Are You Killing Time? Predicting Smartphone Users' Time-killing Moments via Fusion of Smartphone Sensor Data and Screenshots

ANONYMOUS AUTHOR(S)*

Time-killing on smartphones has become a pervasive activity, and could be opportune for delivering content to their users. This research is believed to be the first attempt at time-killing detection, which leverages the fusion of phone-sensor and screenshot data. We collected nearly one million user-annotated screenshots from 36 Android users. Using this dataset, we built a deep-learning fusion model, which achieved a precision of 0.83 and an AUROC of 0.72. We further employed a two-stage clustering approach to separate users into four groups according to the patterns of their phone-usage behaviors, and then built a fusion model for each group. The performance of the four models, though diverse, yielded better average precision of 0.85 and AUROC of 0.76, and was superior to that of the general/unified model shared among all users. We investigated and discussed the features of the four time-killing behavior clusters that explain why the models' performance differ.

CCS Concepts: • **Human-centered computing → Smartphones**; **Ubiquitous and mobile computing systems and tools**.

Additional Key Words and Phrases: Time-killing; Screenshot; Deep Learning; Opportune Moment; Mobile Devices

ACM Reference Format:
Anonymous Author(s). 2018. Are You Killing Time? Predicting Smartphone Users' Time-killing Moments via Fusion of Smartphone Sensor Data and Screenshots. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 25 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Researchers have leveraged smartphones' capabilities to engage individuals in a variety of tasks, including mobile learning exercises [11], just-in-time interventions [17], mobile self-reports [58], and crowdsourcing tasks [16]. In recent years, commercial platforms have also started doing so to obtain crowdsourced data, such as locale information[1] [3, 82] and labeled data[2] [15, 16]. However, given human beings' limited attentional resources, a crucial problem for anyone delivering content to phones is how to make it stand out from the feast of other incoming information. One mainstream approach to achieving this is to predict moments at which users are receptive to such content, e.g., the content related to notifications [55, 62, 65], questionnaires [62], and reading material [19, 62] explored in prior studies.

Moments of "attention surplus" [64] constitute another opportunity for such detection attempts. Pielot et al. [64], for example, attempted to detect one kind of "attention surplus" state – boredom – but reported that it was very challenging to achieve high performance in both recall and precision. One reason for these reported difficulties may be that phone-checking had become a pervasive and habitual behavior [18], thus making it hard to distinguish between the checking due to attention surplus and the checking for specific purposes. Another reason may be that boredom is unobservable by phone sensors. Beyond boredom, however, research has shown that mobile-phone use is not always

[1] https://maps.google.com/localguides
[2] https://play.google.com/store/apps/details?id=com.google.android.apps.village.boond

Manuscript submitted to ACM

associated with a purpose [27], but is often engaged in habitually simply to pass the time [49, 57]. In other words, a considerable proportion of phone usage is either accompanied by, or is primarily, "**time-killing**" behavior: i.e., filling periods that are perceived as free and/or boring [10, 27, 64], such as while waiting for a train to arrive at its destination, or attending an uninteresting speech [35]. In such situations, some people tend to seek stimulation on their phones to alleviate boredom, to achieve a sense of having escaped, or just to pass the time. Therefore, it is logical to assume that during these time-killing moments, individuals will be more receptive than usual to content that researchers, platforms, and others send to their phones.

In light of the above-mentioned challenges, coupled with the compound nature of "attention surplus" itself, we propose to detect time-killing moments, considered as behavioral outcomes of attention surplus, whose patterns may be observable from users' phone activities. Also, given the known difficulty of detecting attention surplus using phone-sensor data alone, our approach to time-killing detection leveraged screenshot data, which we expected would reveal rich temporal, textual, graphical, and topical information about people's phone usage [8].

Accordingly, we developed an Android research application that automatically collected smartphone screenshots and phone-sensor data, and an interface that allowed its users to efficiently annotate time-killing moments on the screenshots. Data collection with 36 participants over 14 days yielded a dataset of 967,466 pairings of annotated phone-sensor data with screenshots, covering 1,343.7 hours of phone usage. Using this dataset, we built a deep-learning-based fusion model that achieved a precision of 0.83 and an Area Under the Receiver Operating Characteristics (AUROC) of 0.71. To further improve the model's performance by taking account of differences in the participants' time-killing behaviors, we employed two-stage clustering that grouped people with similar phone usage behaviors into four groups, and built a fusion model for each group. The four resulting models' collective average precision and AUROC went up to 0.85 and 0.76, respectively: i.e., better than those of the general model (i.e., the one shared among all users). However, the four models achieved quite different performance on many metrics, and to obtain insights into these differences, we delved into the characteristics of each user group's phone-usage behavior as well as the important features learned by their respective models that were positively and negatively correlated with time-killing moments. The results of that investigation help explain both how and why the effectiveness of sensor data and phone screenshots for detecting time-killing moments varied across user clusters.

This paper makes the following three major contributions to the literature on phone-usage behavior.

1. It presents the development of a deep-learning-based fusion model that detects smartphone users' time-killing moments with an AUROC of 0.71.
2. It demonstrates that building such models for user groups clustered according to their phone-usage behaviors can achieve better overall model performance, and that all group-specific models may achieve significantly better performance than the general model.
3. It shows how and why the effectiveness of sensor data and phone screenshots for detecting time-killing moments vary across different time-killing behavioral patterns.

2 RELATED WORK

2.1 Interruptibility, Breakpoint, and Opportune Moment Prediction

Many studies have employed machine-learning techniques to predict interruptible moments, breakpoints, and opportune moments. For instance, Pejovic et al. [60] achieved the predictions of mobile interruptibility with a precision of 0.72. Others have focused on predicting opportune moments for receiving calls and notifications. For example, Fisher et al. [24] built personalized models to predict such moments in the case of incoming cell-phone calls, and achieved an average accuracy above 0.96 (see also Smith et al. [73]); and Pielot et al. [63] applied machine-learning techniques to predict whether users would view an incoming message notification within the next few minutes or not.

Some studies have implemented notification-management systems to reduce interruptions. Mehrotra et al. [52], for instance, proposed a system based on machine-learning algorithms that automatically extracted rules for phone users' preferences about receiving notifications. A similar study by Visuri et al. [81] reported that 81.7% of phone-user interactions with alert dialogs could be accurately predicted based on user clusters.

Among the researchers seeking to identify opportune moments based on breakpoints, Ho et al. [29] detected postural and ambulatory activity transitions in real-time. Iqbal and Bailey [33] showed that scheduling notifications at breakpoints reduced both frustration and reaction times. Okoshi et al. [55], who also developed a breakpoint-detection system for mobile devices, showed that notifications delivered during breakpoints required 33% less cognitive load than those delivered randomly. Later, the same authors [56] showed that delaying notification delivery until an interruptible moment resulted in a significant reduction in user response time. Adamczyk et al. [1] divided breakpoints in tasks into two types, coarse and fine, and showed that delivering notifications at their predicted best points for interruptions consistently produced less annoyance, frustration, and time pressure. Adopting the same definition of breakpoint granularity, Iqbal et al. [32] applied it to statistical models that mapped interaction features to each breakpoint type, based on task-execution data and video footage. And Park et al. [59] used built-in sensors to detect social contexts, which in turn enabled them to identify four distinct types of breakpoints, all of which were deemed suitable for the delivery of deferred smartphone notifications.

Detecting moments when device users want to engage with content has also been a focus of considerable research effort. Sarker et al. [70], for example, sought to identify moments for delivering notifications that would result in maximum engagement. Similarly, Choi et al. [17] built a mobile intervention system for preventing prolonged sedentary behaviors, and showed that contextual factors and cognitive/physical states were good predictors of decision points. Turner et al. [78] decomposed notification interaction into three stages – reachability, engageability, and receptivity – and developed models for predicting when phone users reached each of them. Pielot et al. [62] built a model that predicted whether their participants would engage with different types of content they were offered, which achieved a success rate 66.6% higher than the baseline. A few other detection studies have been focused on notification recipients' attention. For example, Steil et al. [74] predicted whether people's primary attentional focus was on their handheld mobile devices, and proposed "attention forecasting", which is similar in spirit to user-intention prediction.

Another strand of research on attention prediction involves identifying "attention surplus" moments and timing the delivery of specific content and tasks accordingly. Such content and tasks have thus far included reading material [20, 62], learning material [11, 21, 31], interventions [17, 53, 71], questionnaires [28, 62], and crowdsourcing tasks [16], among others. For example, Pielot et al. [64] deemed moments of boredom to be moments of attention surplus, and detected them using phone logs: an approach that achieved 0.83 AUROC. However, they obtained a high number of false positives, which they felt would lead to user annoyance, and therefore tuned their model to strike an optimal balance between recall and precision. Based on boredom levels detected via phone-sensor data, Dingler et al. [21] delivered micro-learning reminders to language learners, and their results suggested the feasibility of identifying moments of boredom as mobile learning opportunities. Cai et al. [11] developed WaitSuite, which detects various types of moments when its users are waiting for something to happen, and delivers micro-learning tasks during them. Similarly, Inie and Lungu [31] detected when users were about to become unproductive due to visiting time-wasting websites, blocked such visits, and delivered learning exercises instead.

In this paper, we aim to predict time-killing moments, i.e., ones in which people do things to pass or fill time using their smartphones. Killing time, though conceptually similar to boredom, is nevertheless discernibly different from it. Specifically, boredom is an individual's psychological state, which is unobservable, and can exist within a task if that task is causing fatigue and/or is mundane or routine [37]. Killing time, on the other hand, is an explicit and observable behavior and is usually performed when people are bored or micro-waiting. As such, instead of detecting boredom – which can take place at any point, even in the middle of a person's primary task, when notification delivery may be inopportune – our aim is to detect moments at which a phone is being used explicitly to kill time [31], which are *ipso facto* opportune for content delivery.

2.2 Phone-usage Research

The prevalence and abundance of smartphone apps have drawn researchers' attention to identifying specific patterns of phone usage. One of the two main strands of such research focuses on such patterns as a source of insights into phone users' other behaviors, while the other uses computational approaches to distinguish them and then uses that data to predict specific forms of phone use.

Several studies have utilized self-report methods such as interviews and diaries. For instance, Palen et al. [58] investigated mobile usage via a voicemail diary study. However, because self-report methods are subject to recall biases [22, 25], quantitative analysis of phone-usage logs is becoming increasingly popular [23, 85, 87]. For example, Böhmer et al.'s [7] large-scale study based on logged application usage found that news applications were most popular in the morning; and that game-playing mostly occurred at night. Xu et al. [85] also found differential patterns by app type, e.g., that sports apps were more frequently used in the evening. Falaki et al. [23] distinguished between two broad types of intentional use activities-user/phone interaction, and app use-and found that strong diversity in users' behavior was linked to different purposes for using phones. Canneyt et al. [80] revealed how app-usage behavior was disrupted during major political, social, and sporting events. And Li et al. [47] studied the long-term evolution of mobile-app usage, and found that the diversity of app-category usage declined over time, whereas the diversity of the individual apps used increased.

Lukoff et al. [49] identified situations in which people felt a lack of meaning while using their phones, which prominently included passively browsing social media, consuming entertainment, and habitual use. They also discovered that some users did not always use their phones for a purpose, but rather, as micro-escapes from negative situations. Hiniker et al. [27] likewise reported "ritualistic" uses of phones, which tended to be habitual. Another habitual phone usage is "phubbing", i.e., the habit of snubbing someone in favour of a mobile phone. As Al-Saggaf et al. [5] have suggested, individuals engage in phubbing while they are experiencing negative emotions such as boredom, loneliness, and fear of missing out. In a different study, Al-Saggaf and colleagues [4] reported that trait boredom could predict phubbing frequency.

A growing body of work involves attempts to construct models of phone usage. Kostakos et al. [43], for instance, developed a Markov state transition model of smartphone screen use. Jesdabodi et al. [36] identified phone users' behavioral states, and showed that morning and evening routines were both mostly marked by communication and gaming activities. The same study also found that the usage of timer apps was less apparent on weekend mornings than on weekday mornings. Some other work has focused on understanding differences in usage features across distinct user clusters. Zhao et al. [89] studied app usage with a two-step clustering approach and revealed clusters of users including "night communicators", "evening learners", and "screen checkers", among others. Jones et al. [38], on the other hand, identified three clusters of users: "checkers", "waiters" and "responsives". And Katevas et al. [39], based on a combination of phone-use log data and experience-sampling method data, identified five types of mobile-phone use: “limited use”, “business use”, “power use”, “personality-induced problematic use”, and “externally induced problematic use”.

Finally, because log data are limited to system events like screen events and app states, some researchers have used screenshots and video recordings to study phone usage. For example, Brown et al. [9] combined screen-captures of iPhone use with recordings from wearable video cameras, and showed that video data illuminated various aspects of people’s interactions with their phones. Subsequently, Brown et al. [10] collected screen recordings of phone use and audio recordings of ambient talk, and identified various situations in which people engaged in phone usage with their "free" attention and another activity simultaneously, e.g., during television viewing. Another such situation was killing time. For example, they found users engaged in quick games or social-media checking while waiting for a friend to arrive or for an event to start. Reeves et al. [68] showed how screenshots could be used to unobtrusively collect valuable data on individuals’ digital life experience: e.g., switching among content categories and devices across a day. Later, Reeves et al. [8] explored how textual and graphical features changed during sessions. For instance, they measured aggregate-level trends in word count, and aggregate-level stability in image complexity throughout the day, and found that word and image velocity both decreased late at night. However, some of their participants interacted with more image-based content during the overnight hours.

Some other researchers have used deep-learning models trained on large amounts of Graphical User Interface (GUI) data to detect screenshots. For instance, Beltramelli’s [6] Pix2Code applies an end-to-end neural image captioning model to generate code from a single input image, with better than 0.77 accuracy across various platforms. Similarly, Chen et al. [14] utilized a CNN-RNN model to generate GUI skeletons from screenshots. Other work focused on locating UI elements on screens, such as by White et al. [84] , has used YOLOv2 [67] to automatically identify GUI widgets in screenshots. Chen et al. [13] built a gallery of large scale of GUI designs by applying a Faster RCNN model [69] ; and Zhang et al. [88] proposed an on-device model capable of detecting UI elements.

Unlike any the studies reviewed above, however, our work focuses on detecting time-killing moments using a fusion of phone-sensor and screenshot data. In the remaining of the paper, we present our methodology and results.

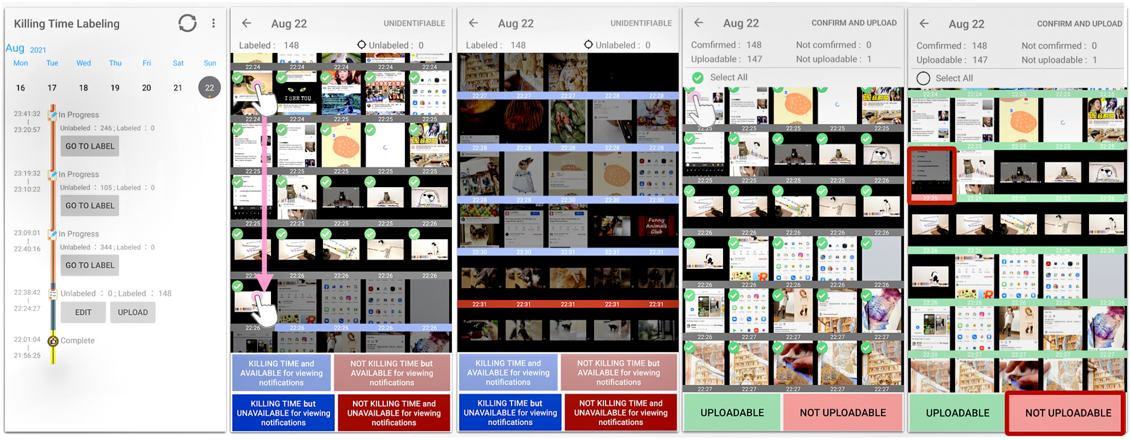
3 DATA COLLECTION

3.1 Input-data Selection

Screenshot collection has become a popular method in HCI research, because it allows researchers to collect quantitative and qualitative data simultaneously [40, 44, 45, 76] in high granularity and rich detail [8]. Along with information about people’s interactions with their phones, it can help researchers reconstruct both moment-to-moment phone use and wider usage patterns [51, 66, 68, 86]. Due to these advantages, we aimed to leverage screenshot data, along with phone-sensor interaction information (including user/phone interaction and phone status), to extract features that characterized our participants’ app usage and switching patterns. We then attempted to associate such usage information and patterns with time-killing vs. non-time-killing moments.

3.2 Research Instrument

We developed an Android research application, called Killing Time Labeling (KTL), to collect annotated screenshots and phone-sensor data (i.e., Android accessibility events, screen status, network connections, phone volume, application usage, and type of transportation). KTL also captures the notifications its users receive, the times at which they receive



(a) Main screen (b) Labeling screen (c) Screen after labeling (d) Upload selection (e) Screen after selection

Fig. 1. User interfaces for the main functions of the Killing Time Labeling application

them, and how they are dealt with. The background service that automatically collects data is activated within a 12-hour timeframe every day, the default being from 10:00 a.m. to 10:00 p.m., but the start time and end time are both user-adjustable, meaning that the data might be collected for more than 12 hours per day in some cases. During whatever 12+-hour window the user has chosen, his/her phone-sensor data is collected every five seconds. Screenshots are also captured every five seconds, but only when the phone screen is on.

We designed a user interface for KTL that allowed our participants to easily select groups of screenshots via drag-and-drop for data labeling (see Fig. 1). A detailed demonstration of this data-labeling procedure is provided in our supplemental video. The participants were instructed to review and annotate screenshots in accordance with the situations in which they were taken. For each screenshot, participants had five annotation options: 1) killing time and available for viewing notifications; 2) not killing time but available for viewing notifications; 3) killing time but unavailable for viewing notifications; 4) not killing time and unavailable for viewing notifications; and 5) unidentifiable, i.e., the participant could not be certain of his/her time-killing state or had forgotten it. Each time s/he manually selected and annotated a series of screenshots, the participant was to report his/her actual activities[3] at the time those screenshots were taken. We instructed the participants to annotate them as “killing time” as long as they felt that their mobile-phone usage at the time was to pass time, and otherwise to annotate it as "not killing time". Regarding the availability label for viewing notifications, we instructed them to annotate screenshots as “unavailable for viewing notifications” if they positively did not want to be interrupted or to see any notifications when using the app, and otherwise to annotate them as “available”. Because KTL invalidated screenshots after two days, meaning they could no longer be annotated, we also instructed the participants to complete their labeling before going to bed every day.

All screenshots were reduced in size and temporarily stored in the local storage of the participants’ respective phones before they were reviewed, labeled, and manually uploaded to our server. The participants had the right not to upload any given screenshot, e.g., because it contained sensitive information. Phone-sensor data, on the other hand, was automatically uploaded by KTL whenever a participant’s phone was connected to the Internet, to avoid such data taking up too much storage space. Also, to avoid impacting the participants’ data plans, KTL only did so via WiFi networks,

[3]This question was adopted from previous research [46].

unless a user overrode this feature and chose to upload using the cellular network. The participants were informed of all these rules in a pre-study meeting (the other purposes of which are detailed in section 3.3, below).

KTL also delivered notifications linked to experience sampling method (ESM) questionnaires and to various other types of content. That other content consisted of 1) crowdsourcing tasks[4] [15, 16], 2) non-ESM questionnaires[5] [62], 3) advertisements [62], and 4) news items [61, 62, 64]. KTL only sent such notifications within the user's chosen 12+-hour timeframe and only when his/her screen was on. Each notification was randomly selected from among the four types listed above, and delivered at random intervals of not less than one or more than three hours. Five minutes after each notification arrived, an ESM questionnaire was also sent, asking the participant to report his/her awareness of and receptivity to that notification, as well as what context s/he was in at the moment it had arrived.

3.3 Study Procedure

Prior to data collection, due to the COVID-19 pandemic, we allowed our participants to choose between remotely and physically attending a pre-study meeting, during which the researchers helped them install KTL on their phones, explained how to use it, and walked them through the study procedure. We told them that we expected them to annotate all screenshots that were automatically captured by KTL every day, and that 14 days of active participation were needed for their data to be useful to us. Thus, for each day they did not provide annotated screenshots, their participation was extended by one day. On their respective final days of participation, to aid future analysis, they completed four additional questionnaires that measured their boredom proneness [75], smartphone addiction [48], inattention [41], and perceived acceptability of time-killing detection being deployed on their phone. In addition, we invited all participants to two optional semi-structured interviews, the first of which was held after they had contributed data for seven full days, and the second, after their participation was complete. In those interviews, we asked them about their labeling processes, time-killing behaviors and preferences, and how they killed time (both typically and during the study). Those who completed 14 days of data collection were paid NT$1,350 (approximately US$44). Those who participated in the mid-study interview were paid an additional NT$150 (US$5), and those who were interviewed after the study, another NT$250 (US$8). The study was approved by our university's Institutional Review Board (IRB).

3.4 Recruitment and Participants

We selected participants with various occupations, in the expectation that they would have different time-killing patterns. Also, to ensure that sufficient data were collected, we selected participants who used their mobile phones more than one hour a day, according to their self-reporting in a screening questionnaire. We recruited participants primarily via several Facebook groups aimed at matching researchers with study participants in our country, but also posted a recruiting message on Facebook pages for the local community in the hope of further diversifying our subjects' backgrounds. Through this process, a total of 55 participants were recruited, including 12 who participated in a pilot study. Of the remaining 43 participants, one withdrew before data collection commenced, two did not complete the experiment, and four others were excluded as being outliers (i.e., they had annotated more than 95% of their data as "killing time"). As a result, data from 36 people were used for training our time-killing detection model. Of those 36, 32 took part in both optional interviews, two only in the mid-study interview, and two others, only in the post-study

[4]The crowdsourcing questions were inspired by Google Crowdsource and Local Guide, two platforms that aim to improve Google Maps and various other Google services through user-oriented training of multiple algorithms.
[5]The questionnaire was inspired by Google Opinion Rewards, which offers rewards to its users who answer surveys and opinion polls on a variety of topics.

Table 1. Summary of data collection

Labels	Uploaded	Not uploaded	Total
Killing time and available for viewing notifications	606,760 (51.1%)	29,160 (2.5%)	635,920 (53.6%)
Killing time but unavailable for viewing notifications	135,380 (11.4%)	2,101 (0.2%)	137,481 (11.6%)
Not killing time but available for viewing notifications	202,327 (17.1%)	17,081 (1.4%)	219,408 (18.5%)
Not killing time and unavailable for viewing notifications	118,313 (10.0%)	9,071 (0.8%)	127,384 (10.7%)
Unidentifiable	0 (0.0%)	66,152 (5.6%)	66,152 (5.6%)
Total	1,062,780 (89.6%)	123,565 (10.4%)	1,186,345 (100.0%)

interview. All 36 participants were aged between 20 and 54 (M = 27.4, SD = 6.8), with 16 identifying as male and 20 as female. Half were students, and the other half in employment.

3.5 Data Collection

Most participants provided data on 12 hours of phone usage per day, but six voluntarily extended this to 13-15 hours; one, to 17.5 hours; and another, to the whole day. In total, 1,186,345 screenshots were annotated (per-participant M = 32,954.0, SD = 15,557.9), which represented approximately 1,633.8 hours of phone use. Among these 1,186,345 annotated data points, 1,062,780 (89.6%) screenshots were uploaded; a per-participant average of 29,521.7 screenshots (SD = 13,544.9). Thus, the initial dataset that we collected for analysis consisted of 1,062,780 annotated screenshots and the phone-sensor data associated with the moments at which they were captured. Two-thirds (n = 773,401) of uploaded and non-uploaded screenshots were annotated as "killing time", and somewhat over a quarter (n = 346,792) as "not killing time", with the remaining 5.6% (n = 66,152) being "unidentifiable" (see Table 1). The above distribution cannot perfectly represent the participants' actual phone usage, insofar as some screenshots were not annotated and/or not uploaded. Nevertheless, we are confident in its general outlines, e.g., that there were more time-killing moments than non-time-killing ones, and that the participants more often self-reported being available for viewing notifications than otherwise.

Because the focus of this paper is on how to predict time-killing moments, it will not systematically discuss the interview data, collected notification data, ESM results, or the results of the three questionnaires that were not related to our approach's user acceptance. Those other datasets will instead be used in future research.

3.6 Feature Selection and Extraction

To predict time-killing moments, we extracted two kinds of feature sets from the phone-sensor data: phone context and user interactions. For each of these feature sets, we created two temporal ranges, one describing the phone at the moment when a screenshot was taken, and the other, the characteristics of the phone-use session during which it was taken. We defined a phone-use session as a continuous use of the phone during which any brief screen-off interval was not longer than 45 seconds, based on the findings of van Berkel et al. [79], that using the 45-second threshold separating two sessions was more accurate than the others. Thus, if more than 45 seconds had passed since the last screen-off event, the current usage was considered as a new session. In addition, inspired by our interview data and prior research findings [64] suggesting that some phone events or user actions occur intensively during time-killing, we created features that measured the frequency of various types of phone and interaction events during nine past-time windows, ranging from a minimum of 30 seconds to a maximum of 3,600 seconds (e.g., frequency of scrolling within the previous 30 minutes). We excluded data from the first hour of each person's participation day, because a large

Table 2. The sensor features used in the study

Phone Context	Current Characteristics	Current session characteristics (accumulated up to the current screenshot record)
Transportation Mode	Physical activity (i.e., not moving, on foot, in vehicle, or on bicycle)	Cumulative time of {not moving, on foot, in vehicle, on bicycle}
	Was moving (i.e., on foot, in vehicle, or on bicycle)	Majority of physical activity
Type of Day	Day of the week (0-6)	
	Was weekend (i.e., Saturday, Sunday)	
Time of a Day	Hour of the day in 24-hour notation (0-23)	
	Was meal time (11:00 a.m.-12:59 p.m., 5:00 p.m.-6:59 p.m.)	
Battery Status	Phone battery level	{AVG, STD, MIN, MAX, MED} Phone-battery level
	Phone was charging / not charging	Charging count
	If charging over AC or USB	Cumulative charging time
Screen Time		{AVG, STD, MIN, MAX, MED, SUM} Screen time
Screen Orientation	Portrait / landscape mode	
Foreground App	Name of the app in the foreground	Count and frequency of app switches
	Package name of the app in the foreground	Count of used apps
	Category of the app in the foreground	Cumulative usage time of the 15 most frequently used app categories and all remaining app categories combined into one category group.
Network Info	{WiFi, Mobile} network was available / unavailable	Cumulative time the phone was connected to the {WiFi, Mobile} network
	{Type, operator} of the network the phone connected to	Cumulative time the phone was not connected to any network
	Was connected to the network	
Ringer Mode	Silent / vibrate / normal	Cumulative time of {silent, vibrate, normal}
		Was adjusted
Audio Mode	Ringing / in call / in communication / normal	Cumulative time of {ringing, in call, in communication, normal}
		Call count
Stream Volume	Volume of streams, e.g., music playback, notification, phone calls, phone ring, system sounds	{AVG, STD, MIN, MAX, MED} Volume of stream {music playback, notification, phone calls, phone ring, system sounds}
		Volume of stream {music playback, notification, phone calls, phone ring, system sounds} was adjusted
Call Status	Device call state: idle / off-hook / ringing	
Usage	**Current Characteristics**	**Current session characteristics (accumulated up to the current screenshot record)**
Screen-on Events	Count of Screen-on events during the past 180/300/600/900/1,800/3,600 seconds	{count, frequency} of screen-on events
Accessibility Events	Count of {clicking, long-clicking, scrolling, hover enter/exit, setting-input focus, changing-the-text, selecting} events during the past 30/60/180/300/600/900/1,800/3,600 seconds	{count, frequency} of {clicking, long-clicking, scrolling, hover enter/exit, setting-input focus, changing-the-text, selecting} events

Note. * All time-related calculations were in seconds

portion of such data could not allow us to compute these features. As a result, the final dataset for developing the model consisted of 967,466 annotated screenshots, from which 183 features were derived, as shown in Table 2. The 1,181 apps used during the study by our participants were placed in 56 categories based on their Google Play Store categorizations and prior literature [89].

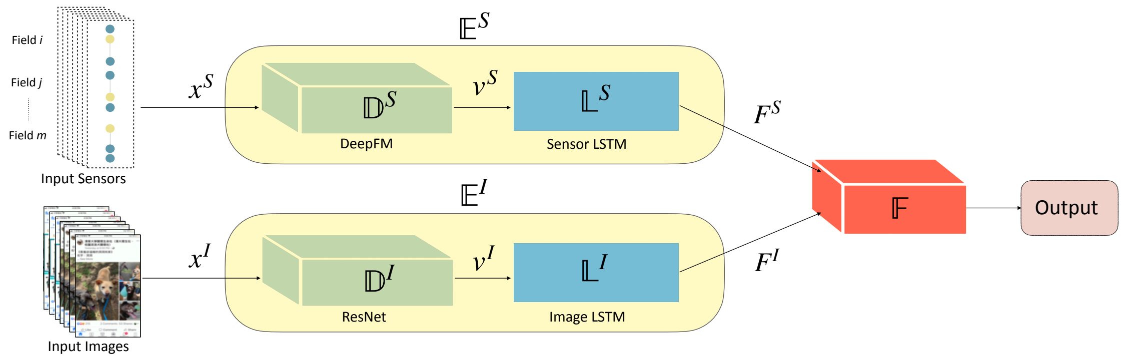


Fig. 2. Illustration for the architecture of our proposed model, which takes the input composed of the phone-sensor data and the screenshots (collected within a certain time window, e.g., 30 seconds) and predicts the user's intention on time-killing.

4 MODEL DESIGN

The goal of our proposed method is to leverage the rich information embedded in the phone-sensor data and screenshots to detect participants' time-killing moments. We adopt deep-learning, which learns the pattern in an end-to-end manner. Specifically, our proposed model (shown in Fig. 2) is composed of three main subnetworks: 1) an encoder $\mathbb{E}^S$ built upon DeepFM [26] and an LSTM [30] that extract *sensor features* from phone-sensor data, 2) an encoder $\mathbb{E}^I$ based on the ResNet and an LSTM that encode the sequences of screenshots into *visual features*, and 3) a fusion subnetwork $\mathbb{F}$ that adopts an attention mechanism followed by fully-connected layers to fuse the sensor features and the visual features into the final prediction outcome, i.e., time-killing vs. non-time-killing. More details of these subnetworks are provided in the following sections.

4.1 Encoder $\mathbb{E}^S$ of Phone-sensor Data

Given a sequence of phone-sensor data collected at several time steps within a certain time window (ideally these time steps are evenly distributed within a given time window), denoted as $\mathcal{X}^S = \{x_k^S\}_{k=1}^K$, where K is the number of time steps, the encoder $\mathbb{E}^S$ which is built upon a DeepFM module $\mathbb{D}^S$ and a 3-layer LSTM module $\mathbb{L}^S$ turns $\mathcal{X}^S$ into the sensor feature $\mathcal{F}^S$. As our phone-sensor data x_k^S contain both continuous and categorical values (e.g., a phone battery level is a continuous value, whereas a ringer mode is a categorical value), our DeepFM module $\mathbb{D}^S$ adopts the DeepFM [26] framework that extracts a feature representation $v_k^S = \mathbb{D}^S(x_k^S)$ for each x_k^S. Note that the architecture of our DeepFM module $\mathbb{D}^S$ is almost identical to the one proposed in [26], except that it uses a 128-dimensional vector in the last fully-connected layer in order to fit into the size of v_k^S. Specifically, the feature vectors $\{v_k^S\}_{k=1}^K$ extracted from the sensor data $\{x_k^S\}_{k=1}^K$ are sequentially fed into the LSTM module $\mathbb{L}^S$ to model the temporal variations in $\{x_k^S\}_{k=1}^K$, which then generates a 256-dimensional sensor-feature vector $\mathcal{F}^S$.

4.2 Encoder $\mathbb{E}^I$ of Screenshots

The visual encoder $\mathbb{E}^I$ which extracts the visual feature $\mathcal{F}^I$ from a stack of K screenshots $\mathcal{X}^I = \{x_k^I\}_{k=1}^K$ is composed of a ResNet module $\mathbb{D}^I$ and a 3-layer LSTM module $\mathbb{L}^I$. All the screenshots are resized to 224×224 pixels, regardless of whether they were taken horizontally or vertically; then they are fed into the ResNet module $\mathbb{D}^I$ to extract the feature representation $v_k^I = \mathbb{D}^I(x_k^I)$, where $\mathbb{D}^I$ adopts the ImageNet-pretrained Resnet-101 backbone and the size of v_k^I

is $7 \times 7 \times 2048$. Similar to the procedure of encoding phone-sensor data, these extracted features $\{v_k^I\}_{k=1}^K$ are taken as a sequential input for the LSTM module $\mathbb{L}^I$ to derive their visual feature $\mathcal{F}^I$ (which is 256-dimensional) of $\mathcal{X}^I$. For both LSTM modules $\mathbb{L}^S$ and $\mathbb{L}^I$, the dimensions of all the hidden state, cell state, and the hidden layer are set to 512 respectively. Note that although $\mathbb{L}^S$ and $\mathbb{L}^I$ have a similar architecture, they are trained independently and do not share any weight.

4.3 Fusion Subnetwork $\mathbb{F}$ over Sensor and Visual Features

After obtaining the sensor feature $\mathcal{F}^S$ and visual feature $\mathcal{F}^I$ from phone-sensor data $\mathcal{X}^S$ and screenshots $\mathcal{X}^I$, respectively , we used a fusion subnetwork $\mathbb{F}$ that jointly considers the high-level information from these two features in order to detect participants' time-killing behaviors. To achieve this, instead of concatenating two features and utilizing a simple classifier to perform a multi-modal fusion, we introduced an additional multi-fusion layer that takes both features as inputs to predict the reweighting coefficients α^S and α^I (i.e., analogous to the importance) for both feature dimension $\mathcal{F}^S$ and $\mathcal{F}^I$; The reweighted features, denoted as $\tilde{\mathcal{F}}^S = \alpha^S \otimes \mathcal{F}^S$ and $\tilde{\mathcal{F}}^I = \alpha^I \otimes \mathcal{F}^I$, are then concatenated with the original $\mathcal{F}^S$ and $\mathcal{F}^I$, which are further intertwined by several fully-connected layers to generate the final classification outcome of time-killing or not.

Training Details. We adopted a stage-wise training procedure, in which we first trained the encoders, $\mathbb{E}^S$ and $\mathbb{E}^I$, independently, followed by training the fusion subnetwork. Specifically, we first attached a fully connected layer to the end of the encoder $\mathbb{E}^S$ and $\mathbb{E}^I$ individually. Then, the layer maps the sensor feature $\mathcal{F}^S$ and the visual feature $\mathcal{F}^I$ to the output of time-killing detection respectively, i.e., the whole encoder together with the attached fully connected layer becomes a classification model and can be pre-trained via using our collected dataset and a classification objective of cross-entropy. After pre-training both encoders till they converged, we removed the attached fully connected layers and fixed the weights of encoders. Then we trained the fusion subnetwork $\mathbb{F}$ via the cross-entropy loss. We chose to follow a stage-wise training procedure because it performs better than training from scratch. We adopted the Adam optimizer [42] for training the model. In pretraining the encoder $\mathbb{E}^S$, we set the batch size 512 and the learning rate 10^{-3}, while for pretraining the encoder $\mathbb{E}^I$, we set a batch size 196 and the learning rate 10^{-5}. Lastly, for training the fusion subnetwork $\mathbb{F}$, we set a batch size 196 and the learning rate 10^{-5}. Our model is implemented with PyTorch and trained using 8 Tesla V100 GPU cores.

5 THE FUSION MODEL FOR PREDICTING TIME-KILLING MOMENTS

In the first subsection below, we describe our experimental environment, configuration, and evaluation metrics. In the second, we report on the performance of our fusion model for predicting time-killing moments, as compared to models that used only phone-sensor data and only screenshot data, respectively. Lastly, subsection 5.3 discusses how phone-sensor and screenshot data complemented each other in the fusion model.

5.1 Experiment

5.1.1 Dataset. We paired each labeled screenshot with phone-sensor data according to the time at which that screenshot was taken. To predict whether a screenshot was labeled as time-killing or non-time-killing, we used features derived from the screenshots and their paired sensor data 30 seconds (i.e., six screenshots) prior to the predicted one. In other words, a sequence of data including both the predicted screenshot and the data for predicting it contained seven data pairs. We made sure that such sequences did not overlap with one another; and that, if a sequence contained fewer than seven data pairs, we padded it to that length seven by using zero padding, i.e., a whole black image.

Each participant contributed a different amount of data. Therefore, to prevent our model being overly biased towards particular participants who contributed much more data than others did, we sampled 20,000 screenshots from each participant to create our training dataset. Such sampling was random, except insofar as we ensured that it contained 1) data collected on both weekends and weekdays, and 2) exactly equal numbers of time-killing and non-time-killing instances. For the testing dataset, on the other hand, we did not seek to strike this balance, but instead followed the original distribution, such that the evaluation of the model would more accurately reflect the time-killing distribution that one would observe in the real world.

5.1.2 Evaluation Metrics. Our testing dataset had more time-killing instances than non-time-killing ones, in the ratio 7:3. We made many computations to compare model performance, but here, we will focus on ROC-curve (Receiver Operating Characteristics) and PR-curve (Precision Recall). The ROC curve plots the true positive rate against the false positive rate at various classification thresholds for time-killing classification, and AUROC, i.e., the area under the ROC curve, indicates better performance where its values are higher. The PR-curve allowed us to observe the precision score against the recall score at various classification thresholds. We prioritized the precision of the prediction over recall, because the higher the former is, the fewer non-time-killing moments will be falsely predicted as time-killing moments, and thus, fewer notifications will be mistakenly sent to the user at these moments. For the same reason, we also assessed specificity, which measures the prediction's true negative rate.

5.1.3 Model Evaluation. To evaluate the performance of the model, we performed three-fold cross-validation on the dataset. As noted earlier, two-thirds of the data from each participant were used for re-sampling, and formed a training dataset, with the rest forming the test dataset. We made sure that when we divided the dataset, the order among the screenshot and phone-sensor pairs was maintained. In evaluating the performance of the fusion model for predicting time-killing moments, we also compared it against two other models, which respectively used only phone-sensor data and only screenshot data. We describe all three models in more detail below.

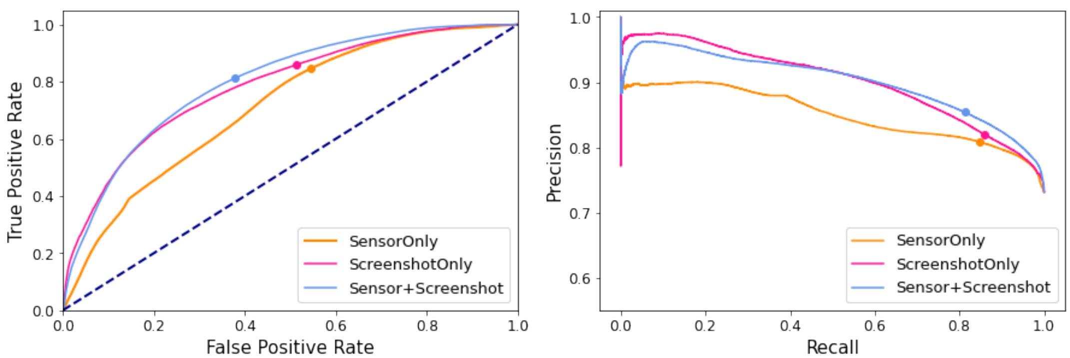
- ***Fusion (Sensor+Screenshot)*** - Used both phone-sensor data and screenshot data; model design as described earlier.
- ***SensorOnly*** - Used the phone-sensor data encoded by $\mathbb{E}^S$ to perform time-killing prediction, with an additional fully connected layer attached to $\mathbb{E}^S$ acting as the linear classifier.
- ***ScreenshotOnly*** - Used phone-screenshot data encoded by $\mathbb{E}^I$ to perform time-killing prediction, with an additional fully connected layer attached to $\mathbb{E}^I$ as a linear classifier.

5.2 Result

The models' overall performance metrics are presented in Table 3, which uses a classification threshold of 0.5. Fig. 3a and 3b show their ROC curves and PR curves. Overall, the fusion model achieved the best AUROC among the three models, as shown in both Table 3 and Fig. 3a. The fusion model's prediction of a given moment as being a time-killing one was the most accurate among the three models. Moreover, as shown by the PR curves, the fusion model achieved higher precision with high recall than the other two models, and its specificity score was also significantly higher than theirs. These results imply that taking account of both sensor data and screenshot data makes it less likely to falsely predict a non-time-killing moment as a time-killing one than when only one source or the other is considered. The *SensorOnly* model achieved the lowest performance across all metrics except recall. As shown in both Fig. 3a and Fig. 3b, it had notably lower precision across classification thresholds than the other two models, suggesting that many of the

Table 3. The three models' time-killing prediction task performance

Model	Accuracy	Precision	Recall	AUROC	Specificity
Fusion (Sensor+Screenshot)	0.76	0.83	0.81	0.72	0.62
SensorOnly	0.74	0.8	0.85	0.65	0.45
ScreenshotOnly	0.76	0.81	0.86	0.67	0.49



(a) ROC Curves (b) PR Curves

Fig. 3. Two performance measurements of our proposed fusion model (i.e., *Sensor+Screenshot*), its variants (i.e., *SensorOnly* and *ScreenshotOnly*). Note. Point on the curves represents a classification threshold equal to 0.5.

moments it predicted as time-killing were incorrect. This was because some phone states or interactions that occurred mainly during time-killing by one group of users often occurred during the non-time-killing-moments of another group, making it difficult to differentiate these two kinds of moments across users with different behavior patterns: a phenomenon that will be explored in the Section 6. The *ScreenshotOnly* model, on the other hand, had a better ability to distinguish between them, suggesting that phone-screenshot data were more informative about time-killing moments than sensor data were. That being said, the inclusion of phone-sensor data improved the performance of the fusion model.

5.3 Examples of How Fusing Phone-sensor Data and Screenshots Helped us Recognize Time-killing vs. Non-time-killing Behaviors

In our view, the fact that fusing phone-sensor data and screenshots yielded the best performance in detecting time-killing moments implies that these two data sources to some extent complemented each other. To explore this possible phenomenon, we inspected cases in our test dataset in which a time-killing moment was correctly detected by the fusion model, but incorrectly detected by either or both of the *SensorOnly* and *ScreenshotOnly* models.

To facilitate this exploration and our sense-making of these cases, we created attention maps from the final convolution layer of the *ScreenshotOnly* model, using a popular technique called Grad-CAM [72]. These attention maps helped us to identify regions in the screenshots that the fusion/*ScreenshotOnly* model considered influential on its time-killing behavior detection. For instance, the top row of Fig. 4 provides examples in which both the *ScreenshotOnly* and fusion models correctly recognized a time-killing moment that was mistaken as a non-time-killing one by the *SensorOnly* model. We suspect that the *SensorOnly* model incorrectly recognized such sequences of data because a series of text

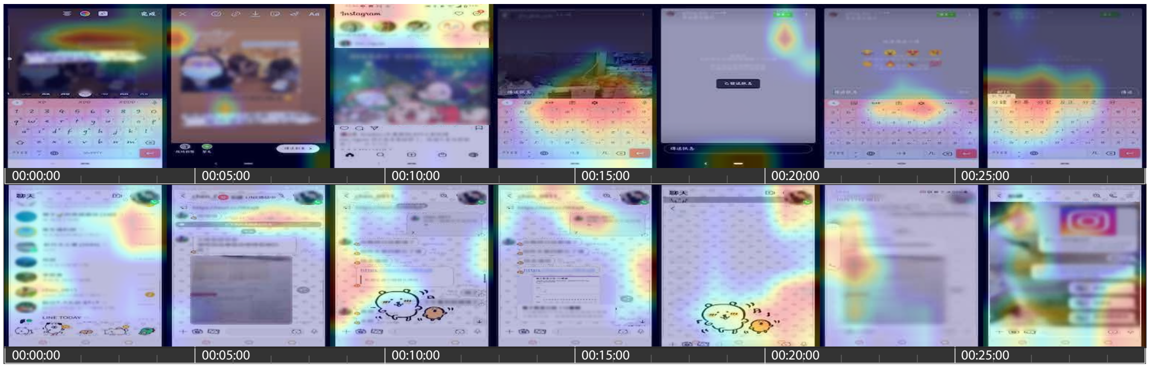


Fig. 4. Example attention maps, produced by Grad-CAM [72] and the *ScreenshotOnly* model, comprising a sequence of time-killing screenshots in the top row, and a sequence of non-time-killing ones in the bottom row. Images have been blurred for privacy reasons.

changed events were detected, which was more likely to occur when not killing time. On the other hand, we suspect that the *ScreenshotOnly* model detected it correctly because it recognized the layout of the user interface of Instagram's Story feature, which tended to be associated with time-killing moments. In other words, although the first two screenshots showed a Story post feature on Instagram, and the last three, participants replies to others' stories, the model knew the layout of the Story feature, and thus stuck to its prior prediction that time-killing was taking place. The *SensorOnly* model, in contrast, could only know that an Instagram application was currently in use, and that typing was occurring, not the specific feature of Instagram the participants were using (i.e., post, story, or direct message).

The bottom row in Fig. 4, meanwhile, shows a distinctive case in which both the *SensorOnly* and fusion models correctly predicted a non-time-killing moment that was incorrectly predicted by the *ScreenshotOnly* model as a time-killing one. We suspect that the *ScreenshotOnly* model misinterpreted this screenshot sequence as a time-killing moment because it recognized the layout of LINE, a popular instant-messaging, social-media and portal service in Taiwan. In this case, the participant was discussing an assignment with others via text conversation; however, the participant was talking to her friend (prompted by the communication icon in the upper-right corner) while, which was often associated with time-killing moments. The *ScreenshotOnly* model did not attend to the communication icon in all sequences of the screenshots, but instead relied mostly on the layout of the chat room. Nevertheless, we observed that the relevant information was captured in the user's phone-sensor data: specifically, by the call status and the change of the call volume (as the sixth screenshot shows). Knowing these pieces of information enabled the fusion model to correctly recognize this moment as a non-time-killing rather than a time-killing one, in contrast to the *ScreenshotOnly* model. There were many similar instances; however, these two vivid examples should suffice to explain why the fusion model performed best at detecting time-killing moments across nearly all metrics.

6 TAILORING FUSION MODELS TO USERS CLUSTERED BY PHONE-USAGE BEHAVIOR

Inspired by our interview data, we decided to build a prediction model tailored to varied phone-usage behaviors. Specifically, we learned from the interviews that various distinct time-killing patterns existed among our participants, who could be grouped based on similarities in their phone interactions, task choices, task switching, audio modes, and so on. Because we could not group participants based on their time-killing behaviors, assuming that during system runtime such a label might not be obtainable, we instead grouped them based on their phone-usage behavior, which could be

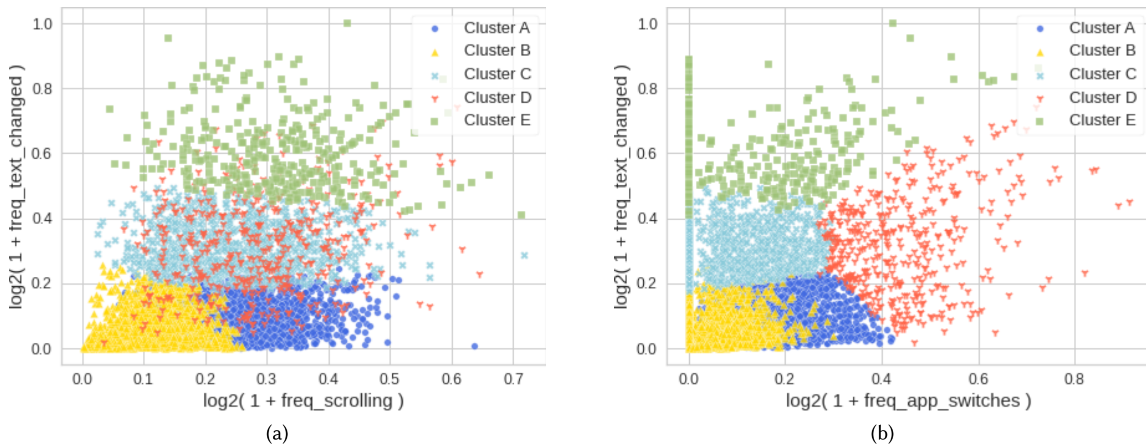


Fig. 5. Scatter plot of session clusters, grouped based on in-session behavioral characteristics

obtained during runtime. Despite the fact that grouping users would inevitably reduce the dataset for training each individual fusion model, we assumed that a user-group-based model was likely to achieve better overall performance than general model. Below, we present the group-based model we arrived at using clustering, followed by model evaluation and our observations about the features of these individual models.

6.1 Clustering Participants Based on their Phone-usage Behavior

We employed two stages of the k-means method [50] to group users hierarchically. First, inspired by Isaacs et al. [34], we employed clustering to identify distinct phone-usage behavioral patterns. Then, we clustered participants according to how often their use of the phone belonged to each of the identified phone-usage patterns, based on an assumption that a user was likely to display more than one such pattern.

6.1.1 Clustering Phone-usage Behavior. Inspired by previous work [34] that used the concept of *sessions* to cluster phone usage, we generated participants' sessions based on the rule suggested by van Berkel et al. [79]: that is, we divided pairs of sessions using a separation threshold of 45 seconds. This approach resulted in a total of 5,266 phone-usage sessions. For each of them, inspired by our interview, we computed nine features: 1) session duration, 2) screen-switching frequency, 3) application-switching frequency, 4) scroll-event frequency, 5) text-change event frequency, 6) maximum and 7) minimum gap durations for scroll events, and 8) maximum and 9) minimum gap durations for text-change events. We then applied k-means to these sessions, and used the Elbow method [77] to determine the number of clusters. This revealed the optimal number of clusters as five. The 5,266 phone-usage sessions were grouped into these five clusters, named A, B, C, D, and E in descending order by cluster size,, whose sizes were 1,882, 1,664, 942, 417 and 361, respectively.

The five groups mainly differed in terms of how actively their members used their phones. For example, Fig. 5a shows the distribution of the frequency of the participants' scrolling by the frequency of text-changes in a session, colored according to the cluster they belonged to; and Fig. 5b, the distribution of the same frequency by the frequency of app switching. For example, cluster B contained inactive phone-usage sessions, which involved low frequencies of text-changes, scrolling, and app switching. The sessions in Cluster A, on the other hand, were also marked by

Table 4. Experimental Results: Clustering Participants by Behavioral and Temporal Characteristics

Group	Accuracy			Precision			Recall			AUCROC			Specificity		
Group 1	0.70	0.73	0.73	0.81	0.81	0.88	0.85	0.81	0.80	0.68	0.76	0.77	0.38	0.54	0.59
Group 2	0.75	0.77	0.77	0.85	0.89	0.91	0.84	0.87	0.84	0.70	0.75	0.78	0.46	0.44	0.55
Group 3	0.80	0.77	0.78	0.91	0.95	0.93	0.87	0.82	0.82	0.68	0.75	0.72	0.39	0.48	0.50
Group 4	0.72	0.74	0.77	0.71	0.74	0.77	0.78	0.78	0.79	0.70	0.73	0.77	0.63	0.69	0.74
Average	0.74	0.75	0.76	0.82	0.84	0.87	0.83	0.82	0.81	0.69	0.75	0.76	0.47	0.54	0.60
General model	0.74	0.76	0.76	0.80	0.81	0.83	0.85	0.86	0.81	0.65	0.67	0.72	0.45	0.49	0.62

Note. The white, light gray, and dark gray backgrounds indicate the results for *SensorOnly*, *ScreenshotOnly*, and Fusion (*SensorOnly+ScreenshotOnly*) models, respectively.

Table 5. The 15 non-category features most highly correlated (either positively or negatively) with time-killing moments, by user group

Group 1	corr.	Group 2	corr.	Group 3	corr.	Group 4	corr.	General Model	corr.
call_count	-0.25	screen-on_past_900s	-0.22	T_photography_apps	-0.18	battery_level	-0.40	T_vibration	-0.17
is_adjusted_vol_noti	-0.25	screen-on_past_600s	-0.22	scrolling_past_3600s	0.15	AVG_battery	-0.40	scrolling_past_3600	0.15
is_adjusted_vol_ring	-0.25	screen-on_past_300s	-0.21	screen-on_past_600s	-0.15	MED_battery	-0.40	call_count	-0.15
T_Silent	0.24	screen-on_past_1800s	-0.21	screen-on_past_1800s	-0.15	MIN_battery	-0.39	scrolling_past_1800s	0.14
is_adjusted_vol_voicecall	-0.24	call_count	-0.21	screen-on_past_900s	-0.14	MAX_battery	-0.37	T_InComm.	-0.14
is_adjusted_vol_sys	-0.24	screen-on_past_3600s	-0.21	scrolling_past_1800s	0.14	MAX_vol_music	0.36	MIN_battery	-0.14
T_game_apps	0.24	screen-on_past_180s	-0.21	T_normal_ringer	0.14	AVG_vol_music	0.35	T_ringer_silent	0.13
MAX_vol_ring	-0.21	T_InComm.	-0.19	screen-on_past_300s	-0.14	MED_vol_music	0.33	MED_battery	-0.13
MAX_vol_noti	-0.21	T_normal_audio	0.19	T_map_apps	-0.13	MIN_vol_ring	0.32	AVG_battery	-0.13
MAX_vol_sys	-0.20	T_ringtone	-0.16	scrolling_count	0.13	strm_vol_music	0.32	scrolling_past_900s	0.13
STD_vol_sys	-0.19	MAX_vol_sys	-0.16	long-clicking_count	0.13	AVG_vol_ring	0.31	T_photography_apps	-0.12
STD_vol_noti	-0.19	MAX_vol_noti	-0.16	T_social_apps	0.13	MED_vol_ring	0.31	scrolling_past_600s	0.12
STD_vol_ring	-0.19	T_mobile_network	0.15	scrolling_past_900s	0.13	strm_vol_ring	0.31	battery_level	-0.12
MIN_vol_voicecall	0.18	freq_text_changed	-0.15	scrolling_past_600s	0.13	AVG_vol_sys	0.31	focus_event_past_3600s	0.12
T_InComm.	-0.16	MAX_vol_ring	-0.15	screen-on_past_180s	-0.12	MAX_vol_sys	0.30	MAX_vol_music	0.12

Note. The T prefix indicates the cumulative time; the green and blue backgrounds indicate positive and negative correlations, respectively, with darker colors indicating higher correlations.

low-frequency text-changes and relatively low-frequency app switching, but high-frequency scrolling; and those in cluster D exhibited the highest-frequency app switching of any cluster.

6.1.2 Clustering Users by the Proportions of Five Behavioral Outcomes. Having clustered similar phone-usage behaviors as described above, we observed that most users performed all five behaviors, but in varying proportions. Therefore, to group users with similar overall mobile-phone usage, we calculated the proportions of each user's five outcome behaviors, and used those proportions to cluster users. The same k-means and Elbow methods as described above were performed, and the resulting k value for user clustering was 4. Thus, we separated our participants into four groups, in which the numbers of participants were 11, 11, nine, and five. The positive (time-killing) and negative (non-time-killing) instance ratios of those four groups were 13:6, 3:1, 81:19, and 3:2, respectively.

6.2 Overall Performance of the Cluster-based Models

We built the same fusion model for each of the four user groups, and examined each one's average performance separately via the same three-fold cross-validation approach mentioned in Section 5.1. Table 4, which presents the respective performance of those four models along with their average performance, shows that both their average AUROC (0.76) and precision (0.87) were higher than those of the general model (AUROC: 0.72, precision: 0.83). In terms of individual model performance, all four models' AUROC values were at least as good as that of the general model, with three significantly higher than it; and three models' precision values were also higher than the general model's. These results suggest that dividing users into groups according to their phone-usage behavior and building a time-killing prediction model for each such user group is beneficial.

We also looked at the correlations between time-killing moments and phone-sensor features for each of these user groups separately. Table 5 shows the 15 non-category features most highly correlated (either positively or negatively) with time-killing moments, by user group. In each such group, some features were more correlated with time-killing moments than their counterparts in the general model, suggesting that clustering users into behavioral groups was also beneficial to time-killing prediction: i.e., doing so revealed features correlated with time-killing moments specifically for certain participants, which would not have been revealed had they not been divided into groups. That being said, the results in Table 4 also show that the performances of the four models varied, suggesting that some user groups' time-killing moments might be more difficult than the others' to predict. We discuss each user group's model performance and time-killing behaviors in the next section.

6.3 Model Performance and Behavior by User Group

First, Group 2's fusion model achieved the best AUROC among the four user groups. It is also worth noting that Group 2's *ScreenshotOnly* model achieved better performance than its *SensorOnly* model for all metrics except specificity, suggesting that it was accurate in predicting time-killing moments but less so in predicting non-time-killing moments. When observing features correlated with time-killing moments in Group 2, we found that screen-on events, number of calls, and volume of communication and ringtone were all negatively correlated with the members' time-killing moments. In other words, when participants in this group were not killing time, they tended to increase the audio volume of their phones and frequently turned their screens on and off. Their switching to normal ringer mode was also positively correlated with time-killing moments; this reflected their higher usage of the two relatively quiet modes, vibrate and silent, when they were not killing time. All of this implies that these participants' non-time-killing moments were more often associated with making calls. As prior research has reported a high association between quiet ringer modes and proactive phone-checking behaviors [12], the Group 2 behaviors we observed could have indicated participants checking their phones frequently to avoid missing calls and/or notifications. The fact that these behaviors might have been captured better by sensor data than by screenshot data could explain why – in this group alone – the *SensorOnly* model performed better at identifying non-time-killing moments (i.e., higher specificity; true negative rate) than the *ScreenshotOnly* model did.

Secondly, Group 1's and Group 4's fusion models both achieved AUROCs of 0.77, but the reasons for these two models achieving this same value differed dramatically, as shown by the significant differences in their other performance metrics. Specifically, whereas Group 1's fusion model achieved significantly higher precision (0.88) than Group 4's fusion model did (0.77), Group 4's fusion model performed particularly well in specificity (0.74): significantly higher than any of the other models. In other words, Group 1's fusion model was better at predicting its members' time-killing moments, whereas Group 4's fusion model was better at predicting its members' non-time-killing moments. As shown in Table 5, Group 4's key features for prediction were predominantly battery-related ones, which were negatively correlated with time-killing moments. Also, while the feature number of charging events is not displayed in Table 5, its correlation was -0.27 – higher than many other features in other user groups – suggesting that this group's members' non-time-killing moments were associated with high values of battery-related features, very likely linked to battery-charging at non-time-killing moments. We further observed the app-usage distribution of Group 4's members, as shown in Fig. 6, and found that they played games much more often during non-time-killing moments than during time-killing

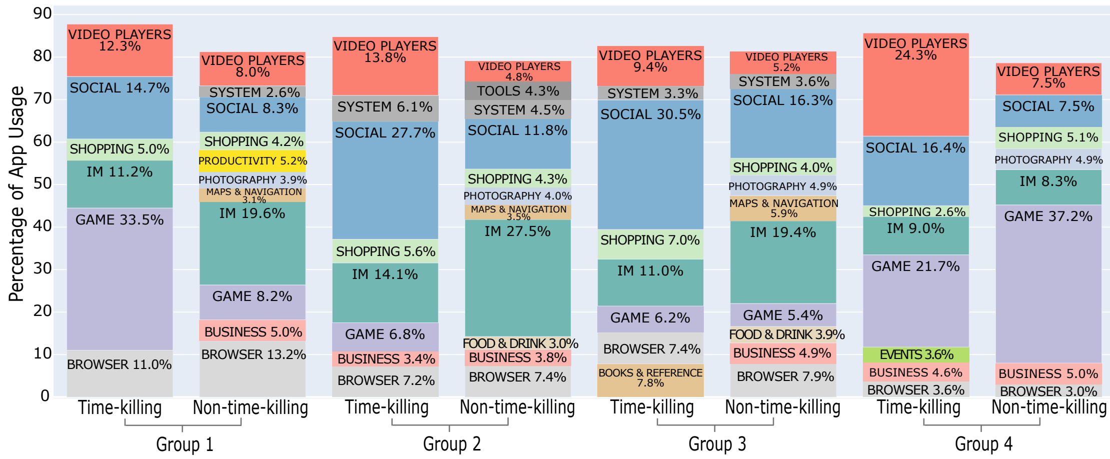


Fig. 6. Percentage of application categories used by each user group when killing time and not killing time Note. Categories 1) related to the launcher and 2) with percentages <2.5% are not displayed.

ones (37.2% vs. 21.7%); this percentage was also the greatest among the four groups. When we took a closer look at the games they played, we found that 88.6% of their game time during non-time-killing moments was taken up by Pokémon Go, and 95% of the time, they were correctly predicted by the model to be non-time-killing moments. Possibly because of the large quantity of this distinctive behavior during non-time-killing moments, the Group 4 fusion model's true negative rate was particularly high. Interestingly, Group 1 was another group whose members spent considerable time playing games, but in contrast to the Group 4 members, they were much more likely to do so during time-killing moments, and rarely did so in non-time-killing ones. The Group 1 participants also often used social-media applications, watched videos, and engaged in IM during their time-killing moments, but seldom did so during their non-time-killing moments. It is noteworthy that Group 1's *SensorOnly* model achieved much poorer specificity than its *ScreenshotOnly* model, suggesting that the fusion model relied heavily on screenshot data to recognize non-time-killing moments.

Finally, Group 3's fusion model achieved the lowest AUROC (0.72) among the four groups' fusion models, an outcome even worse than that of its *ScreenshotOnly* model (0.75). This was because, despite having the highest precision among the four groups, it had a particularly low true-negative rate. In part, this distinctive characteristic of the model might be attributed to it having the most unbalanced dataset: 80% of the instances were time-killing moments, and this might have made it tend to predict Group 3 members' moments as time-killing ones. The chief reason this user group's dataset was unbalanced was that its members used their phones mainly for killing time. Notably, correlations between features and time-killing moments were also lowest for Group 3, suggesting that its members' time-killing behaviors tended to be diverse and not associated with strong patterns. Also, when we looked into the Group 3 members' app-usage distribution in their time-killing vs. non-time-killing moments, we found it to be likewise highly diverse and evenly distributed. In short, a lack of clear patterns in phone usage during time-killing moments might explain the relatively low performance of this user group's *SensorOnly* model, which in turn seemed to lead the fusion model astray.

7 DISCUSSION

In the hope that time-killing moments might be leveraged for delivering content to smartphone users, we built models to predict such moments and examined their performance. We found that a deep-learning model fusing screenshot and phone-sensor data could achieve a precision of 0.83 and an AUROC of 0.72. However, there are two even more important takeaways of our results.

First, leveraging both phone-sensor and screenshot data in time-killing detection can achieve significantly better performance than using either of these data sources by itself, and particularly good at distinguishing non-time-killing moments from time-killing-ones. This is a vital capability that could help prevent a future commercial system from sending users digital content at falsely detected time-killing-moments. Therefore, fusion-model based systems for time-killing detection are likely to be more desirable, insofar as they are less likely than sensor-based ones to cause disruption through incorrectly assuming a non-time-killing period is a time-killing one. Crucially, the fusion model has this capability because, to a large extent, sensor features and the visual information extracted from screenshots complement each other effectively. For example, while screenshots do not inform us about various aspects of phone status such as battery, voice, and network, and are thus unhelpful in recognizing certain time-killing moments characterized by these features, they contain rich and unambiguous contextual information about the activity a user is undertaking during time-killing and non-time-killing-moments alike. We believe this complementary nature of the two data sources will be helpful not only in the detection of time-killing behaviors, but also possibly in the detection of other behavior/moments on phones and other devices, such as interruptible moments [2, 54, 56, 83], moments of boredom [64], mirco-waiting [11, 35], and/or breakpoint [1, 29, 55]. In addition, we believe that our approach can usefully be employed in future research, not only on opportune moments and interruptibility, but also more generally in fields that have already leveraged screenshot data to analyze broader patterns of behavior, such as smartphone users' media consumption [23].

The second key takeaway of our results is the benefits of clustering users according to their phone-usage behaviors and then tailoring fusion models to the resulting clusters. In our own experiment, this resulted not only in better overall performance than a general model that was built based on all users' data, but also better performance than that of most *SensorOnly* and *ScreenshotOnly* model. We attribute the superior performance achieved via this group-based approach to the diverse time-killing patterns of our participants, which sometimes were even opposite to each other, confusing the general model. A vivid example of this phenomenon was that participants in Group 1 tended to play games during time-killing moments, whereas those in Group 4 tended to do so at non-time-killing ones. Unsurprisingly, after these participants were separated, both their groups' respective models achieved significantly higher AUROC than the general model did.

The profound benefits of building user-cluster-based models were even manifested in the complementarity between sensor data and screenshot data. This was because some participants' behavior changes were associated more with changes in sensor data than phone-screen data, others' were opposite. For example, Groups 1, 2, and 4 exhibited phone-usage behavior that was clearly associated with time-killing moments (see Table 5). Thus, the extra information from sensors complemented that from screenshots, because each captured some aspect(s) of time-killing moments that the other missed. In contrast, Group 3's fusion model achieved lower AUROC than its *ScreenshotOnly* model. This may provide an example of conflicting instead of complementary information provided by the two data sources: i.e., the sensor information collected from this group of participants did not assist the fusion model in distinguishing time-killing moments from non-time-killing ones. This can also be seen from the low correlations between sensor features and this group's time-killing behaviors.

These results suggest that the effectiveness of phone sensor data for predicting time-killing moments depends heavily on phone users' behavior patterns. They also imply that decisions about whether it is worthwhile to engage in the privacy-intrusive and phone-resource-demanding process of capturing of users' screenshots should take account of the objective of such detection. For example, the *SensorOnly* models of both Group 1 and Group 3 achieved higher recall than their fusion models; so, if one's objective were to capture as many time-killing moments as possible, capturing only sensor information on the phones of users of the Group 1 and Group 3 types would be adequate to purpose. On the other hand, if one's main aim was to reduce falsely detected time-killing moments, leveraging screenshot data would generally be more helpful.

In sum, we believe the approach we have presented in this paper will help researchers and practitioners interested in leveraging screenshot data for predicting or detecting specific smartphone-user behavior and moments. In particular, we expect it to be useful for those interested in detecting time-killing moments for delivering content to which people may not be receptive at other moments.

8 LIMITATION

This research has several limitations. First, its study design was inherently reliant on the participants' in-the-wild annotations, which may not be always reliable. Indeed, our observations of the dataset indicated that some screenshots were mistakenly labeled, which could account for some of our models' apparent inaccuracies. Second, although we strove to ease our participants' screenshot-annotation burdens – on the grounds that otherwise, their compliance would have been much lower – it is possible that the user-friendly drag-and-drop interface we developed to address this problem facilitated mislabeling. That is, some subjects might have considered it more efficient, at least in some cases, to label a whole block of data at once. Third, our dataset was established based on a small (n=36) sample of smartphone users in Taiwan; all our participants were under 55 years old, and half of them were students. As a result, it is unclear whether our models' detection performance can be generalized to populations that display even more diverse time-killing behaviors or different phone-usage patterns. For example, we believe that such behaviors may be clustered into more types than the four that our small sample suggested. Thus, longer-term and larger-scale data collection could lead to more reliable results. Finally, although we collected other aspects of the participants' tendencies and characteristics that might have affected their time-killing behaviors, such as their demographic characteristics and occupations, we did not include them in this paper. We also did not analyze their notification-attendance behavior during time-killing moments. These aspects should be given greater attention in future studies.

9 CONCLUSION

In this paper, we leveraged both phone-sensor and screenshot data to predict time-killing moments using deep-learning techniques. We developed an Android app for collecting labeled time-killing data, and conducted data collection with 36 participants over 14 days, resulting in a total of 967,466 pairs of annotated phone-sensor data and screenshots for training our time-killing models. We have shown that phone-sensor and screenshot data each have their advantages in such detection tasks; and that, due to them being complementary to each other, integrating these two data sources can yield better model performance than using either of them by itself can. We also have shown that separating users into groups according to their phone-usage patterns and building individual time-killing models for each group can achieve strong overall performance, with most group-specific models also achieving better performance than a general model. Additionally, we have provided insights into how and why the effectiveness of sensor data and phone screenshots as a basis for detecting time-killing moments vary across different user groups. We believe this paper offers a good starting point for researchers and practitioners who are interested in leveraging both screenshot and sensor data in their prediction tasks, and that it will be especially useful for practitioners who want to incorporate time-killing detection into their applications.

REFERENCES

[1] Piotr D. Adamczyk and Brian P. Bailey. 2004. *If Not Now, When? The Effects of Interruption at Different Moments within Task Execution.* Association for Computing Machinery, New York, NY, USA, 271–278. https://doi.org/10.1145/985692.985727

[2] Piotr D. Adamczyk, Shamsi T. Iqbal, and Brian P. Bailey. 2005. A method, system, and tools for intelligent interruption management. In *Proceedings of the 4th international workshop on Task models and diagrams (TAMODIA '05)*. Association for Computing Machinery, New York, NY, USA, 123–126. https://doi.org/10.1145/1122935.1122959

[3] Elena Agapie, Jaime Teevan, and Andrés Monroy-Hernández. 2015. Crowdsourcing in the field: A case study using local crowds for event reporting. In *Third AAAI Conference on Human Computation and Crowdsourcing (HCOMP '15)*.

[4] Yeslam Al-Saggaf, Rachel MacCulloch, and Karl Wiener. 2019. Trait Boredom Is a Predictor of Phubbing Frequency. *Journal of Technology in Behavioral Science* 4 (09 2019). https://doi.org/10.1007/s41347-018-0080-4

[5] Yeslam Al-Saggaf and Sarah B O'Donnell. 2019. Phubbing: Perceptions, reasons behind, predictors, and impacts. *Human Behavior and Emerging Technologies* 1, 2 (2019), 132–140.

[6] Tony Beltramelli. 2018. Pix2code: Generating Code from a Graphical User Interface Screenshot. In *Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Paris, France) *(EICS '18)*. Association for Computing Machinery, New York, NY, USA, Article 3, 6 pages. https://doi.org/10.1145/3220134.3220135

[7] Matthias Böhmer, Brent Hecht, Johannes Schöning, Antonio Krüger, and Gernot Bauer. 2011. Falling Asleep with Angry Birds, Facebook and Kindle: A Large Scale Study on Mobile Application Usage. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services* (Stockholm, Sweden) *(MobileHCI '11)*. Association for Computing Machinery, New York, NY, USA, 47–56. https://doi.org/10.1145/2037373.2037383

[8] Miriam Brinberg, Nilam Ram, Xiao Yang, Mu-Jung Cho, S Shyam Sundar, Thomas N Robinson, and Byron Reeves. 2021. The idiosyncrasies of everyday digital lives: Using the Human Screenome Project to study user behavior on smartphones. *Computers in Human Behavior* 114 (2021), 106570.

[9] Barry Brown, Moira McGregor, and Eric Laurier. 2013. *IPhone in Vivo: Video Analysis of Mobile Device Use.* Association for Computing Machinery, New York, NY, USA, 1031–1040. https://doi.org/10.1145/2470654.2466132

[10] Barry Brown, Moira McGregor, and Donald McMillan. 2014. 100 Days of IPhone Use: Understanding the Details of Mobile Device Use. In *Proceedings of the 16th International Conference on Human-Computer Interaction with Mobile Devices & Services* (Toronto, ON, Canada) *(MobileHCI '14)*. Association for Computing Machinery, New York, NY, USA, 223–232. https://doi.org/10.1145/2628363.2628377

[11] Carrie J. Cai, Anji Ren, and Robert C. Miller. 2017. WaitSuite: Productive Use of Diverse Waiting Moments. *ACM Trans. Comput.-Hum. Interact.* 24, 1, Article 7 (March 2017), 41 pages. https://doi.org/10.1145/3044534

[12] Yung-Ju Chang and John C. Tang. 2015. Investigating Mobile Users' Ringer Mode Usage and Attentiveness and Responsiveness to Communication. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Copenhagen, Denmark) *(MobileHCI '15)*. Association for Computing Machinery, New York, NY, USA, 6–15. https://doi.org/10.1145/2785830.2785852

[13] Chunyang Chen, Sidong Feng, Zhenchang Xing, Linda Liu, Shengdong Zhao, and Jinshui Wang. 2019. Gallery D.C.: Design Search and Knowledge Discovery through Auto-Created GUI Component Gallery. *Proc. ACM Hum.-Comput. Interact.* 3, CSCW, Article 180 (Nov. 2019), 22 pages. https://doi.org/10.1145/3359282

[14] Chunyang Chen, Ting Su, Guozhu Meng, Zhenchang Xing, and Yang Liu. 2018. From UI Design Image to GUI Skeleton: A Neural Machine Translator to Bootstrap Mobile GUI Implementation. In *Proceedings of the 40th International Conference on Software Engineering* (Gothenburg, Sweden) *(ICSE '18)*. Association for Computing Machinery, New York, NY, USA, 665–676. https://doi.org/10.1145/3180155.3180240

[15] Pei-Yu Peggy Chi, Matthew Long, Akshay Gaur, Abhimanyu Deora, Anurag Batra, and Daphne Luong. 2019. Crowdsourcing Images for Global Diversity. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services* (Taipei, Taiwan) *(MobileHCI '19)*. Association for Computing Machinery, New York, NY, USA, Article 79, 10 pages. https://doi.org/10.1145/3338286.3347546

[16] Chia-En Chiang, Yu-Chun Chen, Fang-Yu Lin, Felicia Feng, Hao-An Wu, Hao-Ping Lee, Chang-Hsuan Yang, and Yung-Ju Chang. 2021. "I Got Some Free Time": Investigating Task-Execution and Task-Effort Metrics in Mobile Crowdsourcing Tasks. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, Article 648, 14 pages. https://doi.org/10.1145/3411764.3445477

[17] Woohyeok Choi, Sangkeun Park, Duyeon Kim, Youn-kyung Lim, and Uichin Lee. 2019. Multi-Stage Receptivity Model for Mobile Just-In-Time Health Intervention. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 2, Article 39 (June 2019), 26 pages. https://doi.org/10.1145/3328910

[18] Tilman Dingler and Martin Pielot. 2015. I'll Be There for You: Quantifying Attentiveness towards Mobile Messaging. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Copenhagen, Denmark) *(MobileHCI '15)*. Association for Computing Machinery, New York, NY, USA, 1–5. https://doi.org/10.1145/2785830.2785840

[19] Tilman Dingler, Benjamin Tag, Sabrina Lehrer, and Albrecht Schmidt. 2018. Reading Scheduler: Proactive Recommendations to Help Users Cope with Their Daily Reading Volume. In *Proceedings of the 17th International Conference on Mobile and Ubiquitous Multimedia* (Cairo, Egypt) *(MUM 2018)*. Association for Computing Machinery, New York, NY, USA, 239–244. https://doi.org/10.1145/3282894.3282917

[20] Tilman Dingler, Benjamin Tag, Sabrina Lehrer, and Albrecht Schmidt. 2018. Reading Scheduler: Proactive Recommendations to Help Users Cope with Their Daily Reading Volume. In *Proceedings of the 17th International Conference on Mobile and Ubiquitous Multimedia (MUM 2018)*. Association

for Computing Machinery, New York, NY, USA, 239–244. https://doi.org/10.1145/3282894.3282917

[21] Tilman Dingler, Dominik Weber, Martin Pielot, Jennifer Cooper, Chung-Cheng Chang, and Niels Henze. 2017. Language Learning On-the-Go: Opportune Moments and Design of Mobile Microlearning Sessions. In *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Vienna, Austria) *(MobileHCI '17)*. Association for Computing Machinery, New York, NY, USA, Article 28, 12 pages. https://doi.org/10.1145/3098279.3098565

[22] Trinh Minh Tri Do, Jan Blom, and Daniel Gatica-Perez. 2011. Smartphone Usage in the Wild: A Large-Scale Analysis of Applications and Context. In *Proceedings of the 13th International Conference on Multimodal Interfaces* (Alicante, Spain) *(ICMI '11)*. Association for Computing Machinery, New York, NY, USA, 353–360. https://doi.org/10.1145/2070481.2070550

[23] Hossein Falaki, Ratul Mahajan, Srikanth Kandula, Dimitrios Lymberopoulos, Ramesh Govindan, and Deborah Estrin. 2010. Diversity in Smartphone Usage. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services* (San Francisco, California, USA) *(MobiSys '10)*. Association for Computing Machinery, New York, NY, USA, 179–194. https://doi.org/10.1145/1814433.1814453

[24] Robert Fisher and Reid Simmons. 2011. Smartphone Interruptibility Using Density-Weighted Uncertainty Sampling with Reinforcement Learning. In *2011 10th International Conference on Machine Learning and Applications and Workshops*, Vol. 1. 436–441. https://doi.org/10.1109/ICMLA.2011.128

[25] Jon Froehlich, Mike Y. Chen, Sunny Consolvo, Beverly Harrison, and James A. Landay. 2007. MyExperience: A System for in Situ Tracing and Capturing of User Feedback on Mobile Phones. In *Proceedings of the 5th International Conference on Mobile Systems, Applications and Services* (San Juan, Puerto Rico) *(MobiSys '07)*. Association for Computing Machinery, New York, NY, USA, 57–70. https://doi.org/10.1145/1247660.1247670

[26] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

[27] Alexis Hiniker, Shwetak N. Patel, Tadayoshi Kohno, and Julie A. Kientz. 2016. Why Would You Do That? Predicting the Uses and Gratifications behind Smartphone-Usage Behaviors. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Heidelberg, Germany) *(UbiComp '16)*. Association for Computing Machinery, New York, NY, USA, 634–645. https://doi.org/10.1145/2971648.2971762

[28] Bo-Jhang Ho, Bharathan Balaji, Mehmet Koseoglu, Sandeep Sandha, Siyou Pei, and Mani Srivastava. 2020. Quick Question: Interrupting Users for Microtasks with Reinforcement Learning. *arXiv:2007.09515 [cs]* (July 2020). http://arxiv.org/abs/2007.09515 arXiv: 2007.09515.

[29] Joyce Ho and Stephen S. Intille. 2005. *Using Context-Aware Computing to Reduce the Perceived Burden of Interruptions from Mobile Devices*. Association for Computing Machinery, New York, NY, USA, 909–918. https://doi.org/10.1145/1054972.1055100

[30] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[31] Nanna Inie and Mircea F Lungu. 2021. Aiki - Turning Online Procrastination into Microlearning. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 369, 13 pages. https://doi.org/10.1145/3411764.3445202

[32] Shamsi T. Iqbal and Brian P. Bailey. 2007. Understanding and developing models for detecting and differentiating breakpoints during interactive tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 697–706. https://doi.org/10.1145/1240624.1240732

[33] Shamsi T. Iqbal and Brian P. Bailey. 2011. Oasis: A framework for linking notification delivery to the perceptual structure of goal-directed tasks. *ACM Transactions on Computer-Human Interaction* 17, 4 (Dec. 2011), 15:1–15:28. https://doi.org/10.1145/1879831.1879833

[34] Ellen Isaacs, Alan Walendowski, Steve Whittaker, Diane J. Schiano, and Candace Kamm. 2002. The Character, Functions, and Styles of Instant Messaging in the Workplace. In *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work* (New Orleans, Louisiana, USA) *(CSCW '02)*. Association for Computing Machinery, New York, NY, USA, 11–20. https://doi.org/10.1145/587078.587081

[35] E. Isaacs, N. Yee, D. Schiano, Nathaniel Good, Nicolas Ducheneaut, and V. Bellotti. 2010. - 1-Mobile Microwaiting Moments : The Role of Context in Receptivity to Content While on the Go. https://www.semanticscholar.org/paper/1-Mobile-Microwaiting-Moments-%3A-The-Role-of-Context-Isaacs-Yee/cafaaa823351104bddb3f95071241866b3993dbe

[36] Chakajkla Jesdabodi and Walid Maalej. 2015. Understanding Usage States on Mobile Devices. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Osaka, Japan) *(UbiComp '15)*. Association for Computing Machinery, New York, NY, USA, 1221–1225. https://doi.org/10.1145/2750858.2805837

[37] Jing Jin and Laura A. Dabbish. 2009. *Self-Interruption on the Computer: A Typology of Discretionary Task Interleaving*. Association for Computing Machinery, New York, NY, USA, 1799–1808. https://doi.org/10.1145/1518701.1518979

[38] Simon L. Jones, Denzil Ferreira, Simo Hosio, Jorge Goncalves, and Vassilis Kostakos. 2015. Revisitation Analysis of Smartphone App Use. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Osaka, Japan) *(UbiComp '15)*. Association for Computing Machinery, New York, NY, USA, 1197–1208. https://doi.org/10.1145/2750858.2807542

[39] Kleomenis Katevas, Ioannis Arapakis, and Martin Pielot. 2018. Typical Phone Use Habits: Intense Use Does Not Predict Negative Well-Being. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Barcelona, Spain) *(MobileHCI '18)*. Association for Computing Machinery, New York, NY, USA, Article 11, 13 pages. https://doi.org/10.1145/3229434.3229441

[40] Jürgen Kawalek, Annegret Stark, and Marcel Riebeck. 2008. A New Approach to Analyze Human-Mobile Computer Interaction. *J. Usability Studies* 3, 2 (Feb. 2008), 90–98.

[41] Ronald C Kessler, Lenard Adler, Minnie Ames, Olga Demler, Steve Faraone, EVA Hiripi, Mary J Howes, Robert Jin, Kristina Secnik, Thomas Spencer, et al. 2005. The World Health Organization Adult ADHD Self-Report Scale (ASRS): a short screening scale for use in the general population. *Psychological medicine* 35, 2 (2005), 245–256.

[42] Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs.LG]

[43] Vassilis Kostakos, Denzil Ferreira, Jorge Goncalves, and Simo Hosio. 2016. Modelling Smartphone Usage: A Markov State Transition Model. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Heidelberg, Germany) *(UbiComp '16)*. Association for Computing Machinery, New York, NY, USA, 486–497. https://doi.org/10.1145/2971648.2971669

[44] Philipp Krieter. 2019. Can I Record Your Screen? Mobile Screen Recordings as a Long-Term Data Source for User Studies. In *Proceedings of the 18th International Conference on Mobile and Ubiquitous Multimedia* (Pisa, Italy) *(MUM '19)*. Association for Computing Machinery, New York, NY, USA, Article 23, 10 pages. https://doi.org/10.1145/3365610.3365618

[45] Philipp Krieter and Andreas Breiter. 2018. Analyzing Mobile Application Usage: Generating Log Files from Mobile Screen Recordings. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Barcelona, Spain) *(MobileHCI '18)*. Association for Computing Machinery, New York, NY, USA, Article 9, 10 pages. https://doi.org/10.1145/3229434.3229450

[46] Hao-Ping Lee, Kuan-Yin Chen, Chih-Heng Lin, Chia-Yu Chen, Yu-Lin Chung, Yung-Ju Chang, and Chien-Ru Sun. 2019. *Does Who Matter? Studying the Impact of Relationship Characteristics on Receptivity to Mobile IM Messages.* Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3290605.3300756

[47] Tong Li, Mingyang Zhang, Hancheng Cao, Yong Li, Sasu Tarkoma, and Pan Hui. 2020. "What Apps Did You Use?": Understanding the Long-Term Evolution of Mobile App Usage. In *Proceedings of The Web Conference 2020* (Taipei, Taiwan) *(WWW '20)*. Association for Computing Machinery, New York, NY, USA, 66–76. https://doi.org/10.1145/3366423.3380095

[48] Yu-Hsuan Lin, Li-Ren Chang, Yang-Han Lee, Hsien-Wei Tseng, Terry BJ Kuo, and Sue-Huei Chen. 2014. Development and Validation of the Smartphone Addiction Inventory (SPAI). *PloS one* 9, 6 (2014), e98312.

[49] Kai Lukoff, Cissy Yu, Julie Kientz, and Alexis Hiniker. 2018. What Makes Smartphone Use Meaningful or Meaningless? *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 1, Article 22 (March 2018), 26 pages. https://doi.org/10.1145/3191754

[50] James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Vol. 1. Oakland, CA, USA, 281–297.

[51] Donald McMillan, Moira McGregor, and Barry Brown. 2015. From in the Wild to in Vivo: Video Analysis of Mobile Device Use. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Copenhagen, Denmark) *(MobileHCI '15)*. Association for Computing Machinery, New York, NY, USA, 494–503. https://doi.org/10.1145/2785830.2785883

[52] Abhinav Mehrotra, Robert Hendley, and Mirco Musolesi. 2016. PrefMiner: Mining User's Preferences for Intelligent Mobile Notification Management. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Heidelberg, Germany) *(UbiComp '16)*. Association for Computing Machinery, New York, NY, USA, 1223–1234. https://doi.org/10.1145/2971648.2971747

[53] Varun Mishra, Florian Künzler, Jan-Niklas Kramer, Elgar Fleisch, Tobias Kowatsch, and David Kotz. 2021. Detecting receptivity for mhealth interventions in the natural environment. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 2 (2021), 1–24.

[54] Christopher Monk, Deborah Boehm-Davis, and J. Trafton. 2002. The Attentional Costs of Interrupting Task Performance at Various Stages. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 46 (Sept. 2002). https://doi.org/10.1177/154193120204602210

[55] Tadashi Okoshi, Julian Ramos, Hiroki Nozaki, Jin Nakazawa, Anind K Dey, and Hideyuki Tokuda. 2015. Attelia: Reducing user's cognitive load due to interruptive notifications on smart phones. In *2015 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 96–104. https://doi.org/10.1109/PERCOM.2015.7146515

[56] T. Okoshi, Kota Tsubouchi, Masaya Taji, Takanori Ichikawa, and H. Tokuda. 2017. Attention and engagement-awareness in the wild: A large-scale study with adaptive notifications. *2017 IEEE International Conference on Pervasive Computing and Communications (PerCom)* (2017), 100–110.

[57] Antti Oulasvirta, Tye Rattenbury, Lingyi Ma, and Eeva Raita. 2012. Habits Make Smartphone Use More Pervasive. *Personal Ubiquitous Comput.* 16, 1 (Jan. 2012), 105–114. https://doi.org/10.1007/s00779-011-0412-2

[58] Leysia Palen and Marilyn Salzman. 2002. Voice-Mail Diary Studies for Naturalistic Data Capture under Mobile Conditions. In *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work* (New Orleans, Louisiana, USA) *(CSCW '02)*. Association for Computing Machinery, New York, NY, USA, 87–95. https://doi.org/10.1145/587078.587092

[59] Chunjong Park, Junsung Lim, Juho Kim, Sung-Ju Lee, and Dongman Lee. 2017. Don't Bother Me. I'm Socializing! A Breakpoint-Based Smartphone Notification System. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing* (Portland, Oregon, USA) *(CSCW '17)*. Association for Computing Machinery, New York, NY, USA, 541–554. https://doi.org/10.1145/2998181.2998189

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

[61] Martin Pielot, Linas Baltrunas, and Nuria Oliver. 2015. Boredom-Triggered Proactive Recommendations. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct* (Copenhagen, Denmark) *(MobileHCI '15)*. Association for Computing Machinery, New York, NY, USA, 1106–1110. https://doi.org/10.1145/2786567.2794340

[62] Martin Pielot, Bruno Cardoso, Kleomenis Katevas, Joan Serrà, Aleksandar Matic, and Nuria Oliver. 2017. Beyond Interruptibility: Predicting Opportune Moments to Engage Mobile Phone Users. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 3 (Sept. 2017), 91:1–91:25. https://doi.org/10.1145/3130956

[63] Martin Pielot, Rodrigo de Oliveira, Haewoon Kwak, and Nuria Oliver. 2014. Didn't You See My Message? Predicting Attentiveness to Mobile Instant Messages. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. Association for

Computing Machinery, New York, NY, USA, 3319–3328. https://doi.org/10.1145/2556288.2556973

[64] Martin Pielot, Tilman Dingler, Jose San Pedro, and Nuria Oliver. 2015. When attention is not scarce - detecting boredom from mobile phone usage. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. Association for Computing Machinery, New York, NY, USA, 825–836. https://doi.org/10.1145/2750858.2804252

[65] Benjamin Poppinga, Wilko Heuten, and Susanne Boll. 2014. Sensor-based identification of opportune moments for triggering notifications. *IEEE Pervasive Computing* 13, 1 (2014), 22–29.

[66] Nilam Ram, Xiao Yang, Mu-Jung Cho, Miriam Brinberg, Fiona Muirhead, Byron Reeves, and Thomas N Robinson. 2020. Screenomics: A new approach for observing and studying individuals' digital lives. *Journal of adolescent research* 35, 1 (2020), 16–50.

[67] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 779–788.

[68] Byron Reeves, Nilam Ram, Thomas N Robinson, James J Cummings, C Lee Giles, Jennifer Pan, Agnese Chiatti, Mj Cho, Katie Roehrick, Xiao Yang, et al. 2021. Screenomics: A framework to capture and analyze personal life experiences and the ways that technology shapes them. *Human–Computer Interaction* 36, 2 (2021), 150–201.

[69] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems* 28 (2015), 91–99.

[70] Hillol Sarker, Moushumi Sharmin, Amin Ahsan Ali, Md. Mahbubur Rahman, Rummana Bari, Syed Monowar Hossain, and Santosh Kumar. 2014. Assessing the Availability of Users to Engage in Just-in-Time Intervention in the Natural Environment. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Seattle, Washington) *(UbiComp '14)*. Association for Computing Machinery, New York, NY, USA, 909–920. https://doi.org/10.1145/2632048.2636082

[71] Hillol Sarker, Moushumi Sharmin, Amin Ahsan Ali, Md. Mahbubur Rahman, Rummana Bari, Syed Monowar Hossain, and Santosh Kumar. 2014. Assessing the Availability of Users to Engage in Just-in-Time Intervention in the Natural Environment. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Seattle, Washington) *(UbiComp '14)*. Association for Computing Machinery, New York, NY, USA, 909–920. https://doi.org/10.1145/2632048.2636082

[72] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In *IEEE International Conference on Computer Vision (ICCV)*.

[73] Jeremiah Smith, Anna Lavygina, Jiefei Ma, Alessandra Russo, and Naranker Dulay. 2014. Learning to Recognise Disruptive Smartphone Notifications. In *Proceedings of the 16th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Toronto, ON, Canada) *(MobileHCI '14)*. Association for Computing Machinery, New York, NY, USA, 121–124. https://doi.org/10.1145/2628363.2628404

[74] Julian Steil, Philipp Müller, Yusuke Sugano, and Andreas Bulling. 2018. Forecasting user attention during everyday mobile interactions using device-integrated and wearable sensors. *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Sep 2018). https://doi.org/10.1145/3229434.3229439

[75] Andriy A Struk, Jonathan SA Carriere, J Allan Cheyne, and James Danckert. 2017. A short boredom proneness scale: Development and psychometric properties. *Assessment* 24, 3 (2017), 346–359.

[76] John C. Tang, Sophia B. Liu, Michael Muller, James Lin, and Clemens Drews. 2006. Unobtrusive but Invasive: Using Screen Recording to Collect Field Data on Computer-Mediated Interaction. In *Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work* (Banff, Alberta, Canada) *(CSCW '06)*. Association for Computing Machinery, New York, NY, USA, 479–482. https://doi.org/10.1145/1180875.1180948

[77] Robert L Thorndike. 1953. Who belongs in the family. In *Psychometrika*. Citeseer.

[78] Liam D. Turner, Stuart M. Allen, and Roger M. Whitaker. 2017. Reachable but not receptive: Enhancing smartphone interruptibility prediction by modelling the extent of user engagement with notifications. *Pervasive and Mobile Computing* 40 (2017), 480–494. https://doi.org/10.1016/j.pmcj.2017.01.011

[79] Niels van Berkel, Chu Luo, Theodoros Anagnostopoulos, Denzil Ferreira, Jorge Goncalves, Simo Hosio, and Vassilis Kostakos. 2016. A Systematic Assessment of Smartphone Usage Gaps. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 4711–4721. https://doi.org/10.1145/2858036.2858348

[80] Steven Van Canneyt, Marc Bron, Andy Haines, and Mounia Lalmas. 2017. Describing Patterns and Disruptions in Large Scale Mobile App Usage Data. In *Proceedings of the 26th International Conference on World Wide Web Companion* (Perth, Australia) *(WWW '17 Companion)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 1579–1584. https://doi.org/10.1145/3041021.3051113

[81] Aku Visuri, Niels van Berkel, Chu Luo, Jorge Goncalves, Denzil Ferreira, and Vassilis Kostakos. 2017. Predicting Interruptibility for Manual Data Collection: A Cluster-Based User Model. In *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Vienna, Austria) *(MobileHCI '17)*. Association for Computing Machinery, New York, NY, USA, Article 12, 14 pages. https://doi.org/10.1145/3098279.3098532

[82] Heli Väätäjä and Paul Egglestone. 2012. Briefing news reporting with mobile assignments: perceptions, needs and challenges. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work (CSCW '12)*. Association for Computing Machinery, New York, NY, USA, 485–494. https://doi.org/10.1145/2145204.2145280

[83] Dominik Weber, Alexandra Voit, Gisela Kollotzek, and Niels Henze. 2019. Annotif: A System for Annotating Mobile Notifcations in User Studies. In *Proceedings of the 18th International Conference on Mobile and Ubiquitous Multimedia* (Pisa, Italy) *(MUM '19)*. Association for Computing Machinery, New York, NY, USA, Article 24, 12 pages. https://doi.org/10.1145/3365610.3365611

[84] Thomas D. White, Gordon Fraser, and Guy J. Brown. 2019. Improving Random GUI Testing with Image-Based Widget Detection. In *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis* (Beijing, China) *(ISSTA 2019)*. Association for Computing Machinery, New York, NY, USA, 307–317. https://doi.org/10.1145/3293882.3330551

[85] Qiang Xu, Jeffrey Erman, Alexandre Gerber, Zhuoqing Mao, Jeffrey Pang, and Shobha Venkataraman. 2011. Identifying Diverse Usage Behaviors of Smartphone Apps. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference* (Berlin, Germany) *(IMC '11)*. Association for Computing Machinery, New York, NY, USA, 329–344. https://doi.org/10.1145/2068816.2068847

[86] Xiao Yang, Nilam Ram, Thomas Robinson, and Byron Reeves. 2019. Using Screenshots to Predict Task Switching on Smartphones. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI EA '19)*. Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/3290607.3313089

[87] Nalingna Yuan, Heidi M Weeks, Rosa Ball, Mark W Newman, Yung-Ju Chang, and Jenny S Radesky. 2019. How much do parents actually use their smartphones? Pilot study comparing self-report to passive sensing. *Pediatric research* 86, 4 (2019), 416–418.

[88] Xiaoyi Zhang, Lilian de Greef, Amanda Swearngin, Samuel White, Kyle Murray, Lisa Yu, Qi Shan, Jeffrey Nichols, Jason Wu, Chris Fleizach, Aaron Everitt, and Jeffrey P Bigham. 2021. *Screen Recognition: Creating Accessibility Metadata for Mobile Applications from Pixels*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3411764.3445186

[89] Sha Zhao, Julian Ramos, Jianrong Tao, Ziwen Jiang, Shijian Li, Zhaohui Wu, Gang Pan, and Anind K. Dey. 2016. Discovering Different Kinds of Smartphone Users through Their Application Usage Behaviors. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Heidelberg, Germany) *(UbiComp '16)*. Association for Computing Machinery, New York, NY, USA, 498–509. https://doi.org/10.1145/2971648.2971696