates the training data from the available information on the device and performs balancing of the data by downsampling the majority class appropriately. This reduces the bias due to overfitting. A classifier is then generated for each cluster size k = {10,15,20,25,30} and each classifier is evaluated using 10-fold cross-validation. We use a combination of correctly classified instances, ROC-area, number of false negatives and false positives, and Kappa to compare between evaluation results. The classifier with the best evaluation score is then stored alongside the training data, and the generated word bins (clusters) for selected cluster size k. The process of iterating through different k (word bin) values, generating the word bins, and training and evaluating each created classifier takes approximately 1-5 minutes (measured during our piloting phase), depending on the amount of training data and device capabilities. The user is presented with statistics (such as overall accuracy and the estimated probability of important notifications being falsely hidden) of the generated model, and a summary of words that were either considered important and unimportant.

Arriving notifications are classified using the currently stored classifier and the word bins associated with each classifier. The current device usage context is extracted when the notification arrives, and the notification contents are mapped to the corresponding word bins. The notification instance is then classified as *show* or *hide* by the stored machine learning model and the decision is sent to the OS in case the notification is deemed as unwanted and should be discarded. The notifications classified as *hide* are automatically discarded from the notification tray by Notification Diary. The process of handling an arriving notification is detailed in Figure 7.

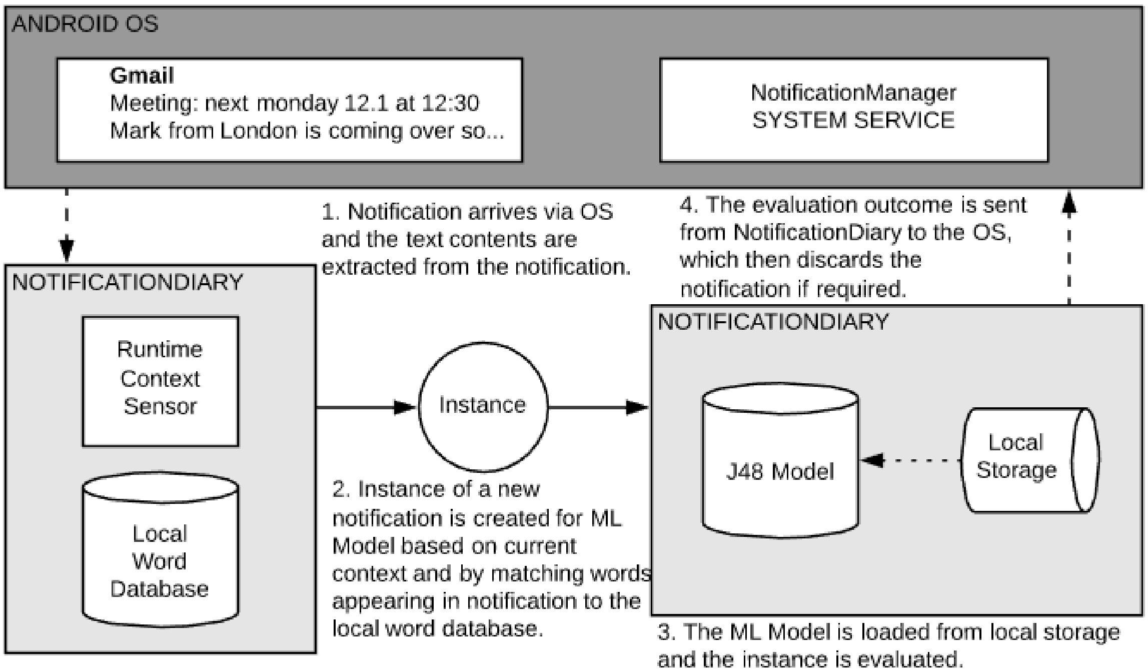


Figure 7. Stages of the prediction analysis from the beginning (new notification arrives in the notification tray) to the end (the notification is handled).

However, Android's NotificationListener class only allows access to notifications that are pending by other applications. The user still receives cues of these incoming notifications if unattended - since Notification Diary only observes them after they are already posted with any accompanied cues. When the prediction mode is enabled, by default Notification Diary mutes all alarms, vibrations, and sounds, and plays corresponding sound cues or vibration when needed (*e.g.,* if a notification with a cue arrives, or if there is an incoming call), similar to [34]. This approach ensures that the user does not receive these cues when notifications arrive, and unwanted notifications can then be silently discarded.

6.3 Predicting Notification Relevance

As enabling the prediction mode in the Notification Diary application partly interferes with normal smartphone use by silencing the device, not every user felt comfortable using this mode. A total of 33 users generated a total of 313 machine learning models (M = 9.28, SD = 15.75) during their use, with an average of 215.69 (SD = 289.55) training data points. Classifier accuracy and ROC describe the overall accuracy of the model, while the Kappa statistic indicates how much better the model performs compared to a random guess (0...1, higher is better). The mean classifier accuracy is 91.1% (SD = 5.8%), ROC 81.1% (SD = 15.4%), and Kappa .65 (SD = .25).

Table 4 shows the summary of the different groups, and the generated models and their accuracy. Analysing the accuracy characteristics of different user groups with the use of one-way ANOVA we can identify significant ($F = 3.88$, $p < .05$) differences in the Kappa values across the groups. In this case, lower Kappa values indicate overfitting to displaying notifications too frequently, and this occurs more in models generated by members of Group C and Group D – the groups of users who generally received fewer notifications. As both clicked notifications, indicating user preference to said notification, and the user-provided labels are used to train the models, we can also see the added influence of user-given labels on the models' performance. This effect is highlighted on group A on the ROC-area and kappa statistic, which are more reliable indicators of model performance than accuracy, as the accuracy can be more strongly influenced by overfitting.

While our models are accurate, the false positive rates of either *showing* or *hiding* a notification indicate differences in performance. The false positive rate (FPR) of showing an unwanted notification is significantly higher than the false negative ratio ($0.29 > 0.04$, $t = 17.85$, $p < .05$ using Student's *t*-test), indicating that the generated models are likely biased towards showing notifications. This can be due to multiple reasons, of which one likely candidate is the user indicating high content importance for the majority of the notifications, causing the training data to be overfitted for the *show* class. The best performing models, however, are the ones where such bias does not exist and FPR is significantly lower. There is a strong negative correlation between both Kappa and FPR ($r = -.95$, $p < .05$) and ROC and FPR ($r = -.87$, $p < .05$), which indicates the importance of FPR in overall model performance. The overall model accuracy is highlighted by measuring the number of clicks for notifications shown without predictions, compared to those shown when the prediction model and automatic filtering is enabled. Chi-Squared indicates significant ($p < .05$, x^2=34376) increases in click ratios ($.07 > .01$, $N = 27385$, $N = 85812$, respectively) for automatic filtering. It should be noted that in our experiment we are unable to verify whether the user experiences that a notification was correctly hidden or not, or if a notification was hidden via the application (whether the notification hiding option was enabled when a notification arrives), but the increase in click ratios when the models are generated (prediction mode is enabled) strongly indicates its influence.

	Group A	Group B	Group C	Group D	Total
Total # of users	5	5	17	13	40
Total # of notifications	16599	12121	20995	63482	113197
Total # of labelled notifications	1658 (10.0%)	130 (1.1%)	1252 (6.0%)	1503 (2.4%)	4543 (4.0%)
Mean # of daily notifications	1185.643	527.0	134.58	377.87	313.4
Machine Learning Model Accuracy					
Total # of created models	18	86	166	41	313
Mean # of training data per model	302 ± 779	221 ± 69	128 ± 90	300 ± 488	186 ± 272
Mean accuracy	90.6% ± 5.4%	90.4% ± 4.9%	92.0% ± 5.6%	89.6% ± 7.0%	91.1% ± 5.8%
Mean ROC-area	87.0% ± 11.4%	82.7% ± 8.0%	79.3% ± 18.0%	83.0% ± 15.9%	81.1% ± 15.4%
Mean Kappa	0.74 ± 0.21	0.70 ± 0.16	0.62 ± 0.30	0.66 ± 0.28	0.65 ± 0.27

Table 4. Summary of collected variables and statistics of generated machine learning models.

Exploring the table and the previously reported differences in interaction frequencies in more detail, it becomes evident that the models generated by Group A are most accurate likely due to a) high number of labelled notifications indicating in more detail how users perceive the notification content, and b) high interaction frequency, again enabling more detailed training data. The lower labeling frequencies cause the generated models to likely suffer from overfitting. If most the information gained in the training data are the clicked notifications, *i.e.* desired notifications, this leads to lower values in kappa statistic and artificially high accuracy.

7 Discussion

Intelligent notification management systems traditionally assess the user's situation via usage context for delivering notifications. The importance of individual notifications (and their delivery context) is measured via click-ratios under the binary assumption that clicked notifications are desired and important, while dismissed notifications are not seen as important. This notion has been the basis of multiple works [18,25,29], and while the importance of individual notifications' contents has been revealed to hold more information about the user's preference than the situation, it has proven difficult to effectively train autonomous intelligent management mechanisms to understand the importance of individual notifications. We set out to investigate this binary nature of notification interactions in more detail, hoping to both verify the validity of previous assumptions and to collect more detailed information about notification interactions in general.

Previous work suggests [8,35] that a notification's source plays a big role in the user's preference to see a notification. While certainly true, individual details play a much larger role in the user's preferences, as indicated by the data collected in our study. The ratio of clicking, dismissing, or 'ignoring' notifications vary significantly across different notification source categories, and even for individual notifications that are labeled according to their perceived importance. Thus, drawing conclusions on notification importance solely from the interactions *or* the source is not supported by our results. The interaction choice is likely a result of a much larger set of features including the notification source, notification contents, perceived importance of the type of notifications, and the enveloping context (*i.e.*, the situation in which the smartphone was used).

The notion of click ratios as evaluators of user's preference or attentiveness to notifications is warranted in some cases – namely when the users opt in to certain notifications, *e.g.*, to particular news items or prompts in [25,29] – but can be inefficient when attempting to comprehensively manage notifications. Such systems require the knowledge of which notifications are considered *unwanted*. Considering the binary categorisation of clicked notifications as inherently desired, and the ambiguous nature of dismissed (or ignored) notifications, it is not possible to correctly assess which notifications are actually unwanted (and thus should be opted out). More details are required to correctly assess which notifications should be filtered out.

7.1 Enhancing Automatic Notification Filtering with Semantic Analysis

Previously generated and neglected notifications often ending up taking unnecessary space in the notification tray and portion of users neglect to interact with notifications *at all*, leading to the notification tray becoming overcrowded, and severely diminishing the quality of future information. When multiple notifications are persistently bundled together (*'You have 37 messages in 4 chats'*) the information provided by a new notification is minimal, as it only adds up to the bundle, and the item on the notification tray offers no detailed information. Similarly, too many individual items in the notification tray can hide portions of the information. This begs for ways to automatically filter out the unnecessary notifications.

Poppinga *et al.* [3] used contextual variables to predict opportune moments to interrupt the user by presenting a notification and reached a reasonable accuracy of 77%. Including semantic analysis of the notification's content can increase accuracy by 14.8 percentage points, resulting in an average accuracy of 91.1% in our experiment. Okoshi *et al.* [6] deployed a similar model in a real-world application combining both notification contents (Yahoo news) and contextual analysis to assess moments for delivering new items. Their work reports that deferring the notifications accurately decreases the click delay and that their approach continuously increased the click rate throughout the experiment. We can observe similar results in increased click rates (from .01 to .07) with the prediction mode enabled.

In [7], a comparison was performed between user-provided rules and personalised models, with the use of user given labels to notifications, as well as the user's social circles. Our results show an increase in comparison to the user-defined rules with the use of Random Forest, which reached approximately 61% accuracy in filtering out unwanted predictions. The computationally generated rule-based approach used in PrefMiner [5] analysed the **contents** and **source** of each notification, and while being highly sensitive (*i.e.*, was careful not to hide important notifications) reduced the number of unwanted notifications by 48% overall. The measurement used in PrefMiner [5] determined how many of the dismissed notifications could be pre-filtered as unwanted. Combination of these approaches results in a significantly higher prediction accuracy, and our best performing models highlight a low False Positive Ratio (FPR), indicating that the users would receive significantly less unwanted notifications.

7.2 User Types and Interaction Burden

We showcase improvements in prediction accuracy using a combination of contextual features, and semantic analysis of notification contents based on the *varying content importance*. The user perception of notifications is based on details of individual notifications, as well as *personal preferences* - after all, most notifications contain highly personal content. It is known that smartphone users show diversity in their application selections [23], application use [24], and responsiveness to prompts [25]. Our assumption is, this should be true for interacting with arriving notifications as well. Our analysis reveals four distinct user groups, and we are able to show diversity in *the number of notifications* different smartphone user groups receive, as well as *how different user groups interact* with the incoming notifications.

Distinct groups receive different amounts of notifications, and their trend of interaction ratios (clicking or dismissing a notification) follows a common observation: the potential information overload caused from *too many* notifications tends to decrease both click ratios (significantly) and dismiss ratios (less significantly). Generally speaking, this means the burden caused by notification overload and the end outcome of both *reduced user experience*, and deduction in the *received information*. The threshold for reduced click ratio seems to be at around 100-120 daily notifications (including the new notifications arriving in notification *stacks*), after which the click frequency drops significantly. For dismissing notifications, users begin to feel burdened at around 140-160daily notifications.

During our pilot testing of the Notification Diary application, we noticed one of the researchers used as test subjects habitually ignoring all incoming notifications and leaving them present in the notification diary for extended periods of time. We thought this behaviour was peculiar, but surprisingly, this behaviour also existed within our dataset. Both Group B and D users habitually neglect to interact with arriving notifications, indicated by the high *replaced* ratio, meaning notifications or notification stacks remain in the notification tray until they become updated. Part of the explanation could be the presence of a communication app sending constant flow of messages, but the total number of notifications arriving for these groups (especially Group D) does not indicate that they received exceedingly many notifications - note that the Android OS uses an internal threshold (measured in seconds) to block certain applications from obsessive notification spam and does not send cues for *all* arriving notifications. Other culprits could be *e.g.*, group chats with content that is generally deemed unimportant. These types of notifications clearly signify the need for notification management overall, as they cause unnecessary overhead and depreciation of the quality of information presented by notifications. After all, the notification tray has limited space and while the applications can request priority (and OS can assign priority) to notifications that should be shown at the top, if the notifications are not handled (*i.e.*, dismissed by the user, dismissed by the system, or filtered automatically), the notification tray will quickly become overpopulated by unimportant content. This signifies the importance of content analysis done on an individual notification-by-notification basis when filtering out unwanted notifications. Two of our application users contacted us during the experiment via email, and wanted to emphasise the usefulness of our approach - even if the approach for hiding notifications and notification cues of our application could be considered somewhat crude: ”*I really liked the idea of the application and it was the reason why the joined the study. An application that can hide unwanted notifications and* ***can understand notification contents would be extremely useful****.”*

7.3 Do not Block, Clean, and Going Forward

This issue of crowded notification trays and the users' frequent neglect to manually filter out unwanted notifications should definitely lead to new methods for notification management. Especially for users who habitually ignore and do not filter out and interact with notifications themselves, it becomes increasingly important to manage their notifications to reduce information overload - in order to ensure new and important notifications do not simply get lost in the notification tray

The current ranking system for displaying notifications in order of importance is limited, as the priority value can be specified by the application, and not by the notification contents - which is essentially the thing that matters the most, especially when the importance is highly contested between similar notifications. A better approach would be to both a) *filter out unnecessary notifications* (methodology most commonly researched), and to also b) *ensure that the notification tray is not overloaded* by **limiting the number of shown notifications** – *i.e.,* cleaning when the tray becomes overcrowded. If the user only has two notifications showing, there is no immediate need to filter anything out, since the user has access to all presented information. But when the number of concurrent notifications increases, precedence should be given to the ones with important content. This approach would also allow the notification management system to *filter out old notifications*, which are no longer considered high priority, but was displayed because there was no immediate need for cleaning. Lastly, newer notifications could also be given preference over notifications that have already existed on the notification tray for longer periods of time with the information already likely consumed.

Android 8.0 (Oreo) offers developers a new method for designing notifications, *i.e.*, *notification channels*, allowing developers to discern notifications' importance and group them by content similarity. While promising customisability, it still lacks an understanding of *what the user ultimately deems as important*. In other words, it should not be the developer imposing the rules, but the user! Admittedly, understanding the importance of notification contents to individual users without relying on user feedback is challenging, however core to our findings. Relying solely on binary interactions like the click or dismiss ratios is not sufficient: dismissing notifications does not indicate low importance (and that most notifications are dismissed anyway). Thus, new metrics for evaluating the perceived importance need to be considered. The new application-side management methods, presented by the new notification channels, potentially offer solutions to this as users are able to interact with notifications with more extensive methods and developers can design notifications with more details.

8 Conclusion

We collected smartphone notification data in combination with user-labelled information on the importance and timing of notifications. Our results highlight that previous work, which assumed that user's perceived importance of a notification correlates with the notification's interaction, is unfounded in generating knowledge for automatically filtering out unwanted notifications. Many users frequently and habitually dismiss or ignore the majority of their notifications – regardless of their perceived importance. This further complicates notification filtering mechanisms relying solely on user interaction. Understanding notification content preference via semantic analysis increases the accuracy of prediction models aimed at automatically detecting unwanted notifications. Our work challenges researchers of notification management systems to understand user's personal preferences of notification contents and interaction choices more accurately. Future work must focus on developing user-driven notification management systems.

Acknowledgements

This work is partially funded by the Academy of Finland (Grants 286386-CPDSS, 285459-iSCIENCE, 304925-CARE, 313224-STOP), and Marie Skłodowska-Curie Actions (645706-GRAGE).

9 References

[1] Niels van Berkel, Denzil Ferreira and Vassilis Kostakos. 2017. The Experience Sampling Method on Mobile Devices. *ACM Comput. Surv.*, 50 (6). 1-40. DOI: https://doi.org/10.1145/3123988

[2] Herbert H Clark. 1996. *Using language*. Cambridge university press.

[3] Luigi De Russis and Alberto Monge Roffarello. 2017. On the Benefit of Adding User Preferences to Notification Delivery.

[4] Tilman Dingler and Martin Pielot. 2015. I'll Be There for You: Quantifying Attentiveness Towards Mobile Messaging. In *17th International Conference on Human-Computer Interaction with Mobile Devices and Services*, ACM, 1-5. DOI: https://doi.org/10.1145/2785830.2785840

[5] 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*, ACM, 91-100. DOI: https://doi.org/10.1145/2628363.2628367

[6] Joel E Fischer, Chris Greenhalgh and Steve Benford. Year. Investigating episodes of mobile phone activity as indicators of opportune moments to deliver notifications. In *Proceedings of the 13th international conference on human computer interaction with mobile devices and services*, ACM, 181-190.

[7] Joel E. Fischer, Chris Greenhalgh and Steve Benford. 2011. Investigating Episodes of Mobile Phone Activity As Indicators of Opportune Moments to Deliver Notifications. In *International Conference on Human Computer Interaction with Mobile Devices and Services*, ACM, 181-190. DOI: https://doi.org/10.1145/2037373.2037402

[8] Joel E. Fischer, Nick Yee, Victoria Bellotti, Nathan Good, Steve Benford and Chris Greenhalgh. 2010. Effects of Content and Time of Delivery on Receptivity to Mobile Interruptions. In *International Conference on Human-Computer Interaction with Mobile Devices and Services*, Lisbon, Portugal, ACM, 103-112. DOI: https://doi.org/10.1145/1851600.1851620

[9] 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. DOI: https://doi.org/10.1145/2750858.2804290

[10] 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. DOI: https://doi.org/10.1145/1656274.1656278

[11] Starr R Hiltz and Murray Turoff. 1985. Structuring computer-mediated communication systems to avoid information overload. *Communications of the ACM*, 28 (7). 680-689.

[12] 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. DOI: https://doi.org/10.1145/1054972.1055100

[13] 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. DOI: https://doi.org/10.1145/1357054.1357070

[14] 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. DOI: https://doi.org/10.1145/2371574.2371617

[15] Hugo Lopez-Tovar, Andreas Charalambous and John Dowell. Year. Managing smartphone interruptions through adaptive modes and modulation of notifications. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, ACM, 296-299.

[16] Akhil Mathur, Nicholas D Lane and Fahim Kawsar. Year. Engagement-aware computing: Modelling user engagement from mobile contexts. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ACM, 622-633.

[17] 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. DOI: https://doi.org/10.1145/2750858.2807530

[18] 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*,

[19] Abhinav Mehrotra, Mirco Musolesi, Robert Hendley and Veljko Pejovic. Year. Designing content-driven intelligent notification mechanisms for mobile applications. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ACM, 813-824.

[20] Abhinav Mehrotra, Mirco Musolesi, Robert Hendley and Veljko Pejovic. 2015. Designing Content-driven Intelligent Notification Mechanisms for Mobile Applications. In *International Joint Conference on Pervasive and Ubiquitous Computing*, ACM, 813-824. DOI: https://doi.org/10.1145/2750858.2807544

[21] Abhinav Mehrotra, Veljko Pejovic, Jo Vermeulen, Robert Hendley and Mirco Musolesi. 2016. My Phone and Me: Understanding People's Receptivity to Mobile Notifications. In *Conference on Human Factors in Computing Systems*, Santa Clara, California, USA, ACM, 1021-1032. DOI: https://doi.org/10.1145/2858036.2858566

[22] Abhinav Mehrotra, Veljko Pejovic, Jo Vermeulen, Robert Hendley and Mirco Musolesi. Year. My phone and me: understanding people's receptivity to mobile notifications. In *Proceedings of the 2016 CHI conference on human factors in computing systems*, ACM, 1021-1032.

[23] Jochen Meyer, Merlin Wasmann, Wilko Heuten, Abdallah El Ali and Susanne CJ Boll. Year. Identification and Classification of Usage Patterns in Long-Term Activity Tracking. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, ACM, 667-678.

[24] 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. DOI: https://doi.org/10.1109/PERCOM.2015.7146515

[25] Tadashi Okoshi, Kota Tsubouchi, Masaya Taji, Takanori Ichikawa and Hideyuki Tokuda. Year. Attention and engagement-awareness in the wild: A large-scale study with adaptive

notifications. In *Pervasive Computing and Communications (PerCom), 2017 IEEE International Conference on*, IEEE, 100-110.

[26] Veljko Pejovic and Mirco Musolesi. 2014. InterruptMe: Designing Intelligent Prompting Mechanisms for Pervasive Applications. In *International Joint Conference on Pervasive and Ubiquitous Computing*, ACM, 897-908. DOI: https://doi.org/10.1145/2632048.2632062

[27] Veljko Pejovic, Mirco Musolesi and Abhinav Mehrotra. Year. Investigating the role of task engagement in mobile interruptibility. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*, ACM, 1100-1105.

[28] 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. DOI: https://doi.org/10.1145/2632048.2632060

[29] Martin Pielot, Bruno Cardoso, Kleomenis Katevas, Joan Serrà, Aleksandar Matic and Nuria Oliver. 2017. Beyond interruptibility: Predicting opportune moments to engage mobile phone users. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1 (3). 91.

[30] Martin Pielot, Karen Church and Rodrigo de Oliveira. 2014. An In-situ Study of Mobile Phone Notifications. In *International Conference on Human-computer Interaction with Mobile Devices & Services*, ACM, 233-242. DOI: https://doi.org/10.1145/2628363.2628364

[31] 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. DOI: https://doi.org/10.1145/2556288.2556973

[32] Martin Pielot, Tilman Dingler, Jose San Pedro and Nuria Oliver. 2015. When Attention is Not Scarce - Detecting Boredom from Mobile Phone Usage. In *International Joint Conference on Pervasive and Ubiquitous Computing*, ACM, 825-836. DOI: https://doi.org/10.1145/2750858.2804252

[33] Benjamin Poppinga, Wilko Heuten and Susanne Boll. 2014. Sensor-Based Identification of Opportune Moments for Triggering Notifications. *Pervasive Computing, IEEE*, 13 (1). 22-29. DOI: https://doi.org/10.1109/MPRV.2014.15

[34] Stephanie Rosenthal, Anind K Dey and Manuela Veloso. Year. Using decision-theoretic experience sampling to build personalized mobile phone interruption models. In *International Conference on Pervasive Computing*, Springer, 170-187.

[35] Alireza Sahami Shirazi, Niels Henze, Tilman Dingler, Martin Pielot, Dominik Weber and Albrecht Schmidt. 2014. Large-scale assessment of mobile notifications. In *Conference on Human Factors in Computing Systems*, ACM, 3055-3064. DOI: https://doi.org/10.1145/2556288.2557189

[36] Hillol Sarker, Moushumi Sharmin, Amin Ahsan Ali, Mahbubur M. Rahman, Rummana Bari, Syed Monowar Hossain and Santosh Kumar. 2014. Assessing the Availability of Users to Engage in Just-in-time Intervention in the Natural Environment. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ACM, 909-920. DOI: https://doi.org/10.1145/2632048.2636082

[37] Cheri Speier, Joseph S Valacich and Iris Vessey. 1999. The influence of task interruption on individual decision making: An information overload perspective. *Decision Sciences*, 30 (2). 337-360.

[38] Aku Visuri, Zhanna Sarsenbayeva, Niels van Berkel, Jorge Goncalves, Reza Rawassizadeh, Vassilis Kostakos and Denzil Ferreira. 2017. Quantifying Sources and Types of Smartwatch

Usage Sessions. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems - CHI '17*, Denver, USA, ACM, 3569-3581. DOI: https://doi.org/10.1145/3025453.3025817

[39] Aku Visuri, Niels van Berkel, Chu Luo, Jorge Goncalves, Denzil Ferreira and Vassilis Kostakos. Year. Predicting Interruptibility for Manual Data Collection: A Cluster-Based User Model. In *MobileHCI'17*, Vienna, Austria, ACM. DOI: https://doi.org/10.1145/3098279.3098532

[40] Tilo Westermann, Sebastian Möller and Ina Wechsung. Year. Assessing the relationship between technical affinity, stress and notifications on smartphones. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*, ACM, 652-659.

[41] Tilo Westermann, Ina Wechsung and Sebastian Möller. Year. Smartphone Notifications in Context: A Case Study on Receptivity by the Example of an Advertising Service. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, ACM, 2355-2361.

**For Editor only: Author Agreement Form*

International Journal of Human - Computer Studies

AUTHOR AGREEMENT FORM

Manuscript Title:

Understanding Smartphone Notifications' User Interactions and Content Importance

List of All Authors:
Aku Visuri, University of Oulu, Finland
Aku.visuri@oulu.fi

Niels van Berkel, The University of Melbourne, Australia
n.vanberkel@student.unimelb.edu.au

Tadashi Okoshi, Keio University, Japan
slash@ht.sfc.keio.ac.jp

Jorge Goncalves, The University of Melbourne, Australia
jorge.goncalves@unimelb.edu.au

Vassilis Kostakos, The University of Melbourne, Australia
vassilis.kostakos@unimelb.edu.au

Corresponding Author:
Aku Visuri

This statement is to certify that that the author list is correct, all Authors have seen and approved the manuscript being submitted and agree to its submission to the *International Journal of Human - Computer Studies.* The Authors also confirm that this research has not been published previously and that it is not under consideration for publication elsewhere. On behalf of all Co-Authors, the Corresponding Author shall bear full responsibility for the submission.

All authors agree that the author list is correct in its content and order and that no modification to the author list can be made without the formal approval of the Editor-in-Chief. All authors accept that the Editor-in-Chief's decisions over acceptance, rejection or retraction (the latter in the event of any breach of the Principles of Ethical Publishing in the International Journal of Human - Computer Studies being discovered) are final.

***Highlights (for review)**

- Smartphone users exhibit different styles to interact with notifications
- This work uncovers four distinct methods for interaction
- Users frequently neglect to filter away unneeded smartphone notifications
- The importance of notification contents can not be derived from interaction
- Combining user-reported importance with context to machine learning training data improves notification filtering accuracy drastically