Decomposing responses to mobile notifications

A thesis submitted in partial fulfilment

of the requirement for the degree of Doctor of Philosophy

Liam D. Turner

2017

Cardiff University
School of Computer Science & Informatics

Declaration

This work has not been submitted in substance for any other degree or award at this or any other university or place of learning, nor is being submitted concurrently in candidature for any degree or other award.

Signed ..... (candidate) Date .....

Statement 1

This thesis is being submitted in partial fulfilment of the requirements for the degree of PhD.

Signed ..... (candidate) Date .....

Statement 2

This thesis is the result of my own independent work/investigation, except where otherwise stated, and the thesis has not been edited by a third party beyond what is permitted by Cardiff University's Policy on the Use of Third Party Editors by Research Degree Students. Other sources are acknowledged by explicit references. The views expressed are my own.

Signed ..... (candidate) Date .....

Statement 3

I hereby give consent for my thesis, if accepted, to be available online in the University's Open Access repository and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed ..... (candidate) Date .....

To Sophie, for your patience and support.

Acknowledgements

I would like to thank my supervisors Professor Stuart Allen and Professor Roger Whitaker for their continued support and guidance. I would also like to thank the staff and PhD students of the School of Computer Science and Informatics, both past and present, for their support and feedback.

Abstract

Notifications from mobile devices frequently prompt us with information, either to merely inform us or to elicit a reaction. This has led to increasing research interest in considering an individual's interruptibility prior to issuing notifications, in order for them to be positively received. To achieve this, predictive models need to be built from previous response behaviour where the individual's interruptibility is known. However, there are several degrees of freedom in achieving this, from different definitions in what it means to be interruptible and a notification to be successful, to various methods for collecting data, and building predictive models.

The primary focus of this thesis is to improve upon the typical convention used for labelling interruptibility, an area which has had limited direct attention. This includes the proposal of a flexible framework, called the decision-on-information-gain model, which passively observes response behaviour in order to support various interruptibility definitions. In contrast, previous studies have largely surrounded the investigation of influential contextual factors on predicting interruptibility, using a broad labelling convention that relies on notifications being responded to fully and potentially a survey needing to be completed.

The approach is supported through two in-the-wild studies of Android notifications, one with 11,000 notifications across 90 users, and another with 32,000,000 across 3000 users. Analysis of these datasets shows that: a) responses to notifications is a decision-making process, whereby individuals can be reachable but not receptive to their content,

supporting the premise of the approach; b) the approach is implementable on typical Android devices and capable of adapting to different notification designs and user preferences; and c) the different labels produced by the model are predictable using data sources that do not require invasive permissions or persistent background monitoring; however there are notable performance differences between different machine learning strategies for training and evaluation.

Contents

List of Publications

This thesis includes work introduced in the following publications:

Turner, L. D., Allen, S. M., and Whitaker, R. M. (2015a). Interruptibility prediction for ubiquitous systems: Conventions and new directions from a growing field. In *Proc. UbiComp'15*, pages 801–812. ACM

Turner, L. D., Allen, S. M., and Whitaker, R. M. (2015b). Push or delay? decomposing smartphone notification response behaviour. In *Human Behavior Understanding*, volume 9277 of *Lecture Notes in Computer Science*, pages 69–83. Springer International Publishing

Turner, L. D., Allen, S. M., and Whitaker, R. M. (2017b). Reachable but not receptive: Enhancing smartphone interruptibility prediction by modelling the extent of user engagement with notifications. *Pervasive and Mobile Computing*

Turner, L. D., Allen, S. M., and Whitaker, R. M. (2017a). Behaviour patterns in managing stacks of mobile notifications

List of Figures

List of Tables

Chapter 1

Introduction

Over the last decade the rise of the smartphone has had a profound effect on society, providing opportunities for ubiquitous information retrieval and delivery. Interactions with mobile devices have shifted from being predominately instigated by the user, to also include responses to interruptions instigated by applications (apps). This has extended into the rise of other mobile devices, including tablets and smart wearables.

The *app* culture has expanded the diversity and frequency of interruptions from phone calls, alarms, and SMS messages to include notifications - snippets of information from diverse information sources, intended to inform, persuade, or prompt reaction. The concept of a notification is not limited to mobile devices, but a common thread exists in their intention to augment daily life with information. However, inappropriately timed interruptions from notifications can have a negative effect, at best being an annoyance and at worst a dangerous distraction. The ubiquitous nature of mobile devices and the continual evolution of notifications make this a timely issue, with ramifications for both applications that interrupt and the cognitive demands placed on individuals.

Assessing another person's interruptibility prior to interaction with them is a natural human behaviour that is easily handled by the human brain [16, 34, 49, 126, 69]; for example we naturally assess the likely ramifications of engaging someone before initiating conversation. However, creating such capability in the context of a machine (such as a mobile application issuing notifications) is a significant challenge. Towards this, a central theme in interruptibility research with mobile notifications (and interruptions

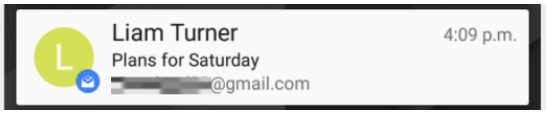


Figure 1.1: An example Android notification.

more broadly) has been to learn from how users interact with the interrupting content (if at all). However, variation in notification design, as well as in definitions of what makes someone "interruptible" and a notification successful, has left the area fragmented with study conclusions that are tightly coupled with specific experiment scenarios [121]; where the boundaries of wider applicability are often unclear [107].

1.1 Mobile notifications

The development of use cases for notifications (and their design) has been an evolutionary process. Historically, interruptions from mobile devices were limited to communication prompts or alarms. Notifications have absorbed these into a flexible platform for delivering and presenting snippets of information. However, they are autonomous in that they can be generated by individual applications at any time without any consideration of interruptibility, and contain any information relevant to that application. An example (email) notification is shown in Figure 1.1, with further examples, such as an instant messaging notification, a media player notification and a calendar reminder shown in Figure 1.2. This has enabled other applications that have previously not attempted to interact with the user in this manner, such as games, to adopt notifications as a means of attracting attention.

The implementation of notifications is similar across different mobile devices and operating systems, with some degrees of freedom in their individual appearance and interruptive design. Content has traditionally been a short piece of text, however this has been extended over recent years to also enable other types of content and interactive features, such as images, lists, and actionable buttons. Notifications can also adopt

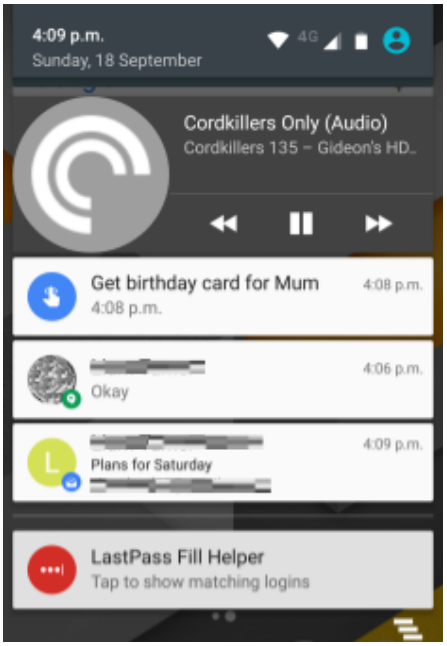


Figure 1.2: Multiple notifications in a notification drawer.

mechanisms to interrupt the user, in addition to merely appearing on the device's user interface; using combinations of audio tones, vibration patterns, and visual cues (e.g., a flashing LED pattern). Finally, notifications have evolved to have variability in their persistence (i.e., whether a user can remove them) and priority (e.g., whether they pop-up on the screen), making the concept suitable for a broad range of use cases.

1.1.1 Interacting with notifications

While notifications are tightly associated with individual applications, they are isolated from their user interfaces. Notifications operate in a *push* based manner, where an application dictates when they are created and made known to the user (assuming that the user has not disabled an application's ability to push notifications). Therefore, a user does not need to be using the application, or even using the device, in order for notifications to arrive. In some use cases this is directly influenced by external sources, for example, a SMS application will push notifications when the device receives the message. This is the opposite of user-driven interactions with applications, where a user

Figure 1.3: An example notification icon shown on the Android top bar.

may open and interact with the user interface in order to *pull* information from it (e.g., perform a Google search or browse Twitter).

As with interruptions in general [73, 75], the response process towards notifications is a sequential process of pausing the current task and pursuing the content. On Android devices, after a notification has been generated, it is placed in the *notification drawer* (shown in Figure 1.2), where it remains until it is removed and permanently destroyed. Access to a notification can require the user to traverse a series of screens, particularly if the device is not in use. Along the way, the user can become aware of various information about the notification (e.g., the originating app), and decide whether to pursue further. For example, Figure 1.3 shows a notification icon being displayed along the top (left) of Android's user interface.

As well as presenting information, the notification design may also encourage the user to act upon the notification and perform some direct action in response. However this is largely use-case dependent, for example, an email notification may have actions for immediately replying to the email or deleting it, whereas a weather summary notification may not have any direct response actions and simply display information.

1.1.2 Coexisting notifications

Notifications are designed, delivered, and responded to independently of one another, however they can coexist together. Multiple notifications that display different information for different purposes can be present at any given time (as shown in Figure 1.2), with any interruptions queued if they arrive in quick succession. On arrival, notifications are added to a stack where a user can interact with them individually. Historically, notifications in the stack have been sorted by arrival time, however this has evolved

to also be influenced by other operation system dependent factors (e.g., an assigned priority [3]).

However, the nature of mobile operating systems dictate that applications are sandboxed [6], including their notifications, where individual applications are typically not aware of the notifications produced from other applications.

1.2 Problem definition

Notifications bring important utility to daily life, in both alerting a user of information they would likely otherwise look for (e.g., emails), as well as information they may not have otherwise considered (e.g., recommendations). However, a user will likely only find an individual notification useful in isolated contexts, yet they can receive notifications about any topic at any time. The filtering of this usefulness is largely reliant on the user, creating a cognitive burden that is accelerated with the increasing frequency and diversity of notifications [85].

Similarly to other information consumption environments, such as browsing social media feeds (e.g., [22, 125]) and email clients (e.g., [38]), current implementations of notification delivery create noise, where the information is not necessarily useful at the point in time it is seen. This not only diminishes the effectiveness of individual notifications, but any interruptions could also produce a negative effect in environments where focus is key (e.g., when driving [59]) or contribute to negative states of mind (e.g., stress [70]).

Therefore, limiting notification delivery to moments where it could be more useful is a highly desirable but challenging problem. This leads to the following question as the motivation for this thesis:

Can a notification delivery system assess and act upon an individual's interruptive state in a similar manner to the social conventions that humans typically adopt?

This question has motivated the research space in general, with studies largely focusing on improving predictive models of interruptibility through examining the accuracy improvements that different contextual factors can bring. However, approaches for collecting and labelling interruption behaviour in order to build accurate predictive models with this contextual data (i.e., the procedure for determining whether the user was interruptible or not) have had limited direct attention, with the typical conventions used having a number of limitations that are susceptible to under-representing interruptibility. Discussed further in the Chapters 2 and 3, these limitations include: a heavy reliance on human annotation that is assumed to be reliable and accurate; a common focus on just observing interactions with the notification in isolation, rather than the interrupting device in general; and strict design assumptions in what makes an interruption successful.

The central approach adopted by this thesis is to embrace the fragmentation and variability seen across existing studies. Motivated by the sequence of decisions that a user makes when receiving and responding to a notification [75], this thesis proposes that deconstructing how a response is made (from the point of delivery) can improve the labelling of interruptibility by separating where possible, a representation of a user's physiological interruptive state from their sentiment towards the notification content. This intends to improve upon existing conventions for labelling interruptibility, such as relying solely on explicit user annotation through surveys, or merely knowledge that a notification has been removed (as is common in previous studies, e.g., [99, 92]), by providing a flexible basis to collect behaviour and predict interruptibility for different definitions and use cases.

1.3 Contributions

The overarching contribution of this thesis is the improvement of the typical mechanism for measuring the effectiveness of interruptions (such as mobile notifications). In

doing so, this enables both consumer facing mobile applications, as well as research applications that issue interruptions (for example, experience sampling surveys for mood/well-being, e.g. [66, 11]) to define their own definition of a successful interruption and learn to deliver content at times where a successful response is likely to occur. This is formed from several individual contributions (summarised below), as a result of a survey of the literature and analysis of two in-the-wild empirical studies surrounding Android mobile notifications. Towards this thesis, these contributions have formed and extended a number of peer-reviewed research papers:

[121] Turner, L. D., Allen, S. M., and Whitaker, R. M. (2015a). Interruptibility prediction for ubiquitous systems: Conventions and new directions from a growing field. In *Proc. UbiComp'15*, pages 801–812. ACM

[122] Turner, L. D., Allen, S. M., and Whitaker, R. M. (2015b). Push or delay? decomposing smartphone notification response behaviour. In *Human Behavior Understanding*, volume 9277 of *Lecture Notes in Computer Science*, pages 69–83. Springer International Publishing

[124] Turner, L. D., Allen, S. M., and Whitaker, R. M. (2017b). Reachable but not receptive: Enhancing smartphone interruptibility prediction by modelling the extent of user engagement with notifications. *Pervasive and Mobile Computing*

[123] Turner, L. D., Allen, S. M., and Whitaker, R. M. (2017a). Behaviour patterns in managing stacks of mobile notifications

Contributions

C1 A survey of the fragmented research area, developing open research questions by highlighting limitations and gaps in existing methodologies and conventions for collecting, labelling, and predicting interruptibility.

C2 A flexible model for labelling interruptibility for different definitions, the Decision-On-Information-Gain (DOIG) model, that deconstructs the observable behavioural

trace in a response to a notification.

C3 Analysis into the natural decision behaviour underpinning interactions with notifications, using data collected in-the-wild.

C4 Analysis into the predictability of response behaviour using past behaviour that is labelled using the DOIG model, including examining the effect of various machine learning strategies on predictive performance.

C5 A demonstration of the flexibility of DOIG the model for different notification designs and device preferences, using additional in-the-wild data.

C6 An exploration of where the DOIG model sits amongst wider notification behaviour on the device.

Contribution C1 is relevant to [121]; C2, C3, and C4 to [122, 124]; and C5 and C6 to [123]. Across these contributions, a primary output is the proposal and validation of a labelling framework (primarily C2, C4, and C5). However the passive nature of the data collection enables further related contributions to be formed from the resulting datasets and analyses (C3 and C6).

1.4 Thesis Structure

The outline for the remainder of this thesis is as follows:

Background and research gaps: A survey and meta-analysis of relevant interruptibility literature, exposing key conventions used in collecting and studying interruption behaviour. Within this, a collection of research questions are proposed from gaps and limitations in the conventions exposed. A subset of these then shape the scope of the subsequent chapters. The associated chapter (2) creates contribution C1.

Proposal and implementation of a new labelling method for notifications: A flexible model for inferring and labelling interruptibility using the observable behavi-

oural trace towards interruptions is developed; called the decision-on-information-gain (DOIG) model. An empirical study using an Android application is then used to demonstrate an example implementation of the model and the benefit it brings over the existing convention for labelling interruptibility. Additionally, the procedure and challenges associated with collecting the behaviour and correlating contextual data on typical Android hardware is discussed. The associated chapters (3 and 4) create contributions C2 and C3.

Predicting decision behaviour towards notifications using the DOIG model: An analysis into the relative differences in predictive performance for: a) the different labels produced by the DOIG model that represent various definitions of interruptibility, and b) different machine learning strategies, including: classifier algorithms, training strategies, and evaluation criteria. The associated chapter (5) creates contribution C4.

Examining the flexibility of the DOIG model: An empirical investigation (using a second Android application) into the practical flexibility of the DOIG model when different notification design characteristics and device preferences are used, through examining how these can modify the response process that the DOIG model can capture for labelling. The associated chapter (6) creates contribution C5.

Wider notification behaviour and implications on the DOIG model: An exploration of notification behaviour from the wider viewpoint of the notification stack, in order to determine further support for the DOIG model. With analysis surrounding: a) whether decision making in responses to notifications can also be seen from this viewpoint, and b) whether potential impacting factors in the wider behaviour may impact responses to individual notifications. The associated chapter (7) creates contribution C6.

Conclusions and future work: The thesis concludes with a reflection on the contributions made, as well as a discussion of areas of future work.

Chapters 3 through 7 discuss the empirical studies conducted. These chapters are uniformly structured to firstly give an extended rationale for the chapter, discussing the

explicit limitations in the approaches seen in literature that are focused on, along with a section outline. From this, each group of analysis ends with a summary of the primary findings, before a closing chapter conclusion that summarises the extent to which the limitations have been improved upon and the resulting primary contributions towards this thesis.

Chapter 2

Background

The following survey explores the wider research area of machine-to-human interruptibility, using studies surrounding interruptions from computational devices in general. The reason for this is that the research area is heavily fragmented with investigations concerning specific interruptions in specific scenarios. However broadly speaking, this does not create an entirely new research problem from that surrounding mobile notifications, with similar conventions used in study design, data collection, and data analysis.

The purpose of this chapter is twofold, firstly, to expose and classify key conventions used in the literature for: defining interruptibility, collecting data, labelling behaviour, and predicting interruptibility. Secondly, to provide insight of the wider research space, which is used as a basis for reflecting on how the contributions of the rest of this thesis (surrounding an improved framework for labelling interruptibility) may extend to other types of interruptions and environments.

The survey is organised as follows. Firstly, the broad overarching themes in conducting research in this area are discussed, followed by an examination of the differences seen in how interruptibility has been defined across the literature. From this, the typical linear paradigm of empirical studies is explored (shown in Table 2.1), concerning: defining a scenario, collecting data, and building predictive models. Within each area, the typical design choices, assumptions, and implementation practices used are discussed, as well as the capabilities and limitations in generalising approaches.

1. Scenario Selection	2. Data Collection	3. Prediction
Decide on the: - interruptions used - interruption environment (e.g., offices, or everyday life) - intended objective (e.g., predict all moments of interruptibility, or just in isolated tasks)	Decide on what types of data to use in representing interruptions and response behaviour Collect data and extract feature vectors, including choosing the: - raw data traces to sample - feature variable extraction process Label the extracted feature vectors Aggregate the data to form a dataset for prediction	Perform pre-processing Build predictive models, including choosing the: - training environment (e.g., offline or online) - training data (e.g., use personal or aggregate data) Evaluate the predictive performance

Table 2.1: The typical linear paradigm of interruptibility studies, including subcomponents. Table extended from [121].

As part of this, 10 open research questions for the broader research space are proposed (as in [121]). The collective breadth of these questions goes beyond the scope of this thesis, however a subset of the questions help form the focus of the subsequent chapters, surrounding improving the conventions for labelling interruptibility (discussed further at the end of the survey in Section 2.6).

2.1 Overarching themes in studying interruptibility

Broadly speaking, interruptions from computational devices often result in a cognitive burden being placed on the recipient to individually assess and decide on a course of action [75]. Offloading this to systems that can proactively assess interruptibility before issuing interruptions (similar to what a human would prior to engaging in conversation, e.g., [34, 49, 126, 69]) is therefore highly desirable and forms an overarching focus

across the literature.

Towards this, reviews and empirical studies of interruptibility have surrounded two distinct approaches; inline with the typical study process shown in Table 2.1. Firstly, studies have encompassed interruptibility within the concepts and visions of creating attention-aware systems (e.g., [95, 101, 81, 107]). The second investigates specific relevant practices towards these systems, such as influential contextual features (e.g., [43, 97, 92, 76]) or the effectiveness of human/machine data collection (e.g., [12, 64]).

In doing so, it is assumed that inappropriate interruption has a human cost (e.g., annoyance or cognitive burden), as does the lack of a legitimate interruption (e.g., opportunity cost). With interruptions in the right context able to augment some task-oriented environments [47] or even provide productivity stimulus when self initiated [54]. Different use cases of interruptions are likely to have different priorities associated with these costs, and this is reflected in the literature with studies sometimes focusing on one or the other (discussed further in Sections 2.3.3).

Despite this, a consistent standpoint exists in the supportive role of intelligent technology and the need for accurate interruptibility prediction in order to improve this. This has resulted in increased academic interest from a wide range of disciplines including: psychology [79], human-computer interaction [75], and ubiquitous computing [101, 20, 93], as well as diverse application areas including medical [100, 62, 112] and safety [59] domains.

2.2 Variability in the definitions of interruptibility used

The purpose of inferring interruptibility is to identify (typically in situ) whether it is a suitable moment to introduce a stimulus that the user may choose to act upon. Thus, to minimise disruption and maximise timely response rates, interruptions should ideally occur at the most convenient or opportune moments. However there are degrees of freedom in what it means to be interruptible, such as being physically interruptible,

whether the interruption will (or will not) adversely affect a task, or whether the interruption content is considered useful. This is highlighted through the 2005 survey conducted by Ho and Intille [43], who report at least 8 definitions of interruptibility across the literature. More generally, studies can be broadly categorised under 3 groups (as defined in [121]), where the focus concerns either the:

- *physiological ability* to switch focus;
- *cognitive effect* on task performance;
- or *user sentiment* towards the interruption.

Studies focusing on the *physiological ability* to switch focus surround the assessment of the cognitive workload of an individual at the time of interruption, and their capacity to receive it (i.e., whether they are physically interruptible). At the very lowest level, this can be assessed with the aid of EEG [71] or pupil size events [9, 13], although achieving this outside of controlled conditions is currently not a practical basis for measurement.

Studies focusing on the *cognitive effect* on task performance surround the assessment of the likely effects the interruption will have on task performance. This has typically been adopted in task-oriented environments through identifying breakpoints where disruption is minimised (e.g., [50, 80]). These studies may not predict a user's interruptibility, but instead the effect it has on the task. A common measure used for this is the elapsed time to regain focus after the interruption, referred to as *resumption lag* (e.g., [50, 8, 80, 51]).

Studies focusing on the *user sentiment* towards an interruption surround the assessment of the user's desire to react and consume the interruption (rather than the effect on the current task). This can involve more subjective opinions captured using self reports, termed *experience sampling methods* (ESM) (e.g., [99, 92]). However, degrees of freedom can be seen within this, with some studies distinguishing between *attentiveness* (e.g., [97, 92]) towards an interruption (i.e., whether the whether the user decides to attend to the interruption) and *receptiveness* towards the interruption content (e.g., [29]).

Table 2.2 classifies relevant works of note under these categories and Figure 2.1 visual-

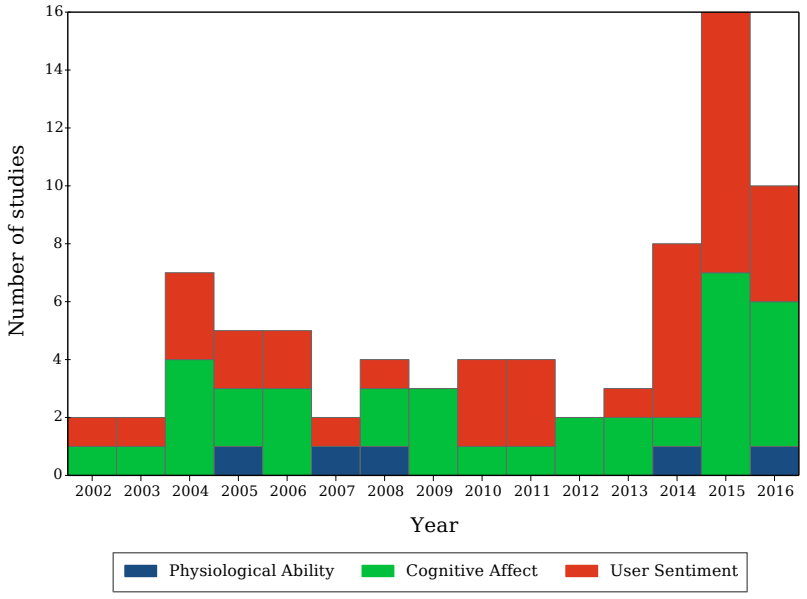


Figure 2.1: Definition groups over time. Figure extended from [121].

ises their prominence over time. Within this, some studies consider multiple categories and others do not define an explicit definition of interruptibility; in these cases a judgement is made from the information provided. Overall, there is clear fragmentation across the groups with all three being continually used over time to some degree. The small number of studies examining the physiological ability to switch focus is notable, but likely due to the complexities of data collection (e.g., requiring hardware to enable EEG readings [71]). Additionally, research into human physiology and cognitive interruptions is also present in other domains such as neuroscience (e.g., [25, 27]), which is not included in this meta-analysis.

The high proportion of studies focusing on the cognitive effect on task performance or user sentiment remains consistent over time. A likely cause for this is the different experiment scenarios being used, in which only a specific definition may be relevant. For example, nurses working in an emergency facility are more likely to be concerned with the effect on task performance rather than their desire to receive it (e.g., [100]), while office environments (and mobile notifications in general) are relevant to both user sentiment and the cognitive affect on workload, depending on the interruption content.

		Definition Focus			Environment			Contextual Data			Labelling		
	Year	PA	CE	US	COE	EI	II	EUS	RS	SS	EOS	IIO	RSL
[80]	2002		X		X				X			X	
[47]	2002			X	X			X			X		
[44]	2003		X		X				X		X		X
[48]	2003			X	X					X	X		X
[82]	2004		X			X		X	X		X		
[46]	2004			X	X				X		X		
[8]	2004		X	X	X			X	X		X	X	
[32]	2004				X				X		X		
[15]	2004		X		X				X				
[42]	2004		X		X				X			X	
[57]	2004			X	X				X		X		
[33]	2005		X		X				X			X	
[9]	2005	X			X				X			X	
[31]	2005			X	X					X	X		X
[45]	2005		X		X				X			X	
[43]	2005			X		X			X		X		
[50]	2006		X		X				X			X	
[101]	2006		X										
[14]	2006		X	X	X			X	X		X	X	
[58]	2006			X	X				X		X		
[71]	2007	X				X			X		X		X
[119]	2007			X		X		X			X		
[51]	2008		X	X	X				X		X	X	
[13]	2008	X	X		X				X			X	
[104]	2009		X										
[17]	2009		X										
[54]	2009		X		X			X					X
[29]	2010			X		X			X		X		
[53]	2010		X			X			X			X	X
[130]	2010			X					X		X		
[36]	2010			X		X		X			X		
[113]	2011			X	X				X		X		
[118]	2011		X		X				X		X		
[30]	2011			X		X			X			X	
[102]	2011			X		X			X			X	X
[65]	2012		X			X			X			X	
[55]	2012		X		X				X			X	
[81]	2013			X									
[107]	2013		X										
[37]	2013		X		X			X	X		X	X	
[115]	2014			X		X			X				

		Definition Focus			Environment			Contextual Data			Labelling		
	Year	PA	CE	US	COE	EI	II	EUS	RS	SS	EOS	IIO	RSL
[97]	2014			X			X		X			X	
[99]	2014			X			X		X		X		
[111]	2014			X			X		X			X	
[105]	2014	X		X		X		X	X			X	
[92]	2014			X		X		X	X		X		
[18]	2014		X		X				X			X	
[91]	2015			X		X		X			X		
[59]	2015		X		X				X			X	
[122]	2015			X			X		X			X	
[19]	2015		X		X								
[89]	2015		X			X			X		X		
[21]	2015			X		X			X		X	X	
[114]	2015		X		X				X			X	
[129]	2015		X	X	X	X		X	X		X		
[61]	2015		X	X		X			X		X		
[94]	2015			X		X		X			X		
[67]	2015			X	X					X			X
[76]	2015			X		X		X	X		X		
[87]	2015		X		X	X		X	X		X		
[116]	2015			X		X			X		X		
[72]	2016	X	X		X	X			X		X	X	
[106]	2016			X	X				X			X	
[83]	2016		X			X		X				X	
[24]	2016			X		X			X		X	X	
[108]	2016			X		X		X	X		X		
[77]	2016		X	X			X	X	X		X	X	
[56]	2016		X		X				X			X	

Table 2.2: A decomposition of the approaches used across studies for: defining interruptibility, the experiment environment, collecting contextual data, and collecting interruptibility labels, sorted ascending by year. Some studies can include the use of multiple types of approaches (e.g., if different experiments are performed). PA=Physiological Ability, CE=Cognitive Effect, US=User Sentiment as defined in Section 2.2. COE=Controlled environment, EI=Explicit "in-the-wild" environment, II=Implicit "in-the-wild" environment as discussed in Section 2.3.2 and defined in Table 2.3. EUS=Explicit User Surveys (i.e., ESM), RS=Real world machine data sources, SS = Simulated data sources, as discussed in Section 2.4.2.1 and shown in Figure 2.4. EOS=Explicit opinion from in situ surveys (i.e., ESM), IIO=Implicit in situ observations of behaviour, RSL=Retrospective labelling, as discussed in Section 2.4.1 and shown in Figure 2.3. Table extended from [121].

Overall, there is a clear disparity in what constitutes interruptibility. The ability to categorise this is useful for synthesising studies, however a problem still remains in that the choice here impacts upon the rest of a study (i.e., data collection and labelling) and ultimately the conclusions made. This issue is a motivation in the focus of the remaining chapters of this thesis surrounding a flexible framework for labelling interruptibility for different definitions where possible; this is discussed further at the end of the survey in Section 2.6.

2.3 Scenarios and interruptibility

The first dimension of interruptibility studies is defining the scenario. At its highest level, this captures the scope, by defining a *channel of interruption* (such as mobile notifications), the *study environment* (which addresses the physical context in which the interruption is studied), and the *objective for the study*.

2.3.1 Choice of interruption

In general, studies typically investigate using a single type of interruption from a single source. This ranges from messaging communications (e.g., instant messaging [97, 37, 21] or email [53, 56, 61]); to audio recordings (e.g., [32, 31]); to pop-up messages during device usage (e.g., [33, 116, 129, 85]); to phone calls (e.g., [30, 111, 106]); and to mobile notifications in general (e.g., [92, 99, 83, 76, 114, 24]). However, our daily lives typically involve multiple devices that can interact with us in more than one way, these devices may have multiple means of interaction, they may be restricted by place or time, and multiple devices can exist at the same time. Exploring how interruptibility can be affected by issuing interruptions through different channels and devices (i.e., predicting *how* to best interrupt, as well as *when*) has been a relatively unexplored area, which leads to the following an open research question:

(RQ1) How can different channels of interruption (and potentially devices) be used in combination and to the best effect?

Towards addressing this, Sarter [107] reviews interruption management in a multi-modal context, and proposes the use of different interruptive cues based on characteristics of the current activity and the type of interruption, however does not test these with empirical experiments. Additionally, Okoshi et al [89] experiment with introducing mobile interruptions onto smart watches in addition to the smartphone. However more empirical work in this area is needed, particularly involving direct comparisons of delivering interruption content through different cues and devices. This goes beyond the scope of this thesis, which surrounds mobile notifications on a single device, however it remains a direction of future work.

2.3.2 Choice of study environment

Experiment environments have ranged from all moments of daily life (e.g., through a personal smartphone [92, 115]) through to a more specific focus, such as those with high social costs (e.g., during collaborative working [40, 60]) or where task disruption is likely to occur, (e.g., in offices [31, 82]). More generally, experiment environments are either *controlled* or *in-the-wild*, as defined in Table 2.3, with variability in what constitutes an in-the-wild environment. While controlled environments have traditionally involved a laboratory setting (e.g., [51, 55, 8]), static office settings may also fall into this category. For example, in cases where a third party observer is present (e.g., [54]) or when cameras are added to an existing environment (e.g., [31, 58]).

Table 2.2 classifies existing literature into these different types of environments, with Figure 2.2 visualising this over time. There is a clear recent increase in experimenting *in-the-wild*, this likely due to the spatial and temporal freedom that ubiquitous technologies such as the smartphone have enabled. However, controlled environments still remain a popular design choice for experiment scenarios involving set tasks over a finite time

Type	Definition
Controlled environment	The experiment typically takes place in a single static location (e.g., a laboratory setting), involving simulations of activities and interruptions. Participants are typically compensated for their time, but not always.
Explicit in-the-wild	The experiment takes place *in situ* around the daily lives of the participants. However, the user is continually aware of the experiment (e.g., if a dedicated mobile application is used to issue interruptions surveying interruptibility [92]). The participants are typically incentivised through compensation for their time, but not always.
Implicit in-the-wild	The experiment takes place *in situ* around the daily lives of participants. The experiment is often embedded through other features that the participant finds useful (e.g., if a mobile app is used that also offers additional features to the user, such as a mood diary [99]), providing more natural incentive than explicit compensation.

Table 2.3: Common types of experiment environments used. Table extended from [121].

period (e.g., [37]), and where additional technologies need to be introduced into the environment (e.g., external cameras [91]).

2.3.3 Choice of study objective

The objective for a study concerns what is trying to be predicted in relation to the choice of interruption and environment. For example, some works focus on classifying any given moment as either suitable for a particular interruption or not (e.g., [30, 99, 92, 76, 122, 72]), while others focus on exploring the effects of interruptions (e.g., [50, 13]). There are also studies with a more specific focus, such as predicting the timeliness of instant messages being read (e.g., [97]) or the time it takes for a user to resume to their previous task (e.g., [50]).

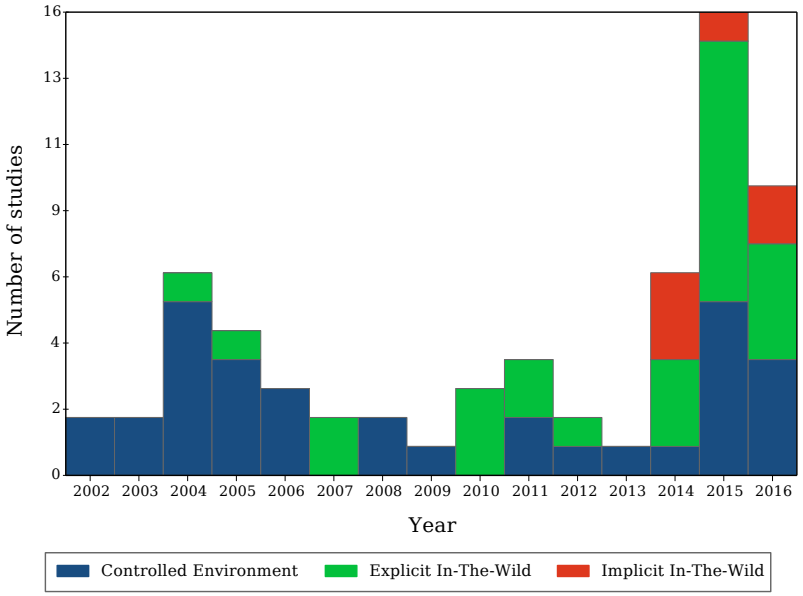


Figure 2.2: Use of different experiment environments over time. Figure extended from [121].

Overall, the scenarios for studying interruptibility can be seen as being heavily domain and interruption type specific. This creates a problem in that the choices made here have a profound effect on the later stages (e.g., what data is collected) and ultimately on interruption prediction systems in the evaluation criteria chosen (i.e., prioritising the minimisation of false positive or false negative predictions, or both). This creates uncertainty in assessing the wider applicability for other scenarios [107], which could require costly implementation and testing to determine. Therefore, another open research question remains in whether a one-size-fits-all framework can be achieved, or whether conclusions are limited to being tightly coupled with specific scenarios:

(RQ2) Given the diversity of potential scenarios, when are generalised and interoperable solutions for interruptibility sufficient, and when are domain specific solutions necessary?

Little progress has so far been made towards addressing this issue directly, with works either presenting either broad frameworks that represent an interruption as a general concept (e.g., the Interruption Management Stage Model [73, 75]) or isolating their

investigations to specific channels of interruption (e.g., all smartphone notifications [99, 92, 76]). This research question inspires the focus in the remainder of this thesis, in considering multiple definitions of interruptibility in labelling, through to assessing the relative differences in predictive performance for different labels with different evaluation criteria.

2.4 Data collection and labelling

Towards predicting interruptibility, empirical studies require the collection of a dataset of previous interruptions where the interruptibility of the user is known (or the effects task performance are known). In this data, each interruption is typically represented by a vector of variables that capture the context at a given moment, and a label representing some categorisation of interruptibility (e.g., interruptible, or not interruptible). However within this, there are considerable degrees of freedom in how this is achieved.

2.4.1 Strategies for labelling interruptibility

The label used to denote interruptibility is often tightly coupled with the definition of interruptibility used and the objective of a study. However these labels ultimately represent interruptibility as either a binary state (e.g., [105]) or on a scale (e.g., [113, 92]) with some threshold then used to convert it to a binary state. This then represents whether the interruption should or should not have occurred.

In order to retrieve this label, some form of labelling task needs to take place after an interruption has been issued. However accomplishing this accurately and reliably can be problematic. Three different approaches are dominant in the literature, *explicit* and *implicit* labelling by the user being interrupted (performed in situ, e.g., [116, 122, 92]), or *retrospective* labelling that occurs after the data collection (e.g., [48, 67]).

Explicit labelling is typically performed directly by the user in situ through self-reporting (ESM) (e.g., [43, 31, 36, 30, 129, 116]). For example, Choy et al [24] ask the user the binary question "Are you interruptible?", whereas others ask the user provide a response on a scale, e.g., Pejovic and Musolesi [92] ask "Is this a good moment to interrupt?". With retrospective labelling also formed from user opinion (e.g., [48, 57]). However, it is questionable whether a user can accurately and consistently quantify their interruptibility [111], either in real time (e.g., [43, 31]) or retrospectively (e.g., [91]). Additionally, a user may be interruptible, but not to the extent that they wish to complete the labelling task [48, 72] (e.g., Pejovic et al. [94] only had 36% of their surveys completed), or they may find doing so undesirable (e.g., if they respond and fill in a survey, but can state that they are not at all interruptible [116, 129, 92]); leading to some dedicated studies that focus on finding opportune moments to issue surveys (e.g., [83, 78]).

Alternatively, *implicit labelling* (e.g., [24]) involves observing user actions and making deductions (e.g., [30, 97]) rather than relying on user annotation. For example, for studies focusing on user sentiment towards mobile interruptions (discussed in Section 2.2) this has included observing whether a phone call is answered (e.g., [111]) or a notification is tapped on (e.g., [97]). However this loses the benefit of human opinion and may not be feasible in environments where this behaviour cannot be observed by machine sources.

The extent that these different types of labelling methods occur in the literature is shown in Table 2.2, with Figure 2.3 visualising this over time. From this, it is clear that retrospective labelling has not been widely used in recent years, likely due to technological advances enabling participant feedback in situ as opposed to relying on video recordings (e.g., [44, 67]). Interestingly, the debate of using ESM or implicit observations of behaviour is reflected in the consistent use of both techniques over time.

Across all of the methods used, a primary limitation has been to rely on some final action being performed by the user. For example, if human annotation is used, this relies on the

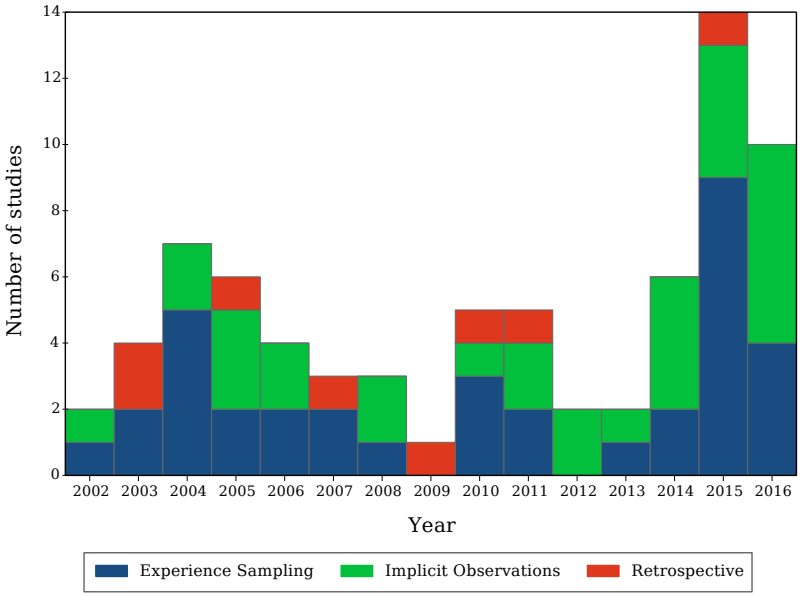


Figure 2.3: Use of different labelling strategies over time. Figure extended from [121].

individual being willing to complete a survey (e.g., [99, 92, 116, 129, 24]). If implicit observations are used, this has typically relied on some action being reached, such as a mobile notification being tapped on, or application opened (e.g. [97, 18, 111]). However, in some scenarios, the response process could involve multiple steps. Additionally, as with Android mobile notifications, it may be the case that not all information is available to the user initially (e.g., the source application or exact content) until the user performs additional interactions with the device.

This could therefore result in responses that are started but then abandoned, where arguably some degree of interruptibility is shown, i.e., the user was physically reachable for interruptions in general, but not receptive to the particular interruption [21]. These cases may be incorrectly classified as not interruptible because, for example, the user did not complete the survey. Investigations into the importance of incomplete responses has received little attention, leading to the following research question:

(RQ3) Can including the extent of a response to an interruption provide additional semantic value for inferring the user's attentiveness towards it?

Across the literature there is a foundation of key works introducing relevant concepts. Firstly, McFarlane and Latorella show that the act of interrupting and responding is a decision process [74], however this stops short of modelling the response process between switching to the interruption and returning to the previous task. Additionally, as discussed in Section 2.2, several works (e.g., [97, 92, 29]) have proposed concepts such as *attentiveness* and *receptivity* in order to separate willingness to consume an interruption and liking the content. However, there has been little empirical investigation into the viability of a framework that labels interruptibility using different definitions from a trace of response behaviour, and by extension the impact that this has on prediction. This forms a key focus in the subsequent chapters of this thesis.

2.4.2 Capturing contextual data

The usefulness of the interruptibility label produced is ultimately tied to its ability to be predicted from contextual data. Therefore a key design consideration is the choice of what data to capture and how, which can include data representing the current moment (e.g., [72, 99]) as well as historical activity (e.g., [24]).

2.4.2.1 Common contextual data traces collected

Capturing signals to represent the current context is an essential component in predicting interruptibility (or the effects of interruptibility). Table 2.4 details the types of data traces commonly collected in the literature, classified as being from either external sources to an individual or more latent. These can loosely be described as capturing *what is currently happening* and *how the user feels* respectively. Ideally, this data should be as rich as possible, however resource constraints and scenario environments typically dictate a subset of these being used (as shown in Table 2.4). Collection of this data can involve the use of explicit human annotation through surveys (likewise to labelling), real world machine sources, or in some cases, simulated data; with Table 2.2 and Figure 2.4

Source	Example data traces and studies
External	Smartphone sensors: such as hardware sensors (e.g., [115, 59, 97, 99, 105, 92, 30, 102, 89, 72, 24, 108, 76, 77, 83]) and/or software APIs (e.g., [65, 115, 97, 111, 30, 18, 102, 77, 106, 21]) Other physiological sensors: capturing physical state (such as heart rate) (e.g., [59, 105]) or activity (e.g., [105, 57, 43]) Other environmental sensors: such as sound or motion in a room (e.g., [82, 32, 48, 15, 44, 45]) or car (e.g., [59]) Software events: e.g., active windows, keyboard and mouse activity (e.g., [50, 51, 55, 82, 32, 118, 80, 42, 15, 44, 46, 45, 33, 116, 61, 85, 72]) Calendar schedules: (e.g., [113, 115, 44, 106]) Temporal logs: e.g., of user actions (e.g., [55, 65, 97, 45, 106]) Spatial logs: e.g., GPS (e.g., [113, 115, 111, 105, 30, 102]) or connections to antennas (e.g., [82, 99, 111, 92, 46])
Latent	Self reports: experience sampling (e.g., [29, 91, 82, 119, 105, 92, 47, 51, 57, 43, 94, 76, 108]) or post-experiment surveys (e.g., [8, 37, 54]) Qualitative feedback: e.g., post-interviews (e.g., [29, 47]) Third party observer reports: e.g., in situ observation (e.g., [54]) or video annotations (e.g., [48, 57, 31, 67]) Physiological sensors: e.g., mental state or workload (e.g., [71, 9, 105, 13, 72, 129])

Table 2.4: A categorisation of commonly captured data traces. Table extended from [121].

showing the use of these different practices over time.

Advances in ubiquitous sensing (such as mobile devices) is a likely cause of the rise in the use of real world machine sources, such as sensors (e.g., [99, 15]) and experience sampling (e.g., [92]) over simulated sensors (e.g., [48, 31]) in recent years (Figure 2.4). Additionally, the personal relationship between these devices and their user has been argued to allow more “ecologically valid data" [79], rather than using peripheral devices, such as external cameras (e.g., [48, 31]) or wearable accelerometers (e.g., [43, 57]). However there is still disagreement over whether sensors should be used [92, 99, 115] or not over user annotation (e.g., through ESM), due to accuracy and reliability concerns [113, 64, 105], resource requirements [111], and limitations for measuring latent variables.

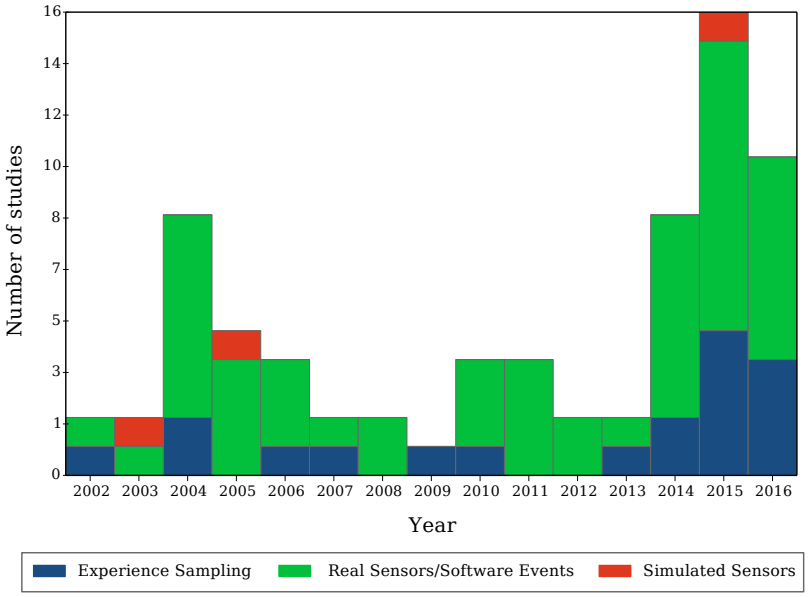


Figure 2.4: Use of different data collection strategies over time. Figure extended from [121].

However, human involvement either from a third party observer (e.g., [48, 57]) or by the participant themselves (e.g., [47, 119, 91]) has also been argued to suffer from similar issues (e.g., [82, 91, 72]). On one hand, it has benefits including being highly flexible in what can be asked, having a low cost overhead in terms of technical resources, and allowing the collection of latent variables (e.g., mental state) which aren't easily observable by readily available sensors [64]. However, the use of ESM for interruptibility research specifically has been controversial due to the additional interruption cost it places on the user [30, 72, 78]. Overall, likewise to labelling, the use of explicit human annotation and/or implicit machine sources remains a contested issue.

Beyond this debate, a trend in recent years is the consolidation of technologies used to collect this contextual data, such as only using a smartphone (e.g., [92, 99, 30]). However the emergence of networked pervasive technologies in the environment (i.e., the Internet of Things) and upon the person (i.e., smart wearables), could lead to this becoming unconsolidated once more with these technologies augmenting existing data traces. For example, a light sensor in a room may be more consistent and accurate

than a smartphone equivalent. Additionally this could extend the possibilities of what contextual data is possible to collect - leading to the question:

(RQ4) How can emerging sensor-equipped ubiquitous technologies (such as wearables) improve sampling accuracy and reduce collection and processing complexity in-the-wild?

Tapping into these technologies does not form part of the scope for this thesis, however forms a direction for future research; with their presence becoming more natural and accepted (like the smartphone), rather than the presence of foreign peripheral devices introduced just for experiments.

2.4.2.2 Extracting feature variables from the raw traces

After collecting data traces, feature variables are extracted from the raw data to create a vector representing the context for predicting interruptibility from (or the effect of the interruption). A common first step is to apply smoothing techniques to the data, in order to remove noise (e.g., [71]). However, in conducting the meta-analysis, broadly speaking there is little evidence of widely adopted conventions within interruptibility studies - likely due to scenario differences in the use of different data traces and hardware.

The extracted feature variables can be categorised as representing either the: user, environment, interruption, or the relationships between these. A previous survey by Ho and Intille [43] detailed 11 measures/variables that have previously been considered to influence interruptibility. However, due to the volume and breadth of studies since their work, Table 2.5 extends their observations of the types of features commonly used across the literature. It should be noted that the variables included here were identified where they were either explicitly stated or could be confidently inferred. While some features are likely scenario dependent (e.g., location), there are still large differences across works in the features used, with only a few reoccurring often. Again this supports that comparing and building from interruptibility works is challenging [107].

Type	Example types of features
User Features	Pupil size event statistics (e.g., [9, 13]), EEG event statistics (e.g., [71, 72, 129]), emotion (e.g., [105, 92, 36, 108]), learning style (e.g., [115]), personality (e.g., [115], demographics (e.g., [72]), time until next calendar event (e.g., [46, 113, 106]).
Environment Features	Location (e.g., [119, 91, 99, 105, 92, 115, 113, 82, 47]), other people present (e.g., [48, 31, 92]), states e.g., door open/closed (e.g., [31, 32]), cell tower id (e.g., [111]), connectivity (e.g., [111, 76, 46]), nearby Bluetooth (e.g., [92]), smartphone ringer state [97, 30, 72]), smartphone screen covered (e.g., [97, 99, 30, 77, 72]), smartphone orientation or position (e.g., [99, 67]), ambient noise (e.g., [30, 15]), light intensity (e.g., [72, 122, 24]).
Interruption Features	Content e.g., text or phone number (e.g., [111, 29, 102, 76]), task complexity (e.g., [37, 19]), number of queued interruptions (e.g., [97]), time between interruptions (e.g., [44]).
User and Environment Features	Time of day (e.g., [31, 115, 113, 82, 46, 97, 111, 99, 102]), day of the week (e.g., [115, 46, 97, 111, 105, 102]), user is in conversation (e.g., [119, 46, 31, 105, 46, 45, 108]), user's current activity (e.g., [31, 82, 115, 105, 92, 54, 57, 102, 83]), user is present (e.g., [31, 15, 44, 43, 48]), software event statistics (e.g., [97, 46, 82, 65, 118, 42, 18, 15, 54, 32, 44, 45, 33, 86]), unusual environment to be in (e.g., [91]), frustration level (e.g., [115, 8, 51]), stress (e.g., [105]), level of annoyance (e.g., [14]) respiration (e.g., [105]), ambient sound (e.g., [82, 46, 44, 108]), car movement (e.g., [59]), human motion (e.g., [59, 43, 117]), smartphone motions or acceleration (e.g., [99, 30]), PC active and inactive time (e.g., [46]), user head position and posture (e.g., [116]), device use statistics (e.g., [72, 89, 24, 77, 106]).
User and Interruption Features	Social relation (e.g., [119, 37, 102, 36, 10]), interruption frequency (e.g., [37]), content desirability (e.g., [91]), perceived mental effort (e.g., [8, 37]), perceived task performance (e.g., [37, 8]), resumption lag (e.g., [50, 8, 65, 80, 51]), perceived timeliness of delivery (e.g., [91]), number of primary task errors (e.g., [55, 14]), primary task duration (e.g., [8, 65, 46, 14]), elapsed time to switch to interruption (e.g., [97, 105, 45, 51]), primary task complexity (e.g., [115, 37]), interruption time (e.g., [111, 72]), interruption duration (e.g., [65, 8, 105, 14]), perceived time pressure (e.g., [8]), previous or next task cue presented (e.g., [55]), elapsed time before user reaction (e.g., [42]), influence from social contexts (e.g., [36]).

Table 2.5: A categorisation of common features. Table extended from [121].

Additionally, in highly constrained and volatile environments such as the smartphone, this transformation process (e.g., from raw microphone readings to the level of ambient noise) could be costly in terms of computational resources. Choosing appropriate and technically feasible data sources to create features from is common at the design phase, however reflections on the cost of transforming these into features (where relevant) has received little attention; yet could bring valuable design considerations for future studies and applications:

(RQ5) Can the utility of potentially influential variables be standardised by considering the trade-off between accuracy and sampling / processing complexities?

Several works have touched on this within wider domains, however, this is not common practice for individual interruptibility studies. For example, Lathia et al explore the issues relating to smartphone sensor sampling stability [64]. A future research direction could involve the formulation a standardised framework for quantifying the cost of retrieving individual feature variables on specific hardware, or investigations into the difficulties of doing. This is not a direct focus in the remainder of this thesis, however the stability of using typical Android hardware to support the data collection used is discussed in Chapter 4.

2.4.3 Datasets and participation

Across the literature, datasets have predominantly involved either a small number of subjects (up to approximately 20 participants) as seen in [32, 111, 92, 76, 116], up to approximately 100 as seen in [102, 37, 97, 99, 108, 72], with larger analysis of thousands of users being an uncommon and recent occurrence, as seen in [103, 65, 24]. Establishing guidelines for suitable dataset size and diversity has received little attention, however the importance of longitudinal data, in order to observe interruptibility habits over time has been stated (e.g., [71, 92, 55, 111]).

Additionally, there has been little attention towards the scalability and sustainability of

the architecture to collect datasets (or to support real-world interruptibility prediction systems). In early works, forming a dataset typically involved manual retrieval of the data from each participant (e.g., [43]), whereas the introduction of technologies such as the smartphone has enabled a more autonomous client-server model, supporting *in-the-wild* studies (e.g., [99]). With data traces potentially becoming more diverse and representative of our daily lives (as noted by RQ4), this raises another open question:

(RQ6) What architectural barriers remain in enabling the collection, storage, and processing of detailed sensor data and interruptibility behaviour at scale? More specifically, what roles should sensors, personal devices and servers play to minimise connectivity and processing bottlenecks?

Several architectural frameworks have been proposed that encompass wider intelligent interruption systems (e.g., Syke's "Interaction Management System Architectural Model" [115] and Iqbal and Bailey's "Oasis" framework for scheduling interruptions around tasks [52]). However, there is a lack of empirical evidence in the literature that these are feasible at scale and practical beyond controlled experiments.

Extending from this is the social, ethical and privacy standpoint for architectures and the resulting datasets, which leads to the following question:

(RQ7) What consent and anonymisation measures are appropriate for applications and researchers to know how interruptible someone is, and how does this balance with the potential for bias from the knowledge of behaviour monitoring?

This area has also received little attention but is fundamental to the viability of interruptibility research for real-world applications. With this in mind, there are currently are also no widely adopted conventions to provide "open data", impeding reproducibility of results. Further to this, the use of other datasets for benchmarking is also not a widely adopted convention; likely impeded by the different scenario and data collection choices seen across studies. Addressing RQ6 and RQ7 goes beyond the scope of this thesis, however remain future research directions.

2.4.3.1 Incentivising study participation

Additionally, obtaining quality data from participants requires user engagement, which in turn requires incentivisation. However, if incentives cause deviations from natural behaviour they can adversely affect a study and its conclusions. The balance of informed consent and behavioural bias extends beyond interruptibility into the wider research space of observing and learning from human behaviour (e.g., [79]). Popular methods within interruptibility studies for addressing bias and incentivisation include: using monetary compensation (e.g., [33, 91, 50]); providing feedback and visualisations to the user (e.g., [76, 78]); or providing an additional utility (e.g., mood diary features [99]).

The convention of experimenting *in-the-wild* (e.g., [92, 103, 71]) also addresses this bias to an extent by removing the locality limitations of a controlled experiment, promoting natural behaviour [79]. Ubiquitous technologies such as the smartphone are enablers for this as the experiment can operate within environments and conventions that the user is already comfortable with, such as mobile applications (e.g., [99]). However this only mitigates some data quality issues. For example, in many cases participants in such studies are self-selecting (e.g., [76, 77, 124, 99]), which can be challenging to control both the quantity and the quality of data.

2.5 Predicting interruptibility

Machine learning has been commonly used for producing predictive models of interruptibility. However, there is wide disparity across the literature within the components involved, from feature selection, through to classifier choice, training environments, and evaluation criteria. Likewise to the definition of interruptibility, scenario selection, and data collection, a key theme is the limited consideration of wider applicability of the choices and results beyond the confines of individual studies. It should also be noted that not all works study prediction; some simply explore frequency statistics and apply

statistical tests (e.g., [65, 55, 114, 83, 61, 21]) to determine whether certain factors correlate with interruptibility labels.

2.5.1 Finding influential features

It is plausible to assume that some chosen feature variables may provide more predictive power than others. Discovering this can be considered to be a pre-processing step before the training predictive models, as this can result in uninfluential features being removed. This can be referred to as *feature selection*, this step, which is not performed by all studies, primarily aims to balance the number of features used to build a predictive model and its accuracy. Additionally, this step can aid with determining and correcting for issues such as model *overfitting* (e.g., [68, 129, 24]); where a predictive model is heavily influenced by outliers in the underlying training data, which results in poor performance when tested with unseen data.

Common techniques for this include a statistical correlation-based approach (e.g., [31, 119, 129, 24]) where correlating features are considered influential to some extent, and a wrapper-based approach (e.g., [32, 31, 105]), where subsets of features are evaluated to quantify their effect on classification performance. Feature ranking is another technique, which ranks features using a defined measure, which has included measures such as *information gain* (e.g., [31, 76]) and the number of classifications that become incorrect after removing a feature (e.g., [97]). Direct comparisons of these techniques are uncommon in the literature, however, Fogarty et al [31] showed no significant difference between correlation and wrapper based methods for accuracy in their study, but the fewer features typically selected in a wrapper-based approach was deemed favourable. However, whether this is reflective more broadly is unclear.

Across these methods, a feature's importance is often measured using the effect it has on predictive performance. However, given the potential environments of practical interruptibility-aware systems, such as on smartphones, this may not be the only ap-

propriate measurement. An assumption is often implicitly made across the literature that the features extracted from the underlying data traces are accurate and reliable; however there has been investigations that this may not to be the case (e.g., [64, 79]). An extension of RQ5 could be to also include factors such as reliability and resource costs of the underlying data traces into the feature selection process.

2.5.2 From datasets to training sets

Predictive models are typically trained from a subset of a dataset, with the remainder then used for testing the model. The most common technique in the literature to achieve this is cross-validation (e.g., [58, 32, 71, 129, 72]). This involves splitting the dataset into a training set and a testing set multiple times and using the mean performance, mitigating potential skewness from using a single training set.

However, likewise to feature selection (Section 2.5.1), an additional task that can be performed is balancing the size of the training set with predictive performance (e.g., [32, 30]). The motivation behind this process is to reduce the overall complexity of the model and improve the viability of recreating the model in real-world applications by reducing the expected storage and processing requirements. In practice however, studies have had varying success. For example, Fogarty et al [32], showed evidence of diminishing returns in the accuracy that more training data brings, when using more than 40% of the original dataset. However, Fisher and Simmons [30] show clear fluctuations in the accuracy as more training data is considered, across several classifiers. In addition to this, more bespoke methods have been used to reduce training requirements. For example, Sarker et al [105] attempt to reduce the training data needs by using groups of cases at opposite polarities (in this case the 6 quickest and the 6 slowest responses to represent the user being “available” and “unavailable” respectively). However, it is unclear whether this would be feasible beyond scenarios where response speed is the primary concern.

Likewise to feature selection, it is arguable whether balancing the size of the training set (like the number of features) is the only appropriate measure. The impact of other factors such as temporal representation and diversity within the training data has not been widely explored within interruptibility studies, but could offer useful insight into the training data requirements of future studies and applications, prompting the following:

(RQ8) How do training dataset characteristics affect the diminishing returns of prediction performance?

Some aspects of the current literature are relevant to this question. For example, Smith et al consider *concept drift* in their analysis [111], where the values for some features may only appear in the test data, hindering the opportunity for an optimal model, motivating the need for diversity guidelines, if viable. This issue influences the focus of this thesis somewhat in the experiment of training from personal or aggregated data in the empirical analyses (Chapter 5), however remains an issue for further direct attention.

2.5.3 Training environment: offline vs online

Interruptibility studies have involved two distinct approaches to training environments: offline and online environments. Offline training environments are the most prominent across interruptibility studies (e.g., [119, 48, 72, 122, 24, 108, 116]), and involves building predictive models from all data, typically after data collection has taken place. In contrast, online learning (e.g., [76, 92, 111]) refers to a predictive model being retrained as more data becomes available, creating a feedback loop for relearning interruption behaviour over time. Within the literature, this approach is often utilised to improve upon the issue of having a lack of available training data initially, which can be mitigated by instead retraining models frequently.

Across these, studies have implemented these training environments using different hardware. Typically, predictive models are trained (and evaluated) after data collection,

without being deployed into a real-world environment (e.g., [124, 76, 129, 24, 108, 67, 31]). For studies that include the integration of predictive models into the interruption environment, this has been achieved through either implementing training capabilities into the interruptive device itself (e.g., a smartphone app [87, 92]) or through sending the data to a server for training and then sending the model back to the device (e.g., [115]).

However establishing guidelines on the suitability of each type of environment for interruptibility studies is still arguably in its infancy, leading to the following research question that extends RQ6:

(RQ9) When should intelligent interruption systems adopt online and offline learning, and what factors in the scenario and data collection influence this choice?

Generally speaking, the majority of current studies largely focus on a single technique, with only a few recent studies directly comparing performance (e.g., [111, 92]). This forms part of the focus of this thesis with the analyses conducted in Chapter 5 considering both together alongside different labels of interruptibility. Nevertheless, it remains a question in need of further direct contributions alongside RQ6.

2.5.4 Training data: aggregate vs personal

As well as the training environment, the type of training data can also vary, between either aggregating data from multiple users in order to build a single model (e.g., [31, 99, 105, 129, 67]) or keeping training data personal in order to build models for individual users (e.g., [58, 102, 72]). The debate of using either approach often concludes in favour of personalisation when compared together (e.g., [92, 76]). This is can be attributed to the variety of environments, activities, interruptions and preferences across users in their daily life. Personalised interruptibility models are also typically used alongside online learning (e.g., [92, 111]), because personal data to train from will likely be lacking initially in real-world use cases (e.g., newly installed mobile

applications).

Due to the fragmentation of works in terms of scenarios and features, it is hard to draw strong conclusions on which technique is more appropriate for which use cases. Further to this, mixing personal and aggregated data as a hybrid approach has also been suggested [32, 128]. In this case, aggregated data from other users could provide the initial model, which could then be removed as personalised data becomes available. As with offline and online learning, there is an absence of guidelines across the literature of when each is more appropriate, leading to a similar research question to RQ9:

(RQ10) Do personalised models mean better performance and how does this balance with increased complexity? Could a hybrid approach using personal and aggregated data reduce the training requirements for new users?

As prediction involves the use of at least one of these approaches, all works that investigate prediction (e.g., those highlighted in Table 2.6) provide a basis for addressing this research question. Within this however, the number of comparative works is currently limited (e.g., [92, 76]). Additionally there has been recent empirical investigation into a framework for facilitating a hybrid approach to training data for mobile notifications (e.g., [128]).

Likewise to RQ9, the analyses conducted in this thesis (Chapter 5) provide a basis towards addressing this question in addition to the above works, with comparisons made between using personalised and aggregated training data. However, further empirical studies (particularly beyond mobile notifications) are needed in order to make wider design considerations.

2.5.5 Classification and evaluation

Across interruptibility studies, several machine learning algorithms and statistical methods have been used (shown in Table 2.6), with the most common types being: tree, rule, function or Bayesian based classifiers. A typical convention has been to experiment

with multiple classifiers and determine which has the highest performance; sometimes supplemented with using statistical analysis (e.g., [31]). With the introduction of machine learning suites such as Weka [32, 31, 111, 92, 119] and MOA [92] providing a convenient means to facilitate this.

However, as with feature selection (Section 2.5.1), accuracy is arguably not the only metric that could be considered, particularly if training occurs on resource sensitive technologies such as the smartphone. In these cases, the computational complexity associated with generating and storing the model (or the connectivity requirements for sending the data to and from a server) are relevant but have not been widely considered. Some works consider these factors when choosing classifiers (e.g., [102]), however this is not a widely adopted convention across the literature.

Similarly there is widespread variation in the evaluation metrics used. Some studies offer confusion matrices (e.g., [32, 105, 82]), which aid in determining the wider applicability of the results, however more commonly, explicit evaluation metrics are chosen. Examples of these include, precision and recall (e.g., [99, 92, 116]), specificity and sensitivity (e.g., [76]), F-measure scores (e.g., [105, 108, 129, 72]), Kappa statistics (e.g., [105, 24]) and area under curve values (e.g., [99, 71]). In evaluating the wider applicability of results, this raises uncertainty, especially when alongside the use of different conventions in data collection and training, and likewise to those areas, remains an area in need of further direct attention to be improved upon, where possible.

In addition to evaluating the predictive performance as an absolute value, performance has also been evaluated by comparing the relative performance of a model in comparison to baselines. Some studies compare predictive performance against a baseline of classifying all moments as not interruptible (e.g., [32, 31, 99]) or interruptible (e.g., [119]), whereas others have compared performance against human estimators (e.g., [48, 31, 113]).

Towards enabling more comparability across works, an opportunity exists to construct a framework for evaluating and presenting the performance of predictive models of inter-

Environment	Algorithm/Technique	Example Studies
Offline	Naïve Bayes	[48, 32, 129, 31, 30, 97, 111, 92, 24, 119, 82, 76, 33, 72]
	Support Vector Machines	[48, 31, 30, 97, 111, 105, 24, 72]
	Decision Trees (e.g., J48/C4.5)	[48, 31, 30, 99, 87, 86, 24]
	Random Forests	[97, 59, 76, 67, 24, 72]
	Adaboost	[48, 31, 92, 72, 76]
	Bayesian Network	[44, 92, 46, 108]
	Logistic Regression	[102, 97, 72]
	Nearest Neighbour (e.g., k)	[30, 111]
	Neural Networks	[99]
	JRip	[99]
	RUSBoost	[111]
	Genetic Programming	[111]
	Association Rule Learning	[111]
	Adaptive Neuro Fuzzy Inference System	[115]
	Partial Least Squares	[37]
Online	Naïve Bayes	[92, 111, 76]
	Adaboost	[76]
	Random Forests	[76]
	Nearest Neighbour (e.g., k)	[111]
	Support Vector Machines	[111]
	RUSBoost	[111]
	Hoeffding Tree	[92]
	Ozaboost	[92]

Table 2.6: An overview of techniques and algorithms used for interruptibility prediction. Table extended from [121].

ruptibility. Within this would be conventions that cater for different scenario objectives in regards to evaluation metrics to use and baselines to compare performance against. Extending this would be the promotion of within-study discussion, or further empirical work, in how the reported performance may change beyond the interruptibility scenario of an individual study. As an extension of RQ2, this is a non-trivial problem, where the solution may not be a unified approach to conducting and evaluating interruptibility studies, but means in which the likely boundaries of wider applicability can be better

estimated. Addressing this is not a direct focus of this thesis, however an overarching theme in the forthcoming analyses is the consideration of these areas where possible (e.g., in the use of multiple evaluation criteria (Chapter 5)).

2.6 Conclusions

The ability to perceive the interruptibility of another human being has a fundamental influence on our ability to communicate effectively [16, 34, 126, 69]. The introduction of pervasive technologies capable of interruption, such as the mobile devices, has extended the impact of a machine's inability to do so into our daily lives. As a result, research has focused on building towards intelligent systems for predicting interruptibility. However, despite conventions existing in how interruptibility is defined, scenarios are selected, data is collected, and predictive models created, there remains wide disparity in these areas, creating challenges in generalising individual conclusions beyond the confines of individual studies. The intention of this chapter has been to highlight this issue, creating the following primary contribution:

C1 A survey of the fragmented research area, developing open research questions by highlighting limitations and gaps in existing methodologies and conventions for collecting, labelling, and predicting interruptibility.

2.6.1 Thesis scope

The remainder of this thesis focuses on the improvement of a subset of the conventions exposed, primarily surrounding the labelling of interruptibility and building predictive models from these. The primary focus is the development and validation of an improved framework for capturing and labelling response behaviour towards interruptions (such as mobile notifications), called the decision-on-information-gain (DOIG) model. The framework is supported through the analysis of the datasets from two large-scale in-the-

wild empirical studies of Android notifications; which also provide further empirical exploration of the conventions exposed in this chapter. Where possible, the bounds of generalisation of the analysis are also discussed.

Additionally, while the open research questions proposed are intended to be a standalone contribution of this thesis, this scope is relevant to a subset these:

RQ2: Chapters 3-4 and 6-7 contribute towards addressing RQ2 by developing the DOIG model to be flexible to different interruption scenarios, definitions of interruptibility (through the ways in which a user can respond), and interruption designs; as well as validating the need for the framework to be flexible.

RQ3: Chapters 3 and 7 also contribute towards RQ3 while investigating the validity of the DOIG model, by deconstructing and examining the natural decision-making behaviour in notification responses.

RQ's 8, 9, and 10: Chapter 5 contributes towards addressing these by exploring the relative performance differences between predictive models built for the labels produced by the DOIG model, using various machine learning strategies for training and evaluation.

The limitations of these analyses, as well as the remaining research questions, serve as a basis for future work (discussed in Chapter 8, Section 8.2.3) beyond the scope of this thesis.

Chapter 3

Interruptibility behaviour as a decision process

Interruptibility can be fundamentally represented as a binary classification problem, where given a moment in time, an intelligent system (e.g., as part of a mobile app) needs to decide whether or not to interrupt and deliver information (e.g., through a notification). This is achieved through a predictive model built from a dataset of previous response behaviour, typically a set of feature vectors (built from contextual data) with associated labels of whether the user was interruptible or not. Approaches for determining these labels has received little direct attention, in comparison to explorations of influential contextual data (e.g., [43, 58, 81, 99, 76]) and prediction strategies (e.g., [111, 92, 78]).

In this chapter a new method is proposed for capturing and labelling interruption behaviour. Referred to as the *decision-on-information-gain* (DOIG) model, the approach extends the existing convention to label response behaviour from how a response is made (as well as if), enabling different definitions of interruptibility to be represented. It can also be implemented passively without a reliance on surveys, demonstrated through an in-the-wild study.

The rationale behind the approach is discussed further in Section 3.1, and formally defined in Section 3.2. The practical applicability is then examined in Section 3.3 using an in-the-wild study of 11,346 passively captured responses towards Android notifications. Section 3.4 then uses the dataset to show how individuals can respond to

Android notifications in different ways and demonstrates how the DOIG model offers improvement over the existing convention for labelling, by considering the impact of partial responses.

3.1 Decomposing response behaviour for labelling interruptibility

While previous empirical studies have been successful in predicting interruptibility labels (e.g., [30, 102, 76]), a common assumption has been to represent the response process as a single large decision, where the recipient either responds (fully) or not at all. As a result, the label produced is often determined through whether the recipient of the interruption completes a set task (as discussed in Chapter 2, Section 2.4.1). The exact task has varied across individual study scenarios, ranging from explicit tasks such as filling in a survey (e.g., [99, 92, 76]) to passive observations of whether the interrupting content is consumed (e.g., [13, 51, 52, 97]). However common limitations exist with this approach which help frame the development of the DOIG model, including[1]:

L3.1 This assumes that if the user is interruptible, then they will complete the labelling task. This presents data quality issues in the labels produced, particularly in in-the-wild environments. In many cases this involves a counter intuitive approach to data collection of interrupting the individual to ask how interruptible they are.

L3.2 This does not consider the potential variability in the extent of a response (no response started, partial, or complete), resulting in labels potentially being formed from an under-representation of the response that a recipient gave.

L3.3 This does not account for the subjectively in what response behaviour signifies a successful interruption, or variability in the content and design of interruptions, beyond the confines of individual study scenarios.

[1] labelled using: L(imitation){chapter number}.{enumeration}

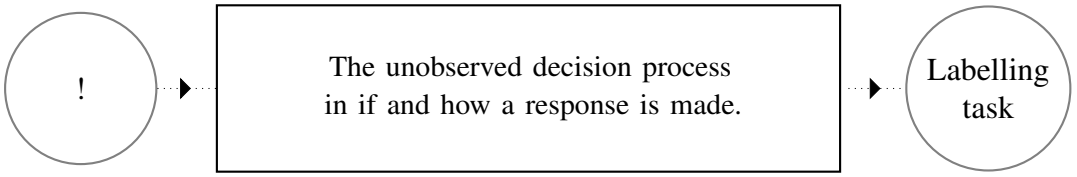


Figure 3.1: An abstract representation of the common black-box convention for labelling interruptibility. After being interrupted (!), if a user chooses to respond and eventually reaches and consumes the content (e.g., by tapping on a mobile notification), a labelling task is then performed by the user (either explicitly through a survey or passively).

L3.4 Where an explicit labelling task is used (e.g., surveys), implementation in real-world applications is often impractical due to the additional intrusiveness and requirements that this places on recipients.

As a result, existing empirical studies can be broadly grouped together as using a *black-box* approach to labelling [122, 124], where the focus is on the user completing a specific end-goal behaviour that denotes interruptibility (visualised in Figure 3.1). This motivates adapting the existing labelling convention to improve upon these limitations, where possible, in order to improve the quality of the training data used for predicting interruptibility.

While the black-box approach is useful in that it can be wrapped around any interruption, it under-represents scenarios where information surrounding the interruption is presented in a step-wise manner and the user has degrees of freedom in how they respond. In these cases, the recipient could still be considered interruptible if they start to respond but do not perform the labelling task. For example, in the case of Android mobile notifications, an application could consider the notification to be a success if the user at least reacts and notices the notification, even if they do not physically tap on and consume it; or they may wish to at least differentiate between these cases and where no response is started at all.

Towards improving upon these limitations, several key contributions in the literature can be considered. Firstly, the work of McFarlane and Latorella [75] is an early contribution

that sought to understand the different ways in which individuals handle interruptions. This included the proposal of an abstract representation of the interruption process for machine-to-human interactions (Interruption Management Stage Model), which models a series of decision-making steps from pausing the current task through to returning back to it. However, this work stops short of modelling the decision processes between the initial decision to switch focus and finally returning to the previous task (i.e., what the user does inside the black-box depicted in Figure 3.1 after deciding to respond).

Additionally, within the context of mobile notifications, various works have proposed sub-components of interruptibility that isolate various definitions, including the *attentiveness* towards a notification (e.g., whether a user responds or not [97, 92]), and the *receptivity* towards it (e.g., did the user like what they were interrupted with [76]). However, while these works show that interruptibility towards mobile notifications can be defined differently, the resulting fragmentation in these definitions have not been synthesised together into a flexible framework for labelling response behaviour. This forms the focus of this work, in synthesising these sub-components with an extension of the ideology of modelling an interruption response as a decision process.

3.2 The Decision-On-Information-Gain (DOIG) model

The *decision-on-information-gain* model is a proposed framework for supporting the capture and labelling of response behaviour towards interruptions [122, 124]. While the focus of the empirical work in this thesis surrounds Android mobile notifications, the concept can be generalised into an abstract model, consistent with the previous work by McFarlane and Latorella [75]. This approach models the extent to which the recipient pursues information about the interruption from the point of interruption (i.e., what the user does inside the black-box visualised in Figure 3.1). From this, interruptibility can be labelled flexibly on a per-application basis, by enabling the interrupting application to define what type of response behaviour (i.e., the extent to

which a user responds) signifies that the interruption was successful (discussed further in Sections 3.2.1 and 3.2.2). The model is therefore the most appropriate for cases where information surrounding interruptions is revealed in a step-wise manner, such as with Android mobile notifications. However it is flexible to other scenarios by reverting back to the existing convention of a single decision in the worse case if all information is instead revealed at the point of interruption.

3.2.1 Abstract model

The DOIG model follows the decisions a user *must* engage with (either consciously or subconsciously) in response to an interruption, dictated by the interruption environment. These decisions are defined as points in the response where the user must choose whether to act further to gain more information about the interruption, or exit the interruption response.

The initial decision is whether to switch focus after being prompted, e.g., after an audio alert. Subsequently there can be k points where extra information is provided (such as the identity of the interrupter or the subject topic). This produces a set of $k + 1$ sequential decisions that are required for a complete response to the interruption, D=$\{d_1, d_2, ...d_{k+1}\}$ - where decision d_i precedes d_{i+1}. While the exact number of decisions may vary based on the interruption characteristics, a general rule is that a decision will occur each time the user is given new information as they respond. It is important to note that this approach intends to observe the natural decisions that are already being made and that this does not change the response process in any way.

A sub-sequence $\{d_1, d_2, ...d_i\}$, where $i \leq (k + 1)$, captures the extent of the users response, with d_i indicating the decision the responder exited the response. An application can then use this to determine whether the user responded enough (in their opinion) to be considered interruptible. In comparison, a black-box approach assumes that for an interruption to be successful, a complete response must be performed, that is while all

decision steps $d_1, \ldots, d_{k+1}$ are assumed to be carried out, only the final decision d_{k+1} is typically assessed. Consequently the black-box approach is inherently susceptible to under-representing the choices that a user makes during the response as they are presented with more information about the interruption. This is particularly useful for applications that can consider an interruption to be successful at an earlier decision than d_{k+1}, e.g., a partial response to a mobile notification, where it is noticed but not tapped on and consumed.

3.2.2 Applying the DOIG model to Android notifications

Conceptually, the DOIG model is suitable for scenarios using Android notifications. The nature of Android notifications causes the responder to discover information about a notification in stages. This enables decisions to be made on whether to continue on towards consuming a notification, or abandon the response part way through. Rather than making an assumption on what point in the response behaviour correctly signifies being interruptible (i.e., the measure of success), which will likely change on at least a per-application basis, a spectrum of potential responses can be considered (shown in Figure 3.2); these being:

- **Null Responses** - Cases where the user does not show any observable response behaviour, either because the user was not physically interrupted or did not want to switch tasks for any notification, from any application.
- **Partial Responses** - Cases where the user begins to respond, but abandons after further information. For example, they turn the screen on, discover the notification relates to an email (e.g., through the icon displayed on the top bar, Figure 1.3) but exit at that point (or after unlocking the device and reading the sender or subject).
- **Complete Responses** - Cases where the user consumes the notification and completes a response. For example, tapping on the notification and reading an email or filling in a survey.

Given that a response can be null, partial, or complete, the potential measures for success can be described as whether the user is at least *reachable* [122], willing to *engage* [122] to some extent, or is *receptive* [29, 122, 76] to what they are interrupted with. These independent measures fit together under the wider umbrella of mobile notification interruptibility, providing flexibility for labelling interruption behaviour for different definitions. From this the following terms can be defined:

- **Reachability**, which indicates whether a response will at least be started (i.e., not null), or not.
- **Engageability**, which indicates whether a response will be started but abandoned without formally consuming the notification (i.e., partial response), either because merely noticing the notification is sufficient, or it is undesirable to pursue it further.
- **Receptivity**, which indicates whether the user is receptive to the notification content and either consumes it by removing it in some way (i.e., complete response).

Modelling a range of response behaviour (as shown in Figure 3.2) means that different definitions of what constitutes a successful notification can be accounted for. It may be that an application considers a notification to be a success if the user was reachable (i.e., the response is not null), such as reminders. Whereas others may require the user to reach a specific later stage in the response (i.e., at least engageable), or consume it completely and open the application (i.e., receptive).

This is contrary to the wider research space, that typically labels interruptibility using a strict measure of success (e.g., just receptivity [29]), which often relies on the user to open the interrupting application (e.g., [97]) or fill in a survey (e.g., [92, 99, 108, 94, 76, 129]).

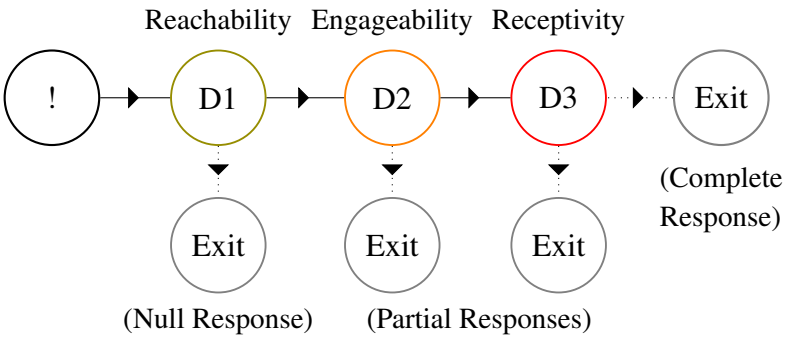


Figure 3.2: A visualisation of the linear sequence of decisions made during a typical response to an Android notification ($k = 3$). After the interruption occurs (!), at each point new information is given (e.g., the application icon) the user must decide (e.g., D1) whether to continue on to the next decision (e.g., D2), (up until either the notification is consumed) or exit at a particular decision. Figure from [124].

3.2.2.1 Flexibility and limitations in applying the model for Android and other notification systems

Several uncontrollable factors can impact what unique decisions are observable in response to an Android notification. The DOIG model is flexible to these limitations by not defining a set number of decisions or strict methods for observing decisions being made, allowing the model to remain usable if the Android notification ecosystem evolves over time.

For example, due to technical restrictions imposed by the Android operating system, some relevant UI events (e.g., accessing the Notification Drawer, shown in Figure 1.2) are not observable by third party applications without privacy-sensitive Accessibility permissions. This limits which decisions are observable, particularly when the device is in use. Discussed further in Appendix Tables A.1 and A.2, if the device is not in use when the notification is delivered, decision outcomes for D1 and D2 (Figure 3.2) can be observed through the process of the user turning on the screen (indicating reachability) and unlocking the device (indicating engageability) to discover more information. However, if the device is already in use, while a decision process occurs, there are currently no observable system events for D1 and D2.

Secondly, technical limitations of these devices prevent the ability to distinguish between the reasons why a user may not be reachable. For example, it could be that an individual is not physically interrupted in order to make the first decision to begin responding or not, or it could be that they were and chose not to switch from their current task. This is a challenge for interruptibility studies in general, leading to some studies investigating the physiological ability to switch focus explicitly as discussed in Chapter 2 (e.g., using EEG readings [71]). However from the perspective of labelling notifications using the DOIG model, this issue is mitigated as in either case, the outcome remains the same in that the individual was not reachable and the notification was ineffective.

Thirdly, the example decision process visualised in Figure 3.2 represents a typical Android notification. While the notification convention is standardised and imposes design constraints, some variability remains for individual applications in what information can be presented, when, and how. Additionally some more recent versions of Android can enable the user to show some notifications on the lock-screen. In both of these cases, the number of observable decisions (k) will need to be adapted, with Chapter 6 addressing this explicitly. For example, D1 and D2 may be merged if the audio tone used for interruption is distinguishable for a given application.

Beyond Android, other mobile operating systems (such as iOS) or environments (such as PC tasks [118, 116]) have slightly different implementations of notifications. For example, on iOS devices, notifications can turn on the screen without explicit user interaction. However, as the intention here is to observe and not change how a notification is presented and responded to, these variations require a flexible model (and for any future changes), which the DOIG model provides. Finally, in the worse case, the DOIG model falls back to capturing the same information as a black-box approach (i.e., interruption interaction events).

3.3 Applying the DOIG model: ImprompDo Android application

A bespoke Android application was developed, called *ImprompDo* [122, 124], to quantify empirically whether the DOIG model brings a useful utility in capturing and representing response behaviour towards Android notifications that would otherwise be missed with a typical black-box approach. ImprompDo captures context data and response behaviour to productivity notifications in-the-wild (discussed further in Sections 3.3.3 and 3.3.4) and was distributed freely through the Google Play Store (shown in Figure 3.3) for devices running Android 4.0 to 4.4 (inclusive), which covered ~85%-94% of the market distribution at the time of the study from July 2014 - January 2015[2].

Participants of the study were self-selecting and remained anonymous, with the application marketed through social media and online news outlets (e.g., [120]). The application received generally positive reviews from media outlets (e.g., Lifehacker UK [35]) and users[3], but some users did suffer from issues such as device specific bugs that were challenging to test for, and users being unable to set up the productivity side of the application successfully. However, this is arguably reflective of Android applications in general due to device manufacturer variability and serves as an example of the challenges of performing in-the-wild research studies remotely on these devices.

3.3.1 Rationale for an in-the-wild application design

ImprompDo is an example of an implicit in-the-wild study of interruptibility (as defined in Chapter 2, Table 2.3). It represents an example of a real world application where an intelligent interruption system would be suitable, which has been a common design

[2] As per Google's "Dashboards" at the time of the study - `https://developer.android.com/about/dashboards/index.html`

[3] ImprompDo received an average Google Play Store rating of 4.1/5 by the end of the study

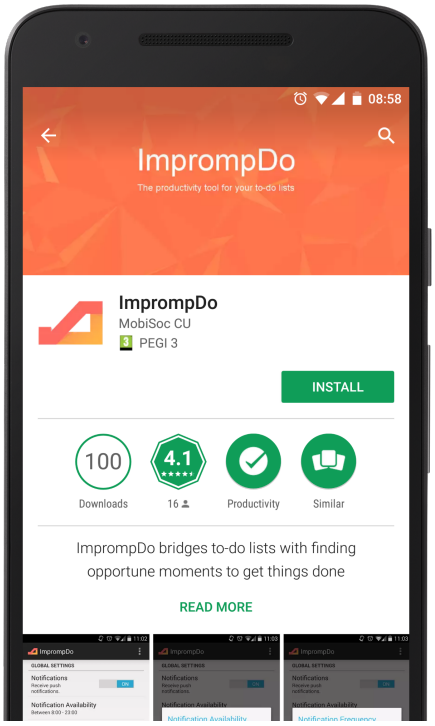


Figure 3.3: The ImprompDo app listing on the Google Play Store.

choice of similar empirical studies in the area, including mood diaries (e.g., [99]), and apps issuing news stories and weather updates (e.g., [76]). However, it should be noted that the application's productivity focus and in-the-wild nature could therefore result in a bias in user participation towards individuals that are more productivity driven. To incentivise participation, the app was designed to perform as a useful productivity tool, rather than through monetary compensation. This aimed to promote natural behaviour, in comparison to relying on volunteers merely willing to be interrupted, as seen in previous studies (e.g., [92, 76]). Additionally, the application was developed in accordance with the ethical research requirements and processes of Cardiff University and followed Android's official design guidelines at the time of creation.

Ideally, a dataset should contain response behaviour which represents all possible notifications. In reality, notifications are diverse in design and purpose, and experimenting with a one-size-fits-all notification would not be possible beyond a controlled research study. Additionally, interrupting the user without a purpose in an effort to be more gen-

eric would make the notification unrepresentative of all practical scenarios. Therefore the notifications produced by the application (discussed further in Section 3.3.3) were made to be as generic as possible in their interruptive characteristics, in order to remain broadly representative.

3.3.2 Installation and setup

Through the Google Play Store, self-selecting participants could discover and install the application onto their Android device in the same manner as other applications. After installing the application, it remained dormant until the user completed a setup process. Upon opening ImprompDo for the first time, users were presented with the setup process and provided with links to a disclaimer, EULA, and privacy policy. Before being able to progress any further, the user was asked to provide consent to the anonymised data collection. After granting consent, these documents were then made available for review at anytime through a menu on the main user interface.

After completing the setup process, a single user interface is then used to manage the application, shown in Figure 3.4. Users were able to set a time range of hours in which they were happy for notifications to occur (by default this was 9am to 9pm) and a maximum frequency within this period (by default this was once per hour). Additionally, users were required to enable access to at least one of the following one to-do list services in order for notifications to occur: Google Tasks or Todoist. After which, the application was able to use the web APIs of the service(s) to retrieve random to-do list items as the final content shown in the notifications (discussed further in Section 3.3.3). Finally, this interface could also be used to pause participation indefinitely, alternatively users were free to uninstall the application at any time.

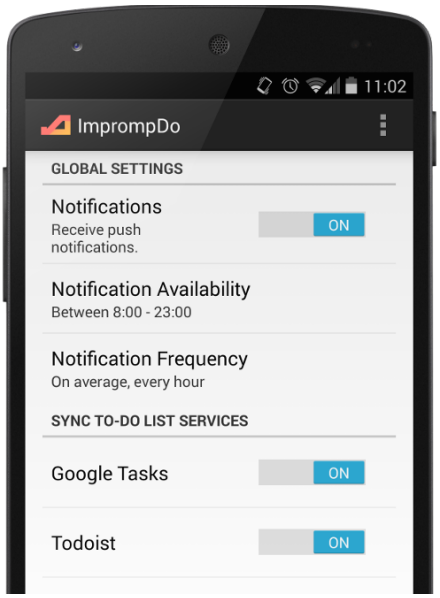
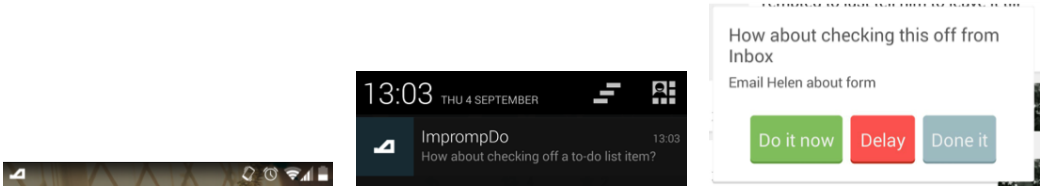


Figure 3.4: The main UI screen for the ImprompDo app after initial setup. Figure from [122].

3.3.3 Interruptions: Android notifications

The application periodically interrupted the user through Android's notification framework. Notifications used the device's default tone, vibration pattern and visual cues, while adhering to the device's global volume settings in situ. If the user is interrupted, they respond in the same way as any other Android notification (shown in Figures 3.5a through 3.5c). That is, assuming the user decides to continue at every decision point (shown in Figure 3.2), they turn on the screen, unlock the device (unless it is already in use), access the notification drawer and tap on or dismiss the notification. The user is then presented with a random item from their to-do list and buttons to manage it.

Each user is interrupted by the notifications periodically using 1 of 4 randomly selected triggers, while respecting preferences in available hours and frequency (shown in Figure 3.4). Inspired from the conclusions of related works (e.g., [43, 59]), these are: at a random time; at the end of a period of acceleration; an X in 10 chance to occur at a random time, where X increments or decrements each time a notification in that hour on previous days is consumed or not; and a binary Logistic Regression model trained



(a) The application icon shown for an example ImprompDo notification

(b) The Notification Drawer showing an ImprompDo notification

(c) The application content shown if the ImprompDo notification is consumed

Figure 3.5: The Android notification response process used. Figures from [122, 124].

from whether notifications were fully consumed in similar contexts in the previous 7 days. Appendix Table A.3 describes these in greater detail.

As the focus of this study was on near-real time interruptibility, notifications were removed after 30 seconds if the user did not remove the notification. This allows the immediate interruptibility of the user to be assessed in various contexts and minimise the likelihood of a response being the result of a coincidental interaction with the device at a later time. While response behaviour beyond this timeframe is interesting in its own right, this is beyond the scope of this analysis. However this forms part of the scope of Chapter 7, in investigating behavioural patterns in notification behaviour across the device.

3.3.4 Data collection: response behaviour and contextual data

As notifications are delivered, data samples are taken from hardware sensors and software APIs on the device using a background service; starting from 5s before and running until the notification is removed in some way or 30 seconds has past after delivery (shown in Figure 3.6). Further details of the sampling process are outlined in Chapter 4, Section 4.2.1. This *passive sampling* provides a trace of how the user interacted with the device in response to the notification (e.g., turning the screen on, unlocking the device, etc), enabling interruptibllity to be labelled in the same manner as described in Section 3.2.2). Additionally, the data provides an in situ representation of

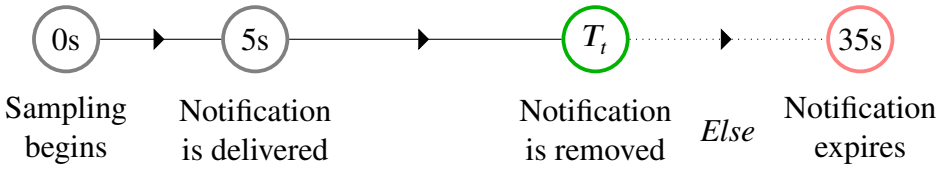


Figure 3.6: Visualisation of the the data collection process, from 5 seconds before delivery up until the notification is consumed (at T_t ($5s < T_t < 35s$)) or it expires. Figure from [122].

the device and its environment (similarly to [99, 97]), enabling features to be extracted for predicting interruptibility from (discussed further in Chapter 4).

In comparison to the typical conventions used in previous studies (discussed in Chapter 2, and in [121]), this removes limitations such as: relying on the user to provide information and labelling through surveys (e.g., as used in [92]); permissions that are privacy invasive and out-of-place for most applications (e.g., as used in [77]) and needing persistent monitoring of device state changes (e.g., as used in [97]).

3.3.4.1 Collecting response behaviour for labelling

To determine the response behaviour for creating the label, changes in the screen state, lock state, and notification interaction events are used; which occur as a by-product of the user conducting the response to each notification (as described in Section 3.2.2). For example, if the device is not in use, if the user turns the screen on to begin interacting with the device, this indicates reachability, with unlocking the device to investigate further indicating engageability. Appendix Tables A.1 and A.2 describe the use of these events in greater detail.

However, as stated in Section 3.2.2.1, the decisions that are able to be captured varies depending on whether the device is in use or not at the time the interruption occurred. To determine this, readings from the screen state API can be used, if the screen is off, the device can be deemed to be not in use, otherwise it is considered in use. This is consistent with the approach used by other works investigating smartphone behaviour

(e.g., [90, 110]). As data readings occurred at irregular intervals, this is calculated from the reading taken closest to the time of the notification's delivery, within $\pm.5$ seconds.

3.3.4.2 Collecting contextual data

To gain contextual data for prediction (discussed in Chapters 4 and 5), a bottom-up approach of collecting from a variety of different data sources was used. To maintain wider applicability beyond the scope of this study application, data sources were chosen that: are present on the majority of devices; do not require additional privacy invasive permissions that would not be consistent with what is expected for most applications (e.g., microphone, location, calendar), which may also introduce a behavioural bias even if the user accepts them [63]; require persistent monitoring of the device (e.g., device usage data or detailed activity recognition [97]); or require a fundamental change to how a user interacts with an application (e.g., in needing to answer surveys [92, 76]). As a result, data was collected from the following sensor and software API sources (in addition to the current timestamp):

- Linear acceleration (accelerometer)
- Gravity (accelerometer)
- Light sensor
- Proximity sensor
- Battery charging state
- Rotation vector
- Gyroscope
- Pressure sensor
- Magnetic field sensor

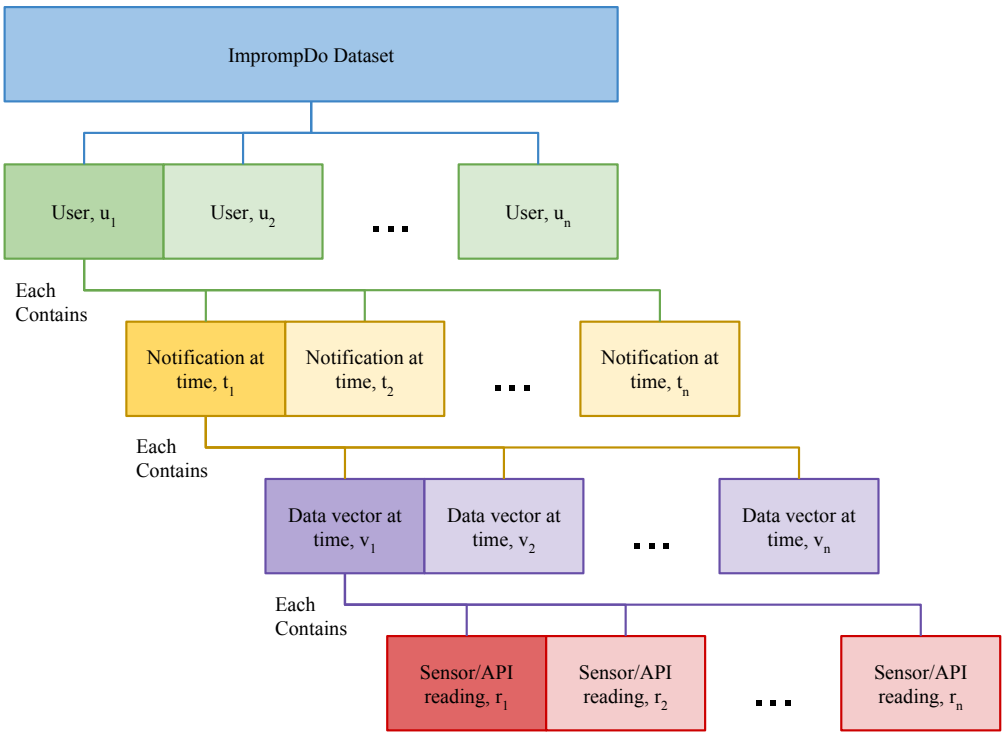


Figure 3.7: A visualisation of the ImprompDo dataset structure.

- Device screen state (on/off) and lock state (locked/unlocked)
- Device volume state (silent, audible, or vibrate)

The sampling process and reliability of these data sources is discussed further in Chapter 4, along with the features extracted from these data sources. The remainder of this chapter uses the resulting dataset of this application to compare the DOIG model against the current black-box convention for labelling interruptibility. This is to firstly show a benefit to using the DOIG model, before moving forward towards predicting reachability, engageability, and receptivity using features extracted from the contextual data.

3.3.5 Dataset

The resulting dataset from the study contains 11,346 notifications, each with an associated set of data vectors containing raw sensor and software API data (visualised in Figure 3.7). This was collected over 178 days between July 2014 and January 2015, with 224 participants installing the application over the period and 93 (41.5%) providing data for at least 1 notification - producing a relatively large population in comparison to similar studies (as discussed in Chapter 2, Section 2.4.3, and in [121]). Participants used the application for an average of 26.5 days ($Min = 1$, $Max = 129.6$, $SD = 35.6$), received an average of 122 notifications ($Min = 1$, $Max = 781$, $SD = 175.3$), with each notification having an average of 65.3 data vectors ($Min = 0$, $Max = 840$, $SD = 7.6$). Further breakdowns of the dataset can be found in Appendix Tables A.4-A.8.

The remaining analysis of this chapter uses this dataset as the basis for determining whether the DOIG model is comparatively more suitable for labelling interruptibility than the existing convention. From this, it is also used to explore the feasibility of passive data collection on Android devices in greater detail (Chapter 4), whether prediction is worthwhile to pursue (through whether contextual data correlates to the different labels produced by the model, also in Chapter 4), and in exploring different machine learning strategies for predicting the labels (Chapter 5).

Beyond this analysis, the versions of Android that were dominant at the time of the study are limited in their ability to enable the observation of wider notification behaviour (e.g., what other notifications occur on the device). While the analysis in this chapter and 4 through 5 is inline with similar works in the area (e.g., [102, 92, 76]), the fast moving nature of Android's development has enabled the collection of additional data from a device-wide viewpoint. This motivates and enables Chapters 6 and 7, which use further in-the-wild data to a) explore the flexibility of the DOIG model, and b) examine decision making in the response behaviour beyond an individual notification.

3.4 DOIG model versus black-box labelling

The purpose of this analysis is to examine the extent to which null, partial, and complete responses occurred in the dataset, and subsequently examine the benefit of the DOIG model in labelling this behaviour in comparison to the existing convention. As fewer decisions can be observed if the device was in use than not, the data was split into two groups. However, data collection issues prevented either the in use state or overall response behaviour to be observed for 1287 notifications (11.4%), which were excluded from the analysis.

A hypothesis can be made in that extending the black-box approach using the DOIG model captures additional information that is useful for labelling. To determine this empirically, a comparison can be made between the number of responses captured by the DOIG model where the user at least partially responded to the notification (i.e., the user was reachable, engageable, or receptive), against those responses that would be captured by a typical black-box approach (i.e., receptive only). As described in Section 3.2.2, a response is considered partial if the user is at least reachable, but is not receptive (i.e., the user at least turns the screen on, but does not tap on and consume the notification, it either expires or is dismissed). Applying a black-box approach to the same data would typically not capture this data, however, it should be noted that this could also include the capture of notification dismissals. Therefore the analysis is conducted for both cases, those that include dismissals and those which do not.

The results show that 1317/10,059 (13.1%) of all cases were partial responses if dismissals are included, or 802/10059 (8%) if not. These cases would be missed by a commonly used black-box approach, which would misclassify these cases as a null-response (i.e., not interruptible at all). By combining partial responses and complete responses, the total number of cases where at least some degree of interruptibility was shown increases from 1056 with a black-box approach to 2373 with the DOIG model if dismissals are considered as partial responses - a substantial 124.7% increase. Alternatively, if dismissals are captured by a black-box approach, this increases the total

from 1571 to 2373, a 51.1% increase. These results show that using the DOIG model to capture user interactions with the device, and subsequently observe the decisions being made, isolates responses that are: not started, i.e., null (unreachable) responses; those that are started but abandoned, i.e., partial (engaged) responses; and those which consume the notification, i.e., complete (receptive) responses.

From a usability standpoint, this suggests that observing the response process using the DOIG model is more worthwhile for applications than solely relying on notifications being consumed. For example, the ImprompDo application represents a use case where knowing that the user was at least reached is useful as this is indicative that the user made a decision regarding their productivity. This is not exclusive to to-do list applications and applies to other applications which issue single purpose notifications (e.g., in hydration or exercise reminders) where merely seeing that a notification has arrived may have the desired effect, even if the notification is then not consumed.

Alternatively, for other applications which require the user to completely consume the notification to be considered successful, the DOIG model still provides a useful utility in being able to distinguish between cases where the user did not respond at all and those where they partially did (i.e., they were at least reachable). From a practical standpoint, the data collection application itself also serves as evidence that passive observation using the DOIG model is feasible, and without privacy sensitive permissions or a persistent background service which has commonly been used (e.g., [97]).

3.4.1 Exploring response time

In collecting the ImprompDo dataset, an assumption was made that if a user were to respond immediately as a result of being interrupted, they would do so within 30 seconds. As this assumption could impact the above comparison of the DOIG model and the black-box convention, the suitability of this threshold value is explored. Figure 3.8 visualises the response times of notifications that were either consumed or dismissed

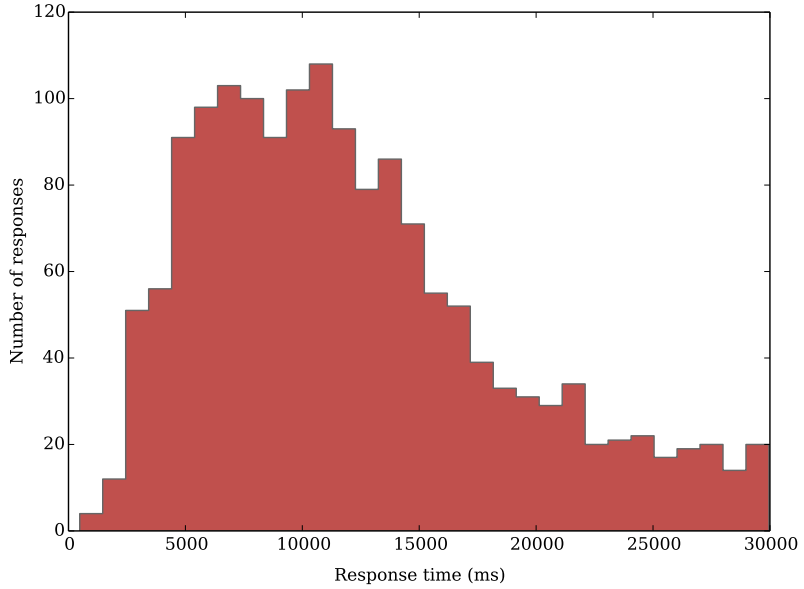


Figure 3.8: Histogram of response times for notifications that were either consumed or dismissed, using a bin size of 1000 milliseconds.

(as a complete response represents the longest response time a user would need).

The results support that the threshold of 30 seconds is suitable to allow a user to consume the notification, with the majority of responses occurring between 3 and 17 seconds (millisecond statistics: $M = 12,188.8$; $Min = 481$; $Max = 29,958$; $SD = 6513.9$; $N = 1571$). When splitting the data between those in use and not, there were only minor changes to the distribution. The primary difference is a quicker mean response time when the device was in use ($M = 897.4$; $Min = 481$; $Max = 29,504$; $SD = 6039.1$; $N = 575$), which is expected given that the user does not have to go through the process of unlocking the device. With the mean response time for cases where the device was not in use taking longer, but still less than half of the 30s timeout ($M = 14,046.7$; $Min = 3,798$; $Max = 29,958$; $SD = 6036.5$; $N = 996$).

Summary: Observing decision making behaviour in interruption responses is worthwhile

The above results can be summarised by the following primary findings:

- Individuals are often not immediately interruptible for notifications (i.e., within 30 seconds);
- When they are, users respond to notifications to different extents (i.e,. partial as well as complete responses);
- The DOIG model offers improvement over the black-box convention for capturing these cases and labelling interruptibility.

Overall, the high number of null responses emphasises the need for interruptibility aware notification systems, with the number of partial responses suggesting that this behaviour should be observed and considered. The DOIG model therefore offers a useful extension to the existing black-box convention that can help applications in labelling interruptibility behaviour. However, the model is arguably only useful if the labels it produces are predictable, which forms a focus of the following chapters (Chapters 4 and 5).

3.5 Conclusions

While considerable progress has been made concerning capturing and predicting interruptibility across the literature, the research area is fragmented with specific solutions for specific interruptions and environments [43, 122]. Despite this, a broad convention has been to treat the response process towards an interruption as a single decision, with interruptibility labels determined around whether the user consumes the interruption content or not - creating a number of limitations as outlined in Section 3.1.

However, depending on the interruption environment, degrees of freedom can exist in

the response process that result in the user needing to make decisions during the response itself. For example, in environments where information surrounding the interruption is delivered in stages that require further interaction to access (as is the case with Android mobile notifications). Subsequently, partial responses can occur that can indicate some degree of interruptibility, even if the interruption content is not physically consumed.

In this chapter, a new approach to labelling is proposed (that extends the existing convention) to expose and capture this behaviour where possible, called the decision-on-information-gain model. Through an in-the-wild case study of Android notifications, partial responses were shown to feature prominently in comparison to complete responses (Section 3.4), demonstrating the utility of the model for labelling interruptibility. This forms the following contribution to this thesis:

C2 A flexible model for labelling interruptibility for different definitions, the Decision-On-Information-Gain (DOIG) model, that deconstructs the observable behavioural trace in a response to a notification.

The model improves upon the limitations of the existing convention outlined in Section 3.1. Limitations L3.1 and L3.2 are improved upon by the model deconstructing how a response is made into the conscious or subconscious decision steps that a user makes; with the worst case performance being the same as the existing convention (e.g., in cases where technical constraints exist in observing decision behaviour). L3.3 is improved upon by enabling individual applications to infer what decision behaviour (i.e., the extent to which a user responded) makes the user interruptible and their interruption successful. Finally L3.1 and L3.4 are improved upon by utilising passive collection over human annotation, with the ImprompDo case study providing as a demonstration of the model being feasible for real-world applications, and without a fundamental change to notification design or user experience.

As the DOIG model does not alter the response process to notifications, the finer granularity of observation that it enables produces a second contribution that is related, but independent to C2, using the ImprompDo dataset:

C3 Analysis into the natural decision behaviour underpinning interactions with notifications, using data collected in-the-wild.

This analysis has examined what kinds of behaviour can occur inside the "black-box" (Figure 3.1) of how a response is made, and while this motivates the DOIG model for labelling, the variety of responses is interesting in its own right.

Subsequent chapters of this thesis build upon this by investigating the practical feasibility of the DOIG model for implementation on Android devices further, by exploring whether contextual factors are reliable to sample from and correlate with the labels produced by the DOIG model (Chapter 4), towards building predictive models (in Chapter 5). Finally, the flexibility of the model is demonstrated for other, more customised Android notification designs and the variability that can exist in device preferences (Chapter 6), as well as whether the decision making behaviour observed in this chapter can be seen for other notifications across the device (Chapter 7).

Chapter 4

Sampling context and decision data

The decision-on-information-gain model has been shown to be a useful aid in labelling interruptibility from response behaviour (Chapter 3). However, its utility is ultimately tied to the ability to predict the labels it helps to produce. In order to motivate the building of predictive models of the reachability, engageability, and receptivity labels produced using the DOIG model, the purpose of this chapter is to examine the feasibility of passively sampling relevant data on Android devices that can be used for either creating the labels, or for forming contextual features to use as the basis for prediction. This also extends the observations of potentially influential contextual data seen in previous studies (discussed in Chapter 2 and in [121]) by considering both multiple labels of interruptibility and the technical feasibility of retrieving the data together.

Firstly, the rationale behind this analysis is discussed further in Section 4.1. The data sampling strategy for the ImprompDo application introduced in Chapter 3 is detailed in Section 4.2. From this, the practical feasibility of collecting sensor and software API data from Android devices is examined in Section 4.2.1. Finally, investigations into correlations between features extracted from this data and different interruptibility labels is examined in Section 4.3, resulting in preliminary suggestions that the labels produced by the model are likely to be predictable.

4.1 Investigating sampling stability and usefulness

The primary rationale for this analysis is to determine the feasibility of using machine sources (rather than explicit human annotation) to passively implement the DOIG model on Android devices, in terms of helping to produce labels of interruptibility (as shown in Chapter 3), and in gathering potentially influential contextual data for building predictive models. Towards this, two primary limitations in the literature are used to frame the analysis. Firstly, machines sources have frequently been used across the literature, however this analysis often stops short of exploring the practical feasibility of retrieving the data, in favour of moving forward towards prediction. This creates the following common limitation:

L4.1 The ability for data to be made available when asked for from machine data sources is often assumed.

As the successful operation of the DOIG model is tied to this being a reasonable assumption, the sampling stability of the ImprompDo application is firstly explored, along with strategies to mitigate issues that arise.

From this, before pursuing the predictability of the interruptibility labels produced by the DOIG model using this dataset, the second part of this chapter explores whether this would be worthwhile to pursue. This is achieved through determining whether contextual features extracted from the dataset are correlated to the labels (e.g., whether the battery charging state correlates to being reachable/not reachable). Empirical studies have previously commented on the potential predictive power of different contextual values, however they commonly have the following limitation:

L4.2 Analysis of correlating contextual factors is performed from the perspective of a single definition of interruptibility (e.g. just receptivity), similarly to limitation L3.3 in Chapter 3.

This results in conclusions that are confined to this definition, creating challenges in determining the wider applicability of the results (i.e., for other definitions, such as

reachability). Therefore this analysis is undertaken with multiple labels representing different definitions (reachability, engageability, and receptivity).

4.2 Passively sampling data on Android devices with ImprompDo

In the ImprompDo case study application, data sampling takes place alongside the delivery of notifications (as discussed in Chapter 3, Section 3.3.3), using a number of data sources (listed in Chapter 3, Section 3.3.4.2), including sensors (e.g., from the on-board accelerometer) and software APIs (e.g., the current volume state). Sampling starts 5 seconds before the notification is delivered, until either 30 seconds have elapsed or the notification is removed by the user (visualised in Chapter 3, Figure 3.6), enabling analysis between contextual data just before the interruption and the user's response behaviour.

The sampling process consists of creating sets of data vectors, with each containing a reading from each data source (as visualised in Chapter 3, Figure 3.7). As sensor readings are delivered by Android asynchronously, a time-window is used to listen for data. It is closed when either at least one data reading has been collected from all data sources, or a timeout of 2 seconds has elapsed. If this results in multiple data samples being taken for a given data source while waiting for the others, only the final reading is retained so that the time in-between readings across the data sources is minimised. If no data readings were available after 2 seconds, the reading for that data source is set to null. A new sampling window is then opened immediately, subject to device speed and system stability.

4.2.1 Investigating data availability and sampling regularity

The frequency statistics of the ImprompDo dataset, discussed in Chapter 3, Section 3.3.5, sheds some light on the existence of sampling differences across users. However, the extent of this issue requires further investigation for individual data sources, in order to reflect on the overall suitability. The motivation for this analysis is to examine whether the asynchronous nature of Android sensors results in gaps in the data vectors, as a result of data not arriving within the 2-second window. This section examines this by exploring a) the availability of sensor data when requested, i.e., whether sensor readings exist or not in the data vectors, and b) the reliability of the 2-second window approach, by exploring the interval between the start times.

4.2.1.1 Data availability

This section examines how reliable hardware sensors are at providing readings within the 2-second sampling window, as while a sampling frequency can be suggested by an application, this is neither guaranteed nor predictable. It should be noted that 1 user did not have any raw data vectors collected with their notifications, so was excluded from this analysis.

For each data source sampled, Figure 4.1 shows the number of users where data from each data source was either: *always present* in each data vector; *sometimes present*; or *never present*. The results show that no hardware sensors are uniformly reliable, but that all software API sources were (unsurprisingly) reliable. This suggests that using machine sources such as screen and lock state for capturing the decision behaviour required for labelling is reliable, however the contextual data overall is less reliable. It should also be noted that the *never present* frequencies include devices that may not have that particular sensor.

The large proportion of users that sometimes have sensor data warrants further investigation to see whether these cases can be isolated by operating system version or

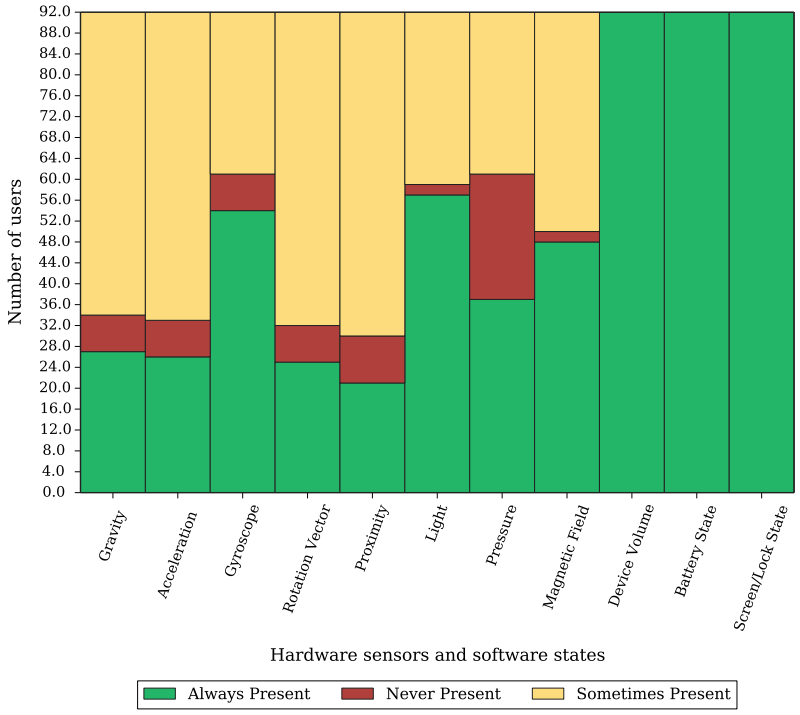


Figure 4.1: The reliability of sensor readings within 2 seconds.

device. However, across users using the same version of Android or same device, the results showed inconsistency across users, with no version or device being consistent in always containing sensor data (shown in Appendix Tables A.9-A.12). Unfortunately, this suggests that simply restricting applications to a given version of Android or device model will not provide guaranteed sampling consistency for sensor data across users. Therefore, whether this issue can be mitigated by considering multiple sequential data vectors together (e.g., all vectors in the 5 seconds before a notification is delivered) is explored in Section 4.2.1.2.

4.2.1.2 Sampling regularity

Data readings need to take place at a reliable rate as environmental context can change and observations of user interactions with the device need to be made quickly. Firstly, the standard deviation in the number of sensor readings taken per notification was 7.6 (as shown in Chapter 3, Section 3.3.5). This suggests that there is some variation in the

M	SD	Min	Max
1000.9	410.5	1.0	33386.0

Table 4.1: Frequency statistics of the (ms) intervals between the start of the sampling time-windows.

number of samples taken, but does not suggest the regularity. Additionally this may be due to the listening window readings being up to 2 seconds, rather than a fixed time, and because sampling stops early if the notification was removed in some way.

The time intervals in-between sampling windows being opened is shown in Table 4.1. The results show that a sampling window typically occurs just over once per second, with an average fluctuation of less than half a second; considerably lower then the 2-second time-out for readings to occur, suggesting that the majority of data vectors contain a complete set of readings. However, there are outlier notifications, which produced a maximum interval of 33.4 seconds. In this case, a single notification had only 2 widely spaced raw data vectors taken over the 35-second period. This shows that while samples are typically taken reliably, there are some minor stability concerns in scheduling sampling to occur.

Overall, this suggests that requests to sample are typically reliable, but the result may or may not contain a reading from every source. This leads to question of whether this reliability can be exploited to mitigate the sensor data availability problems observed (Section 4.2.1.1). If we assume that the context being captured by these sensors is unlikely to change considerably in the 5 seconds before a notification is delivered, then the context before a notification can be built from at least 1 reading across all sampling windows that occurred in that short period. To investigate this, the mean value was taken across each data source to create a new single data vector.

The results showed that 8054/11346 (71%) of notifications now had a value for each data source (an increase of 36.9%). The remaining 3292 (29%) still had some degree of missing data, however, a proportion of this may be due the device not having a particular hardware sensor, with 1525/3292 (46.3%) of these cases consistently missing data from

only 1 sensor. This suggests that taking the mean value can mitigate the reliability problem, however at the loss of temporal granularity.

Summary: Sensors are reliable, if available

From these results, the primary findings on the feasibility of passive sampling can be described as:

- Overall, the results show that passive data collection is viable to support labelling using the DOIG model;
- However data availability issues with the use of on-board sensors presents challenges in retrieving contextual data on demand;
- It is not feasible to ensure reliable sensor data collection through hardware selection, however variability can be mitigated by broadening the temporal granularity of the samples taken and taking mean readings.

Collectively, these findings show support for the use of passive sampling for both contextual data collection and in helping to capture the decision process in response to notifications. However, given the time-sensitive requirements of readings relevant to capturing the response process, such as the screen and lock state readings, it is suggested that for future applications, this data should be sampled in a separate background process to the hardware sensor readings. This will help to avoid crucial information being delayed, such as whether the device is in use. Going forward, the data collected is examined further to determine the likelihood that the contextual data can predict the labels produced by the DOIG model.

4.3 Correlations between contextual data and DOIG labels

To investigate the use of passive sampling for a broad range of notifications as possible, this section focuses on exploring correlations between contextual data and the different response behaviour that can occur up until a typical Android notification is consumed. Further behaviour could occur after this point, such as whether a to-do list item is completed or an email is replied to, however, as these will depend on the individual application, this is not included in this analysis.

The ImprompDo dataset is firstly transformed into a set of *instances*, each representing a notification, containing a *feature vector* and the binary interruptibility labels produced by the DOIG model (Figure 3.2, Chapter 3): reachability, engageability and receptivity. The feature extraction process to convert the contextual data vectors into a feature vector is discussed in Section 4.3.1. With analysis of correlations between individual features before the interruption and reachability, engageability and receptivity labels is explored in Section 4.3.2.

4.3.1 Extracting features from the raw data traces

Given the variable number of data readings, variable amount of missing data and potential for noise in the raw data readings, the sets of raw data vectors need to be converted into a more useful representation. To achieve this, a two-step process was used:

Flattening step: Following the conclusions of Section 4.2.1.2, the mean value for each data source is taken across the set of data vectors that span the 5 seconds before the notification was delivered, creating a single set of values that represent the average readings.

Transformation step: These averaged values are transformed into discrete values as feature variables (features). For example, transforming the proximity sensor value into Screen_Covered: True/False. The list of features and potential values is shown in Table 4.2, with the transformation formulas listed in Appendix Table A.13.

The result of the two-step feature extraction process is a vector of features for each notification, for example:

{***Accelerating***: False, ***Ambient_Light***: Dark, ***Screen_Covered***: True, ***Volume_State***: Silent, ***Orientation***: Flat, ***Charging_State***: True, ***Time_of_Day***: Night, ***Day_of_the_Week***: Sat}

The features chosen were those that were deemed to represent key aspects of the current state of the device which could logically be hypothesised to be relevant to interruptibility, taking into account previous similar studies (explored in Chapter 2 and in [121]). As a result, some less relevant data sources (e.g., atmospheric pressure) were not used.

Statistical analysis procedure

Statistical tests are used to analyse correlations between individual feature variables and the DOIG model labels. The term *correlation* is used here to refer to whether the differences in the underlying distributions are statistically significant (i.e., between cases where users where reachable or not reachable), independent of the type of statistical test used.

Firstly, Kolmogorov-Smirnov tests were performed which determined that the distributions were non-normal, and therefore non-parametric equivalents to *t*-tests were used. For variables with 2 possible values in the distribution, Mann-Whitney U tests are applied. To reduce the likelihood of Type I statistical errors, Kruskal-Wallis one-way ANOVA with Bonferroni-corrected pairwise post-hoc tests were used for variables with more than 2 values. Statistical significance of the results are examined first, by

comparing the p-value against a $\alpha = .05$ threshold. These results are tabularised to provide an overview of which variables may be useful and which are not. For key results with statistical significance, test statistics are reported (χ^2(degrees of freedom, sample size) for Kruskal-Wallis one-way ANOVA tests, U and z (z-score, or standard score) for Mann-Whitney U tests) along with effect sizes (r) [26].

4.3.2 Correlations between features and DOIG model labels

Table 4.2 shows which contextual variables are correlated with which labels. Initial inspection reveals that some features are only correlated for some labels and these differences also extend between whether the device is in use or not. This summary alone suggests that different contextual data may be (consciously or subconsciously) relevant to the user's decision behaviour in their response, providing an initial indicator that prediction is worthwhile to pursue.

While correlation does not imply causation, closer inspection of individual variables reveals logically plausible effects. For example, the "Volume State" is significant for reachability when not in use ($\chi^2(2, 7737) = 202.209, p < .001$). This is expected, as this is a common mechanism to control physical interruptions from the device. Pairwise post-hoc tests reflect this, with statistical significance shown between silent and audible ($p < .001, r = -.170$), and silent and vibrate ($p < .001, r = -.242$) pairs, with medium effect sizes for both. Furthermore, the difference between vibrate and audible is also significant, but with a much smaller effect size ($p < .003, r = .040$). Interestingly, despite the design of the vibration setting intending to lessen the impact of an interruption, which is arguably closer to silent mode, in practice the effect size shows that user behaviour towards interruptions through vibration patterns is more similar to audible tones.

A further example is "Orientation" being significant when the device is in use for receptivity ($\chi^2(2, 2141) = 20.924, p < .001$). Pairwise post-hoc tests revealed the

	Not in use			In use
Feature variables	**Rc**	**Eg**	**Rv**	**Rv**
Accelerating* (False, True)	.186	.458	.072	**.000**
Ambient Light** (Dark, Dim, Light, Bright)	**.000**	**.039**	**.000**	**.000**
Screen Covered* (False, True)	**.000**	.187	**.000**	**.005**
Volume State** (Silent, Vibrate, Audible)	**.000**	**.009**	**.011**	**.000**
Orientation** (Flat, Upright, Other)	**.000**	.098	**.000**	**.000**
Charging State* (False, True)	**.000**	**.001**	.145	.177
Time of Day** (Morning, Afternoon, Evening, Night)	**.002**	.125	.936	**.000**
Day of the Week**	.509	.794	.100	**.000**
Number of cases (n)	7737	1798	1469	2322

Table 4.2: P-values indicating significance of each feature before the interruption and the outcome of each decision [122, 124]. Bold values show significance using $p < .05$. * Mann-Whitney U Test ** Kruskal-Wallis 1-way ANOVA. Rc=Receptivity, Eg=Engageability, Rv=Receptivity. .000 values refer to strong significance $< .001$. Table from [122, 124].

significance pairs to be between groups where the device was flat and those when upright ($p < .001, r = -.087$), and between other orientations and upright ($p < .001, r = .145$). It could be assumed that when a device is being used for active interaction, it will likely be relatively upright in the user's hand, whereas other positions (such as when unlocked flat on a table) may produce false positives. This is reflected in the p-values and effect sizes of these pairwise comparisons, and further supported by the difference between flat and other orientation groups not being significant. Beyond the applicability towards prediction, this suggests that a multi-modal approach, using measures in addition to the screen state, could be used to determine whether the device is in use in the future.

Other variables have more unexpected outcomes, for example, whether the device is "Accelerating" is significant when the device is in use ($U = 482,548, p < .001, z = 3.788, r = .082$) but not when not in use. This is unexpected as if the device is already in use it could be assumed that the user would be more attentive to notifications, regardless of whether they were accelerating. However, this could be explained by the level of focus the user has on an important task when the device is in use. The same argument concerning the current task being performed could also apply to other variables when the device is in use. For example "Screen Covered" ($U = 147,285, p < .005, z = -2.815, r = -0.063$), "Ambient Light" ($\chi^2(2, 2138) = 20.463, p < .001$), and "Volume State" ($\chi^2(2, 2322) = 25.316, p < .001$) are all statistically significant, however for only a small subset of pairs within these (e.g., Dark and Dim ($p < .001, r = -.1$), and Dark and Light ($p < .004, r = -.092$) for "Ambient Light"). Across these the effect size was low, suggesting that the significance may due to cases where the device was not in active use, but the screen remained on.

The significance of temporal variables also differs across the use-states. Firstly, the "Time of Day" was significant for receptivity when the device is in use ($\chi^2(3, 2322) = 27.008, p < .001$), with pairwise-tests revealing the difference between Morning and the other groups having the highest effect sizes (Afternoon ($p < .004, r = -.083$), Evening ($p < .028, r = -.085$), Night ($p < .001, r = -.154$)). This suggests that when the device is in use in the morning, users are typically focused on their current task and are less susceptible to interruptions from notifications. Finally, the "Day of the Week" is also significant for receptivity when the device is in use ($\chi^2(6, 2322) = 24.191, p < .001$), but with only a few significant pairs and low effect sizes, suggesting that it may not have a considerable impact.

Summary: Different correlating factors for different DOIG labels

From the results above, the primary findings from the statistical analyses can be described as:

- Different contextual features before the interruption correlate with different DOIG labels;
- This also extends to differences between whether the device is in use or not at the time the notification occurred.

Collectively, these findings suggest that reachability, engageability, and receptivity are likely to be predictable to some extent from the contextual data collected before the notification is delivered. However, the medium to low affect sizes suggest the likely predictive power of individual features may be small, therefore a multi-modal approach is used going forward in the creation of predictive models. These findings can be further supported by similar findings in the contexts after the interruption (reported in [124]), suggesting that different sets of contexts may also influence response behaviour after a user is at least reachable and begins a response.

4.4 Conclusions

The ability to passively sample relevant contextual data to interruptibility is beneficial for real-world applications, as this removes the reliance and burden upon the user to provide this information. Using the ImprompDo dataset introduced in Chapter 3, this chapter has examined the reliability of sampling hardware and software data sources on Android devices in order to support the implementation of the DOIG model, and whether features extracted from these samples may be useful for predicting different interruptibility labels produced by the DOIG model for typical Android notifications.

Firstly, the reliability of passively sampling data on a variety of Android devices and versions was explored (addressing limitation L4.1). The results show that the data sources discussed in Chapter 3 for labelling interruption behaviour with the DOIG model (e.g., screen and lock state) are reliable. However the results highlight the potential for unreliability in whether sensor data is available within a 2 second period,

which impacts on the ability to create contextual features that can be used as the basis for prediction. While it is unlikely that this can be eliminated through being selective with Android versions and hardware, using mean values across potentially incomplete sets of readings (e.g., over a 5s period) reduces this issue considerably. Secondly, the results show that different contextual features are statistically correlated to different interruptibility labels produced by the DOIG model (addressing limitation L4.2).

The results of these two analyses supplement contribution C3 of Chapter 3, through showing the practical feasibility of implementing the DOIG model. Going forward, the analysis suggests that it may be worthwhile to build predictive models of reachability, engageability, and receptivity with machine learning, using the features extracted from the passively sampled data (Table 4.2). This forms the focus of Chapter 5, in exploring the relative performance differences between models built for each label, along with different machine learning environments and evaluation criteria.

Chapter 5

Predicting decision making behaviour

Studies seeking to predict interruptibility using machine learning typically adopt a single definition of interruptibility (e.g., whether the user is at least reachable, or receptive to the content). As discussed in Chapter 2, this creates a tight coupling between experiment scenarios, datasets, and conclusions of individual studies, limiting the ability to determine the broader applicability. The purpose of this chapter is to explore the predictive performance of models built for different definitions, by creating independent models for the reachability, engageability, and receptivity DOIG labels, with the relative performance differences then examined.

Additionally, degrees of freedom exist in methods for training and testing models, and in evaluating their performance (as discussed in Chapter 2), with studies typically performing limited direct comparison of different strategies. Therefore, this analysis also compares the performance differences of different machine learning strategies for training and evaluation.

Firstly, further rationale behind exploring a variety of training and evaluation strategies for the different interruptibility labels is discussed in Section 5.1. The scope and focus of the analysis is discussed in Section 5.1.1, with initial observations of the performances of a typical user using different classifier algorithms discussed in Section 5.2. A comparison of training from the aggregated data of other users or personal data is then examined in Section 5.3. Finally, the performance of training in an online environment (where the models are retrained at the end of each day with new data) is explored in

Section 5.4.

5.1 Examining machine learning strategies

In the context of this thesis, where interruptibility is considered a binary classification problem, a *predictive model* is a model that takes a feature vector representing the current context as input (e.g., as discussed in Chapter 4, Section 4.3.1), and produces an output that states whether an individual is likely to be interruptible or not in that context. To achieve this, the model is trained from a set of data instances where interruptibility is known, with each instance containing a feature vector and a (class) label (i.e., reachable or not reachable, etc.). The model can then be tested using unseen data, where the accuracy of the outputted predictions can be evaluated.

Several limitations in the existing literature are used to frame this analysis. Firstly, in exploring the relative performance differences of the predictive models built for the reachability, engageability, and receptivity DOIG labels produced, the following common limitation in the literature is improved upon:

L5.1 Prediction is typically performed by individual studies from the perspective of a single definition of interruptibility (similarly to limitation L3.3 in Chapter 3 and limitation L4.2 in Chapter 4).

Secondly, machine learning [39] has been a common means of producing these predictive models. However, for applications wishing to integrate intelligent interruption systems into their applications, the choice of learning strategies within machine learning is arguably a multi-objective problem. As with the sampling of relevant data sources (discussed in Chapter 4), there is a need to consider both utility and practical feasibility. Discussed further in Chapter 2, a common thread across the literature is the limited direct comparison of different methods for training and evaluating models, which forms a secondary rationale for this analysis, with a particular focus on the following limitations:

L5.2 While previous studies have included some comparisons of training from aggregated data or personalised data (e.g., [92]), and between online and offline learning (e.g., [76]), these are typically not considered together.

L5.3 Evaluation is commonly focused on maximising predictive performance (e.g., [99, 97]) rather than considering the diversity in different priorities for different applications (e.g., minimising false-positives or false-negatives), and the practicalities of a real-world implementation (e.g., [128]).

Going forward, as well as examining the relative performance differences across the different labels produced using the DOIG model (improving upon L5.1), this is achieved using different machine learning training methods, environments, and evaluation criteria, in order to suggest additional considerations for future studies and application design.

5.1.1 Machine learning approach

In this chapter, the extent in which reachability, engageability, and receptivity are predictable is explored using multiple contextual features (multi-modal). The feature vectors and labels used for prediction are the same as those created for the analysis in Chapter 4, Section 4.3. The primary aim of the analysis is to explore the relative differences in the predictive performance across the DOIG labels for different machine learning methods, however, where appropriate the scope is refined to prune the worse performing solutions (e.g., classifier choice).

The methods used for each component in the predictive modelling are outlined as follows.

5.1.1.1 Pre-processing

Analysis of the dataset reveals that the label (class) distribution is imbalanced since the majority of notifications are null-responses, i.e., users were often unreachable (as

discussed in Chapter 3, Section 3.4). Without pre-processing, this could lead to false reporting in model performance, for example, if a model always predicts a single class and 80% of the data is labelled with that class, then the model is trivially correct 80% of the time, but practically useless. To prevent this, random-under-sampling (RUS) [41] was used to produce 100 evenly distributed training datasets for each model.

5.1.1.2 Classifier choice

The choice of classifier algorithms to train the models will be examined as part of the initial analysis of predictive performance for a typical user (Section 5.2). This is due to the wide variety of success that has been seen across previous works in using different classifiers (as discussed in Chapter 2). This analysis will also explore the suitability of creating either independent binary classification models for each label and use state (e.g., at least reachable/not, engageable/not, receptive/not) or mutli-class models (e.g., the user is either reachable/engageable/receptive/not at all). The results from this analysis then prune the analysis space for exploring the performance of individual users.

5.1.1.3 Training and testing models

For each DOIG label, three approaches are used for splitting the data where relevant (visualised in Figure 5.1): *Aggregate Trained and Aggregate Tested* (AT-AT), where training and testing data is split from the same aggregated dataset from all users; *Aggregate Trained and Personally Tested* (AT-PT), where for each user, the models are trained from the data of all other users, and tested only against that selected user's data; and *Personally Trained and Personally Tested* (PT-PT), where training and testing data are both from the data of each individual user. However, as the level of participation from individual users varied, some users may not have data for all classes (such as if no notifications occurred when the device was in use), these users are excluded where relevant.

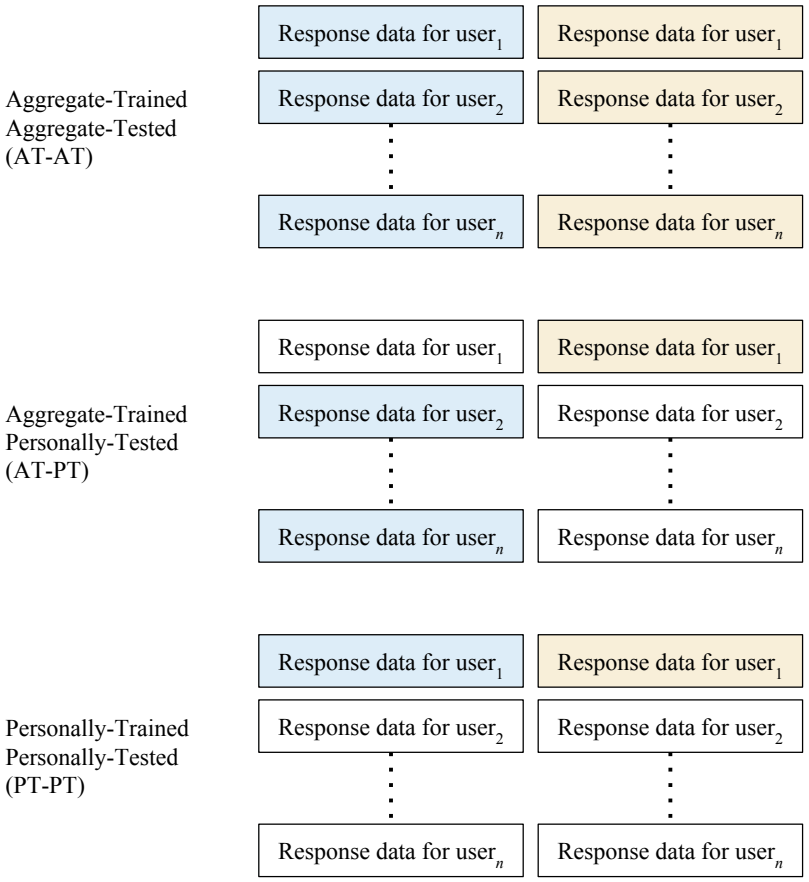


Figure 5.1: Visualisation of the training and testing approaches (as described in Section 5.1.1.3). Personally tested approaches are visualised using an example user ($user_1$). Additionally, each data point cannot be in both training and testing datasets. = the training data used and = the testing data used.

For testing, 10-fold cross-validation was used for the AT-AT and PT-PT models. As AT-PT models use separate training and testing datasets, cross-validation would not be suitable. However, this issue is mitigated as the above analysis is performed on 100 RUS datasets (as defined in Section 5.1.1.1).

5.1.1.4 Evaluating model performance

Different applications may have different priorities on predictive performance (e.g., minimising missed opportunities to interrupt (false-negatives), or minimising ineffective interruptions (false-positives)). To consider this, models are evaluated using different

	Actually True (e.g., reachable)	Actually False (e.g., not reachable)	
Predicted True (e.g., reachable)	True Positive (TP)	False Positives (FP)	PPV = TP/(TP+FP)
Predicted False (e.g., not reachable)	False Negative (FN)	True Negatives (TN)	NPV = TN/(FN+TN)
	Sensitivity = TP/(TP+FN)	Specificity = (TN/FP+TN)	

Figure 5.2: Visualisation of the PPV, NPV, sensitivity and specificity metrics used. Weighted precision is the average between PPV and NPV performance, and weighted recall refers to the average between sensitivity and specificity performance.

standardised metrics, which are derived from the confusion matrix produced in the evaluation (visualised in Figure 5.2):

PPV : The *positive predictive value* (PPV) is a precision metric that refers to the proportion of cases in the testing dataset that were correctly classified as reachable, engageable, or receptive.

NPV : The *negative predictive value* (NPV) is a precision metrics that refers to the proportion of cases in the testing dataset that were correctly classified as not reachable, not engageable, or not receptive.

Sensitivity : The *sensitivity* recall metric refers to the proportion of positive cases (e.g., reachable) that were correctly identified against the total number of cases that exist in the testing dataset. This metric can be paired with PPV.

Specificity : The *specificity* recall metric refers to the proportion of negative cases (e.g., not reachable) that were correctly identified against the total number of cases that exist in the testing dataset. This metric can be paired with NPV.

Weighted Precision : The *weighted precision* value refers to the average of the PPV and NPV metrics, weighted by the number of cases of each class if unbalanced.

Weighted Recall : The *weighted recall* value refers to the average of the sensitivity and specificity metrics, weighted by the number of cases of each class if unbalanced.

Applications that wish to minimise missed opportunities to interrupt (e.g., a game) may focus on PPV and sensitivity in correctly isolating interruptible moments. Whereas applications wishing to avoid interrupting during ineffective moments (e.g., a productivity tool or hydration application) may focus on NPV and specificity in correctly isolating non-interruptible moments. However, applications may also want to perform reasonably well at both priorities and consider all metrics together, with weighted precision and recall metrics offering a summary of these.

5.2 Performance of a typical user (AT-AT)

Firstly, the relative differences between predictive models of reachability, engageability and receptivity are explored for a typical user, using the aggregated dataset from all users (AT-AT) for in training and testing the models.

5.2.1 Classifier performance

Table 5.1 examines the predictive performance across 7 common classifiers used in previous interruptibility works (as discussed in Chapter 2 and in [121]). Firstly, the results show poor performance for the multi-label model in comparison to the individual binary-class models dedicated to predicting a single definition of interruptibility. This suggests that predicting the exact response behaviour is difficult to achieve with the feature variables in this dataset. However as individual applications are likely to want to predict if the user will respond *at least* to the degree that they define the interruption to be successful, the binary class models are suitable. For example, a hydration reminder application may require that a user be at least reachable, with any engageable or receptive behaviour seen as an added positive. From this, only the performances of the individual binary-class models are reported going forward.

While the mean performances of the binary-class models are not very high, the perform-

		Not in use				In use
Classifier	**Metric**	**Rc**	**Eg**	**Rv**	**MC**	**Rv**
AdaBoostM1	Precision	0.6045	**0.6064**	**0.6375**	0.2522	0.5927
	Recall	**0.6026**	**0.6045**	**0.6369**	0.2976	0.5923
BayesNet	Precision	0.5936	0.5873	0.5955	0.2532	0.4997
	Recall	0.5889	0.5831	0.5917	0.2870	0.4996
J48	Precision	**0.6065**	0.5986	0.6316	0.3376	**0.6010**
	Recall	0.6023	0.5957	0.6294	**0.3393**	**0.6002**
Logistic	Precision	0.5719	0.5791	0.6118	0.3217	0.5881
	Recall	0.5718	0.5790	0.6117	0.3272	0.5879
NaiveBayes	Precision	0.5715	0.5816	0.6195	**0.3408**	0.5889
	Recall	0.5702	0.5801	0.6174	0.3372	0.5872
RandomForest	Precision	0.5788	0.5769	0.6250	0.3277	0.5939
	Recall	0.5787	0.5768	0.6246	0.3283	0.5938
SMO	Precision	0.5664	0.5779	0.6036	0.3233	0.5941
	Recall	0.5659	0.5761	0.6017	0.3248	0.5928

Table 5.1: Classifier performance of the aggregated dataset (AT-AT) using weighted precision and recall metrics and different measures of interruption success (reachability, engageability, receptivity). Classifier names are those provided by Weka [39]. MC=the multi-class model. Bold values indicate the highest value across classifiers. Table from [122].

ance is similar to other recent studies (e.g., [92, 118, 116]), including those inferring interruptibility from content data over context (e.g., [76]) and other attentive states (e.g., boredom [98]). Given that participation of individual users varied and that humans can have varying device and interruption habits, this performance (of around 60%) is neither unexpected nor unreasonable.

Crucially, the results show that partial response behaviour (i.e., reachability) can be successfully predicted to a similar degree to complete responses (i.e., receptivity). This is beneficial for real-world implementation as the same classifier can be used for each use-state without a detrimental affect on performance, improving viability as the mobile devices can have limited resources. Overall, these results supplement the conclusions of Chapter 3, in finding that as well as the decision-level labels being worthwhile to

	Not in use			In use
Metric	**Rc**	**Eg**	**Rv**	**Rv**
PPV	0.586	0.582	0.617	0.594
Sensitivity	0.699	0.677	0.684	0.610
NPV	0.627	0.614	0.646	0.600
Specificity	0.505	0.514	0.576	0.582

Table 5.2: Classifier performance (J48) of the aggregated dataset (AT-AT) using unweighted metrics and different measures of notification success (reachability, engageability, receptivity). Table from [124].

extract, they are also reasonably predictable.

5.2.1.1 Reducing classifier choice to decision-trees

As the results show minimal performance differences across various Bayesian, tree, and function based classifiers, the J48 tree (C4.5) classifier is used in further analyses as it offers several advantages beyond performance. Firstly, it has been used successfully in similar studies (e.g., [99, 88, 31, 30, 86, 24]) and the outputs can be easily interpreted. Secondly, models created for when the device is in use and not in use can be merged together simply by adding a top-level node (i.e., in use? $\{true, false\}$), rather than managing multiple models. Finally, storage and traversal of the tree is computationally inexpensive, an important factor for mobile devices with limited resources.

From this, closer inspection of the metrics reveals further patterns relevant to applications that may prioritise minimising either false-positives or false-negatives, rather than the performance of both. Table 5.2 shows of the performance of the J48 algorithm using the finer grained precision and recall metrics. Overall, the models offer slightly higher precision in avoiding untimely interruptions (NPV) than finding opportunities (PPV), suggesting that correctly identifying interruptible moments is more challenging, at least for one-size-fits-all models from aggregated data; however the reverse is true for identifying all of these cases (specificity and sensitivity metrics).

Secondly, for cases where the device is not in use, performance typically increases for

the measures of success that correspond to later points in the response. This suggests that context, as well as content [76], may be a factor that affects receptivity towards the interruption. Another unexpected result is the worse performance for receptivity when the device is in use. This could be explained by the unknown level of engagement that the user had with their device at that time, with task engagement previously been shown to be an additional influential feature (e.g., [88, 50, 52]).

Summary: DOIG labels are similarly predictable for a typical user

The results provide an indication of the expected performance of a one-size-fits-all model built from the aggregated data of all users, producing the following primary findings:

- Predicting that a user is at least reachable, engageable, or receptive (binary-class models) yields higher performance than predicting that the user is only reachable, engageable, or receptive (multi-class models).
- These (binary-class) models produced similar predictive performance across all of the DOIG labels and use-states, suggesting that adopting a particular definition of interruptibility will not result in considerable performance gain or loss.

However, as individual users in the ImprompDo dataset participated for different periods of time, experienced different contexts, and likely have their own interruption habits, this model may not be representative of every user.

5.3 Comparing aggregate and personalised models (AT-PT and PT-PT)

This section explores whether the performance of the typical user model (AT-AT) is representative of the real world; where user participation would be self-selecting and

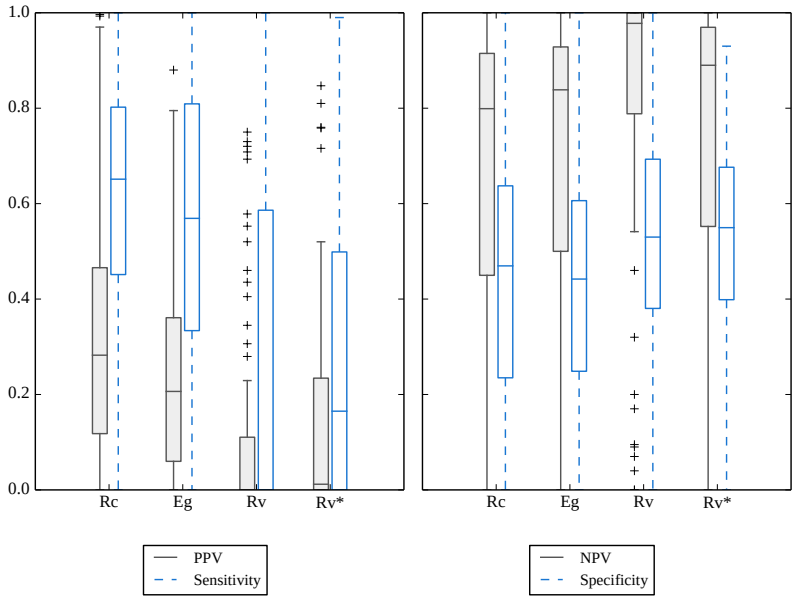


Figure 5.3: User performance for models trained from aggregate data (AT-PT). Rv* refers to receptivity when the device is in use. Y-axis represents prediction performance. Figure from [124].

level of engagement would vary. To investigate the potential effects of this, separate models are built to test each user's data individually. As well as testing at an individual level, a hypothetical application will have to decide what data to train from. While personal training data of interruptibility has previously been successful (e.g., [119, 92]), this data will likely not be available initially (i.e., when a user first installs an application). The analysis is therefore split between training the models using the data of other users (AT-PT) and from the individual's own data (PT-PT).

To examine the results, box-plots are used to visualise the distribution of users (likewise to [23]), offering a wider view of the typical and outlier performances across users, in comparison to the use of standard deviations. Outliers are shown using "+" ticks, with the median performance shown with a horizontal line in the box and the upper and lower quartiles displayed through the position of the top and bottom of the box respectively.

5.3.1 Training from aggregate data (AT-PT)

The first set of models were built where, for each user, the training data consists of the aggregated data of all other users, with the selected user's data used as testing data (visualised in Figure 5.1). This is representative of the performance of new users installing the application where a set of training data from other users already exists.

Figure 5.3 shows the distribution in performance across all individual users (reachability (N=92), engageability (N=92), receptivity (not in use: N=92, in use: N=83)) and Figure 5.4a shows results only for more active users (i.e., those with >10 notifications in the dataset, reachability (N=63), engageability (N=63), receptivity (not in use: N=63, in use: N=41)). The pruned dataset is used for the remaining analysis, as while the effect on the overall distribution and medians is low, this removes outlier performances at the lower and higher quartiles.

The results show that models trained from aggregated data perform very well at correctly predicting that the user is not reachable, willing to engage, or receptive (NPV) for most users (seen in the top right of Figure 5.4a), with receptivity also having much smaller variance. However, these models perform worse at correctly predicting opportune moments (PPV) for most users, across all measures of success (seen in the bottom left of Figure 5.4a).

For the recall metrics (sensitivity and specificity), the median performances are close to the typical user model (Table 5.2) for reachability and engageability (and similar studies, e.g., [92]), with the exception of sensitivity for receptivity (seen in the bottom left of Figure 5.4a); however the variance across users is generally high. Overall, this suggests that individual users are likely to be interruptible in very different contexts, whereas users are not interruptible in similar contexts, which is logical considering cases such as during driving.

In comparison with the one-size-fits-all typical user (AT-AT, Section 5.2), the results highlight the diversity in interruption habits across users, suggesting that the typical

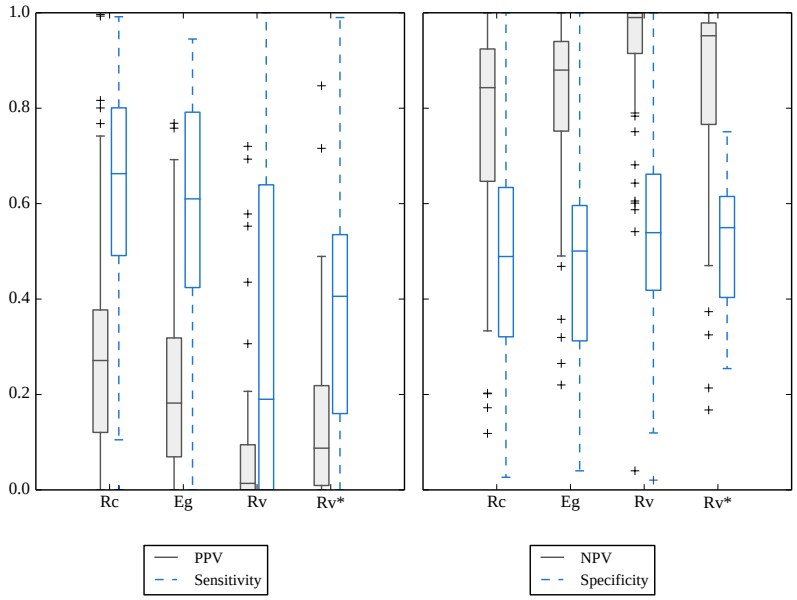
user model predominantly either underestimates or overestimates per-user performance. The suitability of training from an aggregated dataset is therefore largely dependent on an applications desired evaluation priorities.

From the perspective of an application with a higher priority in correctly isolating moments where the user is not interruptible, models trained from aggregated data perform reasonably well (shown on the right hand side of Figure 5.4a), with small differences between models for different the DOIG models and use states. For applications with a higher priority in avoiding missed opportunities to interruption (or wishing to perform similarly at both), the low PPV and sensitivity performance overall (shown on the left side of Figure 5.4a) suggests that these models may not be as suitable, particularly if receptivity is used as the interruptibility label. However, being able to correctly predict the inverse of this, that the user is not reachable, could still be useful.

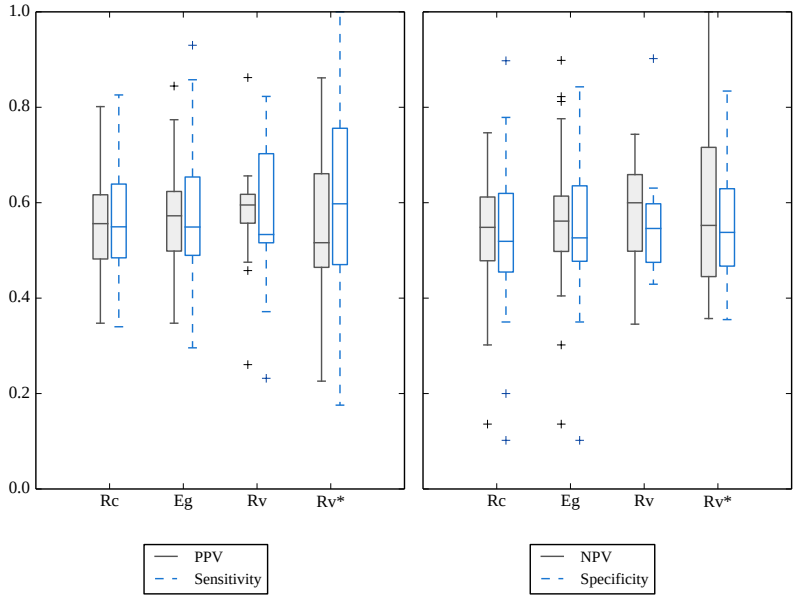
5.3.2 Training from personal data (PT-PT)

The second set of models were trained and tested only using each user's individual data (for those users with enough data). To avoid under or over representing performance, users that produced models for only a single class (i.e., they were always receptive or not) were also removed. Figure 5.5 shows the performance of all users (reachability (N=75), engageability (N=73), receptivity (not in use: N=43, in use: N=45)) and Figure 5.4b shows only those with >10 notifications (reachability (N=43), engageability (N=44), receptivity (not in use: N=17, in use: N=16).). In this case, the pruning operation reduces the variance across users considerably across all labels and use states. As users naturally experienced various contexts, this could be explained by some contexts not being experienced frequently enough to appear in the training data.

For the pruned dataset, the results show that the use of personalised models typically outperforms the aggregately trained models (AT-PT, Figure 5.4a) if the evaluation priority is to isolate opportune moments to interrupt (considering PPV and sensitivity



(a) AT-PT,for users with >10 notifications.



(b) PT-PT, for users with >10 notifications.

Figure 5.4: Predictive performance of AT-PT and PT-PT for more active users. Rv* refers to receptivity when the device is in use. Y-axes represent prediction performance. Figures from [124].

together on the left of Figure 5.4b). However, the models perform worse than the aggregate trained models in avoiding ineffective interruptions (considering NPV and specificity together on the right of Figure 5.4b).

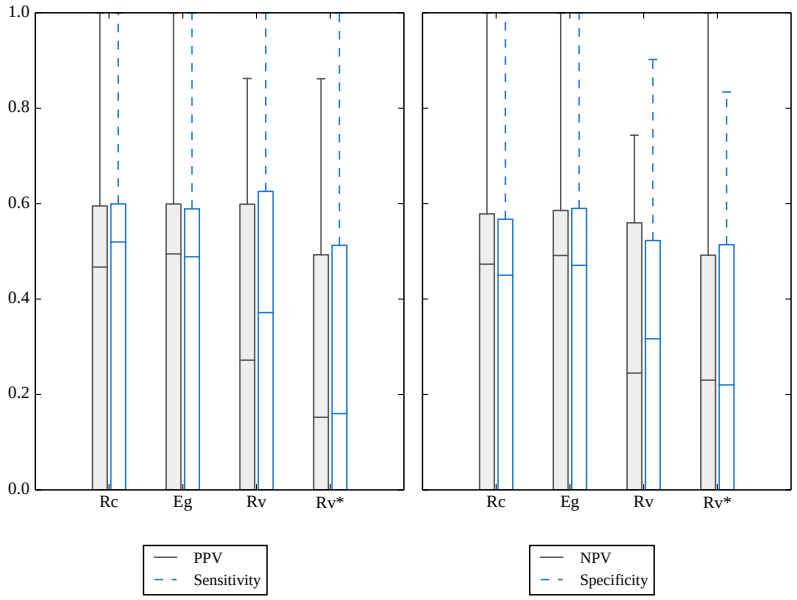


Figure 5.5: User performance for models trained from personalised data (PT-PT). Rv* refers to receptivity when the device is in use. Figure from [124].

This suggests that for applications with a greater priority in avoiding missed opportunities to interrupt (such as the ImprompDo application), or for those wishing to perform reasonably well at both, personalised models are better suited than those aggregately trained. This reflects previous conclusions [92, 76], but also shows that this extends beyond a single measure of success and evaluation metrics.

Closer inspection of the performances shown in Figure 5.4b reveals some slight differences in the distributions of reachability and engageability as compared to receptivity, but not to the extent of AT-PT. When the device is not in use, the variance in user performance is the lowest across all metrics, yet when the device is in use the variance is the largest across all metrics. Despite this, the low variance across users suggests that in comparison to AT-PT, personalised models may be more suitable overall for applications where performance across users needs to be somewhat consistent. However these differences may be due to the fewer number of users for these models.

5.3.3 Comparison with common Android conventions

Analysis of training from aggregate and personalised data revealed differences in the prediction performance across the interruptibility labels and evaluation criteria. Previous studies on inferring other attentive states (e.g., boredom [98]) have found that despite classifier accuracy not being considerably high, the predictive models can still be improvement over having no model at all. To examine this in this use case, the performances of the multi-modal models are compared against typical conventions and mechanisms available on Android devices, using two baselines.

The aim of these baselines is to achieve the following: a) determine whether having an interruptibility model is worthwhile at all, and b) whether a multi-modal model from implicitly observable sensor and API data is worthwhile over only using the user-declared volume state. However, this analysis is only indicative of the features chosen from the dataset and not the suitability of the DOIG model for labelling behaviour.

5.3.3.1 Always interrupt baseline

The first baseline is inspired by the default behaviour of applications, where interruptibility is not considered and a notification is assumed to be appropriate to be delivered at any time. This type of baseline has been used in similar studies (e.g., [31, 99]) to simulate the extent to which assuming interruptibility produces errors. This is achieved by instructing the predictive models to classify each piece of training data as the user being reachable, willing to engage, or receptive, regardless of the training data.

The predictive performance is shown in Figure 5.6. While this captures all moments of interruptibility (i.e., the sensitivity is 100%), the low PPV demonstrates that in most moments for most users, they were not interruptible. This is consistent with the findings of Chapter 3, Section 3.4, and further supports the use of DOIG model to consider partial responses. Additionally, NPV and specificity performance is understandably poor as the models do not predict that the user is not reachable, engageable, or receptive.

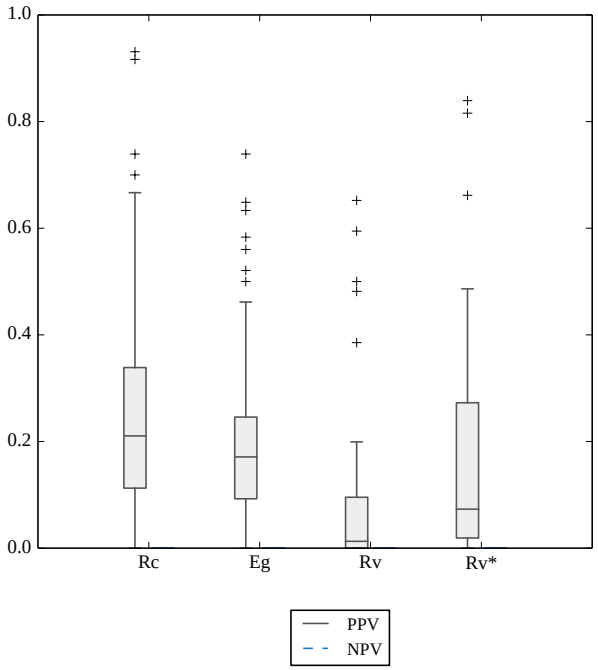
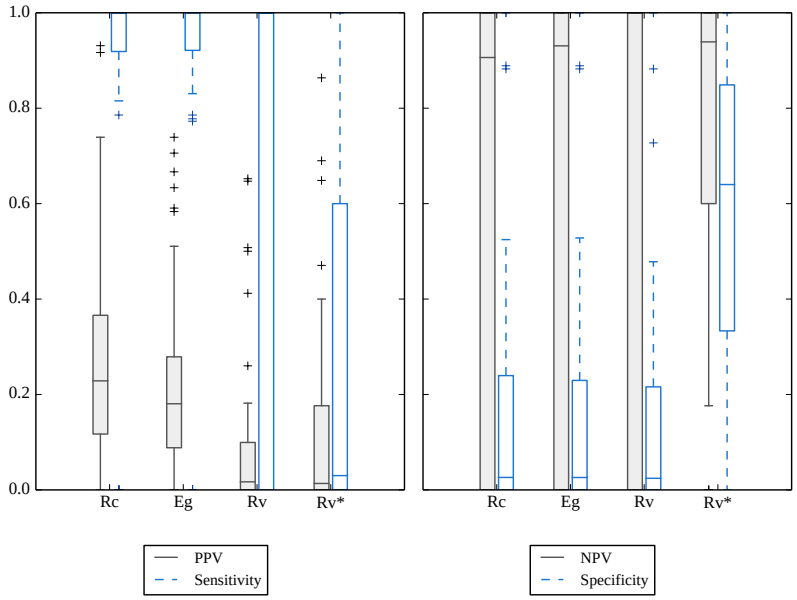


Figure 5.6: Always interrupt baseline PPV performance across users - The user is always interruptible (default application assumption). Sensitivity is 1.0 and 0 for NPV and specificity, across all models. Y-axis represents prediction performance. Figure from [124].

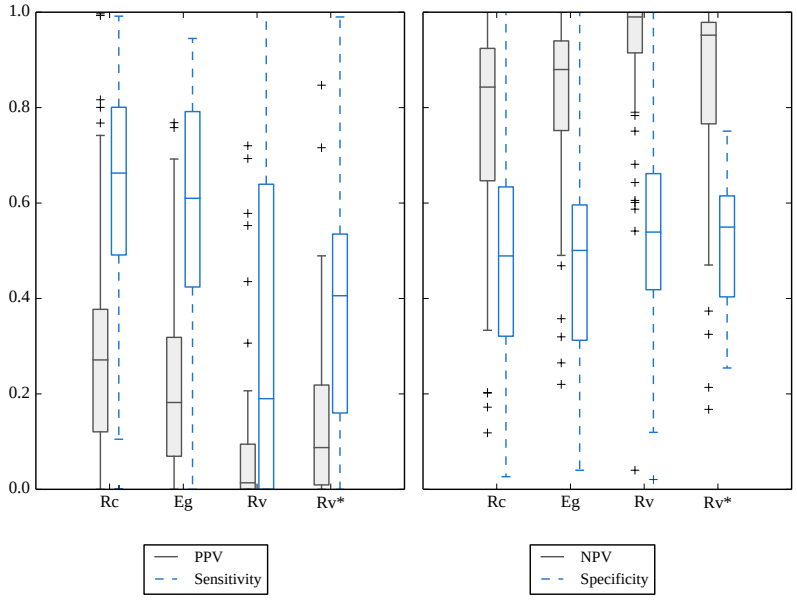
Overall, this suggests that some kind of interruptibility prediction model is worthwhile, with most metrics of the aggregately trained (AT-PT, Figure 5.4a) and personally trained models (PT-PT, Figure 5.4b) outperforming the baseline. However this alone does not indicate whether a multi-modal approach using various contextual features is necessary in comparison to merely considering the volume state of the device (e.g., silent mode).

5.3.3.2 Volume state baseline

As the Android devices in the dataset allow a degree of rule-based interruption management to take place through manually setting the volume state, the second baseline involves training and testing models based only on this feature. For example, a user is conceptually unlikely to be interruptible when the device is in silent mode. The rationale for this baseline is to use it determine whether other contextual features provide



(a) Volume state baseline performance (AT-PT).



(b) Multi-modal model performance (same as Figure 5.4a).

Figure 5.7: A comparison of the volume state baseline against the multi-modal models trained from aggregated data (AT-PT). Rv* refers to receptivity when the device is in use. Y-axes represent prediction performance. Figures from [124].

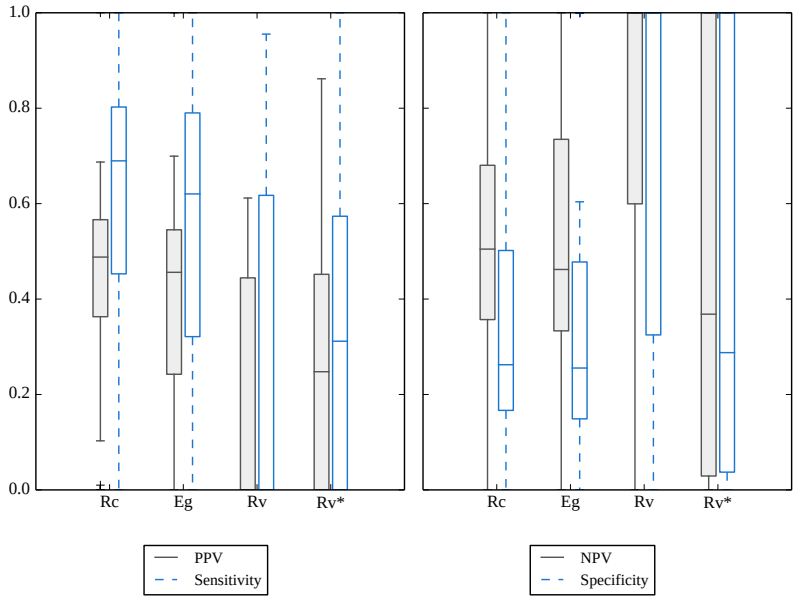
additional utility in reducing the variance between users and in improving typical accuracy.

Figure 5.7a shows the performance of the baseline for AT-PT models. Comparing this with the AT-PT multi-modal model (Figure 5.4a, and shown again in Figure 5.7b for a side-by-side comparison), the baseline performs slightly worse overall at correctly classifying interruptible moments (PPV, shown on the left side) when considering the median and upper quartile values for reachability and engagability and receptivity when in use. With similar performance to the baseline for receptivity when not in use. For sensitivity (also shown on the left side), the baseline performs better for reachability and engagability, but lower for receptivity.

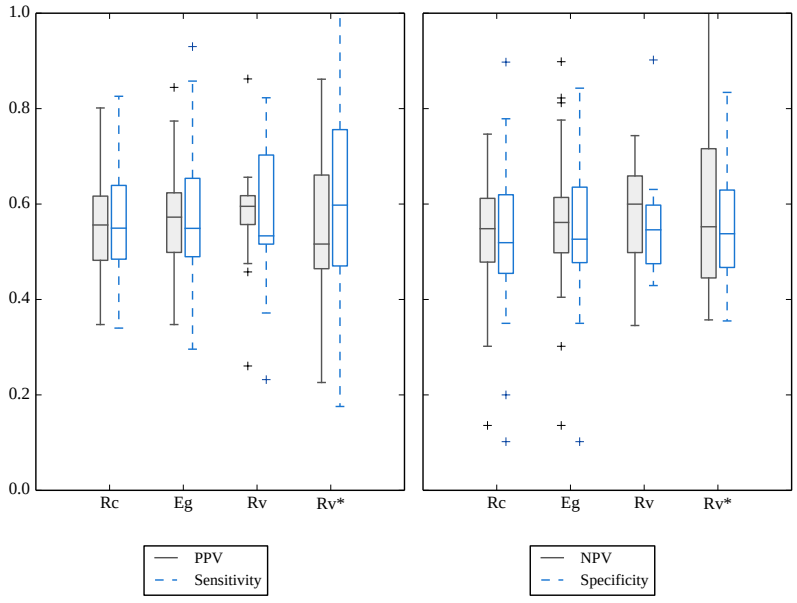
For NPV and specificity (shown on the right side of Figures 5.7a and 5.7b), the general trend in the performance is the inverse of PPV and specificity. The results suggest that the median performance of the baseline for NPV across interruptibility labels and use states is higher or similar to the multi-modal model. This suggests that just using the volume state may be a better choice than a multi-modal approach if this is the sole priority, however the multi-modal approach offers less variation between users and higher specificity. For specificity, the median performances of the baseline when the device is not in use is considerably worse than the multi-modal model and only marginally better when the device is in use.

Overall, these results for AT-PT suggest that users may not always base their decisions in response to a notification purely on the volume state they have set and that the inclusion of other contextual features can aid in correctly predicting opportunities to interrupt. This is useful as while multi-modal AT-PT models were shown to largely under perform against multi-modal PT-PT models, they still offer utility over this baseline.

For PT-PT, Figure 5.8a shows the performance of the baseline. Comparing with the multi-modal PT-PT models (Figure 5.4b, and shown again in Figure 5.8b for a side-by-side comparison), a general trend is that the baseline has much less stability between user performances for all labels and use states. This alone presents a favourable consideration in the use of the multi-modals, as this reduces the variability between different definitions of interruptibility (likewise to the comparison to AT-PT models



(a) Volume state baseline performance (PT-PT).



(b) Multi-modal model performance (same as Figure 5.4b).

Figure 5.8: A comparison of the volume state baseline against the multi-modal models for personalised models (PT-PT). Rv* refers to receptivity when the device is in use. Y-axes represent prediction performance. Figures from [124].

discussed in Section 5.3.2). From this, it could said that users likely manage the volume state differently, and that there may be cases where users unintentionally forget to change the volume state at the exact moment their interruptibility changes, which other

contextual data can help to improve upon.

Additionally, the multi-modal model matches or outperforms the baseline in terms of the median performance for most metrics, interruptibility labels, and use states. With the exceptions being: sensitivity for reachability and engageability (shown on the left side of Figures 5.8a and 5.8b) and NPV for receptivity (shown on the right side of Figures 5.8a and 5.8b) when the device is not in use. Coupling these results with the sole reliance on the human effort required to manage the volume state, the results suggest that the use of a multi-modal trained interruptibility system is more worthwhile, particularly if the objective is to find opportune moments to interrupt; regardless of whether reachability, engageability, or receptivity is used.

Summary: Aggregate and personalised models are useful for different use cases

The primary findings from exploring the use of aggregate and personalised training data can be defined as:

- The relative differences in predictive performance across DOIG labels is larger in comparison to the typical user model (AT-AT, Section 5.2), suggesting that individual differences in interruption habits likely exist between users;
- If a hypothetical application is seeking to predict opportune moments to interrupt, by prioritising true-positive classifications and minimising false-negative classifications, the results showed that personalised models typically outperformed models trained from the data of other users;
- Whereas if an application is seeking to avoid issuing notifications that will not likely produce their desired response behaviour (e.g., being at least reachable) by prioritising true-negative classifications and minimising false-positive classifications, the results showed that models trained from aggregate data typically

outperformed personalised models (but with greater variability between users);

- For applications wishing to perform well at both, personalised models produced the smallest variance between users and across reachability, engageability, and receptivity labels;
- Additionally, a multi-modal provides similar or higher performance when considering all metrics in comparison to having no predictive model in place. While merely relying on the volume state can also over improvement, the use of other contextual features typically reduces the variance between between users.

Collectively, these results provide insight into how different training strategies can impact the performance of models predicting different interruptibility labels. However, while personalised models are often the most suitable choice, a challenge remains in not having personalised data available when a user uses a hypothetical application for the first time. A potential method to overcome this is to adopt the use of online learning, where models can be periodically retrained and used in these early stages of application usage (e.g., [111]), which is examined in the next section.

5.4 Predictive models in an online environment

The evaluation of predictive models in an offline environment (Sections 5.2 and 5.3) has provided a useful indication of the overall predictability, showing particular benefits of building personalised predictive models. However, a hypothetical application will not have the necessary training data when a user first installs it. Online learning, whereby the predictive models are retrained repeatedly with new data, provides a solution to this problem conceptually.

To investigate the predictive performance of this, users in the dataset with at least 21 days worth of data were used. Starting from the second day, the predictive models were retrained daily, using all data from the previous day(s) as the training data (before the

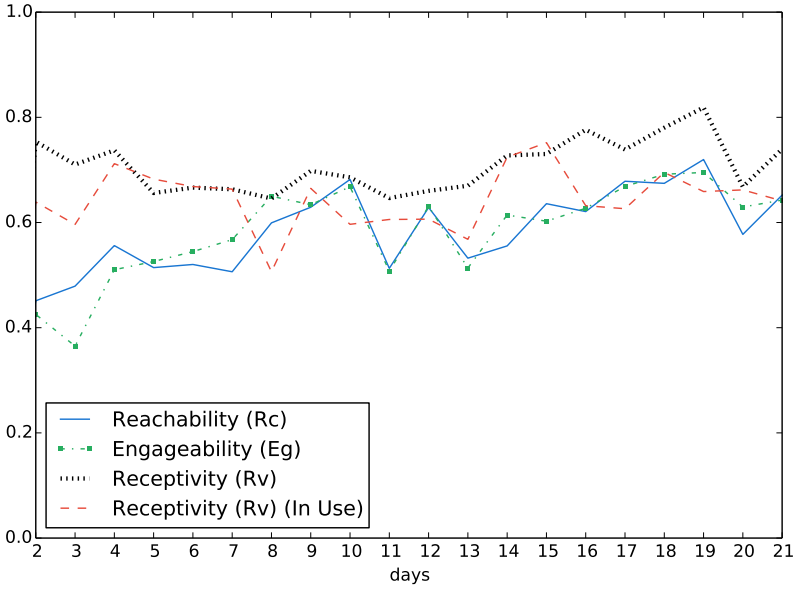
random undersampling pre-processing step) and all the data for that day as the test data (number of users: reachability (N=27), engageability(N=27), receptivity (not in use: N=27, in use: N=18)). This approach has been used in similar studies (e.g., [76]) and allows the examination of how many days of participation a predictive model is likely to need in order to reach peak daily performance.

As the performance across the metrics was similar in an offline environment Figure 5.9 shows the general performance of the models in an online environment, using the mean weighted precision (PPV and NPV) and recall (sensitivity and specificity) across users. The results indicate that for receptivity, the models perform reasonably well initially, with minor fluctuation between days. This is not reflective of similar works, for example Mehrotra et al [76], found that it took up to 9 days of training, however this could be influenced by the use of different contextual features in the datasets.

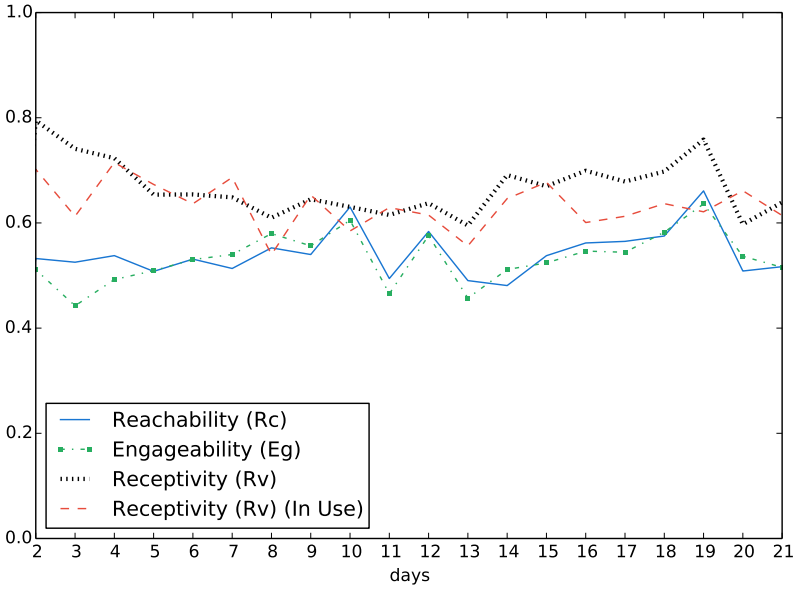
For reachability and engageability models, this is much longer (~1 week). This suggests that these response behaviours may be more sensitive to differences within similar contexts, where several days worth of behaviour is needed to better distinguish between reachable and unreachable, and engageable and non-engageable contexts. This is surprising given that reachability and engageability performed better than receptivity in an offline setting for some models (Section 5.3).

However, examining the unweighted metrics individually, the performance of PPV and sensitivity performed much worse than the weighted values, but with similar consistency across the labels. While this may be influenced by the random-under-sampling pre-processing step, this suggests that it may take several weeks using these features to perform well at correctly identifying reachable, engageable, or receptive moments to delivery notifications.

Overall, the results support the general predictability of labels produced by the DOIG further, and that using features from implicitly sampled contextual data can perform well initially (i.e., when the number of data points will be small). However, as with offline learning, different priorities in the evaluation metrics produce different performance.



(a) Weighted Precision (PPV/NPV)



(b) Weighted Recall (Sensitivity/Specificity)

Figure 5.9: Online learning visualisation for the first 21 days, using the mean value of users with >21 days participation. Y-axes represent prediction performance. Figures from [124].

Considering these results with the offline learning environment (Section 5.3), this could be improved upon in future work by supplementing personalised data in an online learning environment with aggregate data as well (if only for a short period until

sufficient personal data has been collected). This concept has been recently investigated [128] and shown to be a suitable technique.

Summary: Online learning can offer similar performance to offline learning

From the above analysis, the primary findings can be described as:

- Personalised models in an online learning environment produced similar typical performance to an offline environment for weighted metrics after 1 day for receptivity and after ~1 week for reachability and engageability.
- However, online learning performed much worse for the finer grained PPV and sensitivity metrics, suggesting that longitudinal data is needed to perform well at predicting opportune moments in an online environment.

Building predictive models in an online learning environment extends the observations of offline learning (Section 5.3) in finding that different training environments also produce variability in the predictability of reachability, engageability, and receptivity. Overall, hypothetical applications can use these findings to inform the design of their own machine learning strategies, based on their definition of interruptibility (i.e., DOIG label) and priorities in evaluation criteria.

5.5 Conclusions

This chapter examined the predictive performance of the labels produced using the DOIG model for the ImprompDo dataset (i.e., reachability, engageability, and receptivity). In doing so, different machine learning strategies for training and evaluation were explored, including training data selection, training environments, and evaluation metrics. Overall, for future research and the design of intelligent interruption systems using Android

notifications, these results further support the use of the DOIG model, with some models producing >80% precision performance for the majority of users. However the results also highlight the dangers of assuming wider applicability beyond the confines of a single set of labelling, training, and evaluation choices. Producing the following contribution:

C4 Analysis into the predictability of response behaviour using past behaviour that is labelled using the DOIG model, including examining the effect of various machine learning strategies on predictive performance.

This contribution has focused on several common limitations of previous empirical studies (as described in Section 5.1). Sections 5.2 through 5.4 improve upon L5.1 by training and testing predictive models for different interruptibility labels produced by the DOIG model. L5.2 and L5.3 are improved upon by exploring the use of different types of training data in an offline setting in Section 5.3, and the exploration of an online setting in Section 5.4, with different metrics used throughout for evaluating the prediction performance, which correspond to different application priorities.

However, this analysis has some limitations in itself. For example, the dataset and analysis was designed to be as representative of as many different real-world application use cases as possible within a single case study. In doing so, the contextual features used for prediction were limited to those that any Android application could adopt without a fundamental change to their permissions or design (as discussed in Chapter 3, Section 3.3.4). However, other features could also be feasible to use on a per-application basis, with previous works showing predictive power for features such: the time since the last device activity (e.g., [97]), current task data (e.g., [88, 86, 53]), and location (e.g., [92]). Discussed further in Chapter 8, future work could explore the maximisation of predictive performance of the DOIG model labels with these additional features, rather than the primary focus here of examining the relative differences in the performance across the labels.

Additionally, Chapters 3, 4, and 5 have examined each stage of the typical interruptibility

research process (as identified in Chapter 2 and [121]). However, while this is in line with similar studies (e.g., [76, 92, 99, 72]), collectively this has two primary limitations in a) being based on a single dataset with a generic notification design and b) only considering notifications in isolation, rather than as part of a wider process where they can coexist together, and whether decision making can also be seen beyond individual responses to notifications. The following chapters address these using a second dataset containing in-the-wild notifications from multiple applications.

Chapter 6

Model robustness to variability

The decision-on-information-gain model has been shown to assist in the capture, labelling, and prediction of different interruptibility behaviour (Chapters 3, 4 and 5). However, the contributions thus far are based on a single dataset that does not show the extent to which custom notification designs and device variability effects what the model can capture. The purpose of this chapter is to examine the flexibility of the model (which is discussed conceptually in Chapter 3, Sections 3.2.1 and 3.2.2), through analysing a second, larger dataset of notification designs and preferences, collected in-the-wild.

Firstly, the rationale behind this focus is discussed further in Section 6.1, along with the process of collecting further empirical data using a new Android application, Boomerang Notifications. The resulting dataset is then used to examine how the DOIG model can be flexible to different notification design properties that differ from the default properties used in ImprompDo (Chapter 3, Section 3.3) and device preferences in Section 6.2.

6.1 Boomerang Notifications Android application

The intention of this chapter is to support the contributions of Chapters 3 through 5 by exploring the flexibility of the DOIG model using additional in-the-wild empirical data. So far, although the practical usability of the DOIG model has been demonstrated using the ImprompDo dataset, the following limitations have been noted:

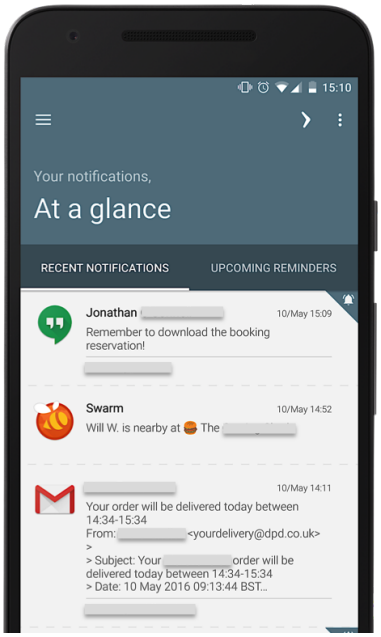
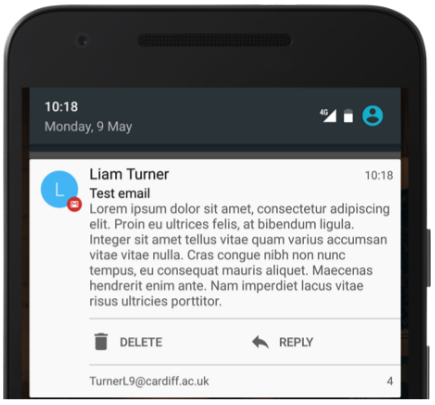


Figure 6.1: An example of the Boomerang Notifications main user interface

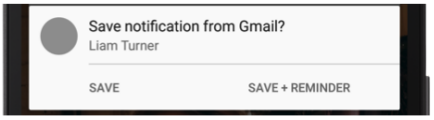
L6.1 The ImprompDo dataset contains response behaviour towards notifications with generic properties in order to be as broadly representative as possible. While using a confined set of interruption design properties is a conventional across interruptibility studies [121], in practice, notifications can deviate from this. Subsequently, while the DOIG model has been formally defined as flexible to customised notifications (Chapter 3, Sections 3.2.1 and 3.2.2), the ImprompDo dataset cannot offer empirical evidence of this.

L6.2 Recent advancements in the Android operating system have allowed for user customisation in when and how notifications are displayed. This is not captured in the ImprompDo dataset due to the study being conducted before these were introduced.

Both customised notification properties and device preferences can affect the number of observable decisions in a response that the DOIG model can capture, and subsequently the distinction between reachability, engageability, and receptivity. To determine the extent to which this customisation occurs, a bespoke Android application, *Boomerang*



(a) An example (email) notification.



(b) The save prompt shown after the notification in Figure 6.2a is removed (assuming that the application is in the user's list of applications to show save prompts for).

Figure 6.2: User feature: the process for saving a notification

Notifications, was developed and released in-the-wild (shown in Figure 6.1). The application harvests characteristics of all notifications that naturally occur on the device, enabling analysis of the variability that can occur (addressing limitation L6.1). It also collects individual user preferences in how and where notifications are made known to the user (addressing L6.2), when notifications are added and removed, and contextual data; discussed further in Section 6.1.2. The application was developed in accordance with the ethical research requirements and processes of Cardiff University.

The application was distributed through the Google Play Store (shown in Figure 6.3) for the public to download and use for free. The application is compatible with Android devices running version 5.0 and higher, which covered ~45% of the market distribution at the time of the study in June - September 2016[1]. To encourage participation, the application enables users them to save and set reminders to review notifications at

[1] As per Google's "Dashboards" at the time of the study - `https://developer.android.com/about/dashboards/index.html`

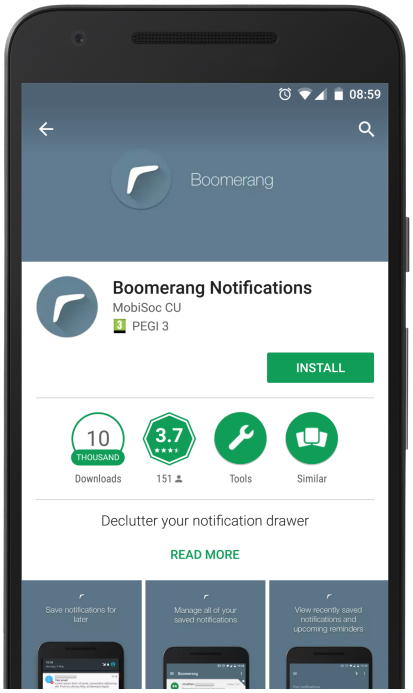
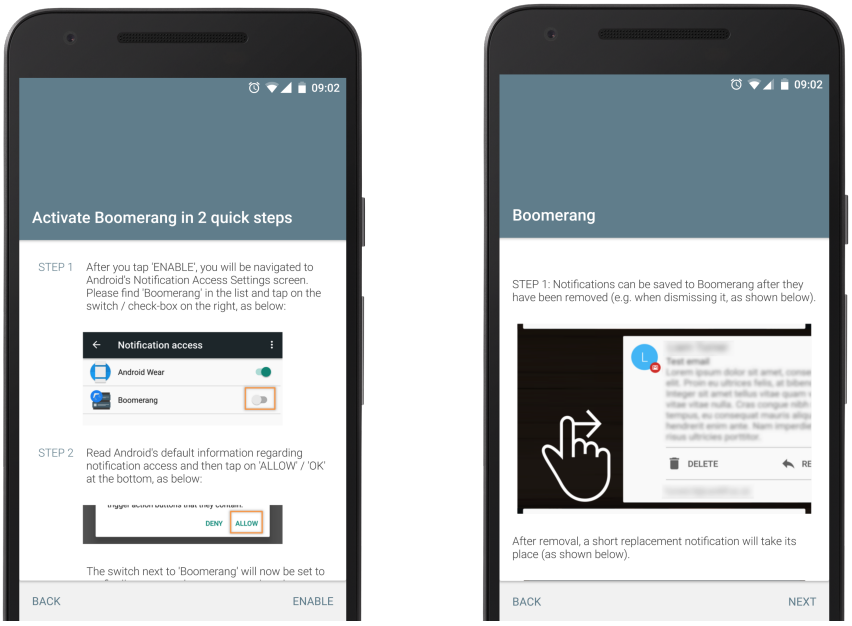


Figure 6.3: The Boomerang Notifications app listing on the Google Play Store.

a later time (shown in Figure 6.2 and discussed further in Sections 6.1.1 and 6.1.2), enabling more natural behaviour to be captured in comparison to the use of monetary compensation [109].

The application received generally positive reviews from online media outlets (e.g., [28, 127]) and users[2], but some users did not like that the app was part of a research study. Additionally, some users suffered similar technical issues to that of ImprompDo that were difficult to capture in testing; for example, device specific bugs that prevented some of the application's features from functioning. However, these issues are only indicative of the challenges of developing research applications that participants wish to use, rather than the ability for a given application to implement the DOIG model (as shown in Chapter 4).

[2] Boomerang Notifications received an average Google Play Store rating of 3.71/5 by the end of the study



(a) An interface in the application setup showing the user how to activate the application.

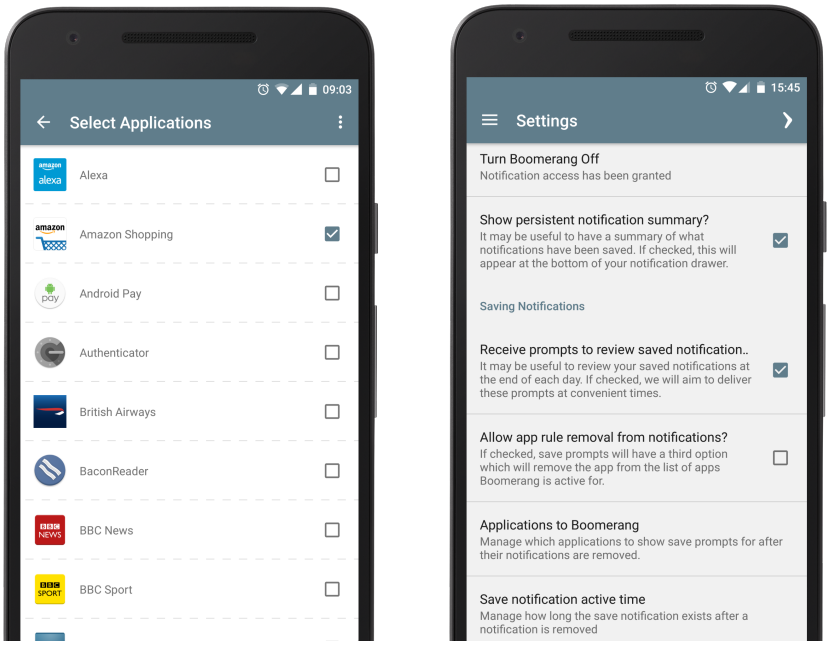
(b) A help interface that shows the user how to save notifications. This screen is shown in the setup and accessed from the main menu thereafter).

Figure 6.4: Screenshots from Boomerang Notification's setup process.

6.1.1 Installation and setup

Self-selecting participants were able to install the application through the Google Play Store in the same manner as other applications. After installation, the application remains dormant until the user opens it and completes a setup process, which must be completed for the application to become functional (as with ImprompDo). The initial screens of this setup describe the application and how it works (e.g., shown in Figure 6.4), along with its research purpose and links to a disclaimer, EULA, and privacy policy. Users are then asked to provide consent to the anonymised data collection before progressing further.

If consent is granted, the user can then progress to activating the application (shown in Figure 6.4a) and setting their preferences of which applications they want save prompts (shown in Figure 6.5a) to occur for, when those application's notifications are removed.



(a) The application rule interface that allows users to select which applications to receive save prompts for after notifications are removed. This screen is shown in the setup and accessed from the main menu thereafter).

(b) Boomerang Notifications' settings interface that is accessible from the main menu. Settings include allowing users to activate/deactivate the application and set additional preferences regarding saving notifications and reminders (outlined in Appendix B, Table B.2).

Figure 6.5: Screenshots of Boomerang Notification's customisation options for the user facing features.

After completing the setup, the user is presented with the home screen of the app, shown in Figure 6.1. From here, the user is shown the most recent notifications saved and those with upcoming reminders that day. A menu is available at the top left of the screen that allows the user to access:

- Details about the study and participation;
- The complete list of saved notifications and reminders;
- App settings for controlling how and when notifications generated by the application occur (shown in Figure 6.5b and outlined in Appendix B, Table B.2);

- A help guide (shown in Figure 6.4b).

6.1.2 Data collection process

The application runs as a continuous background operation in order to enable the passive collection of anonymised research data and for the user facing features to function; with the data periodically sent to a server anonymously at the end of each day. As with ImprompDo, passive background collection from machine sources on the device is used to enable data collection at scale, rather than using experience sampling through surveys. Primarily, the application relies on the NotificationListener API [4] introduced in Android 5.0, which enables third party applications limited access the notifications that occur on the device after the user gives explicit permission. In summary, application implicitly collects details of:

- The properties of all notifications that occurred on the device, such as the originating application, and how many buttons it has. For privacy reasons, the content of the notifications was not collected;
- When notifications were posted, updated, and removed;
- User interactions with the device (e.g., screen on/off, shutdown, boot, etc.);
- Contextual data when the notification and user interaction events occurred, from data sources that did not require invasive permissions, as with ImprompDo (e.g., volume state, battery state, etc.);
- User behaviour with the app's useful features (e.g., notification reminders);
- Device preferences set by the user (e.g., whether pop-up notifications are allowed).

As a result, the dataset produced from the Boomerang Notifications application is considerably richer than that of ImprompDo, as it captures the scale and variety of notifications that naturally occur on the device. The dataset can be examined for a

variety of research hypotheses, however the focus of this chapter and the remainder of this thesis is on improving upon the limitations of the ImprompDo dataset discussed in Section 6.1. Firstly, the flexibility of the DOIG model is explored through examining the variety of notification properties and device preferences that exist in the dataset, as these could affect what decision behaviour can be observed. This forms the scope for the remainder of this chapter, with the user behaviour towards the device and notifications examined in Chapter 7.

Study Limitations

The use of the NotificationListener API presents several challenges and limitations. Firstly, in order for the API to function, the user must explicitly grant Boomerang Notifications access to notifications after install. As participants were self-selecting, the application needed to provide some utility to install and grant this permission. In order to facilitate this, additional notifications are introduced by the application itself (outlined in Appendix B, Table B.2). However the occurrence of these will change based on a user's settings in the application and the natural use of the app's features. Therefore this still offers a natural viewpoint of notification behaviour on the device, and remains arguably more representative than previous studies that issue notifications for experience sampling interruptibility (e.g., [99, 92, 76]).

Secondly, the NotificationListener API provides limited detail in how a notification is removed. The sand-boxed nature of Android results in all interactions with a notification being handled by the application that generated it, which goes beyond the scope of the NotificationListener API. As a result, Boomerang Notifications is aware of what notification were removed and when, but not how (e.g., if the user tapped upon it, or dismissed it, etc.). This is a byproduct of re-purposing Android APIs for purposes they were not originally designed for. However, this limitation is outweighed by the opportunity to retrieve the design properties of notifications across all applications, enabling the primary rationale of this analysis in investigating the flexibility of the

Application	n
Google Maps	3,927,318
Android System UI	2,804,729
Android	1,891,615
Android Downloads Application	1,146,690
Internet Speed Meter Lite	977,378
Light Flow Pro - LED Control	863,592
GPS Status & Toolbox	742,121
WhatsApp Messenger	705,251
Ampere	674,369
Google Play Services	625,902

Table 6.1: The top 10 applications that produced notifications.

DOIG model.

6.1.3 Dataset

The collected dataset contains 32,933,211 notifications (including updates) posted from 7,156 applications, with each application producing an average of 4,602.2 notifications ($SD = 70,292.6,\ Med = 14$). The 10 applications that produced the most notifications is shown in Table 6.1. Notifications occurred from 25 Google Play Store categories (considering games as a single category), with those not present on the store placed in an additional "other" category. Examples of these include system notifications and those that are prohibited on the Google Play Store, such as gambling applications. Each category had an average of 1,266,662 notifications, but with wide variation ($SD = 2,542,582.2,\ Med = 90,662$). Table 6.2 shows the apps that produced the most notifications for a subset of categories as an example.

These notifications occurred across 3,106 users over a 67-day period, with each user using the application for an average of 5.4 days ($SD = 7.8,\ Med = 3$). Each user received notifications from an average of 28.3 applications ($SD = 16.3,\ Med = 27$), with each application issuing an average of 275.2 notifications (per user) ($SD = 553.4,\ Med = 125.6$).

Category	Application	n
Travel and local	Google Maps	3,927,318
	GPS Status & Toolbox	742,121
	Blitzer.de PLUS	29,149
	Waze - GPS, Maps, Traffic Alerts & Sat Nav	9,158
Communication	WhatsApp Messenger	705,251
	Newton Mail - Email & Calendar	259,366
	Gmail	253,790
	Viber Messenger	189,683
Music and audio	Google Play Music	444,667
	Spotify	155,112
	TuneIn Radio Pro - Live Radio	135,858
	TuneIn Radio	30,065
Productivity	DIESEL : App Switcher	300,384
	Inputting Plus: Ctrl + Z/F/C/V	162,780
	MEGA	72,547
	Inbox by Gmail	70,350
Media and video	YouTube	146,352
	Flud - Torrent Downloader	79,813
	Flud (Ad free)	39,745
	tTorrent Lite - Torrent Client	38,292
Lifestyle	Timely Alarm Clock	129,332
	Assistant (by Speaktoit)	21,669
	Morning Routine - Alarm Clock	21,067
	Family Locator - GPS Tracker	14,811
Health and fitness	Strava Running and Cycling GPS	107,269
	Google Fit - Fitness Tracking	88,833
	UP - Smart Coach for Health	86,214
	Twilight	77,419
Social	Glympse - Share GPS location	99,719
	Facebook	45,982
	Instagram	30,129
	Glympse Express	27,189
Weather	YoWindow Weather	59,438
	Weather Timeline - Forecast	53,884
	MyRadar Weather Rada	27,681
	Weather Live	26,956
Game	Integrated Timer For Ingress	24,656
	Real Racing 3	13,890

Category	Application	n
	SimCity BuildIt	5,796
	Asphalt 8: Airborne	4,859
Business	Nine Mail - Best Biz Email App	17,517
	Slack	11,101
	BZ Reminderp	9,518
	OfficeSuite Pro + PDF	5,904
Entertainment	Netflix	15,916
	BBC iPlayer	6,901
	9GAG	6,239
	DIRECTV Remote for Samsung	4,624
Shopping	Slickdeals: Coupons & Shopping	8,080
	eBay	1,225
	TrackChecker Mobile	1,175
	Dealabs: Bon plan & Code promo	919

Table 6.2: Applications that produced the most notifications for 13 example Google Play Store categories.

6.2 Flexibility of the DOIG model

The first analysis of the Boomerang Notifications dataset will focus on further justifying, through empirical data, the need for a flexible labelling methodology for mobile notifications, and how the DOIG model can accommodate this. A principle rule of the DOIG model is that a decision is made after the user gains an additional piece of information about the interruption. Both the notification design and the user's device preferences can impact on when information is made known to the user (e.g., the identity of the application), which subsequently affects what decisions are made when and the ability to capture and separate the spectrum of interruptibility (i.e., from reachability through to receptivity). The intention of this analysis is not to show a complete mapping of combinations (as the Boomerang Notifications dataset is not exhaustive), but to show how (conceptually) decisions can be observable or restricted as a result of the variability.

6.2.1 Variability in notification properties

In addition to content, Android notifications have a number of properties that can impact (to varying degrees) how they are displayed individually and can be interacted with. These can be grouped together into the following:

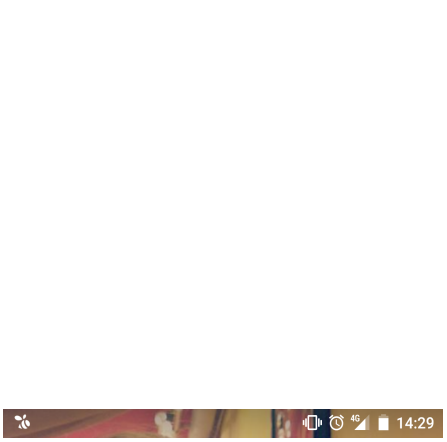
- Grouping and ranking;
- Actions and dismissability;
- Interruptive nature.

For each of these, the frequency distributions within the Boomerang Notifications dataset are discussed, before reflecting on their impact on the usability of the DOIG model.

6.2.1.1 Grouping and priority

An Android design practice is to display the content of similar notifications from an individual application together in a summary notification. However, we find that a small proportion of notifications adopt this behaviour explicitly ($n = 1,777,923,\ 5.4\%$), with the majority shown as standalone notifications ($n = 31,155,288,\ 94.6\%$), however, note that applications can instead update existing notifications with additional information to replicate this grouping.

Applications can also set a priority for the notification [3] that indicates the importance for it to be seen by the user: Maximum (Max), High, Normal (the default priority assigned), Low, and Minimum (Min). This effects where the notification is displayed, with those below-normal priority only shown at the bottom of the notification drawer (Figure 6.6). Normal or higher priority notifications also have an icon shown along the top-left of the screen (Figure 6.6) and can also be shown immediately on the lock-screen (if user preferences and the version of Android allow for it). The distribution in the dataset is: Max ($n = 7,998,188,\ 24.3\%$), High ($n = 3,363,928,\ 10.2\%$), Normal



(a) An example of a Normal priority notification's icon appearing along the top bar.

(b) An example of the notification drawer containing other notifications with below Normal priority that are not shown along the top bar.

Figure 6.6: Icons for notifications with Normal or higher priority are shown along the top (left) of the screen as well as in the notification drawer. Low priority notifications are only shown in the notification drawer.

($n = 12,761,871,\ 38.8\%$), Low ($n = 1,613,201,\ 4.9\%$), Min ($n = 7,168,457,\ 21.8\%$), with a small proportion reporting an unknown priority ($n = 27,566,\ 0.1\%$). Table 6.3 shows the 5 apps with the most notifications for each priority.

6.2.1.2 Actions and remove-ability

Actions can be defined as ways in which a user can interact with the notification (e.g., by tapping on it, dismissing it, or through a button). Actions are performed by the application and could involve opening the application (e.g., after tapping on an email notification) or not (e.g., tapping the delete button on an email notification). For the majority of notifications ($n = 28,867,293,\ 87.7\%$), at least one action could be performed, with explicit action buttons being uncommon by only occurring in a subset of these ($n = 6,787,008,\ 23.5\%$).

Additionally, a large proportion of the notifications could not be individually removed through the conventional swipe ($n = 27,037,362,\ 82.1\%$). This is surprising, but in

Priority	Application	n
Max	Google Maps	3,890,897
	Internet Speed Meter	498,885
	Android System UI	273,462
	Android Phone Call UI	261,659
	DU Battery Saver	246,489
High	Internet Speed Meter Lite	946,811
	Internet Speed Meter	439,964
	WhatsApp	273,024
	Advanced Download Manager	205,769
	Android System UI	193,568
Normal	Android Downloads Application	1,132,322
	Android	1,058,465
	GPS Status & Toolbox	742,121
	Pocket Casts	609,647
	DoggCatcher Podcast Player	525,423
Low	Light Flow Pro - LED Control	705,380
	Light Flow - LED Control	160,727
	Google Play Store	145,056
	Tasker	88,907
	Avast Battery Saver	73,624
Min	Android System UI	2,218,164
	Android	788,055
	Google Play Services	616,436
	VPN by Private Internet Access	425,597
	Google	402,007

Table 6.3: The top 5 applications that produced notifications per notification priority.

some cases these style of notifications can be removed by other means. For example, a media player (which may also produce many updates to the notification) may disable a swipe removal to prevent accidentally stopping playback, and instead offer an action button to remove the notification.

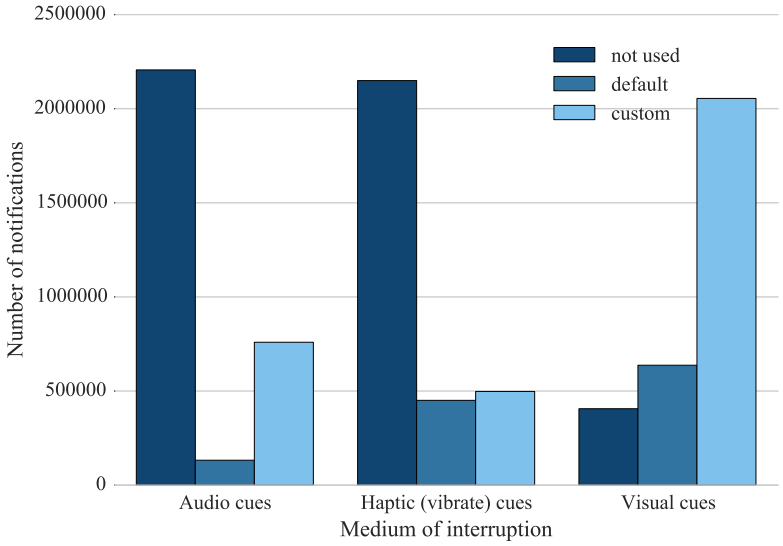


Figure 6.7: Interruptive design across interrupting notifications.

6.2.1.3 Interruptive nature

Notifications can also attempt to interrupt the user in order to draw their attention. To do this, notifications can use a combination of explicit cues, including: an audible tone, haptic vibrate pattern, and visual cues (e.g., flashing LED), or none of these. Surprisingly, 29,835,465 notifications (90.6%) were not designed to be interrupting in any way beyond appearing as a notification. With the remaining 3,097,746 (9.4%) designed to be interrupting in some way (assuming user settings allowed for it to occur).

However, this is arguably reflective of the other design distributions shown, such as those with low priority, those unable to be individually dismissed (e.g., media controls), or those which are updates to notifications showing progress towards some goal (e.g., downloads), which are less likely to be interruptive. Additionally, new notifications can produce a visual cue when using the device by simply appearing (as shown in Figure 6.6).

Figure 6.7 shows the distributions for each type of intrruptive cue, for notifications that adopt at least one of these types. The results show that notifications do not

always use all mediums of interruption, with audio and haptic cues the least used ($n = 2,205,959,\ 71.2\%$, and $n = 2,149,275,\ 69.4\%$ respectively) in comparison to visual cues ($n = 406,066,\ 13.1\%$). Additionally, custom visual cues (e.g., flashing LED patterns) are more prevalent ($n = 2,054,520,\ 66.3\%$) than custom haptic patterns ($n = 497,882,\ 16.1\%$) or custom audio cues ($n = 759,265,\ 24.5\%$), but custom cues in general are used more than the device's default ($n = 637,160,\ 20.6\%$; $n = 450,589,\ 14.5\%$; $n = 132,522, 4.3\ \%$, respectively). Overall, this shows that different notifications use explicit interruptive cues differently, and that applications using the DOIG model should assess this before determining what decisions in the response are observable (discussed further in Section 6.2.1.4).

6.2.1.4 Impact on observable decisions in the DOIG Model

The above frequency statistics illustrate that despite notifications being a standardised convention, several degrees of freedom exist in their design. However, only a subset of these have a notable impact on the response process. The largest impact will arguably be if an application sets distinctly recognisable interruptive cues. This will merge the first two decisions (D1, and D2, shown in Chapter 3, Figure 3.2) that correspond to reachability and engageability, as the information gained that a notification occurred and that it was from a particular application, will occur at the same moment.

However, this assumes that the user is able to remember and distinguish between these and that no other app is also using the same or very similar design (especially for LED pattern and vibration pattern). Determining this will be challenging for an application that does not use an API (e.g., NotificationListener [4]) that allows access to the interruptive properties of notifications from other apps. This will be true for most applications due to the invasive permissions and background monitoring that this entails. In this case, an application will have to determine this from calculating the likely probability that their interruptive properties are unique and distinctly memorable, considering if they use a custom tone, LED pattern, or vibration pattern (Figure 6.7).

Other design properties, such as priority and grouping have small to negligible effects on the model itself, but could impact the need for it at all. For example, the frequency statistics show that a number of notifications operate more passively, such as to merely display the status of something (e.g., system notifications, such as whether the device is plugged in via USB) or progress (e.g., download notifications), using different combinations of design properties, such as: low priorities, the inability to be dismissed, and lack of interruptive cues. A reflection can be made that these notifications may not need to consider interruptibility at all, regardless of definition. Finally, the number of actions that can be performed on a notification will affect the ability to observe of the final decision representing receptivity (D3), as if the notification is not actionable and removable, the user's sentiment towards the content cannot be seen.

6.2.2 Variability in device preferences

The Android operating system also offers users a number of preferences that can impact the way in which they receive and manage notifications. These can be grouped into two areas:

- Notification display preferences;
- Interruption policies.

Likewise to the variability of notification properties (Section 6.2.1) these can change where notifications are displayed on the device and subsequently when information is gained and decisions made. They can also impact whether the design properties of notifications actually occur in practice (e.g., if explicit interruptive cues are suppressed). As for notification properties, the frequency statistics among these areas are firstly discussed, before reflecting on the impact on the DOIG model.

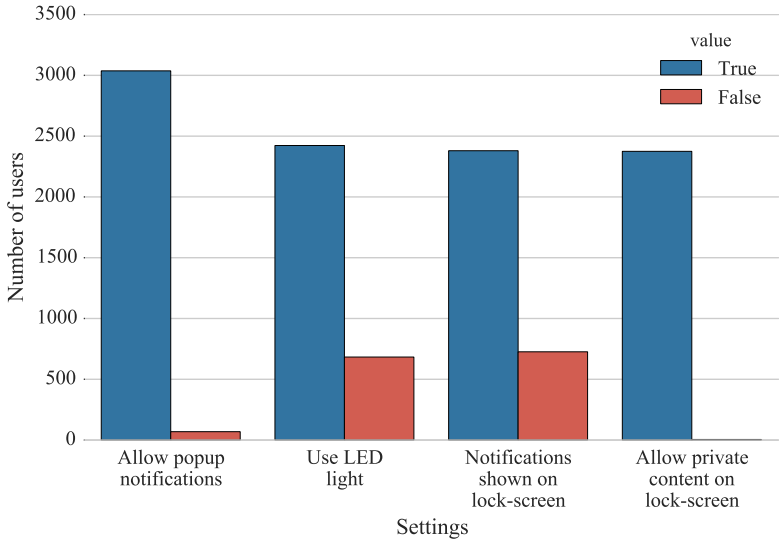


Figure 6.8: Notification display preferences across users.

6.2.2.1 Notification display preferences

Android enables some customisation of when and where notifications are presented to the user through a number of device-wide settings: a) whether the LED patterns on notifications are performed; b) whether notifications are shown on the lock-screen; c) whether private content of notifications is concealed if they are shown on the lock-screen; and d) whether notifications with high priorities that use explicit audible or vibrate cues appear as pop-ups when the device is in use ("heads-up" notifications [1]). Figure 6.8 shows the proportion of users that have these values set to either true or false.

The results indicate that users do make conscious decisions to control the presentation of notifications, with some ($n = 726,\ 23.4\%$) disallowing notifications to be shown on the lock-screen (with 5 users allowing this but with limited private content, which is dictated by the application). Additionally, some users ($n = 683,\ 22\%$) were also conscious about the use of the LED lights and had this feature disabled. Finally, a small number of users ($n = 69,\ 2.2\%$) disallowed pop-up notifications explicitly on a system-wide level. However, this setting cannot be changed through the user interface of most Android devices, instead this can only be performed on a per-application basis.

Per-application data is not available in the dataset due to API restrictions, but the small number of users that did turn this off from a system-wide point of view suggests that this is a setting that some individuals consciously manage.

Overall, these statistics suggest that the DOIG model also needs to be flexible for the settings of individual users as these can change what decisions the user makes in a response and where; this is discussed further in Section 6.2.2.3.

6.2.2.2 Interruption policies

Interruption policies enable "do-not-disturb" capability for notifications during a set time period, extending the historical device-wide silent mode by providing application-level policies. While notifications still arrive as usual, the associated audio, and haptic cues are suppressed. Policies can either silence all notifications or be more selective towards specific applications. Figure 6.9 shows the distribution of users who set an interruption policy at least once.

The results show that the majority of users ($n = 2345,\ 75.5\%$) did not use an interruption policy at all, suggesting that manually managing these may be undesirable. In these cases, users only suppressed interruptions by setting the device to a global vibrate rule (if at all), where the audio cues of all notifications are silenced but vibrate and visual cues still occur. For those that did adopt interruption policies ($n = 761,\ 24.5\%$), the majority of these ($n = 627,\ 82.4\%$) only applied selective policies that only allow the audio and haptic cues from specific notifications (e.g., only alarms). A small number of users ($n = 100,\ 13.1\%$) only used policies that silence all notifications, and a few users used both types ($n = 34,\ 4.5\%$). API restrictions and privacy permissions prevent the exploration of individual rules, however, the results indicate that users are generally not using Android's built in interruption policies.

Going forward, this also raises the question as to what other cognitive mechanisms are being used to manage notifications. This forms a key focus in the analysis of

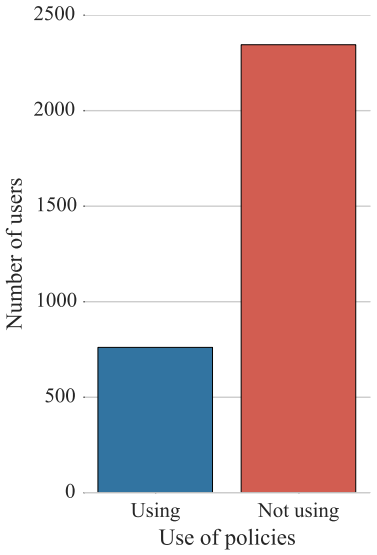


Figure 6.9: Use of interruption policies across users.

notification behaviour in general in Chapter 7. In particular, whether other conscious or subconscious mechanisms can be exposed from behavioural patterns across the independent responses to individual notifications.

6.2.2.3 Impact on observable decisions in the DOIG Model

The variability in device preferences surrounding notifications and interruption policies illustrate that the DOIG model needs to be flexible to these, in addition to the notification design choices of applications. Firstly, the largest potential impact on the DOIG response process comes from users being able to display information about notifications on the lock-screen (Figure 6.8). The ability for this to occur was introduced in Android 5.0 and therefore only needs to be considered for this Android version and above. In this case, the second and third decisions representing engageability and receptivity when the device is not in use (D2 and D3, shown in Chapter 3, Figure 3.2) will need to be merged if the notification priority is normal or higher (otherwise they are not shown immediately on the lock screen). This is because the user no longer needs to unlock the device to view the notification summary. However, if notifications are not shown on the

lock-screen, or if they contain private content that is concealed by the application, then this does not have an effect on the observable decisions (i.e., the response process is similar to Android versions < 5.0).

The other settings have more minor effects on the observable decisions of the DOIG model. Applications that use LED patterns as part of their explicit interruptive cues would need to consider whether the LED is available when determining whether reachability and engageabilty is likely to be distinguishable (as discussed in Section 6.2.1.4). Finally, the ability for high priority notifications to popup on the screen while it is in use has little effect. This is because the decision process is already challenging to observe while the device is in use (as discussed in Chapter 3, Section 3.2.2).

Notification policies can also impact the DOIG model for notifications that have audible or haptic cues. If a policy is in effect that suppresses these cues then this affects whether a response may occur. While it does not change the response process to the same extent as notification display preferences, it does impact whether the user is in a position to be reachable. However, including this consideration is challenging for two reasons. Firstly, even if an interruption policy is not in effect, an application cannot know if a user is simply not reachable or that they were not interrupted (as discussed in Chapter 3, Section 3.2.2.1). Secondly, a typical application will likely not have access to whether a policy is in effect. Notification policy access is an additional permission that is unlikely to fit with the design of most applications, likewise to the notification access through the NotificationListener API.

Summary: Notification characteristics and user preferences are highly variable

Overall, the analysis has highlighted how the DOIG model can be flexible to the variability that can exist in notification design and display preferences, supporting the discussions of the conceptual flexibility in Chapter 3, Section 3.2.2.1. However, in in-

vestigating this, a secondary but independent set of primary findings can be summarised as:

- Notifications are diverse in their design, in addition to their individual content and purpose;
- Notifications are not synonymous with interruptions;
- The use of different on-board notification related preferences suggest that user's wish to control the visibility of notifications;
- The absence of interruption policies for most users suggests that other conscious or sub-conscious management mechanisms may be in effect.

Collectively, these findings highlight the extent to which mobile notifications have evolved from telephonic and alarm based interruptions, and that they now form an integral part of mobile device usage. However, these findings do not highlight the extent to which notifications punctuate our daily lives or the processes used to manage the volume and diversity.

6.3 Conclusions

Android's flexibility in the way notifications operate and can be managed provides degrees of freedom to both application developers issuing notifications and to users receiving them. The strategy adopted in this thesis is to embrace this by creating a flexible labelling framework that is capable of considering this variability and then able maximise, as far as possible, the ability to capture decision making in interruption response behaviour. Chapter 3 has discussed this theoretically, and the analysis of the empirical Boomerang Notifications data set in this chapter supports this further, forming the following thesis contribution:

C5 A demonstration of the flexibility of DOIG the model for different notification

designs and device preferences, using additional in-the-wild data.

Going forward, the findings of variability in the use of interruption policies and notification display settings (Section 6.2.2) suggests that other, wider (conscious or subconscious) management processes for notifications may exist. This forms the primary focus of the next chapter, in exploring the existence of decision making behaviour in how notifications are managed from the wider viewpoint of the notification stack, in order to further support the DOIG model.

Chapter 7

Coexisting Notifications

While individual mobile notifications occur independently of one another, they can coexist and subsequently build up into a "stack" where they compete for attention (shown in Figure 7.1). Using the Boomerang Notification dataset introduced in Chapter 6, the purpose of this chapter is to explore further design considerations for the DOIG model, through analysing the extent to which notifications coexist together, whether decision behaviour in notification responses can be seen from this wider viewpoint, and how this coexistence can impact the decision processes surrounding notification consumption.

The rationale for examining interaction behaviour with notifications that coexist together (as opposed to the viewpoint of individual notifications in isolation) is discussed further in Section 7.1, leading to the introduction of the concept of the notification stack in Section 7.2. Following this, response behaviour towards notifications from the viewpoint of the notification stack is examined in Section 7.3, with the impact of the presence of other notifications on individual responses explored in Section 7.4. Finally, the impact on the DOIG model is discussed in Section 7.5, towards the final conclusions of this thesis.

7.1 Notification Stacks

Although considerable research exists on understanding response behaviour to notifications (e.g., [99, 96, 92]), a common scope in previous studies (e.g., [29, 76]), and the

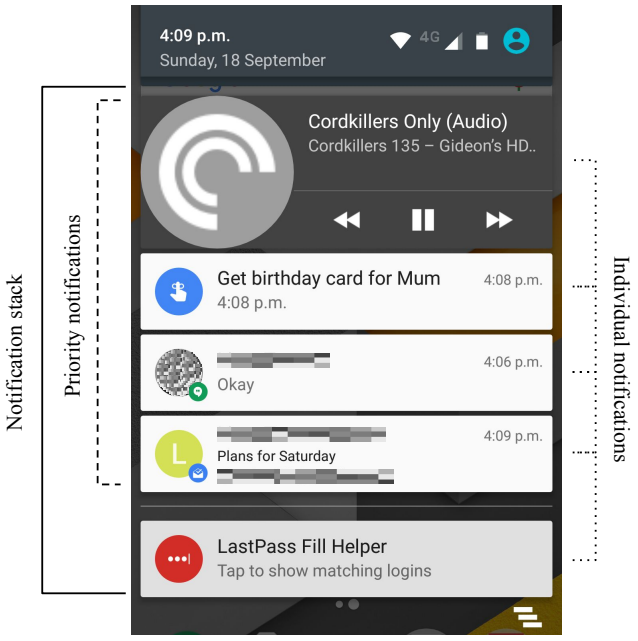


Figure 7.1: A typical Android notification drawer showing an example stack of notifications. Priority notifications refer to those with a priority (set by the application) of at least "Normal" [3], discussed further in Section 7.2.

previous chapters of this thesis, is that while users can receive multiple notifications (e.g., [85]), they interpret and respond to notifications in isolation of one another. This highlights the following limitation in the existing literature:

L7.1 An implicit assumption is made in that the response behaviour (or lack there of) towards a notification is not influenced by the arrival or presence of others.

In an environment where notifications are frequent, users may find it challenging to repeatedly task switch to respond [33, 50, 13, 118] or they may naturally have fewer opportune moments than notifications. As a result, notifications can build up into a *stack* until they are removed by some means, shown in Figure 7.1. The notification stack can be viewed at any time, contain zero or more notifications, and the user can interact with each notification individually in any order.

In reviewing the notification stack, the user undertakes a more burdensome task than has been represented in interruptibility works, with the user (consciously or sub-consciously) needing to make choices on what to prioritise; in a similar manner to information

consumption in other areas, such as social media feeds (e.g., [22]) and email inboxes (e.g., [38, 70]). Whether this changes the perception of notification management in these cases to be a single task, rather than a set of individual tasks treated equally (as has commonly been assumed) has not been widely explored. The intention of this chapter is to examine this through addressing the following questions:

- How often do notifications coexist together?
- Can decision making behaviour in notification consumption be seen at stack level? (e.g., are people often reachable but not receptive?)
- Do aspects from this wider perspective, such as the presence of other notifications, impact individual responses?

These investigations aim to support the DOIG model further through identifying the presence of decision making in responses from the viewpoint of the notification stack, and offer further insight to potentially influential factors on individual responses from other notifications.

7.2 Notifications and usage sessions

Firstly, to determine how often notifications coexist together rather than individually, the following questions are used to frame the analysis:

- How frequently do notifications arrive?
- How frequently do usage sessions occur?
- What are the resulting characteristics of notification stacks? i.e., are users often faced with multiple notifications?

Firstly, analysis of the dataset shows that an average of 2,014.9 new notifications or updates to existing notifications were issued per day ($SD = 3,698.3,\ Med = 963$), 84.0 per hour ($SD = 154.1,\ Med = 40.1$). However, users are likely to not consciously

perceive this rate as this includes notifications that: have no interruptive cues (which is true for most notifications as found in Chapter 6, Section 6.2.1.3), have too low of a priority to appear beyond the notification drawer (e.g., the LastPass Fill Helper notification shown in Figure 7.1), may be presented in groups (e.g., emails), or be relatively minor (e.g., changes in current music track playing). Additionally, the standard deviation and medians for these statistics suggests wide discrepancies across users. Nevertheless, these results highlight the extent to which notifications frequently arrive and compete for our attention.

Users review and address notifications as part of device usage; a *usage session* can be defined as the period between a screen on/boot event and screen off/shut down event (as used in similar works, e.g., [90, 110, 72]), when a user can be assumed to be using their device. Across users, there were 1,097,825 usage sessions after discarding cases of mismatching pairs of screen on/off events (e.g., as a result of the device losing power) and where there were data gaps in the notification meta-data at the start of the session (e.g., as a result of notifications existing before the application was activated), $n = 331,959$. This corresponds to an average of 79.8 usage sessions per user per day ($SD = 110.0,\ Med = 63$), 3.3 per hour ($SD = 4.6,\ Med = 2.6$). As with the frequency of notification arrivals, the standard deviation and median of these distributions indicate wide variability across individuals.

7.2.1 Notification stacks

Analysis of notification stacks at the start of the usage sessions reveals that the notification stacks typically contain an average of 6.4 notifications ($SD = 6.2,\ Med = 5$), with 90.2% containing 2 or more notifications ($n = 990,354$), 8% only a single notification ($n = 87,266,\ 8\%$), and 1.8% containing no notifications at all ($n = 20,205$). This shows that users often face multiple notifications to review (or review again), even if they were only interrupted by one of these.

Chapter 6, Section 6.2.1 highlighted that notifications have highly diverse design properties. This is reflected in the diversity of the notification stack at the start of usage sessions, with these containing notifications generated by an average of 5.1 applications ($SD = 3.6,\ Med = 4$) from 3.4 Google Play Store categories ($SD = 1.9,\ Med = 3$). An average of 1.2 notifications were part of a group ($SD = 2.9,\ Med = 0$), with an average of 2.7 not able to be individually dismissed by swiping ($SD = 2.3,\ Med = 2$). Additionally, each notification in the stack has an average of 1.8 actions that could be performed ($SD = 0.7,\ Med = 1.7$). This shows that as well as being faced with multiple notifications at the start of usage, these notifications will look and function differently.

Priority notifications in the stack

The perceived number of notifications in the stack and their diversity may be lessened by lower priority notifications [3] only being shown in the notification drawer, rather than in icon form along the top bar (discussed in Chapter 6, Section 6.2.1), or immediately on the lock-screen if user preferences allow for it (discussed in Chapter 6, Section 6.2.2). Therefore users may not actively be aware of their presence. To examine the impact of this, the characteristics of stacks are explored when considering only notifications with a priority set to "Normal" or higher, with these types of notifications referred to from this point as being the *priority notifications* of a stack. Some sessions ($n = 160,189,\ 14.6\%$) did not contain any priority notifications in the stack, leaving 937,636 for analysis.

Overall, the characteristics of priority notifications supports the general observation of the entire stack, in that users are often faced with many diverse notifications to review. The notification stack contains multiple priority notifications 78.6% of the time ($n = 736,736$), with an average of 4.2 priority notifications ($SD = 5.7,\ Med = 3$) across 3.6 applications ($SD = 2.9,\ Med = 3$) spanning 2.7 Google Play Store categories ($SD = 1.8,\ Med = 2$). However these notifications are more likely to be actionable in comparison to considering all notifications, with an average of 2.1 possible actions being

able to be performed ($SD = 1.0,\ Med = 2$), and more likely to be able to dismissed, with typically only 1 notification preventing this ($M = 1.1,\ SD = 1.4,\ Med = 1$).

Summary: Notifications arrive frequently and often coexist

Exploring the frequency and diversity of notifications in Boomerang Notifications dataset highlights the extent to which notifications impinge on our daily lives. From the analysis, the primary findings are:

- New and updated notifications arrive frequently, creating a stream of content to assess regularly;
- While notifications are individual and responded to as such, they often coexist at a given time in a diverse stack.

Going forward, this motivates exploring how and when the notification stack as a whole is managed (i.e., notifications are removed), particularly whether selective decision making behaviour (as seen in response to individual notifications in Chapter 3), can be seen from the viewpoint of the wider notification stack as well.

7.3 Selectivity when managing the notification stack

To determine whether decision making can be seen in when notifications are removed from the notification stack, the following questions are used to frame the analysis:

- How frequent are usage sessions with notification removal events?
- How much of the notification stack at the start of a session is typically removed by the end?
- To what extent are notifications kept beyond a session until a later time?

- When do removals occur within sessions? Are there differences in behaviour between notifications present at the start of usage and those that arrive during?

These particular questions are used to show further support for the DOIG model and are not intended to provide a complete representation of user behaviour at stack level, with further pattern analysis extending beyond the scope of this thesis.

7.3.1 Frequency of notification removals

To examine how often sessions with notification removals typically occurred, those sessions with at least 1 removal event are considered, which accounts for just under half of sessions ($N = 487,891,\ 44.4\%$). These sessions occurred frequently, with an average of 33.7 minutes between the end of a session and the start of a new session. ($SD = 145.6,\ Med = 6.4$), with each session typically lasting 5.4 minutes ($SD = 26.4,\ Med = 1.2$). This suggests that users typically adopt an approach of managing the notification stack often, in short bursts.

An average of 4.9 removals occurred in each of these sessions ($SD = 13.8,\ Med = 2$), which suggests that notification management extends beyond individual notifications during usage. However, this does not indicate when this occurs, and whether this typically includes the removal of all of the notifications in the stack or whether some were kept in the notification drawer until a later time; which forms the focus of remaining analysis in this section.

7.3.2 Stack removals and deferment

Figure 7.2 examines the extent to which notifications that were present in the notification stack at the start of usage sessions were removed by the end, by counting the number of sessions where no notifications, some, or all were removed. This is achieved by comparing the sets of unique notification keys at the start and end of sessions, however

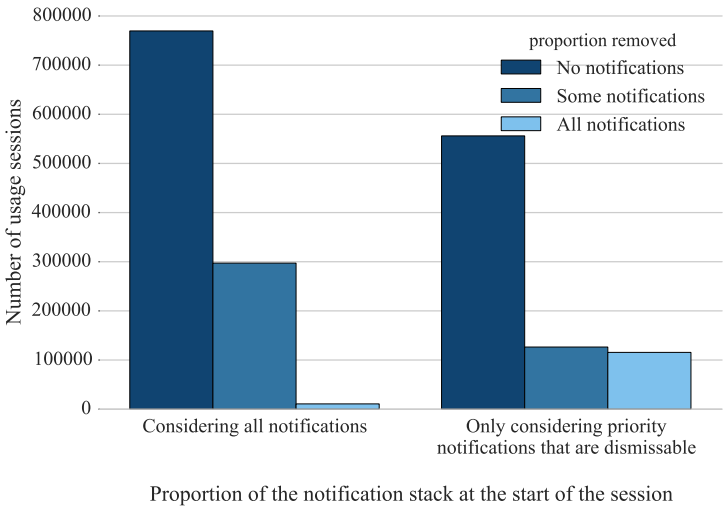


Figure 7.2: Distribution of the number of sessions in which none, some, or all of the notifications present in the starting stack are removed by the end of the session. Considering both: all notifications regardless of their properties (shown on the left, number of sessions = 1,077,518) and only priority notifications that are individually dismissable (shown on the right, number of sessions = 798,358).

in order to consider this, 20,307 additional sessions were removed for either having no notifications in the stack at the start of the session, or where the notification meta-data at the end of the sessions was incomplete.

Figure 7.2 also shows the distribution if only priority notifications that could be dismissed are considered; as these notifications are designed to be the most visible and intended to be removed. In order to consider this, an additional 279,160 sessions were removed as the notifications stacks at the start of these sessions did not contain any of this type of notification. Overall, the results show that while sessions with removal events occur frequently (Section 7.3.1), typically only a subset of notifications present at the start of the session are removed.

Closer inspection of the distributions reveals further insight. Firstly, the distribution of usage sessions where all notifications in the stack were removed in comparison to some is different depending on whether all notifications are considered or only priority notifications. The small amount of cases where all notifications in the stack are removed

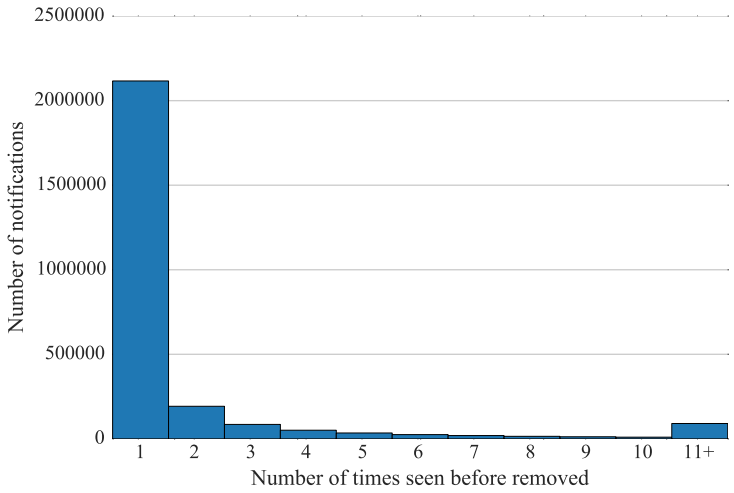


Figure 7.3: The number of usage sessions unique notifications existed within.

is expected due to the notification stacks likely containing low-priority notifications that cannot be individually dismissed (as shown in Section 7.2). Secondly, when considering either all notifications or only priority notifications, there are considerably more cases where no notifications are removed in comparison to those where at least one notification is removed, suggesting that users often use their devices for other reasons than removing notifications (e.g., checking the time, or other app usage).

This can be supported further by examining the notification stack at the end of sessions, irrespective of the starting stack (as notifications can also occur during usage). The results show that 67.5% of sessions ($n = 731,102$) ended with at least 1 dismissable priority notification remaining in the stack ($M = 4.1,\ SD = 5.2,\ Med = 2$); counted over those sessions which either had notifications present at the start (the same number of sessions as the left side of Figure 7.2) or had at least one arrive during (adding an additional 6133 sessions), $N = 1,083,651$.

Overall, this suggests that users selectively leave notifications in the stack until a later time. To examine this from an individual notification's perspective, Figure 7.3 shows the number of sessions each notification was present in before being removed, for those that were present at some point in at least 1 session ($N = 2,653,139$).The results show

that the majority of notifications were removed in the first session they appeared in, however, 20.2% of notifications ($n = 536,748$) persisted across multiple sessions before being removed (($M = 3$ sessions, $SD = 22.3$, $Med = 1$), reflecting Figure 7.2), either due to a conscious choice, or because the user did not notice them. The existence of notifications being deferred until later usage sessions reflects the findings in the earlier chapters of this thesis, in that users can be reachable and engageable to notifications, but not receptive, supporting the DOIG model further.

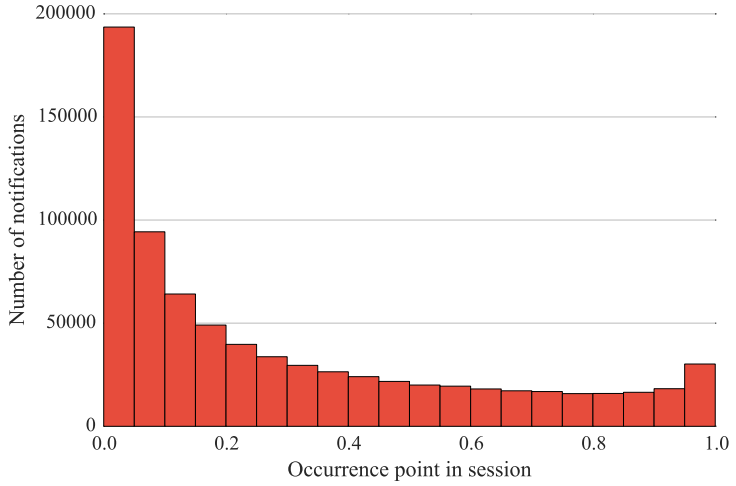
However, this analysis does not examine selectivity in when the notification removals take place during usage sessions, and how it is prioritised over other tasks.

7.3.3 When stack management occurs inside sessions

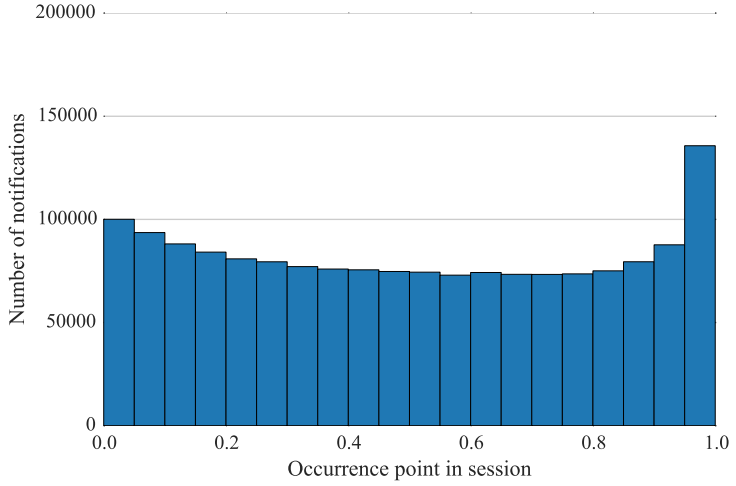
Responding to notifications is only a portion of wider smartphone usage and a user has to prioritise this task against other usage. Figure 7.4 shows the distribution of consumption times of individual notifications, split between those notifications present in the stack at the start of the usage session ($N = 764,788$) and new notifications that were added during a session ($N = 1,649,416$). Overall, the results show that while notifications are removed throughout usage sessions, there is a clear pattern that this occurs at the start and end.

Closer inspection reveals differences in behaviour between notifications present at the start of the session, and those that arrived during. Overall, those present at the start are often removed towards the start of the session; this is unsurprising as interruptions can prompt the user to begin interacting with the device and for most users in the dataset, notifications were shown on the lock screen (as seen in Chapter 6, Section 6.2.2). However, the long-tail distribution suggests that removals of notifications can still occur at any point.

For notifications that arrived during usage, there is also a slight skew towards the start of session. This suggests that despite the user interacting with the device for another



(a) Notifications removed in a session that were present in the stack at the start of the session.



(b) Notifications removed in a session that arrived during the session.

Figure 7.4: When notifications are removed within usage sessions, split between whether the notification was present in the notification stack at the start of the session, or arrived during, using 20 bins with each representing 5% of the usage session.

reason (i.e., other notifications or some other usage), notifications posted soon after usage can direct attention. However, there is also a large proportion of removals at the end of sessions. This is more surprisingly and suggests that users likely manage their notifications after other tasks have been completed on the device. This can also be seen

to an extent for notifications present at the start of a usage session.

This provides insight into the heuristics that users are adopting to manage the notification stack, in avoiding keeping notifications in the stack beyond the current session that no longer serve a purpose. This further supports the premise that decision processes are being used to manage notification responses. However, as this analysis considers the relative point in the session the notification was removed, these distributions could be impacted by long sessions, although most sessions lasted only a few minutes (as discussed in Section 7.3.1).

Summary: Users are selective in removing notifications in the stack

Overall, the results suggest that while notification removals happen frequently, there is high selectivity in what notifications in the stack are removed, and when. The primary findings of this section are:

- The notification stack is managed frequently, often in short bursts;
- Notification stacks are removed to varying extents within usage sessions, but often not completely, with at least a subset of the stack persisting across multiple sessions;
- Within usage sessions, the stack is managed throughout but more so at the start and end; with in situ changes to the stack during usage likely to be reviewed quickly.

Overall, the results show support for the consideration of decision making in response to notifications that underpins the DOIG model introduced in Chapter 3, with selective behaviour also being observed from the viewpoint of the notification stack. However, this leads to the question of whether aspects of wider device usage have an impact on notification response behaviour.

7.4 Influence of wider usage on individual responses

To determine whether the selectivity seen in Section 7.3 may be influenced by wider device behaviour (in addition to the various contextual data (and content) seen in previous interruptibility studies), the following questions are used to frame the analysis:

- Do notification arrivals result in the notification stack being reviewed as a whole?
- Do usage sessions with notification removals occur during interruption policies?
- What impact do the characteristics of the notification stack have on removal behaviour?

7.4.1 Notifications prompt responses to other notifications

To look at the behaviour triggered by individual notifications, sessions that started up to 30 seconds[1] after a notification arrived, and 30 seconds since the previous session, are considered ($N = 291,908,\ 26.6\%$). Analysis of notification removal events in these sessions reveals that other notifications are often removed in addition to (or in lieu of) recent notifications. Firstly, only 9.3% of sessions ($n = 27,151$) had removal events that were limited to only those recent notifications, with 18.7% ($n = 54,691$) also including removals of other notifications that were also present before the usage session. Additionally, in 29% of sessions ($n = 84,856$) involved the user only removing the other notifications. Finally, 21% ($n = 61,230$) had no removal events and 21.9% ($n = 63,980$) only included removals for notifications that occurred after the usage session had started.

Individual notifications can have interruptive cues that prompt the user to review them (as discussed in Chapter 6, Section 6.2.1.3). However as notifications typically exist as part of a stack, this leads to the question of whether interruptive cues have an effect on how the overall stack is managed, in addition to whether the notification it is associated with

[1] Following the conclusions of suitability in Chapter 3, Section 3.4.1

is removed. To examine this, 47% of the sessions ($n = 137,237$) had notifications with explicit interruptive cues within the 30s before it started, with similar a distribution in notification behaviour (sessions: 12,570 removed only recent interruptive notifications, 39,048 others as well, 35,490 only other notifications, 22,549 only notifications after the usage session had started, and 27,580 no notifications).

Overall, the results show that the presence of other notifications can be beneficial for individual notifications by drawing the user into a process of reviewing the notification stack as a whole. Additionally, while the user may have chosen to interact with the device for another task (e.g., to check the time or use an app), this suggests that notifications may influence the user's decisions to use their device, even if not to consume the notifications that have accumulated prior to usage, but this speculative.

7.4.2 Interruption policies are not representative

Interruption policies provide a means to limit notifications from grabbing the user's attention through audio and haptic cues. An interruption policy was in effect throughout 13.2% of usage sessions ($N = 144,406$), which is reflective of the small number of users that used interruption policies, as shown in Chapter 6, Section 6.2.2.

Notifications were removed in 36.8% of these sessions ($n = 53,120$), with an average of 5.1 notifications removed ($SD = 16.9,\ Med = 2$), suggesting that some degree of notification management still takes place. Note that some policies only suppress a subset of applications (as discussed in Chapter 6, Section 6.2.2), however 52.3% of these sessions had notifications removed that were covered by the notification policy in effect ($n = 27,771$).

Overall this shows that notification management (and device usage in general) still occurs during periods of suppressed interruptions. As part of this, the results show that the policies set by users do not always reflect actual behaviour, with notification stack interactions still taking place for suppressed applications. This reflects other findings in

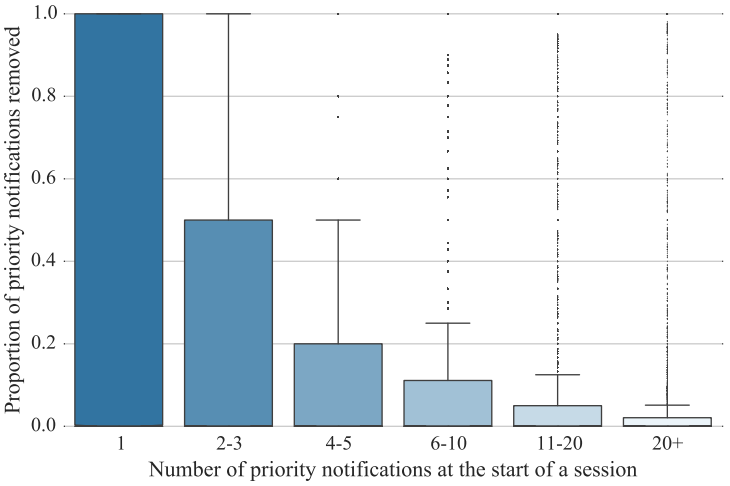


Figure 7.5: The proportion (percentage) of removed priority notifications, grouped by the number of dismissable priority notifications.

the literature, such as users still responding to individual notifications in silent mode (e.g., [77]) and that user's consciously prefer different interruptive cues (e.g., LED, or a vibrate) depending on where the device is in the environment in relation to the user, rather than suppressing them as a measure of self-declared uninterruptibility (e.g., [67]).

7.4.3 Impact from notification stack characteristics

In this section, characteristics of the notification stack itself are examined to determine whether these may influence the perception of the notifications and by extension the response process and DOIG model. To consider this, Figure 7.5 shows (using box-plots) the extent that priority notifications (that are individually dismissable) at the start of a usage session are consumed by the end; using the same sessions as Figure 7.2.

The results show that the proportion of the notification stack consumed does not scale linearly with its size, with typically only a small number of notifications being removed. However there are outliers to this general trend, particularly for large notifications stacks (e.g., 11+ notifications). This suggests that even if the notification stack is very large, users may still review and consume more of the stack, however this is limited to a small

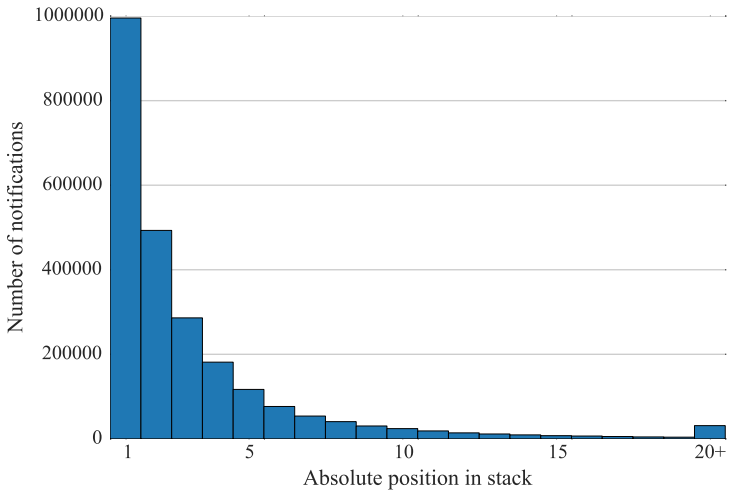


Figure 7.6: The absolute position of notifications that were removed during usage sessions.

proportion of usage sessions in the dataset. Interestingly, this also suggests that users do not often dismiss all notifications and start over when the size gets too big, but rather chip away at it over multiple sessions. From a broader viewpoint, these results provide further indication that the frequency and variety of notifications we receive on a daily basis is challenging to manage and that limited time and cognitive resources are devoted in a given period of usage.

Given that individuals often respond to a small number of notifications irrespective of the size of the notification stack, this leads to the question of whether the ordering of the notifications in the stack influences response behaviour. Figure 7.6 shows the distribution of those removed notifications where the position in the stack was known ($N = 2,412,330$). This shows a long-tail distribution, suggesting that users typically adopt a top-down approach to managing the notification stack. Furthermore, as notifications could have other non-dismissable notifications or a grouped set of notifications above it in the stack, this may explain why the second and third positions are also common (e.g., the user may be removing the highest notification possible).

As notification stacks can vary in size, the distribution using the relative position of the notification in the stack is also explored. The results show a low average position of

21.6% in the stack, ($SD = 24.1\%,\ Med = 14.3\%$). Overall, this supports Figure 7.6 in suggesting that notifications at the top of the stack are more likely to be removed than those towards the bottom.

Summary: Other interruptions and stack characteristics can effect individual responses

This analysis suggests that aspects of wider notification behaviour can have positive and negative effects on individual notification responses, with the primary findings being:

- Interruptions often result in management of the wider notification stack;
- Users do not strictly adhere to their own interruption policies, even those that are selective to individual application rules;
- The size of the notification stack at the start of a usage session has little effect on the number of those notifications removed at some point in the session, which is typically limited to one or two;
- The notification stack is likely reviewed in a top-down manner, with those towards the top of the stack more likely to be removed.

The impact of these on the use of the DOIG model is discussed in Section 7.5.

7.5 Implications and impact on the DOIG model

Overall, the analysis has found that notifications often coexist together and build up as stacks. As a result, users are highly selective in what notifications are removed, reflecting the decision-making processes observed with individual notification responses in Chapter 3, in that users are often reachable (indicated by device usage) but not

receptive. Observing this from the viewpoint of notification stacks further supports that the DOIG model is suitable beyond the findings of the ImprompDo case study.

Additionally, the results show that the current "ready-now-so-push-now" approach to notification delivery creates a means for notifications to inadvertently affect one another. The results show the effects of this can be positive, such as notifications also prompting responses processes for other notifications (Section 7.4.1). However, these can also be negative, particularly in respect to large notification stack sizes and lower positions in the stack amongst other notifications (Section 7.4.3). While this does not effect the application of the DOIG model, it suggests that observing this wider process could provide useful indicators for inferring *why* a user was receptive, or only reachable or engageable.

However, current Android restrictions limit the practical observation of these for most applications, due to the permissions necessary to access this information, as well as the overhead from needing to actively monitor notification activity. Additionally, the NotificationListener API [4] cannot currently offer insight into why a notification was removed (e.g., dismissal), which may offer further value. Therefore, likewise to other contextual data, the usefulness of these as predictive indicators of interruptibility remains a potential focus of future work (discussed further in Chapter 8, Section 8.2); with the conclusions of this chapter being that the decision making behaviour seen for notifications at stack level further supports the premise and suitability of the DOIG model.

7.6 Conclusions

The study of notification management has typically focused on exploring the individual response processes to a set of individual notifications and building design considerations from aggregated findings. However, this does not take into consideration that notifications frequently coexist in a notification stack and the potential decision making that

takes place in choosing what to respond to and when. This analysis of the Boomerang Notifications dataset (introduced in Chapter 6) addresses this limitation directly (L7.1), and finds that decision making behaviour in notification responses can also be seen from this wider viewpoint, supporting the premise of the DOIG model further. This produces the following contribution towards the wider thesis:

C6 An exploration of where the DOIG model sits amongst wider notification behaviour on the device.

The remainder of this thesis outlines and reflects upon the research conclusions made across Chapters 2-7, and discusses potential future directions.

Chapter 8

Conclusions and future work

The study of interruption behaviour focuses on determining when to deliver information so that the interrupter and interruptee enjoy maximum utility. This includes avoiding disruptions at inconvenient moments [52, 18, 76] and making delivery harmonious and relevant to the user's current context [97, 84, 24]. To achieve this, empirical studies have sought to capture, label, and predict the response behaviour towards individual interruptions (such as mobile notifications, [92, 72, 76]). These insights have helped to inform the design of intelligent interruption components in various scenarios, such as mobile application notifications (e.g., [92, 99, 122]).

However, Chapter 2 (and [121]) identified several broad conventions across the research area in need of further direct attention. One particular area is the need for improved mechanisms for capturing and labelling interruptibility from response behaviour (with the limitations in the existing convention outlined in Chapter 3, Section 3.1). This has formed the central focus of this thesis, with the ubiquitous nature of mobile notifications motivating the use of Android apps as a platform for conducting in-the-wild studies. In this chapter, a summary of this thesis and the research contributions made is presented, with a discussion of potential directions for future work.

8.1 Thesis summary

Mobile notifications punctuate our daily lives for a variety of reasons. Each notification can bring varying utility depending on when it is delivered, ranging from highly positive effects to highly negative effects. Assessing the appropriateness of delivering information, which can be highly interruptive for the recipient, is a task that is generally easily handled by the human brain [34, 16, 126, 69]. However, the current convention of mobile notifications is to push content to a user freely, which often results in a backlog of notifications. The management of this, as well as any suppression of their interruptive nature is offloaded from applications and made to be responsibility of the user (e.g., through mechanisms on the device such as silent mode).

The investigation of response behaviour towards notifications and the proposal of intelligent delivery systems that learn from past behaviour has been a popular research area, however several key challenges and limitations exist in the common conventions used (as discussed in Chapter 2). In particular, responses to notifications have largely been considered to be a single large decision, which has led to the use of a black-box approach [122, 124] to labelling interruptibility, whereby a specific labelling task is typically performed after a notification has been consumed (such as filling in a survey). In reality the response process (particularly towards Android mobile notifications) can involve multiple sequential decisions as the content of the notification is pursued through interactions with the device; where a notification could be considered successful and a user interruptible (on a per-application basis) if a user at least partially responds, even if the notification is not tapped on. Additionally, research studies typically only adopt a single definition of interruptibility and conduct research with additional data collection and labelling mechanisms (such as surveys) that are not practical for real world applications. This creates challenges and uncertainty when translating systems from research studies into the real world when the experiment environment does not match a particular application's operation.

The central thesis of this research is the proposal of a framework that decomposes the

natural response behaviour towards notifications into the decisions that can be observed passively; in turn this enables a flexible basis for labelling interruptibility, addressing the limitations of the existing convention. Different applications can use the framework to collect, label, and predict interruptibility using their own thresholds of what makes a notification delivery successful. This is supported through analysis surrounding two empirical datasets collected in-the-wild - ImprompDo, introduced in Chapter 3, and Boomerang Notifications, introduced in Chapter 6, with the datasets containing more users than typical interruptibility studies (as discussed in Chapter 2, Section 2.4.3), providing a rich basis for investigation.

8.1.1 Contributions, key observations, and limitations

The research undertaken forms 6 contributions that can be grouped under three primary areas. Together these provide support for the central thesis that decomposing mobile notification response behaviour is more worthwhile than the existing black-box convention for learning and predicting interruptibility.

8.1.1.1 Current conventions in interruption research

The study of interruption has taken place with many different types of interruptions and environments. A survey of the research area (Chapter 2 and [121]) reveals wide fragmentation in a number of areas: the scenarios explored and definitions of interruptibility; data collection environments; and prediction practices. Despite this however, a number of conventions used across studies in each of these areas have been highlighted. From this, 10 open research questions are proposed that range from improving upon the limitations found in the conventions (such as for labelling), through to areas that have not been extensively explored (such as comparing different machine learning environments). Collectively this produces the first contribution of this thesis:

C1 A survey of the fragmented research area, developing open research questions by

highlighting limitations and gaps in existing methodologies and conventions for collecting, labelling, and predicting interruptibility.

From this, a subset of the proposed research questions help shape the focus of the remainder of this thesis (outlined in Section 8.2), which primarily surrounds improving upon the limitations exposed in the conventions for labelling interruptibility, and ramifications of this in prediction.

8.1.1.2 The decision-on-information-gain model

This thesis proposed a flexible framework for labelling interruptibility, the decision-on-information-gain (DOIG) model (Chapter 3), implemented and validated within the context of Android mobile notifications. The DOIG model extends the existing black-box convention [122, 124] for implicitly capturing and representing interruption response behaviour, enabling per-application flexibility of what it means for someone to be interruptible and a notification successful. This creates the second contribution of this thesis:

C2 A flexible model for labelling interruptibility for different definitions, the Decision-On-Information-Gain (DOIG) model, that deconstructs the observable behavioural trace in a response to a notification.

Despite being limited to observable decisions, support for the model can be seen through an in-the-wild study; with evidence that different response behaviour can be captured, reducing the potential for false-negative labelling of interruptibility in comparison to the existing black-box convention (Chapter 3, Section 3.4). This leads to the following additional contribution:

C3 Analysis into the natural decision behaviour underpinning interactions with notifications, using data collected in-the-wild.

The results demonstrate that the worse case usability of the DOIG model is the same as the best case of the current convention of passively observing response behaviour

(Chapter 3, Section 3.4). It should also be noted that while the use of experience sampling of receptivity was avoided in favour of passive observation, the two practices are not mutually exclusive and both could be hypothetically implemented into future studies, or applications where appropriate. Additionally, the practical feasibility of passively implementing the DOIG model with typical Android hardware was also demonstrated, supporting C2 (Chapters 3 and 4).

In regards to prediction, the analysis in Chapter 4 found that different passively collectable contextual features were statistically correlated to different DOIG labels, suggesting predictability. From this, Chapter 5 adds further support for the DOIG model through finding that the predictive performance of models built with these features for each label were inline with existing works that predict using a single label of interruptibility, and that the models tested with personal data outperform baselines representing the current built-in conventions on Android devices. Additionally, the analysis also presents additional findings in the relative performance differences of using various machine learning strategies and evaluation criteria. Together this produces the following contribution:

C4 Analysis into the predictability of response behaviour using past behaviour that is labelled using the DOIG model, including examining the effect of various machine learning strategies on predictive performance.

These contributions, surrounding data collection, labelling, and the prediction of interruptibility show the usability and performance of the DOIG model with a typical use case. This is made possible by the DOIG model not changing how notifications operate or are responded to in any way, it is a wrapper that is flexible to what behaviour can be observed passively. Additionally, the passive collection of the data creates a representative outlook on the feasibility of interruptibility prediction for real world applications without a fundamental change to application design; this is in contrast to the typical convention of aiming to maximise prediction performance regardless of technical feasibility or changes required to a real world application's operation (as seen in Chapter 2).

8.1.1.3 Robustness: DOIG model flexibility and position among wider behaviour

Chapter 6 presents the Boomerang Notifications dataset, to firstly show how the design of notifications and user preferences vary across the Android application ecosystem. The results support the design choice of creating the DOIG model to be a flexible labelling mechanism (Chapter 3, Section 3.2.1), with the empirical data highlighting the extent that the variability occurs. Within this, the results also show that a large proportion of notifications are not interruptive in nature beyond the visual cue of appearing on the device's user interface. Therefore, not all applications need to adopt an intelligent interruption system, but those that are interruptive can use the DOIG model to label how interruptible the user was, based on the information that is known to the user at each decision point in the response.

The results also show that individuals typically do not adopt the mechanisms built into Android to suppress notifications (Chapter 6, Section 6.2.2). However, individuals do adopt a variety of preferences in where notifications are made known to them (e.g., on the lock-screen). This builds upon the findings of variability in notification design, showing that any labelling methodology needs to be flexible (as the DOIG is) to these preferences. This chapter creates the following contribution from this analysis:

C5 A demonstration of the flexibility of DOIG the model for different notification designs and device preferences, using additional in-the-wild data.

Chapter 7 examines notification behaviour from the wider viewpoint of the notification stack (where notifications can coexist together) using the Boomerang Notifications dataset, and finds that decision making behaviour in notification responses can also be seen from this viewpoint. Additionally, the previous chapters of this thesis have assumed, like other works in the area, that notifications occur in isolation without any cross-notification behavioural affects. However exploration of the dataset reveals that that not only do notifications punctuate our daily lives frequently, behavioural patterns amongst device usage suggest that the notification stack has some degree of influence

on the response behaviour towards individual notifications. For example, the analysis found that users typically respond to notifications at the top of the notification stack, are highly selective in the number of notifications responded to each time, and that notification arrivals can start response processes for other notifications. While this does not directly affect the operation of the DOIG model, the results offer considerations for applications that wish to infer why a user may have responded in a certain way, i.e., they were reachable but not receptive, which may be the result of other notifications being present on the device. The effect on prediction performance is not explored, due to the majority of real world applications not having access to this data, however this does present an option for future work if this limitation changes. These findings produce the final contribution towards this thesis:

C6 An exploration of where the DOIG model sits amongst wider notification behaviour on the device.

While the DOIG model and the other contributions of this thesis have focused on improving upon the limitations found in various common conventions in the wider research space, some limitations remain as discussed above. These form part of the future directions of this work, alongside further directions that remain unexplored (such as the wider research questions proposed in Chapter 2 that were beyond the scope of this thesis).

8.2 Future directions

The contributions made towards this thesis can also be used as a foundation for future work, both in terms of further research investigations into human behaviour, and in the creation and application of intelligent interruptive components into Android applications.

8.2.1 Maximisation of predictive indicators with the DOIG model

Firstly, a key design choice made in the development, implementation, and validation of the DOIG model has been to remain applicable to as many applications as possible. As such, the collection of response behaviour and contextual features used for prediction were limited to components that did not require a fundamental change to either the operation or privacy implications of an application. However, other studies of interruptibility have found other features not considered here, such as location (e.g., [92]) and content (e.g., [76]) to be useful predictive features. While the viability of implementation in real world applications should be retained, an opportunity exists to explore maximising the performance of the predictive models built for the labels produced by the DOIG model using these features.

Building upon this, Chapter 7 found behavioural effects from the characteristics of the notification stack, therefore this exploration of maximising performance could also explore the effect of this behaviour encoded as features on prediction performance. This could also include investigations into personal differences in this behaviour (e.g., analogous to that seen in email management [38]) and prompts for further investigations into behavioural patterns beyond this thesis; such as the effect of application usage on notification management and vice versa.

8.2.2 Real world application and evolutionary learning

A secondary theme of future directions surrounds the development of further real world applications that integrate the DOIG model for labelling, and by extension, prediction. For example, the scope of this thesis has focused on simulating performance of the predictive models through splitting the dataset into training and test cases, likewise to the majority of research studies in this area. Whether the performance is truly reflective of a real world scenario would help to scope the training requirements of labelled notification responses further. This could also include investigating the acceptability and adoption

of an interruptibility-driven notification system into applications, including specific domains such as delivering interventions, as well as general consumer applications.

Building upon this, parts of this thesis, as well as some other studies (e.g., [92]) have investigated the role of online learning to observe the predictive performance from adding new behaviour over time. However the relevance of interruption behaviour over time has not been widely explored, and whether the relevancy of training data can diminish. This may apply for two reasons, firstly the notifications received on a device will likely change over time as applications are added and removed from the device, and secondly, if an individual (or another application) is aware that a predictive model is being used by a given application, they may consciously or subconsciously become aware of a particular pattern and adjust their behaviour. Therefore the role of evolutionary learning of interruption behaviour remains an additional area to explore.

8.2.3 Wider research questions

In addition to these areas, the analysis conducted in Chapters 3 through 7 are relevant to 5 of research questions surrounding the wider interruptibility research space (proposed in Chapter 2 and in [121]). Firstly, investigations into the applicability of the DOIG model and wider behaviour go some way towards addressing the following research questions:

(RQ2) Given the diversity of potential scenarios, when are generalised and interoperable solutions for interruptibility sufficient, and when are domain specific solutions necessary?

(RQ3) Can including the extent of a response to an interruption provide additional semantic value for inferring the user's attentiveness towards it?

RQ2 is somewhat addressed within the context of Android notifications through finding that a flexible labelling model is necessary, because individuals respond to notifications differently (Chapter 3), and that notification designs and device preferences can vary

(Chapter 6), which can effect the response process. Additionally, findings of varying predictive performance for different interruptibility labels across different machine learning training and evaluation strategies (Chapter 5) suggests that a one-size-fits-all predictive model would not be appropriate for all application use cases. However, as discussed in Section 8.2.2, this could be investigated further through additional real world application case studies, and this may not be representative of other types of interruptions.

In regards to RQ3, the finding that individuals are often reachable but not receptive to notifications to consume them (Chapters 3 and 7) suggests that decomposing response behaviour into the decision steps that take place is worthwhile. With Chapter 3, Section 3.4 showing that this can reduce the potential for mislabelling in comparison to relying on complete responses towards notifications. Additionally, the ability to observe this passively on Android devices further supports that this is worthwhile to consider (Chapters 3 and 4). However, across both of these research questions, these conclusions are only suggestive of the wider research space and remain limited within the context of Android smartphone applications.

In addition to this, the analysis into correlating contextual features with DOIG labels (Chapter 4) and predictive performance (Chapter 5) are relevant to some degree towards addressing the following three research questions:

(RQ8) How do training dataset characteristics affect the diminishing returns of prediction performance?

(RQ9) When should intelligent interruption systems adopt online and offline learning, and what factors in the scenario and data collection influence this choice?

(RQ10) Do personalised models mean better performance and how does this balance with increased complexity? Could a hybrid approach using personal and aggregated data reduce the training requirements for new users?

For RQ8, analysis of the ImprompDo dataset suggests the benefits of multi-modal

predictive models over just using the volume state of the device (Chapter 5, Section 5.3.3), along with analysis into the prediction performance over time in online learning (Chapter 5, Section 5.4). Comparisons between offline learning (Chapter 5, Sections 5.2 and 5.3) and online (Chapter 5, Section 5.4) contribute to RQ9 when combining the reported results with the technical considerations in either performing the learning on mobile devices or incorporating a client/server architecture. For RQ10, performance differences between using aggregated versus personal training data (Chapter 5, Section 5.3) are relevant to the first part of this question. This also compliments other recent works that have addressed this question (e.g., [128]). However, whether these findings are reflective of other types of interruptions remains an area to explore further.

Lastly, Chapter 2 proposes 5 other research questions that were not addressed in this thesis, RQ's 1, 4, 5, 6, and 7. These continue to be active areas to explore further, both within the domain of mobile notifications and the wider research space.

8.3 Final remarks

The contributions of this research primarily surround improving upon the limitations of the existing conventions in conducting interruptibility research, particularly surrounding limitations in the conventions for labelling interruptibility and the prediction of these labels. Therefore the intention of the contributions made is to be relevant not only to interruptibility surrounding mobile notifications, but also conceptually for other types of interruptions. Additionally, for real world Android applications wishing to implement interruption components into their notifications, the analysis is undertaken with a broad variety of use cases in mind, from a labelling framework that is flexible to different definitions of interruptibility to performance examinations of different machine learning conventions; supported by the data used being collected by other in-the-wild applications.

For mobile operating systems (e.g., Android), the results suggest that interruptibility

mechanisms would benefit greatly from dedicated APIs for observing interactions with the device and interruptive mechanisms (e.g., notifications), rather than re-purposing mechanisms (where possible) that are intended for other functions on the device. This has been seen to some extent with recent developments in Android creating dedicated APIs for accessing some areas of smartphone usage (e.g., Notification Listener API [4]), however this remains limited to a few areas of usage with limited granularity (e.g., Usage Stats Manager API [7] and Network Stats Manager API [2]).

Bibliography

[1] (2017a). Heads-up notifications. https://developer.android.com/guide/topics/ui/notifiers/notifications.html#Heads-up.

[2] (2017). Networkstatsmanager. https://developer.android.com/reference/android/app/usage/NetworkStatsManager.html.

[3] (2017a). Notification priority. https://developer.android.com/reference/android/app/Notification.html#priority.

[4] (2017b). Notificationlistenerservice. https://developer.android.com/reference/android/service/notification/NotificationListenerService.html.

[5] (2017). Position sensors. https://developer.android.com/guide/topics/sensors/sensors_position.html.

[6] (2017b). Security tips - android application sandbox. https://developer.android.com/training/articles/security-tips.html.

[7] (2017). Usagestatsmanager. https://developer.android.com/reference/android/app/usage/UsageStatsManager.html.

[8] Adamczyk, P. D. and Bailey, B. P. (2004). If not now, when?: the effects of interruption at different moments within task execution. In *Proc. CHI'04*, pages 271–278. ACM.

[9] Adamczyk, P. D., Iqbal, S. T., and Bailey, B. P. (2005). A method, system, and tools for intelligent interruption management. In *Proc. TAMODIA'05*, pages 123–126. ACM.

[10] Anderson, C., Heißler, C., Ohly, S., and David, K. (2016). Assessment of social roles for interruption management: a new concept in the field of interruptibility. In *Proc. UbiComp'16 (Adjunct)*, pages 1530–1535. ACM.

[11] Andone, I., Błaszkiewicz, K., Eibes, M., Trendafilov, B., Montag, C., and Markowetz, A. (2016). Menthal: a framework for mobile data collection and analysis. In *Proc. UbiComp'16 (Adjunct)*, pages 624–629. ACM.

[12] Avrahami, D., Fogarty, J., and Hudson, S. E. (2007). Biases in human estimation of interruptibility: effects and implications for practice. In *Proc. CHI'07*, pages 50–60. ACM.

[13] Bailey, B. P. and Iqbal, S. T. (2008). Understanding changes in mental workload during execution of goal-directed tasks and its application for interruption management. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 14(4):21.

[14] Bailey, B. P. and Konstan, J. A. (2006). On the need for attention-aware systems: Measuring effects of interruption on task performance, error rate, and affective state. *Computers in Human Behavior*, 22(4):685–708.

[15] Begole, J. B., Matsakis, N. E., and Tang, J. C. (2004). Lilsys: sensing unavailability. In *Proc. CSCW'04*, pages 511–514. ACM.

[16] Bennett, A. (1981). Interruptions and the interpretation of conversation. *Discourse Processes*, 4(2):171–188.

[17] Boehm-Davis, D. A. and Remington, R. (2009). Reducing the disruptive effects of interruption: A cognitive framework for analysing the costs and benefits of intervention strategies. *Accident Analysis & Prevention*, 41(5):1124–1129.

[18] Böhmer, M., Lander, C., Gehring, S., Brumby, D. P., and Krüger, A. (2014). Interrupted by a phone call: Exploring designs for lowering the impact of call notifications for smartphone users. In *Proc. CHI'14*, pages 3045–3054. ACM.

[19] Borst, J. P., Taatgen, N. A., and van Rijn, H. (2015). What makes interruptions disruptive?: A process-model account of the effects of the problem state bottleneck on task interruption and resumption. In *Proc. CHI'15*, pages 2971–2980. ACM.

[20] Campbell, A. and Choudhury, T. (2012). From smart to cognitive phones. *Pervasive Computing, IEEE*, 11(3):7–11.

[21] Chang, Y.-J. and Tang, J. C. (2015). Investigating mobile users' ringer mode usage and attentiveness and responsiveness to communication. In *Proc. MobileHCI'15*, pages 6–15. ACM.

[22] Chorley, M. J., Colombo, G. B., Allen, S. M., and Whitaker, R. M. (2015a). Human content filtering in twitter: The influence of metadata. *International Journal of Human-Computer Studies*, 74:32–40.

[23] Chorley, M. J., Whitaker, R. M., and Allen, S. M. (2015b). Personality and location-based social networks. *Computers in Human Behavior*, 46:45–56.

[24] Choy, M., Kim, D., Lee, J.-G., Kim, H., and Motoda, H. (2016). Looking back on the current day: interruptibility prediction using daily behavioral features. In *Proc. UbiComp'16*, pages 1004–1015. ACM.

[25] Cockburn, J. (1995). Task interruption in prospective memory: A frontal lobe function? *Cortex*, 31(1):87–97.

[26] Coe, R. (2002). It's the effect size, stupid: What effect size is and why it is important. Presented at the Annual Conference of the British Educational Research Association.

[27] Corbetta, M. and Shulman, G. L. (2002). Control of goal-directed and stimulus-driven attention in the brain. *Nature reviews neuroscience*, 3(3):201–215.

[28] Crider, M. (2016). 21 new and notable android apps from the last 2 weeks (5/17/16 - 5/30/16). `http://www.androidpolice.com/2016/05/30/21-new-and-notable-android-apps-and-live-wallpapers-from-the-last-2-weeks-51716-53016/`.

[29] Fischer, J. E., Yee, N., Bellotti, V., Good, N., Benford, S., and Greenhalgh, C. (2010). Effects of content and time of delivery on receptivity to mobile interruptions. In *Proc. MobileHCI'10*, pages 103–112. ACM.

[30] Fisher, R. and Simmons, R. (2011). Smartphone interruptibility using density-weighted uncertainty sampling with reinforcement learning. In *ICMLA'11*, volume 1, pages 436–441. IEEE.

[31] Fogarty, J., Hudson, S. E., Atkeson, C. G., Avrahami, D., Forlizzi, J., Kiesler, S., Lee, J. C., and Yang, J. (2005a). Predicting human interruptibility with sensors. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 12(1):119–146.

[32] Fogarty, J., Hudson, S. E., and Lai, J. (2004). Examining the robustness of sensor-based statistical models of human interruptibility. In *Proc. CHI'04*, pages 207–214. ACM.

[33] Fogarty, J., Ko, A. J., Aung, H. H., Golden, E., Tang, K. P., and Hudson, S. E. (2005b). Examining task engagement in sensor-based statistical models of human interruptibility. In *Proc. CHI'05*, pages 331–340. ACM.

[34] Goldberg, J. A. (1990). Interrupting the discourse on interruptions: An analysis in terms of relationally neutral, power-and rapport-oriented acts. *Journal of Pragmatics*, 14(6):883–903.

[35] Grabham, D. (2014). Researchers solve to-do list overload with imprompdo app. `http://www.lifehacker.co.uk/2014/08/07/researchers-solve-list-overload-imprompdo-app`.

[36] Grandhi, S. and Jones, Q. (2010). Technology-mediated interruption management. *International Journal of Human-Computer Studies*, 68(5):288–306.

[37] Gupta, A., Li, H., and Sharda, R. (2013). Should i send this message? understanding the impact of interruptions, social hierarchy and perceived task complexity on user performance and perceived workload. *Decision Support Systems*, 55(1):135–145.

[38] Gwizdka, J. (2004). Email task management styles: the cleaners and the keepers. In *CHI'04 extended abstracts on human factors in computing systems*, pages 1235–1238. ACM.

[39] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18.

[40] Harr, R. and Kaptelinin, V. (2012). Interrupting or not: exploring the effect of social context on interrupters' decision making. In *Proc. NordiCHI'12*, pages 707–710. ACM.

[41] He, H. and Garcia, E. A. (2009). Learning from imbalanced data. *Knowledge and Data Engineering, IEEE Transactions on*, 21(9):1263–1284.

[42] Ho, C.-Y., Nikolic, M. I., Waters, M. J., and Sarter, N. B. (2004). Not now! supporting interruption management by indicating the modality and urgency of pending tasks. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 46(3):399–409.

[43] Ho, J. and Intille, S. S. (2005). Using context-aware computing to reduce the perceived burden of interruptions from mobile devices. In *Proc. CHI'05*, pages 909–918. ACM.

[44] Horvitz, E. and Apacible, J. (2003). Learning and reasoning about interruption. In *Proc. ICMI'03*, pages 20–27. ACM.

[45] Horvitz, E., Apacible, J., and Subramani, M. (2005). Balancing awareness and interruption: Investigation of notification deferral policies. In *User Modeling 2005*, pages 433–437. Springer.

[46] Horvitz, E., Koch, P., and Apacible, J. (2004). Busybody: creating and fielding personalized models of the cost of interruption. In *Proc. CSCW'04*, pages 507–510. ACM.

[47] Hudson, J. M., Christensen, J., Kellogg, W. A., and Erickson, T. (2002). I'd be overwhelmed, but it's just one more thing to do: Availability and interruption in research management. In *Proc. CHI'02*, pages 97–104. ACM.

[48] Hudson, S., Fogarty, J., Atkeson, C., Avrahami, D., Forlizzi, J., Kiesler, S., Lee, J., and Yang, J. (2003). Predicting human interruptibility with sensors: a wizard of oz feasibility study. In *Proc. CHI'03*, pages 257–264. ACM.

[49] Hutchby, I. (2008). Participants' orientations to interruptions, rudeness and other impolite acts in talk-in-interaction. *Journal of Politeness Research. Language, Behaviour, Culture*, 4(2):221–241.

[50] Iqbal, S. T. and Bailey, B. P. (2006). Leveraging characteristics of task structure to predict the cost of interruption. In *Proc. CHI'06*, pages 741–750. ACM.

[51] Iqbal, S. T. and Bailey, B. P. (2008). Effects of intelligent notification management on users and their tasks. In *Proc. CHI'08*, pages 93–102. ACM.

[52] Iqbal, S. T. and Bailey, B. P. (2010). Oasis: A framework for linking notification delivery to the perceptual structure of goal-directed tasks. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 17(4):15.

[53] Iqbal, S. T. and Horvitz, E. (2010). Notifications and awareness: a field study of alert usage and preferences. In *Proc. CSCW'10*, pages 27–30. ACM.

[54] Jin, J. and Dabbish, L. A. (2009). Self-interruption on the computer: a typology of discretionary task interleaving. In *Proc. CHI'09*, pages 1799–1808. ACM.

[55] Jones, S. A., Gould, S. J., and Cox, A. L. (2012). Snookered by an interruption?: use a cue. In *Proc. BCS-HCI'12*. British Computer Society.

[56] Katidioti, I., Borst, J. P., Bierens de Haan, D. J., Pepping, T., van Vugt, M. K., and Taatgen, N. A. (2016). Interrupted by your pupil: An interruption management system based on pupil dilation. *International Journal of Human–Computer Interaction*, 32(10):791–801.

[57] Kern, N., Antifakos, S., Schiele, B., and Schwaninger, A. (2004). A model for human interruptability: experimental evaluation and automatic estimation from wearable sensors. In *Proc. ISWC'04*, volume 1, pages 158–165. IEEE.

[58] Kern, N. and Schiele, B. (2006). Towards personalized mobile interruptibility estimation. In *Location-and Context-Awareness*, pages 134–150. Springer.

[59] Kim, S., Chun, J., and Dey, A. K. (2015). Sensors know when to interrupt you in the car: Detecting driver interruptibility through monitoring of peripheral interactions. In *Proc. CHI'15*, pages 487–496. ACM.

[60] Ko, M., Choi, S., Yatani, K., and Lee, U. (2016). Lock n'lol: Group-based limiting assistance app to mitigate smartphone distractions in group activities. In *Proc. CHI'16*, pages 998–1010. ACM.

[61] Kobayashi, Y., Tanaka, T., Aoki, K., and Fujita, K. (2015). Automatic delivery timing control of incoming email based on user interruptibility. In *Proc. CHI'15 (Extended Abstracts)*, pages 1779–1784. ACM.

[62] Kushlev, K., Proulx, J., and Dunn, E. W. (2016). Silence your phones: Smartphone notifications increase inattention and hyperactivity symptoms. In *Proc. CHI'16*, pages 1011–1020. ACM.

[63] Lane, N. D., Miluzzo, E., Lu, H., Peebles, D., Choudhury, T., and Campbell, A. T. (2010). A survey of mobile phone sensing. *Communications Magazine, IEEE*, 48(9):140–150.

[64] Lathia, N., Rachuri, K. K., Mascolo, C., and Rentfrow, P. J. (2013). Contextual dissonance: Design bias in sensor-based experience sampling methods. In *Proc. UbiComp'13*, pages 183–192. ACM.

[65] Leiva, L., Böhmer, M., Gehring, S., and Krüger, A. (2012). Back to the app: the costs of mobile application interruptions. In *Proc. MobileHCI'12*, pages 291–294. ACM.

[66] LiKamWa, R., Liu, Y., Lane, N. D., and Zhong, L. (2013). Moodscope: Building a mood sensor from smartphone usage patterns. In *Proc. MobiSys'13*, pages 389–402. ACM.

[67] Lopez-Tovar, H., Charalambous, A., and Dowell, J. (2015). Managing smartphone interruptions through adaptive modes and modulation of notifications. In *Proc. IUI'15*, pages 296–299. ACM.

[68] Loughrey, J. and Cunningham, P. (2005). Overfitting in wrapper-based feature subset selection: The harder you try the worse it gets. In *Research and Development in Intelligent Systems XXI*, pages 33–43. Springer.

[69] Malle, B. F. and Knobe, J. (1997). The folk concept of intentionality. *Journal of Experimental Social Psychology*, 33(2):101–121.

[70] Mark, G., Iqbal, S. T., Czerwinski, M., Johns, P., Sano, A., and Lutchyn, Y. (2016). Email duration, batching and self-interruption: Patterns of email use on productivity and stress. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 1717–1728. ACM.

[71] Mathan, S., Whitlow, S., Dorneich, M., Ververs, P., and Davis, G. (2007). Neurophysiological estimation of interruptibility: Demonstrating feasibility in a field context. In *Proc. AugCog'07*, pages 51–58.

[72] Mathur, A., Lane, N. D., and Kawsar, F. (2016). Engagement-aware computing: modelling user engagement from mobile contexts. In *Proc. UbiComp'16*, pages 622–633. ACM.

[73] McFarlane, D. (1997). Interruption of people in human-computer interaction: A general unifying definition of human interruption and taxonomy. Technical report, DTIC Document.

[74] McFarlane, D. (2002). Comparison of four primary methods for coordinating the interruption of people in human-computer interaction. *Human-Computer Interaction*, 17(1):63–139.

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

[76] Mehrotra, A., Musolesi, M., Hendley, R., and Pejovic, V. (2015a). Designing content-driven intelligent notification mechanisms for mobile applications. In *Proc. UbiComp'15*, pages 813–824. ACM.

[77] Mehrotra, A., Pejovic, V., Vermeulen, J., Hendley, R., and Musolesi, M. (2016). My phone and me: Understanding people's receptivity to mobile notifications. In *Proc. CHI'16*, pages 1021–1032. ACM.

[78] Mehrotra, A., Vermeulen, J., Pejovic, V., and Musolesi, M. (2015b). Ask, but don't interrupt: the case for interruptibility-aware mobile experience sampling. In *Proc. UbiComp'15 (Adjunct)*, pages 723–732. ACM.

[79] Miller, G. (2012). The smartphone psychology manifesto. *Perspectives on Psychological Science*, 7(3):221–237.

[80] Monk, C. A., Boehm-Davis, D. A., and Trafton, J. G. (2002). The attentional costs of interrupting task performance at various stages. In *Proc. HFES'02*, volume 46, pages 1824–1828. SAGE Publications.

[81] Moran, S. and Fischer, J. E. (2013). Designing notifications for ubiquitous monitoring systems. In *Proc. PerCom'13 (PERCOM Workshops)*, pages 115–120. IEEE.

[82] Mühlenbrock, M., Brdiczka, O., Snowdon, D., and Meunier, J.-L. (2004). Learning to detect user activity and availability from a variety of sensor data. In *Proc. PerCom'04*, pages 13–13. IEEE Computer Society.

[83] Obuchi, M., Sasaki, W., Okoshi, T., Nakazawa, J., and Tokuda, H. (2016). Investigating interruptibility at activity breakpoints using smartphone activity recognition api. In *Proc. UbiComp'16 (Adjunct)*, pages 1602–1607. ACM.

[84] Oh, H., Jalali, L., and Jain, R. (2015). An intelligent notification system using context from real-time personal activity monitoring. In *Multimedia and Expo (ICME), 2015 IEEE International Conference on*, pages 1–6. IEEE.

[85] Okoshi, T., Nakazawa, J., and Tokuda, H. (2016a). Interruptibility research: opportunities for future flourishment. In *Proc. UbiComp'16 (Adjunct)*, pages 1524–1529. ACM.

[86] Okoshi, T., Nozaki, H., Nakazawa, J., Tokuda, H., Ramos, J., and Dey, A. K. (2016b). Towards attention-aware adaptive notification on smart phones. *Pervasive and Mobile Computing*, 26:17–34.

[87] Okoshi, T., Ramos, J., Nozaki, H., Nakazawa, J., Dey, A., and Tokuda, H. (2015a). Attelia: Reducing user's cognitive load due to interruptive notifications on smart phones. In *Proc. PerCom'15*. IEEE.

[88] Okoshi, T., Ramos, J., Nozaki, H., Nakazawa, J., Dey, A. K., and Tokuda, H. (2015b). Attelia: Reducing user's cognitive load due to interruptive notifications on smart phones. In *Proc. PerCom'15*, pages 96–104. IEEE.

[89] Okoshi, T., Ramos, J., Nozaki, H., Nakazawa, J., Dey, A. K., and Tokuda, H. (2015c). Reducing users' perceived mental effort due to interruptive notifications in multi-device mobile environments. In *Proc. UbiComp'15*, pages 475–486. ACM.

[90] Oulasvirta, A., Rattenbury, T., Ma, L., and Raita, E. (2012). Habits make smartphone use more pervasive. *Personal and Ubiquitous Computing*, 16(1):105–114.

[91] Patil, S., Hoyle, R., Schlegel, R., Kapadia, A., and Lee, A. J. (2015). Interrupt now or inform later?: Comparing immediate and delayed privacy feedback. In *Proc. CHI'15*, pages 1415–1418. ACM.

[92] Pejovic, V. and Musolesi, M. (2014). Interruptme: designing intelligent prompting mechanisms for pervasive applications. In *Proc. UbiComp'14*, pages 897–908. ACM.

[93] Pejovic, V. and Musolesi, M. (2015). Anticipatory mobile computing: A survey of the state of the art and research challenges. *ACM Computing Surveys*, 47(3):47:1–47:29.

[94] Pejovic, V., Musolesi, M., and Mehrotra, A. (2015). Investigating the role of task engagement in mobile interruptibility. In *Proc. MobileHCI'15 (Adjunct)*, pages 1100–1105. ACM.

[95] Petersen, S. A., Cassens, J., Kofod-Petersen, A., and Divitini, M. (2008). To be or not to be aware: Reducing interruptions in pervasive awareness systems. In *Proc. UBICOMM'08*, pages 327–332. IEEE.

[96] Pielot, M., Church, K., and de Oliveira, R. (2014a). An in-situ study of mobile phone notifications. In *Proc. MobileHCI'14*, pages 233–242. ACM.

[97] Pielot, M., de Oliveira, R., Kwak, H., and Oliver, N. (2014b). Didn't you see my message?: predicting attentiveness to mobile instant messages. In *Proc. CHI'14*, pages 3319–3328. ACM.

[98] Pielot, M., Dingler, T., Pedro, J. S., and Oliver, N. (2015). When attention is not scarce - detecting boredom from mobile phone usage. In *Proc. UbiComp'15*, pages 825–836. ACM.

[99] Poppinga, B., Heuten, W., and Boll, S. (2014). Sensor-based identification of opportune moments for triggering notifications. *Pervasive Computing, IEEE*, 13(1):22–29.

[100] Rivera, A. J. (2014). A socio-technical systems approach to studying interruptions: Understanding the interrupter's perspective. *Applied ergonomics*, 45(3):747–756.

[101] Roda, C. and Thomas, J. (2006). Attention aware systems: Theories, applications, and research agenda. *Computers in Human Behavior*, 22(4):557–587.

[102] Rosenthal, S., Dey, A. K., and Veloso, M. (2011). Using decision-theoretic experience sampling to build personalized mobile phone interruption models. In *Pervasive Computing*, pages 170–187. Springer.

[103] Sahami Shirazi, A., Henze, N., Dingler, T., Pielot, M., Weber, D., and Schmidt, A. (2014). Large-scale assessment of mobile notifications. In *Proc. CHI'14*, pages 3055–3064. ACM.

[104] Salvucci, D. D., Taatgen, N. A., and Borst, J. P. (2009). Toward a unified theory of the multitasking continuum: from concurrent performance to task switching, interruption, and resumption. In *Proc. CHI'09*, pages 1819–1828. ACM.

[105] Sarker, H., Sharmin, M., Ali, A. A., Rahman, M. M., Bari, R., Hossain, S. M., and Kumar, S. (2014). Assessing the availability of users to engage in just-in-time intervention in the natural environment. In *Proc. UbiComp'14*, pages 909–920. ACM.

[106] Sarker, I. H., Kabir, M. A., Colman, A., and Han, J. (2016). Predicting how you respond to phone calls: towards discovering temporal behavioral rules. In *Proc. OzCHI'16*, pages 421–425. ACM.

[107] Sarter, N. (2013). Multimodal support for interruption management: Models, empirical findings, and design recommendations. *Proceedings of the IEEE*, 101(9):2105–2112.

[108] Schulze, F. and Groh, G. (2016). Conversational context helps improve mobile notification management. In *Proc. MobileHCI'16*, pages 518–528. ACM.

[109] Scollon, C. N., Prieto, C.-K., and Diener, E. (2003). Experience sampling: promises and pitfalls, strengths and weaknesses. *Journal of Happiness Studies*, 4:5–34.

[110] Shin, C. and Dey, A. K. (2013). Automatically detecting problematic use of smartphones. In *Proc. UbiComp'13*, pages 335–344. ACM.

[111] Smith, J., Lavygina, A., Ma, J., Russo, A., and Dulay, N. (2014). Learning to recognise disruptive smartphone notifications. In *Proc. MobileHCI'14*, pages 121–124. ACM.

[112] Srinivas, P., Faiola, A., and Mark, G. (2016). Designing guidelines for mobile health technology: managing notification interruptions in the icu. In *Proc. CHI'16*, pages 4502–4508. ACM.

[113] Stern, H., Pammer, V., and Lindstaedt, S. N. (2011). A preliminary study on interruptibility detection based on location and calendar information. *Proc. CoSDEO'11*.

[114] Stothart, C., Mitchum, A., and Yehnert, C. (2015). The attentional cost of receiving a cell phone notification. *Journal of experimental psychology: human perception and performance*, 41(4):893.

[115] Sykes, E. R. (2014). A cloud-based interaction management system architecture for mobile devices. *Procedia Computer Science*, 34:625–632.

[116] Tanaka, T., Abe, R., Aoki, K., and Fujita, K. (2015). Interruptibility estimation based on head motion and pc operation. *International Journal of Human-Computer Interaction*, 31(3):167–179.

[117] Tanaka, T. and Fujita, K. (2011a). Interaction mediate agent based on user interruptibility estimation. In *Symposium on Human Interface*, pages 152–160. Springer.

[118] Tanaka, T. and Fujita, K. (2011b). Study of user interruptibility estimation based on focused application switching. In *Proc. CSCW'11*, pages 721–724. ACM.

[119] Ter Hofte, G. H. (2007). Xensible interruptions from your mobile phone. In *Proc. MobileHCI'07*, pages 178–181. ACM.

[120] Turner, L. D. (2014). "is this a good time" - how imprompdo can tell when you're busy. `https://www.software.ac.uk/blog/2016-09-28-good-time-how-imprompdo-can-tell-when-youre-busy`.

[121] Turner, L. D., Allen, S. M., and Whitaker, R. M. (2015a). Interruptibility prediction for ubiquitous systems: Conventions and new directions from a growing field. In *Proc. UbiComp'15*, pages 801–812. ACM.

[122] Turner, L. D., Allen, S. M., and Whitaker, R. M. (2015b). Push or delay? decomposing smartphone notification response behaviour. In *Human Behavior Understanding*, volume 9277 of *Lecture Notes in Computer Science*, pages 69–83. Springer International Publishing.

[123] Turner, L. D., Allen, S. M., and Whitaker, R. M. (2017a). Behaviour patterns in managing stacks of mobile notifications.

[124] Turner, L. D., Allen, S. M., and Whitaker, R. M. (2017b). Reachable but not receptive: Enhancing smartphone interruptibility prediction by modelling the extent of user engagement with notifications. *Pervasive and Mobile Computing*.

[125] Webberley, W. M., Allen, S. M., and Whitaker, R. M. (2016). Retweeting beyond expectation: Inferring interestingness in twitter. *Computer Communications*, 73:229–235.

[126] Whiten, A. (1991). *Natural theories of mind: Evolution, development and simulation of everyday mindreading*. Basil Blackwell Oxford.

[127] Whitwam, R. (2016). Google play app roundup: Boomerang notifications, tiny tower, and crashing season. `http://www.tested.com/tech/android/572624-google-play-app-roundup-boomerang-notifications-tiny-tower-and-crashing-season/`.

[128] Yuan, F., Gao, X., and Lindqvist, J. (2017). How busy are you?: Predicting the interruptibility intensity of mobile users. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 5346–5360. ACM.

[129] Züger, M. and Fritz, T. (2015). Interruptibility of software developers and its prediction using psycho-physiological sensors. In *Proc. CHI'15*, pages 2981–2990. ACM.

[130] Zulkernain, S., Madiraju, P., Ahamed, S. I., and Stamm, K. (2010). A mobile intelligent interruption management system. *Journal of Universal Computer Science*, 16(15):2060–2080.

Appendices

Appendix A

ImprompDo App Design & Dataset

Complete response behaviour	**Exit early behaviour**	**Information gained after the continue behaviour**
The user is interrupted and chooses to begin responding.	The user is not physically interrupted or is not interruptible enough to switch focus towards the notification.	By default, knowledge that a notification has arrived from an unknown source. Specific applications may use a potentially recognisable audible tone, vibration pattern, or LED pattern.
The user turns the screen on.	The user does not interact with the device and continues with their current activity.	The source application that caused the notification (e.g. an email has arrived).
The user unlocks the device.	The user turns the screen off and resumes their previous activity.	The notification summary from accessing the Notification Drawer once unlocked (e.g. the email sender and subject).
The user taps on the notification.	The user dismisses the notification or ignores it resumes their previous activity.	The full notification content (e.g. the email application opens and shows the full email).

Table A.1: The observable interruption and response process to Android notifications for versions up to and including v4.4, when the device is not-in-use at the time the notification is delivered.

Complete response behaviour	Exit early behaviour	Information gained after the continue behaviour
The user is interrupted and chooses to begin responding.	The user is not physically interrupted or is not interruptible enough to switch focus towards the notification.	By default, knowledge that a notification from a specific application (e.g. an email has arrived). Then the notification summary from accessing the Notification Drawer (e.g. the email sender and subject). Specific applications may also use a potentially recognisable audible tone, vibration pattern, or LED pattern.
The user taps on the notification.	The user dismisses the notification or ignores it resumes their previous activity.	The full notification content (e.g. the email application opens and shows the full email).

Table A.2: The observable interruption and response process to Android notifications for versions up and including v4.4, when the device is in-use at the time the notification is delivered.

Trigger	Description
Random (RND)	A random interval at millisecond granularity between now and the next time block. Used as a control group.
End of Acceleration (EOA)	Check for acceleration above a noise threshold every minute until acceleration is detected, then trigger at the point of deceleration under that threshold. The threshold begins as .1 m/s/s, increments by .01 if the user responds and decrements by .01 if no acceleration was detected within an hour.
Temporal Hourly Learning Model (THL)	Each hour has a probability value between 0..1, initially .5. If a notification expires this decrements by .1, consuming increments by .1. The value then determines if the prompt will occur at a random time
Multi-modal Online Learning Model (MML)	A trained Binary Logistic Regression model from data in the past week retrained daily. At minute intervals, a feature vector is created from the first readings and tested against the model. It included: orientation (the axes in which gravity is acting on the most); if the device is accelerating; if the screen is covered; luminescence (raw lux); lock state (screen off, locked or unlocked); volume preference (silent, vibrate or audible); battery charge state and level; current hour, and the weekday. This is then labelled depending on whether the user ignored the interaction (0) or consumed the notification (1); if dismissed, the vector was not included for training.

Table A.3: The randomly chosen triggers used.

Description	Number of notifications (n)
Notifications with a complete data trace	10,059
Notifications with missing data	1287
Reason: unknown if device was in-use	1267
Reason: unknown response behaviour	20

Table A.4: Data completeness in notification responses. This is discussed further in analysis comparing the DOIG model vs typical black-box approaches to labelling in Chapter 3, Section 3.4.

	Expired	Dismissed	Consumed	
Not-in-use	6741	324[a]	672[b]	n=7737
In-use	1747	191[c]	384[d]	n=2322
	n=8488	n=515	n=1056	

Table A.5: Frequency statistics of whether notifications were consumed (tapped on), dismissed, or expired, split between whether the device was in-use or not at the time the notification was delivered. Superscript characters are used for cross-referencing values within the table and with Tables A.6, A.7 and A.8.

Description	Calculation	Result
Notifications removed by user	sum(a,b,c,d)	1571
Notifications removed by user not-in-use	sum(a,b)	996
Notifications removed by user in-use	sum(c,d)	575

Table A.6: Frequency statistics on the number of notifications removed by the user by various means. This is used in the analysis of user response time to notifications in Chapter 3, Section 3.4.1. Superscript characters are used for cross-referencing values within the table and with Tables A.5, A.7 and A.8.

	(At least) Reachable	Not Reachable	
Not-in-use	1798[e]	5939	n=7737

	(At least) Engageable	Not Engageable	
Not-in-use	1469[f]	329	n=1798[e]

	Receptive	Not Receptive	
Not-in-use	672[b]	797	n=1469[f]
In-use	384[d]	1938	n=2322

Table A.7: Frequency statistics of user response behaviour. Only responses that were at least reachable are analysed for engage-ability, likewise only responses where the user was engageable are considered for receptivity. Dismissals are considered "Not Receptive" in this representation. Superscript characters are used for cross-referencing values within the table and with Tables A.5, A.6 and A.8.

Description	Calculation	Result
Interactions considered by a typical black-box model	sum(b,d)	1056
Additional interactions considered by the DOIG model (responses where the user started to respond (reachable) but did not consume the notification by tapping on it)	e - sum(b,c)	1317
Interactions considered by a typical black-box model (including notification dismissals)	sum(a,b,c,d)	1571
Additional interactions considered by the DOIG model (responses where the user started to respond (reachable) but did not consume or dismiss the notification)	e - sum(a,b)	802

Table A.8: Calculations used to compare how many additional responses the DOIG model captures in comparison to typical black-box approaches for labelling interruptibility. This is discussed in Chapter 3, Section 3.4. Superscript characters are used for cross-referencing values within the table and with Tables A.5, A.6 and A.7.

Android	n	grv	prx	prs	lin	rtv	gyr	mag	lgt
4.1.x	2	False	False	False	False	False	False	False	False
4.2.x	3	False	**True**	False	False	False	False	False	False
4.3.x	8	False	**True**	False	False	False	False	False	False
4.4.x	79	False	False	False	False	False	False	False	False

Table A.9: For each version of Android, whether the data vectors were consistent in either always, sometimes, or never containing sensor data. grv=Gravity, prx=Proxmity, prs=Pressure, lin=Linear Acceleration, rtv=Rotation Vector, gyr=Gyroscope, mag=Magnetic Field, lgt=Light.

Android	Sensor	Always present	Sometimes present	Never present
4.2.x	prx	False	**True**	False
4.3.x	prx	False	**True**	False

Table A.10: For those versions of Android that were consistent in some way in Table A.9, whether they were either always, sometimes, or never consistent. prx=Proximity.

Device	n	grv	prx	prs	lin	rtv	gyr	mag	lgt
GT-I9505	5	False	False	False	False	False	False	False	False
Nexus 4	4	False	**True**	False	False	False	**True**	False	**True**
Nexus 5	19	False	False	False	False	False	False	False	False
GT-N7100	4	**True**	**True**	**True**	**True**	**True**	False	False	False
SM-G900F	2	False	False	False	False	False	False	False	False
SM-N9005	5	False	False	False	False	False	False	False	False
HTC One	4	**True**	**True**	**True**	**True**	False	**True**	**True**	**True**
HTC One_M8	2	**True**	False	**True**	**True**	**True**	**True**	**True**	False
XT1032	3	**True**	False	**True**	**True**	**True**	**True**	**True**	**True**
Nexus 7	3	**True**	**True**	**True**	**True**	False	**True**	**True**	**True**
GT-I9300	8	**True**	**True**	**True**	**True**	**True**	**True**	**True**	**True**
Nexus 10	3	**True**	**True**	**True**	False	False	**True**	False	**True**
MI 3W	2	False	**True**	**True**	False	False	**True**	False	False

Table A.11: For devices that were used by at least 2 users, whether the data vectors were consistent in either always, sometimes, or never containing sensor data. grv=Gravity, prx=Proxmity, prs=Pressure, lin=Linear Acceleration, rtv=Rotation Vector, gyr=Gyroscope, mag=Magnetic Field, lgt=Light.

Android	Sensor	Always present	Sometimes present	Never present
Nexus 4	prx	False	**True**	False
Nexus 4	gyr	False	**True**	False
Nexus 4	lgt	False	**True**	False
GT-N7100	grv	False	**True**	False
GT-N7100	prx	False	**True**	False
GT-N7100	prs	False	**True**	False
GT-N7100	lin	False	**True**	False
GT-N7100	rtv	False	**True**	False
HTC One	grv	**True**	False	False
HTC One	prx	False	**True**	False
HTC One	prs	False	False	**True**
HTC One	lin	**True**	False	False
HTC One	gyr	**True**	False	False
HTC One	mag	**True**	False	False
HTC One	lgt	**True**	False	False
HTC One_M8	grv	False	**True**	False
HTC One_M8	prs	**True**	False	False
HTC One_M8	lin	False	**True**	False
HTC One_M8	rtv	False	**True**	False
HTC One_M8	gyr	**True**	False	False
HTC One_M8	mag	**True**	False	False
XT1032	grv	False	False	**True**
XT1032	prs	False	False	**True**
XT1032	lin	False	False	**True**
XT1032	rtv	False	False	**True**
XT1032	gyr	False	False	**True**
XT1032	mag	**True**	False	False
XT1032	lgt	**True**	False	False
Nexus 7	grv	False	**True**	False
Nexus 7	prx	False	False	**True**
Nexus 7	prs	False	False	**True**
Nexus 7	lin	False	**True**	False
Nexus 7	gyr	**True**	False	False
Nexus 7	mag	False	**True**	False
Nexus 7	lgt	**True**	False	False
GT-I9300	grv	False	**True**	False
GT-I9300	prx	False	**True**	False
GT-I9300	prs	False	**True**	False
GT-I9300	lin	False	**True**	False
GT-I9300	rtv	False	**True**	False
GT-I9300	gyr	False	**True**	False
GT-I9300	mag	False	**True**	False

Android	Sensor	Always present	Sometimes present	Never present
GT-I9300	lgt	False	**True**	False
Nexus 10	grv	**True**	False	False
Nexus 10	prx	False	False	**True**
Nexus 10	prs	**True**	False	False
Nexus 10	gyr	**True**	False	False
Nexus 10	lgt	**True**	False	False
MI 3W	prx	False	**True**	False
MI 3W	prs	**True**	False	False
MI 3W	gyr	**True**	False	False

Table A.12: For those devices that were consistent in some way in Table A.11, whether they were either always, sometimes, or never consistent. grv=Gravity, prx=Proxmity, prs=Pressure, lin=Linear Acceleration, rtv=Rotation Vector, gyr=Gyroscope, mag=Magnetic Field, lgt=Light.

Feature	Values	Calculated
Volume State	Silent, Vibrate, Audible	Raw data
Screen Covered	False, True	Whether the proximity sensor reading v is within 5cm*, $v < 5$. *Some hardware only reports binary near/far using a typical 5cm threshold [5].
Ambient Light	Dark, Dim, Light, Bright	Transforming the raw lux value v using: $v < 5$ = Dark, $v < 50$ = Dim, $v < 500$ = Light, $v >= 500$ = Bright
Charging State	False, True	Raw data
Orientation	Flat, Upright, Other	Transforming the raw gravity (m/s^2) value v to it's absolute and using the axis with the highest value if the next highest $u < (v - .5)$. Otherwise multiple axes are considered together. For orientation o, if the z-axis is solely the highest, o = Flat, if y-axis, o = Upright, otherwise o = Other.
Accelerating	False, True	If the raw linear acceleration (m/s^2) value v for at least 1 axis is beyond a noise threshold $v > .1$ or $v < -.1$
Time of Day	Morning, Afternoon, Evening, Night	Transforming the timestamp as: Morning = 06-11, Afternoon = 12-16, Evening = 17-20, Night = 21-05
Day of the Week	Mon, Tues, Wed, Thur, Fri, Sat, Sun	From timestamp
Weekday or Weekend	Weekday, Weekend	From timestamp

Table A.13: Features extracted from the sensor/software API data traces.

Appendix B

Boomerang Notifications App Design

Setting	Description
Turn Boomerang Off	Directs the user to Android's Notification Access screen where the permission to receive notification events can be revoked (this is granted during the setup process of the application)
Show persistent notification summary?	If checked, shows a summary notification of how many notifications have been saved. This is only seen when the notification drawer is open. This is set to true by default.
Receive prompts to review saved notifications at the end of the day	If checked, a notification is posted by the application at the end of each day where at least one notification is saved. This is set to true by default.
Allow app removal from notifications	If checked, notifications which prompt the user to save notifications have a third button, if pressed, this removes the application from the user's application list to show save prompts for. This is set to false by default.
Applications to Boomerang	This directs the user to the interface where the user can select which applications to show save prompts for (shown in Chapter 6, Figure 6.5a).
Save notification active time	This setting produces a pop-up window where the user can set how many seconds the save prompt notifications are displayed for until they are removed by the application. This is set to 10 seconds by default.

Table B.1: Boomerang Notifications' user modifiable settings.

Setting	Description
Summary notification	A summary notification of how many notifications have been saved, which is only seen when the notification drawer is open. This notification is persistent as a result of Google's guidelines that applications running in the background should provide a notification indicating this.
Notification reminders	Notifications that remind the user about a previous notification they have saved. These occur at a time set by the user and contain a single button to remind the user again at a random time later that day (or tomorrow if close to midnight). These notifications use the device's default audible tone, vibrate pattern, and LED pattern.
Prompts to save notifications	Notifications that are shown for a short period (default 10 seconds) after a notification has been removed (shown in Figure 6.2b), but only if that removed notification originated from an application in the user's list of applications to show prompts for (shown in Chapter 6, Figure 6.5a. These notifications have no explicit interruptive properties and have no icon so are only visible inside the notrification drawer (or on the lock-screen if the users device preferences allow notifications to be shown on the lock screen).
End of day review notification	A notification with static text that prompts the user to review the notifications they have saved that day. This notification is pushed at a random time in the evening for days where at least 1 notification was saved.

Table B.2: Notifications produced by Boomerang Notifications.