Designing An Experience Sampling Method for Smartphone based Emotion Detection

Surjya Ghosh*, Niloy Ganguly*, Bivas Mitra*, Pradipta De†
*Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, INDIA
†Department of Computer Science, Georgia Southern University, USA
Email: surjya.ghosh@iitkgp.ac.in, {niloy,bivas}@cse.iitkgp.ernet.in, pde@georgiasouthern.edu

Abstract—Smartphones provide the capability to perform in-situ sampling of human behavior using Experience Sampling Method (ESM). Designing an ESM schedule involves probing the user repeatedly at suitable moments to collect self-reports. Timely probe generation to collect high fidelity user responses while keeping probing rate low is challenging. In mobile-based ESM, timeliness of the probe is also impacted by user's availability to respond to self-report request. Thus, a good ESM design must consider - *probing frequency*, *timely self-report collection*, and *notifying at opportune moment* to ensure high *response quality*. We propose a two-phase ESM design, where the first phase (a) balances between probing frequency and self-report timeliness, and (b) in parallel, constructs a predictive model to identify opportune probing moments. The second phase uses this model to further improve response quality by eliminating inopportune probes. We use typing-based emotion detection in smartphone as a case study to validate proposed ESM design. Our results demonstrate that it reduces probing rate by 64%, samples self-reports timely by reducing elapsed time between self-report collection, and event trigger by 9% while detecting inopportune moments with an average accuracy of 89%. These design choices improve the response quality, as manifested by 96% valid response collection and a maximum improvement of (R3,Q1)24% in emotion classification accuracy.

Index Terms—Experience Sampling Method; Notification; Smartphone; Emotion Detection

1 Introduction

The Experience Sampling Method (ESM) is a widely used tool in psychology and behavioral research for in-situ sampling of human behavior, thoughts and feelings [1], [2]. Ubiquitous use of smartphones and wearable devices helps in more flexible design of ESM, aptly termed as mobile ESM (mESM) [3], [4]. It allows collection of rich contextual information along with behavioral data at an unprecedented scale and granularity, which paves the application of mESM in different domains like personal wellbeing [5], automatic emotion detection [6], [7], social and behavioral studies [8].

Responding to survey requests in ESM is repetitive and can be burdensome, which affects *response quality* in multiple ways. For instance, a high *probing frequency* may cause user fatigue, which makes the users inattentive and respond inaccurately, or simply ignore the probes, thereby impacting label quality [9]. Similarly, if the response is not collected *timely* (e.g. if there is a long time interval between event occurrence and self-report collection), the self-report may suffer from recall bias [10]. Even when probes are generated at appropriate time without high frequency, it may still be possible that a probe request triggered at a time when the user cannot pay attention i.e. she is not *interruptible* [11]. Thus, a mESM design must consider *probing frequency*, *timeliness* and *user interruptibility* together to obtain high quality survey responses.

In state-of-the-art literature, multiple studies attempted to address this challenge by considering notification schedule and interruptibility, two key parameters in mESM design [12]. Typically the notification schedules are policy-based - the policy determined by time, or specific events. Time-driven policies are used to probe at fixed interval often aiming to reduce the *probing frequency* [6], [8], while event-driven policies aim to collect self-reports *timely* by probing as soon as the event occurs [13], [14]. Reducing probe frequency while triggering a probe on every event is inherently contradictory. Hence hybrid schedules are designed that leverage policy-based schedules to balance between probing frequency and timely self-report collection [15] . In spite of these advanced schedules, it is not guaranteed that probes are generated at *interruptible* moments by policy-based schedules. Studies have also investigated interruptibility-aware designs in smartphone notifications [16], [17] and indicated that suitable engagement moments can be detected based on mobile sensors details, contextual and demographic information [18], [19]. Recent studies related to mobile notification demonstrated that carefully selected probing moments lead to better engagement and improved response quality [20], [21], [22].

ESM designs are more stringent in cognitive demand compared to interruptibility aware notifications since a user must actively recall and record, unlike reading an information. Moreover, in case of ESM design, the undergoing user study plays an important role [23]. For example, an appropriate moment to prompt user for a health related intervention is unlikely to be the same as one to notify her of a social network or email update. So, the probing moments in ESM design should not only be opportune, but also *relevant* with respect to the ongoing study. In addition, use of sensor details and contextual information may not be viable in ESM design. Tracking additional sensor and contextual details may not be resource-efficient, also user may not agree to allow to track her senor data (like location

details, call logs) for privacy reasons [24]. As a result, the opportune moments for probing should be learned from the *user inputs* itself.

In this paper, we overcome these limitations in steps by proposing a two-phase ESM design. In first phase, we identify relevant probing moments based on the user study and in second phase, we implement a machine learning model so that the probing is done at opportune moments only. We focus on balancing *probing frequency* and *timeliness* in phase-1, by deploying a policy-based ESM schedule named as *LIHF* (Low-Interference-High-Fidelity) schedule. In parallel, based on user inputs, we construct a machine learning model to detect if the current probing moment is opportune or not. In phase-2, we operationalize this model, which automatically identifies and skips probing at *inopportune* moments. As the model becomes operational, it can improve response quality not only by collecting responses at appropriate moments, but also eliminating the potentially annoying probes (thereby increasing the fraction of valid probes). Thus the proposed design optimizes probing rate, self-report timeliness and user interruptibility together for better response quality, both in terms of number of valid responses and accuracy of these responses.

We use the smartphone based emotion detection, which extensively uses ESM for survey report collection [6], [7], as a case study to evaluate the proposed ESM framework. We design, implement and deploy a typing based emotion detection application *TapSense* for Android platform to determine multiple emotions (*happy, sad, stressed, relaxed*) and validate the efficacy of the proposed method in a 3-week study involving 22 participants. Our major experimental results demonstrate that ESM schedule designed based on proposed framework (a) reduces the probing frequency by 64% (b) collects the self-reports more timely with a reduction of 9% in average elapsed time between self-report sampling and event occurrence (c) detects inopportune moments with an average accuracy (AUCROC) of 89%. The proposed design also helps to improve survey response quality by (a) improving valid response rate to 96% and (b) yielding a maximum improvement of (R3,Q1)24% in emotion classification accuracy (AUCROC) over off-the-shelf ESM schedules, while achieving an average emotion classification accuracy (AUCROC) of 78%.

In summary, the key contributions of this paper are:

- A two-phase ESM schedule design method, which balances survey probe frequency and timeliness of probe generation, while ensuring that the generated probe will be at an opportune moment for the user. The *first phase* implements a hybrid schedule, *LIHF* ESM, to balance between probing rate and timely self-report collection. It also collects data from user responses to construct a machine learning model to predict inopportune probing moments. In the *second phase*, the model is used to ensure probing only at opportune moments.
- A case study of the proposed ESM design in typing-based emotion detection in smartphone, which reveals the efficacy of the proposed method in terms of probing rate, timely self-report collection, and response quality.

The rest of the paper is organized as follows. We present related literature in Section 2. We describe the preliminary field study in Section 3 which leads to the design of the proposed approach in Section 4. We present a case study of typing based emotion detection in Section 5. We discuss the design of *TapSense* along with study procedure and participants details in Section 6. We analyze the collected dataset in Section 7. Experimental evaluation, qualitative assessment and limitations are presented in Section 8. We conclude in Section 9.

2 Related Work

In this section, first we discuss the use of mobile device as a data collection platform for ESM studies. Next, we present literature on notification schedules that balance probing frequency and timely self-report collection, followed by user interruptibility aware ESM designs.

2.1 Smartphone-driven Data Collection

Smartphones can non-intrusively collect sensor information, application usage data, user's contextual information. For example, Device Analyzer collects approximately 300 event details related to telephony, WiFi network, application usage, data usage, sensors etc., and use the same to infer details like mobility pattern, communication trend, WiFi network availability, battery usage and the reliability of smartphone for long-term data collection [25]. The UbiqLog framework was designed to trace life-log events by configuring or adding new sensor details [26]. Multiple studies have explored the collected sensor data and inferred user's context [27], [28], [29]. For example, Ferreira et al. designed open-source platform AWARE to capture, infer and generate context based on sensor data in mobile devices [28]. Similarly, a middleware platform ACE was designed for continuous context sensing in mobile platform by reducing the sensing cost [30]. The dependency on cloud platform for context determinations has been reduced by designing MobileMiner, which identifies frequently co-occurring contexts on mobile device [29].

All these works establish the suitability of smartphone as a data collection platform, which helps to obtain context information from logged data. While these frameworks help in automatic logging of sensor data, self-reports related to various aspects of human life (like emotion) still require explicit input from the user.

2.2 Balancing Probing Rate and Self-report Timeliness

In ESM studies, the participant burden mainly arises from answering survey questions repeatedly. With the proliferation of ubiquitous mobile devices, like smartphones, and other wearable devices, more intelligent and less intrusive survey schedules (e.g. limiting the maximum number of probes, increasing the gap between two consecutive probes) have been designed. Several open source software platforms, like ESP [31], MyExperience [13], PsychLog [32], Personal Analytics Companion [33], are available on different mobile computing platforms to cater to ESM experiments.

Time-based, event-based schedules are most commonly used ESM schedules [10]. Time-based approaches aim to reduce probing rate, while event-driven ones try to collect self-report timely. However, time based approaches do not

guarantee fidelity of the labels as the response may attenuate by the time user records the label. Similarly, although event-based ESMs collect labels close to the events but if the number of events monitored are high in number, there will be large number of probes, resulting into user annoyance. Recently, hybrid ESM schedules are designed combining time-based and event-based ones to trade-off between probing rate and self-report timeliness [15].

2.3 Maintaining Response Quality via Interruptibility-aware Designs

Recent advancements in interruptibility-aware notification management revealed that users receive many probes in a day and all of these are not equally important [34]. Different solutions were proposed to regulate the probing on mobile devices, which primarily leverage on contextual information to infer opportune moments [20], [35]. Ho et al. indicated that interruptions may be considered more positively if placed between two physical activities like sitting and walking [35]. Similarly, Fischer et al. showed that participants react faster to probes when they are delivered immediately after completing a task on mobile such as after finishing a phone call or reading a text message [20]. In [36], authors showed that features like last survey response, phone's ringer mode and user's proximity to the screen can be used to predict whether an intimation will be seen by the recipient within a few minutes. Leveraging on these findings, intelligent notification strategies were developed, which resulted in higher compliance rate and improved response quality [21], [22].

Although these findings indicate that off-the-shelf notification management approaches can be applied in mobile based ESM design, in reality they may not be. The primary reason is that these approaches use different contextual and sensor details, which may not be available during an ESM design because of resource overhead and privacy issue. Moreover, ESM design also depends on underlying user study [23]. As a result, the opportune probing moments are to be extracted based on the target user study.

Schedule	Probing rate	Timeliness	Opportune probing
Time-based (e.g. [6], [8])	✓	✗	✗
Event-based (e.g. [13], [14])	✗	✓	✗
Hybrid (e.g. [15])	✓	✓	✗
Interruptibility-aware (e.g. [16], [17], [20], [21])	✗	✗	✓
Proposed two-phase ESM	✓	✓	✓

TABLE 1: Summarization of the related works reveals the scope of optimizing probing rate, self-report timeliness and probing at opportune moments together in mobile-based ESM design.

We summarize the findings from the literature survey in Table 1. Comparison of different ESM schedule design reveals the scope of optimizing probing rate, self-report timeliness and probing at opportune moments for better survey response quality in mobile-based ESM design, which is addressed in this work.

3 Motivational Study

The objective of this pilot study is to find the limitations of policy-based ESMs in terms of probing rate and timely self-report collection. It also aims to identify how participants respond if probes are issued frequently without considering user attention.

3.1 Experiment Apparatus

In order to conduct this study, we design a virtual keyboard for Android platform and use it as experiment apparatus. We also design a emotion self-report collection UI. We show the keyboard and the self-report collection interface in Fig. 1, 2 respectively. In the self-report collection UI, we include the *No Response* option so that users can skip self-reports if the popup appears at an inopportune moment.

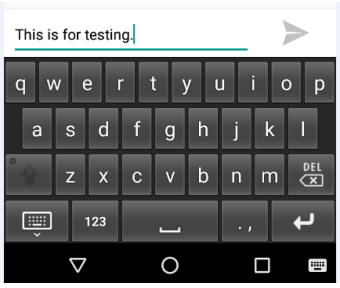


Fig. 1: Smartphone keyboard used in the experiment to capture users' typing activity.

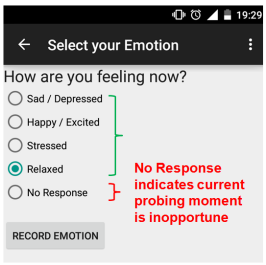


Fig. 2: User Interface for collecting emotion self-reports.

3.2 Study Procedure

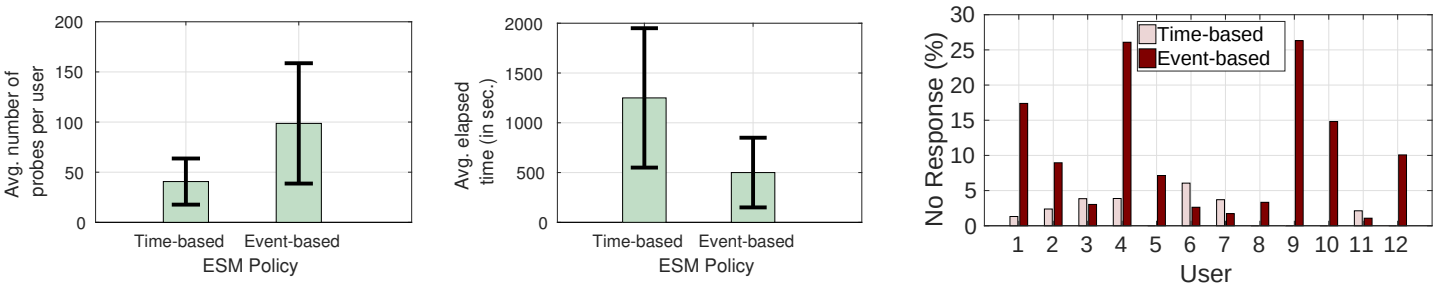
We conduct two studies as described below. In the first study, we deploy a time-based ESM policy [10], which collects emotion self-reports at every 3 hour interval. In the second study, we collect the emotion self-reports using an event-based ESM policy [10], which considers switching from a typing based application as the event and issues probes at the occurrence of every such event.

We installed the ESM application on smartphones of 12 university students (aged between $18-24$ years, 10 male, 2 female) and recorded their emotion labels and typing patterns for 2 weeks using each ESM policy. We instructed the users to make the study keyboard as the default one and asked them to record their emotion in the survey popup. We also instructed them to select *No Response* option and not to discard the pop-up by pressing the back button if they feel that the popup appeared at an inopportune moment.

3.3 Lessons Learnt

We record the response rate of both the policy-based approaches before comparing them. We note $\approx 98\%$ valid self-reports (2% *No Response*) and $\approx 82\%$ valid self-reports (18% *No Response*) for time-based and event-based schedules respectively. Both of these policy-based schedules are compared in terms of (a) average number of probes issued per user (b) average elapsed time between typing completion and self-report collection and (c) the percentage of *No Response* labels. We summarize these results in Fig. 3.

We observe that in case of event-based policy, a high number of probes are issued in comparison to time-based policy, while the event-based policy collects the self-reports more close to the event than time-based one. We also



(a) Avg. number of probes per user (b) Avg. elapsed time (c) Amount of *No Response* labels

Fig. 3: Comparing time-based and event-based ESM schedule in terms of (a) average number of probes issued per user (b) average elapsed time between typing completion and self-report collection and (c) percentage of *No Response* labels for every user

observe that since event-based policy issues a very high number of probes, a large fraction (18%) of these probes are marked as *No Response*.

The aforesaid studies reveal that there is a trade off between probing frequency and timely self-report collection in case of policy-based ESM schedules. They also reveal that if the probes are issued at very high frequency, they are not useful and may be skipped by the users. These observations indicate the need to balance between probing rate and timely self-report collection and regulate the probing at unfavorable moments. These observations motivate us to design an ESM schedule to address these issues together.

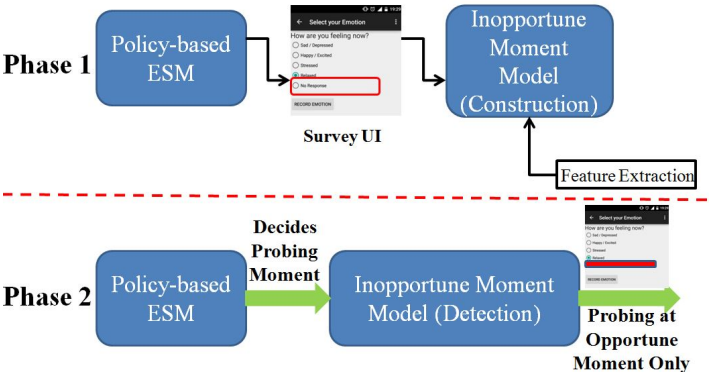


Fig. 4: Schematic of the proposed ESM design method. In phase 1, only policy-based ESM schedule is used to issue probes balancing probing rate and self-report timeliness. In parallel, it constructs the inopportune moment detection model based on user responses. In phase 2, policy-based ESM schedule determines the probing moments, which are checked by the model to ensure probing at opportune moments only.

4 Proposed ESM Design Approach

The proposed ESM design is divided into two phases. We show both these phases in Fig. 4. The first phase is driven by a policy based schedule, which issues probes based on predefined rules. The objective of this phase is to reduce the probing rate and ensure that self-reports are collected timely. Side by side, this phase provides the opportunity to learn the inopportune moments from user responses. The survey response collection UI is provided with a *No Response* option so that the user can indicate that the probe appeared at an inopportune moment. Based on the user reported labels, a machine learning model is constructed to detect the inopportune moments automatically. Finally, in the second phase, the probes generated based on the policy-based module are passed to this inopportune moment detection model. Subsequently, the model decides if the current moment is inopportune, and accordingly the probing is skipped (or issued, otherwise) to the user. Next, we describe each of these phases in detail.

4.1 Phase-1: Hybrid Policy to Balance Probing Rate and Timeliness

In this phase, we have three tasks to perform - (a) balancing between probing rate and self-report timeliness using policy-based schedules (b) collecting self-reports from the users and (c) constructing the inopportune moment detection model.

4.1.1 Balance Probing Rate and Self-report Timeliness

Depending on the nature of the study, we may decide to deploy an appropriate policy-based schedule in this phase. For example, if the focus of the study is to limit number of probes, we may use a time-based schedule with high inter-probing gap. Similarly, if we want to collect the self-reports timely, we may use an event-based schedule. In order to balance both, we use a combination of these two. We name this scheduling policy as LIHF (Low-Interference-High-Fidelity), which tries to strike a balance between probing frequency and timely self-report collection [15].

Algorithm 1: LIHF ESM Schedule

```
Input: EventLog, ESMLog
Output: ESM Probe
while true do
    E ← Detect event of interest
    N ← Set of events in EventLog table
    /* Check if any event of interest has occurred */
    if (E ∈ N) then
        t1 ← Timestamp of last ESM probe
        t2 ← Current timestamp
        /* Check if minimum time has been elapsed since
           last probing */
        if time difference (t2, t1) > W then
            /* Check if the screen is locked */
            if isScreenLocked() then
                Do not fire ESM probe
            else
                1. Fire ESM probe
                2. Update ESM probe timestamp in ESMLog table
    Sleep T seconds;
```

The primary objective of the LIHF policy is to identify the probing moments in such a manner that there is sufficient gap between two ESM probes and at the same time self-reports are collected close to the event of interest. We outline the LIHF policy in Algorithm 1.

This hybrid policy-based approach triggers a survey request only if (a) event of interest E has occurred and (b) a minimum time W has elapsed since the last probe. The algorithm monitors if any event (E) has occurred that belongs to predefined set of events N (*EventLog*). If so, it checks the elapsed time since last probing as maintained in *ESMLog* table. In case the elapsed time is greater than the W, the probe is issued, otherwise not. However, it may so happen that once a probe is about to be issued, the screen is locked which may delay the label collection. We issue the probe as soon as the screen the unlocked but eliminate delayed responses during data processing. We decide not to consider these probes because user response may get attenuated because of the delay induced by the locked screen. The collection of two consecutive events (E) is separated by introducing sleep parameter (T).

4.1.2 Self-report Collection

We use same survey collection UI as shown in Fig. 2. We collect the self-report responses from the participants; notably if any probe response is recorded as *No Response*, it indicates that the probe has been issued at inopportune moment. All other valid responses indicate the probing moment as opportune. Hence, the outcome of *Policy-based ESM Schedule* provides the collection of self-reports, which contain the ground truths regarding the inopportune moments.

4.1.3 Inopportune Moment Model Construction

We leverage on the collected survey responses and information related to the ESM study (say App usages etc) to construct the inopportune moment detection model. This is a two-state classification model which identifies the probing moment as inopportune or not. We extract the features associated from the event of interest and use them to construct a machine learning model to detect the inopportune moments. The evaluation of the model is performed based on the ground truth inopportune moments, collected from the survey responses. The model feature identification step is not generalized and depends on the events specific to the application for which the ESM schedule is being designed. We discuss one case study in section 5.

4.2 Phase-2: Model based Approach to Predict Opportune Probing Moment

At the end of first phase, we have developed the ESM with two capabilities - (1) balance the probing rate and timely self-report collection and (2) ensure that no probe is issued at an inopportune moment based on a machine learning model. In phase-2, the inopportune moment detection model becomes operational. In this phase, the survey UI does not contain any *No Response* label and the model decides whether the probing should be done or not. We outline the second phase in Algorithm 2. We invoke the *Policy-based ESM Schedule* used in phase-1 to generate the probes (line 2). Next, we check if the probing rules have been satisfied and the probe is generated by the policy-based ESM (line 3). If the probe is generated, the model determines whether the current moment is inopportune or not (line 4 – 6). If the model finds that the current moment as inopportune (line 7), it skips the probe (line 8), otherwise the probe is fired (line 10).

Algorithm 2: Phase 2 of ESM Schedule Design

```
   Input: Inopportune moment detection model (M)
   Output: ESM Probe
1  while true do
       /* Determine probing based on policy-based ESM
          first                                          */
2      probe ← isProbeGenerated()
       /* Check if policy-based ESM generates the probe */
3      if (probe == TRUE) then
4          ev ← Identify event of interest
5          [f] ← Extract features from ev
6          pred ← M.predict(f)
7          if (pred == inopportune) then
8              Do not fire ESM probe
9          else
10             Fire ESM probe
```

5 Case Study : Typing based Emotion Detection in Smartphone

In this paper, we focus on typing based emotion detection application in smartphone as case study, as this application heavily relies on ESM for collecting emotion self-reports, which are used as ground truth to build the emotion detection model.

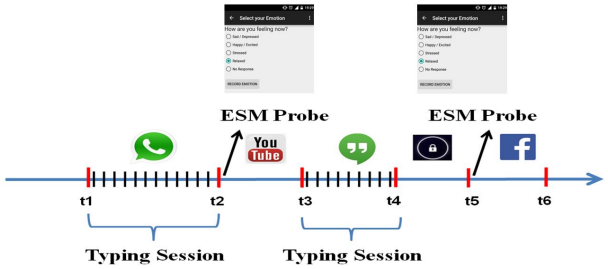


Fig. 5: Schematic of typing based emotion detection scenario. For example, time interval between `t1` and `t2` is considered a *typing session*, where each small bar within this *session* is a key pressing event. ESM probe to collect emotion label provided between `t2` and `t3` is associated with this *typing session*.

We develop *TapSense*, a typing based emotion detection application in smartphone. Fig. 5 shows the scenario of typing based emotion detection. As a user performs typing activity, we extract her *typing sessions*, the time period one stays onto a single application without changing the same. Subsequently, based on the ESM probes, the self-report is collected after each *typing session* and attached with it. We use the same self-report collection UI as shown in Fig. 2. This survey questionnaire provides the option (*happy, sad, stressed, relaxed*) to record ground truth about the emotion experienced by the user while typing. This captures four largely represented emotions from four different quadrants of the Circumplex model [37] as shown in Fig. 6. We select these discrete emotions as their valence-arousal representation is unambiguous on Circumplex plane. Any discrete

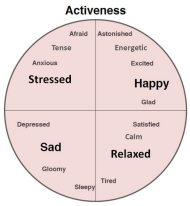


Fig. 6: Circumplex Model of emotion showing four emotions having unambiguous valence and arousal.

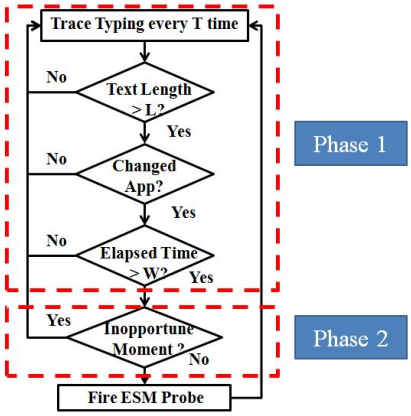


Fig. 7: End-to-end probe generation steps in context of typing based emotion detection when both the phases are operational.

emotion and it's unambiguous representation on valence-arousal plane are equivalent [38]. Moreover selecting emotion states from different quadrants helps user to distinguish them well during self-reporting. We also include the *No Response* option so that the user can select this option to indicate the current probing moment is inopportune.

We customize the proposed ESM schedule and integrate with *TapSense* for self-report collection. In this section, we focus on customizing the proposed ESM schedule. Subsequently, in section 6, we discuss the realization of *TapSense*.

5.1 Implementation of Proposed ESM for *TapSense*

The customization of the proposed two-phase ESM for *TapSense* is summarized in Fig. 7. In phase 1, we combine policy-based schedules to balance probing rate and timeliness and construct the inopportune moment detection model and in phase 2, we make the inopportune moment detection model operational.

5.1.1 Balancing Probing Frequency and Timeliness

We implement the LIHF schedule (Algorithm 1) by customizing in context of typing based emotion detection. In this context, we define the *event of interest* as the end of a *typing session*. This policy would help to collect the labels close to the event, but at the same time probing at the end of every *typing session* would lead to generation of too many probes. In order to trade off these two conflicting requirements, we make sure that there is *sufficient* amount of typing done in a *typing session*. We decide to issue the probe only (a) if the user has performed sufficient amount of typing ($L = 80$ characters) in a *typing session* and (b) a minimum time interval ($W = 30$ minutes) has elapsed since the last ESM probe. In order to ensure the labels are collected close to *typing session*, we use the polling interval parameter ($T = 15$ seconds), which checks at every interval T if the user has performed sufficient amount of typing. We describe the selection of threshold values based on initial field trials in Appendix A[1]. We show the flow-chart of modified version of LIHF schedule in Fig. 7 (Phase 1).

1. Submitted as supplemental material

5.1.2 Inopportune Moment Detection Model

As we are collecting self-reports, we obtain both *No Responses* and valid emotion responses. We leverage on these labels to build the inopportune moment detection model. We use typing duration and the typing length in a *session* as features, since lengthy and longer *typing session* may indicate high user engagement and not be the ideal moment for triggering a probe. In addition, there may be some types of applications like media, games when the users may not be interrupted for probing. So, we include application type also as a feature. We categorize the applications into one of the 7 classes (`Browsing`, `Email`, `Media`, `Instant Messaging (IM)`, `Online Social Network (OSN)`, `SMS` and `Misc`) following the description of the application in the play store. Moreover, we use the label of last ESM probe response as a feature. We use it to determine whether user continues to remain occupied in current *session*, if she marked the previous *session* with *No Response*. During model construction, we can easily get this label from the user self report of the last *session*. However, once the model is deployed and the users stop providing the *No Response* label, we use the predicted value of inopportune moment for last *session* as feature value for the current *session*. Table 2 summarizes the features used to implement the model. We construct an all-user Random Forest based model to detect the inopportune moments. Once the model is constructed, it is augmented with the modified LIHF schedule to detect and eliminate inopportune probes (Fig. 7 (Phase 2)).

Feature Name	Feature Description
Session duration	Duration of the *typing session*
Session length	Length of the text in the *session*
App category	Category of the application
Last ESM probe response	Last ESM probe response

TABLE 2: Features used to detect inopportune moments

6 TapSense: Design and Implementation

In this section, we discuss the design, development and deployment of *TapSense*, an Android based application to determine emotion from typing activity in smartphone. We use it as the experiment apparatus to conduct the field study.

6.1 Experiment Apparatus

We show the architecture of *TapSense* application in Fig. 8. It consists of following key components. *TapLogger* implements an Input Method Editor (IME) [39] provided by Android and we refer it as the *study keyboard*. It is similar like QWERTY keyboards with additional capabilities of tracing all typing details. We do not capture or store any text to preserve user privacy. *ProbeEngine* runs on the phone, generates the notifications and collects the user responses. In first phase of ESM design, it implements *LIHF* as Policy-based ESM schedule and collects emotion self-reports using the survey UI (Fig. 2). The typing details and the associated emotion self-reports available at the server via *Uploader* module once the user connected to Internet. Based on the typing data and self-reports the *inopportune moment detection model* is constructed and integrated with *ProbeEngine* to optimize probing in second phase. In parallel, the *emotion*

detection model is also constructed to determine the different emotion states based on typing. We implement both these models in Weka [40] using Random Forest by using 100 decision trees with maximum depth of the tree as unlimited.

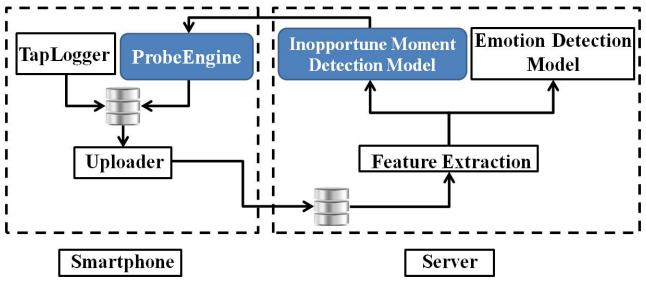


Fig. 8: *TapSense* application architecture. *TapLogger* records user's typing activity and *ProbeEngine* issues probes and collects user responses. Upon receiving these details on the server, features are extracted to build the inopportune moment detection model. Finally, the model is integrated with *ProbeEngine* to optimize probing. In parallel, the emotion detection model is also constructed to detect multiple emotion states from typing.

6.1.1 Emotion Detection Model

The emotion detection model in *TapSense* detects four emotion states (*happy, sad, stressed, relaxed*) based on smartphone typing. The intuition is that if the survey responses collected by proposed ESM are of high quality and capture emotion self-reports correctly, the model can produce high emotion classification performance. We design a personalized multi-state emotion detection model using Random Forest to detect the emotions.

Feature Name	Feature Description
Session typing speed	Average of all elapsed times between two consecutive typing events in the *session*
Session length	Length of the typed text in the *session*
Session duration	Time duration of the *session*
Backspace percentage	Percentage of backspace and delete keys typed in the *session*
Special character percentage	Percentage of special characters in the *session*
Last ESM probe response	Emotion label as provided by the user

TABLE 3: Features used to construct emotion detection model

We summarize the features used for emotion detection in Table 3. We use typing speed as a feature. For every *session*, we compute the time interval between consecutive tap events. We find the mean of all such time intervals present in a *session* and use it as typing speed. We compute the fraction of backspace and delete keys present in a *session* and use it as a feature. Similarly, we use the fraction of special characters in a *session*, *session* duration and length of typed text in a *session* as features. Any non-alphanumeric character is considered special character. We also use last emotion self-report as a feature to build the model, because emotion states persist over time and current emotion may often be influenced by the previous one [41], [42]. During emotion model construction, we obtain this label from the previous emotion self-report. However, when the model is operational, we use the predicted emotion for last *session* as the feature value for the current *session*.

6.2 Field Study

6.2.1 Survey Focus Group

We recruited 28 university students (22 male, 6 female, aged $24-35$ years) to deploy *TapSense*. We installed the application on their smartphones and instructed them to use it for 3 weeks. Three participants left the study in between and other three participants have recorded less than 40 labels. So, we have discarded these 6 users and collected data from the remaining 22 participants (18 male, 4 female).

6.2.2 Instruction and Study Procedure

During the field study, we execute only first phase, where we implement *LIHF* schedule as policy-based ESM Schedule. We instructed participants to select the *study keyboard* as the default keyboard. We informed the participants that when they switch from an application after completing typing activity, they may receive a survey questionnaire as a pop-up, where they can record their emotion. We also advised the participants not to dismiss the pop-up if they are occupied; instead they were asked to record *No Response* if they do not want to record emotion at that moment.

7 Data Analysis and Feature Identification

In this section, first we describe the collected dataset followed by detailed analysis of the dataset in terms of (a) analysis of *No Responses* and (b) feature analysis of the inopportune moment detection model.

Total typing events	942,827
Total typing sessions	4,609
Total typing duration (in Hr.)	199.1
Mean typing sessions (per user)	209 (Sd: 167.2)
Minimum number of typing sessions for a user	46
Maximum number of typing sessions for a user	549

TABLE 4: Final dataset details

7.1 Dataset

During this study period, we have collected $4,609$ *typing sessions*, which constitute close to 200 hours of typing. Out of these *sessions*, we record a total of 642 number of *No Response* sessions, which is nearly 14% of all recorded *sessions*. We summarize the final dataset in Table 4.

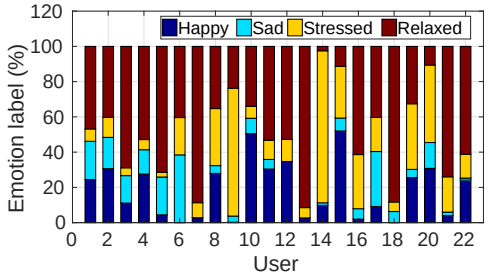
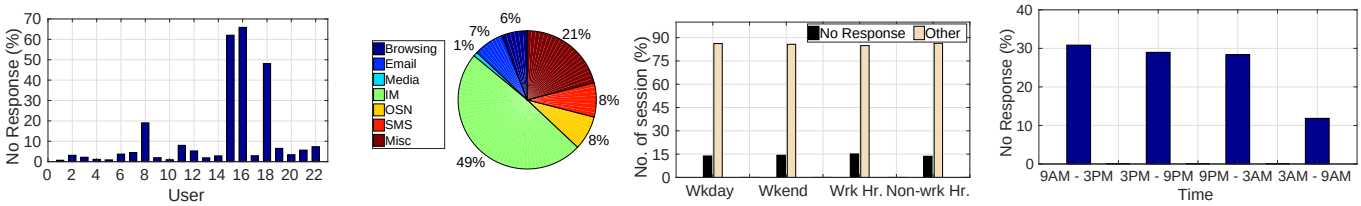


Fig. 9: Distribution of emotion labels for every user. All but 5 users ($U6, U7, U12, U13, U18$) have recorded four emotions.

The users have reported two types of responses - (a) One of the four valid emotions or (b) *No Response* label. We show the distribution of different emotion states for every user in Fig. 9. We have observed that for most of users *relaxed* is the most dominant emotion state. Overall we have achieved $14\%, 9\%, 30\%, 47\%$ sessions tagged with *happy, sad, stressed* and *relaxed* emotion states.



(a) User-wise distribution (b) App-wise distribution (c) Time-wise distribution (d) Hour-wise distribution

Fig. 10: Distribution of *No Response* sessions. (a) User-wise distribution measures fraction of total sessions marked as *No Response* for each user (b) App-wise distribution indicates fraction of total *No Response* sessions collected from each app (c) Time-wise distribution compares fraction of total *No Response* sessions and fraction of total *Other* sessions collected at different times (d) Hour-wise distribution indicates the fraction of total *No Response* sessions collected at specified time window.

7.2 No Response Analysis

We show the user-wise distribution of *No Response* sessions in Fig. 10a. Although for most of the users, the fraction of *No Response* labels are relatively low, for few users it is more than 40%, which can be attributed to personalized self-reporting behavior. We observe the application-wise distribution of *No Response* sessions in Fig. 10b, which indicates that majority of the *No Response* labels are associated with `Instant Messaging (IM)` applications like WhatsApp. We also compare the distribution of *No Response* and other valid emotion labels at weekday, weekend, working hour and non-working hour in Fig. 10c. We compute the percentage of total *No Response* and percentage of total *Other* sessions are recorded at each of these time period. However, in our dataset, we do not observe any difference among these distributions. We also explore the time-wise distribution of *No Response* sessions in Fig. 10d, which indicates that during late night from 3 AM onwards, few number of *No Response* sessions were recorded. This can be attributed to overall less engagement during late night.

7.3 Inopportune Moment Detection Features

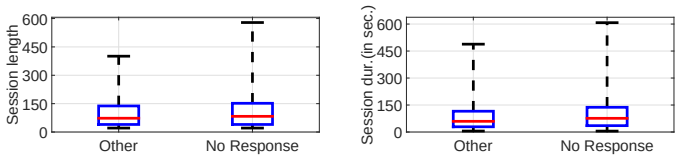
We illustrate the utility of different features used to detect inopportune moments, which are realized by the *No Response* labels.

7.3.1 Typing Session Length and Duration

We compare the session length and session duration for *No Response* and *other* sessions in Fig. 11a and 11b respectively. We observe that *sessions* marked with *No Response* are comparatively lengthy and longer than other *sessions*. We validate these performing a t-test. Before applying t-test, we verify the normality using one-sample Kolmogorov-Smirnov (KS) test [43]. We observe a significant ($p < 0.05$) difference in mean session length of *No Response* and other *sessions*. We observe the similar effect for mean session duration. Intuitively, this finding indicates that when participants are engaged in lengthy and longer typing conversations, they may not like to get the ESM probes and decline the probe by selecting *No Response*.

7.3.2 Application Category

We also compare the distribution of *No Response* and other valid emotion labels for each application category in Fig. 12. We observe that there is a comparatively high number of *No Response* sessions triggered when the participants are



(a) Session length comparison (b) Session duration comparison

Fig. 11: Comparison of session length and session duration for *No Response* and other sessions. Mean session length and session duration are found to be significantly ($p < 0.05$) different between two groups using t-test.

engaged with `IM`, `Email` content. On the contrary, users responded with valid emotion labels if probes are issued during `SMS` or `OSN (Online Social Network)` engagements. We also find that the difference in app usage in *No Response* and *Other* emotion state is significantly ($p < 0.05$) different using chi-square test [df=6, chi-square stat=87.98].

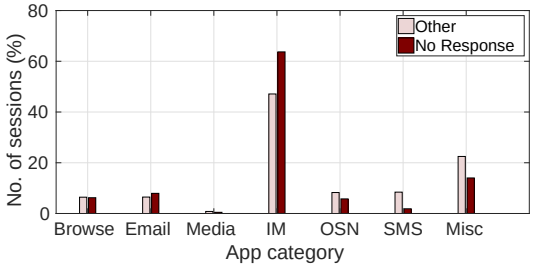


Fig. 12: Comparison of application usage distribution for *No Response* and *other* sessions. The differences in app usage between these two groups are found statistically significant ($p < 0.05$) using chi-square test.

7.3.3 Previous ESM Probe Response

Finally, we investigate if a participant marks the current *session* with *No Response*, how likely she is going to label the next *session* again with *No Response*. This helps to understand if the user is currently occupied, whether she will remain so in near future. We compute the transition probability to *No Response* from other states; i.e. probability of obtaining next state as *No Response* from each of the 5 states (*happy, sad, stressed, relaxed, No Response*). Similarly, we measure the transition probability of any other valid emotion state from each of these 5 states. We plot both these values in Fig. 13. We observe that approximately 78% of cases the next state is labeled as *No Response*, given the current state is labeled with *No Response*. We also perform chi-square test, which reveals that both these transition

probabilities from current state to next state (*No Response* or *Other*) vary significantly ($p < 0.05$) [df=4, chi-square stat=2465.7].

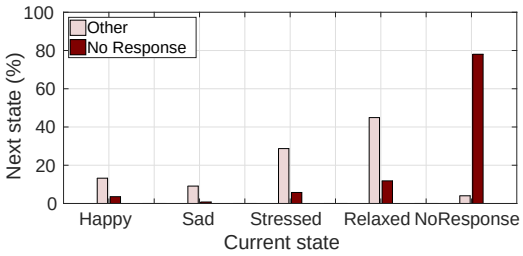


Fig. 13: Transition probability from current state to *No Response* or *Other* as next state. Comparatively high probability of obtaining *No Response* as next probe response if current probe response is selected as *No Response*. The differences in transition probabilities from any state to these two states are found statistically significant ($p < 0.05$) using chi-square test.

8 Evaluation

We evaluate the ESM design with respect to the (a) policy-based ESM schedule and (b) inopportune moment detection model. The metrics used in the evaluation are introduced first.

8.1 Experiment Setup

During field study, we used LIHF ESM schedule for collecting self-reports. However, in order to perform a comparative study across different policies, we require data from time-based and event-based ESM schedules under *identical experimental conditions* from every participant. In actual deployment, identical conditions are impossible to repeat over different time frames. Hence, we generate traces for the other policy-based schedules from the data collected using LIHF ESM. We outline the generation steps in Appendix B[2]. We also show the distribution of emotion labels obtained from different schedules after trace generation in Fig. 14.

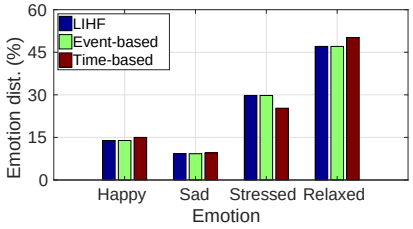


Fig. 14: Frequency distribution of different emotions across various ESM schedules. The distribution is found to be identical.

8.1.1 Baseline ESM Schedules

Different ESM schedules, listed in Table 5, used for comparison are described.

- *Policy-based ESM:* We use three policy-based ESM schedules - *TB, EB* and *LIHF*. In case of *TB*, probes are issued at a fixed interval (3 hours). In case of *EB*, after every typing session a probe is issued, while *LIHF* implements the LIHF policy as outlined in Algorithm 1. These approaches do not use an inopportune moment detection model. Comparison of these schedules help to understand their effectiveness in reducing probing rate and collecting self-reports timely.
- *Model-based ESM:* We use following model-based ESM schedules - *TB-M, EB-M* and *LIHF-M*. These ESM schedules implement *TB*, *EB* and *LIHF* schedule in phase 1 respectively followed by the inopportune moment detection model operational in phase 2. In all these schedules, the model is constructed using same set of features (Table 2) extracted from relevant trace (i.e. for *TB-M* the model is constructed from trace of *TB* and similarly). Comparison of these model-driven schedules helps to understand the efficacy of the model in detecting the inopportune moments and whether applying the model with any off-the-shelf ESM is good enough to improve survey response quality.

ESM Schedule	Phase 1	Phase 2
TB	Time-based	No model is used
EB	Event-based	No model is used
LIHF	LIHF	No model is used
TB-M	Time-based	Inopportune moment detection model
EB-M	Event-based	Inopportune moment detection model
LIHF-M	LIHF	Inopportune moment detection model

TABLE 5: Different ESM schedules based on the policy used in phase 1 and usage of the model in phase 2.

8.1.2 Performance Metrics

We use the following metrics to measure the probing rate reduction, timely self-report collection, inopportune moment identification and survey response quality improvement.

(i) Probe Frequency Index

We compare the probing frequencies of different ESM schedules. We measure it using the metric Probe Frequency Index (*PFI*), defined as follows. Let there be different ESM schedules ($e \in E$) and N_i^e denotes the number of probes issued for user i for an ESM schedule e, then *PFI* for user i for ESM schedule e is expressed as, $PFI_i^e = \frac{N_i^e}{\forall e, max(N_i^e)}$.

(ii) Recency of Label

The timeliness of survey response collection is measured using Recency of Label (*RoL*). *RoL* compares the elapsed time between an event, i.e. completion of *typing session*, and the survey request to collect emotion label. Let there be different ESM schedules ($e \in E$) and d_i^e denotes average elapsed time between typing and probing for user i for an ESM schedule e, then *RoL* for user i for ESM schedule e is expressed as, $RoL_i^e = \frac{d_i^e}{\forall e, max(d_i^e)}$.

(iii) Inopportune Moment Identification

We measure Precision, Recall and F-score for inopportune moment detection. We have two classes *inopportune* and *opportune*, accordingly we measure TP, FP, FN and TN (true positive, false positive, false negative, and true negative). We also use AUCROC (Area under the Receiver Operating Characteristic curve) as the classification metric. We compute weighted average of AUCROC (auc_{wt}) using AUCROC for inopportune and opportune moments as

2. Submitted as supplemental material

follows. Let f_i, auc_i indicate the fraction of samples and AUCROC for class i respectively, then auc_{wt} is expressed as, $auc_{wt} = \sum_{\forall i \in \{inopportune, opportune\}} f_i * auc_i$.

(iv) Survey Response Quality

We measure the survey response quality using (a) emotion classification accuracy and (b) number of valid emotion labels (*happy, sad, stressed, relaxed*) collected.

Emotion Classification Accuracy: The performance of supervised learning algorithms highly depends on quality of labels [44]. The label quality can adversely impact classification accuracy [45], [46]. So, we use classification accuracy to measure survey response quality, as it indicates that randomly reported labels at inopportune moments can impact classification performance, eliminating those probes at inopportune moments can improve quality of labels and overall classification performance in turn. For this purpose, we compare the emotion classification performance obtained using different schedules. We evaluate the emotion detection models for every user using 10-fold cross validation and use AUCROC as the performance metric. We compute the weighted average of AUCROC (auc_{wt}) using AUCROC from four different emotion states. Let f_i, auc_i indicate the fraction of samples and AUCROC for emotion state i respectively, then auc_{wt} is expressed as, $auc_{wt} = \sum_{\forall i \in \{happy, sad, stressed, relaxed\}} f_i * auc_i$.

Valid Response Rate: We compute the percentage of valid emotion labels also. It identifies whether by probing at opportune moments, the number of valid responses has increased or not. Let there be different ESM schedules ($e \in E$) and nr_e denotes the fraction of *No Response* sessions recorded for ESM e, then Valid Response Rate (*VRR*) for ESM e is expressed as, $VRR_e = (1 - nr_e) * 100$.

8.1.3 Evaluation procedure

In policy-based ESM Schedule, we focus on balancing between probing frequency and timely self-report collection. So, we compare *LIHF* ESM with *EB* and *TB* schedules using the above-defined metrics. In order to evaluate the complete ESM design, we focus on ensuring that no probe is issued at inopportune moment. So, we compare *LIHF-M* with *EB-M* and *TB-M* schedules. We perform leave-one-participant-out cross-validation (i.e. for a user we train the model using data from others and test using her data) in each of the three cases and measure the inopportune moment detection performance.

In order to measure the survey response quality, we evaluate how accurately the emotion detection model, constructed using the proposed ESM design, can determine the emotion states. The policy-based ESM schedules (*TB, EB, LIHF*) do not apply the inopportune moment detection model. As a result, there will be *No Response* self-reports attached to different typing sessions. These labels do not reflect any emotion state. But as probes have been issued during those moments, user would have recorded some emotion label against these probes if there was no option to select *No Response*. (R3, Q1) Although it is difficult to know what the user might have responded, we propose the following approach to replace *No Response* by emotion labels, so that distribution of emotion labels before and after replace remains unaltered.

- Compute the frequency distribution of emotion labels (happy, sad, stressed, relaxed) from the original user responses.
- Generate proportionate number of valid emotion labels for the total number of recorded *No Responses*.
- Replace the *No Response* labels with the generated labels, such that the distribution of labels remain unaltered.

Notably, with this approach we obtain an average correlation of 0.99 (std dev. 0.01) between the distribution of emotion labels before and after replacement.

In case of model-based ESM schedules (*TB-M, EB-M, LIHF-M*), the inopportune moment detection model is in place and detects inopportune moments. There will be following possible cases - (a) correctly classifies the inopportune moments (b) correctly classifies opportune moments and (c) incorrect classification, which can be again of two types - identifies opportune moment as inopportune and vice-versa. In first case, the probing will be skipped and we drop the corresponding *No Response* label. In second case, we accept the emotion label as is. In case of an error that opportune moment is identified as inopportune, there will not be any probing and we drop the valid emotion label response. In reverse case, there will be probing and user will record some emotion state. In this case also, we realize the same by replacing the *No Response* label with one of the four emotion states (*happy, sad, stressed, relaxed*) (R3,Q1) based on the *No Response* replacement strategy. Once the emotion labels are obtained, the personalized emotion detection models are constructed for every user and ESM schedule. These models are evaluated similarly using the *survey response quality* metric.

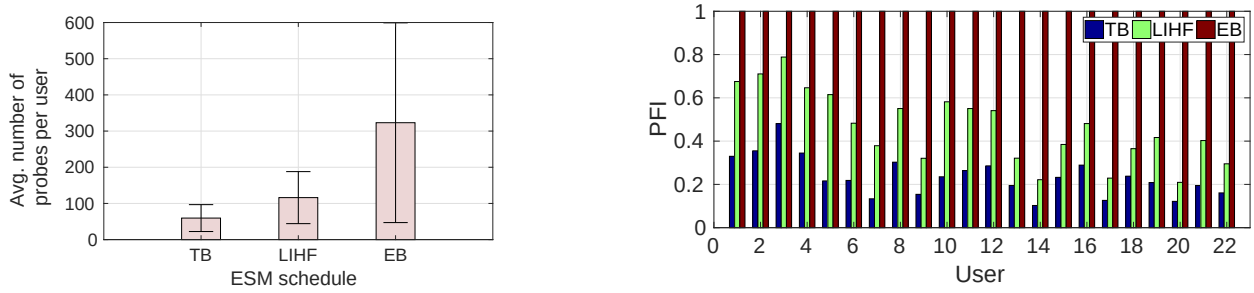
8.2 Probing Rate Reduction

We compare the average number of probes issued by each ESM schedule in Fig. 15a. We observe that time-based ESM (*TB*) issues minimum number of probes, event-based ESM (*EB*) issues maximum number of probes while LIHF ESM lies in between. It is observed that average number of probes is reduced by 64% for LIHF ESM policy.

We also perform user-wise comparison using the Probe Frequency Index (*PFI*) metric in Fig. 15b. For all users, PFI for LIHF ESM is lower than that of event-based ESM. Across all users, there is an average improvement of 54% in PFI. Time-based ESM is the best in terms of PFI, but does not capture self-reports timely, as shown later. As LIHF ESM schedule enforces a minimum elapsed time between two successive probes, it generates less number of probes and reduces probing rate compared to event-based ESM.

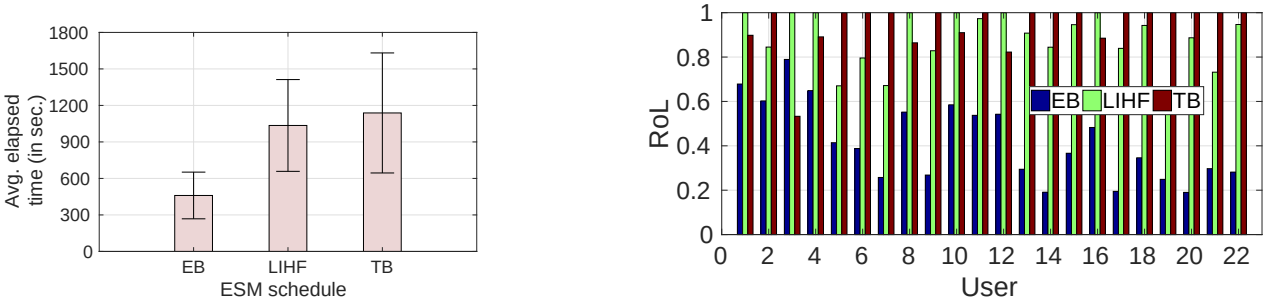
8.3 Timely Self-report Collection

We measure how close to the typing completion, the ESM schedule collects the self-report. It is expected that if the self-reports are collected close to the typing completion, the user will be able to recall the perceived emotion during typing more accurately. We compare the average elapsed time between typing completion and self-report collection for different ESM schedules in Fig. 16a. The average elapsed time is found to be the least for event-based ESM, highest for time-based ESM, while for *LIHF*, it lies in between. Average elapsed time for label collection is reduced by 9% for *LIHF*.



(a) Average number of probes (b) PFI comparison

Fig. 15: Comparing probing rate across ESM schedules in terms of average number of probes and PFI. Event-based schedule has the highest probing rate and time-based schedule has the least one, while LIHF has the probing rate in between.



(a) Average elapse time (b) RoL comparison

Fig. 16: Comparing timely self-report collection across ESM schedules in terms of average elapsed time and RoL. Time-based schedule has the highest elapsed time and event-based schedule has the least one, while LIHF has the elapsed time in between.

We also compare the recency of labels using RoL in Fig. 16b. We observe that for every user RoL is minimum for *EB* and for most of the users RoL is maximum in case of *TB* while for *LIHF* the RoL lies in between. Since, in case of *EB*, we issue the probe as soon as the typing event is completed, it can collect self-reports very close to the event, resulting in lowest RoL. On the contrary, in case of *TB*, we perform probing at an interval of 3 hours, as a result there is often large gap between typing completion and self-report collection, resulting in high RoL. But in case of *LIHF*, we keep on accumulating events and separate two consecutive probes by at least half an hour, therefore we compromise to some extent in recency but not as much as in case of *TB*. (R3,Q5) However, we observe that for few users (e.g. 1, 3, 4), some labels in *LIHF* are collected after a significant time is elapsed after typing completion. As a result, when we compute average elapsed time for every probe, the value becomes large thereby having high RoL in *LIHF* than that of *TB* for these users. However, for most of the users, the value of RoL is highest in *TB* indicating timely self-report collection may not be ensured by *TB* schedules.

8.4 Inopportune Moment Detection

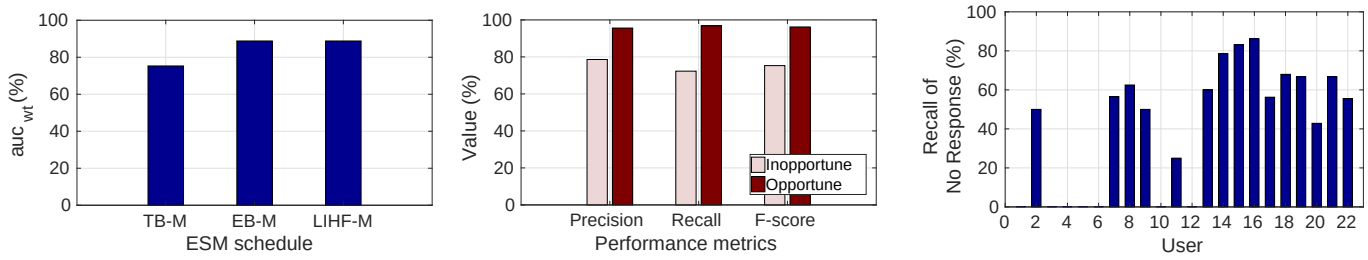
We compare the inopportune moment classification performance of three model based approaches in Fig. 17a. We observe that the *LIHF-M* attains an accuracy (AUCROC) of 89%, closely followed by *EB-M*, while *TB-M* performs poorly. (R3,Q7) We obtained an AUCROC of 88% and 75% for EB-M and TB-M respectively. We also note the precision, recall and F-score values of identifying inopportune moments in Fig. 17b using *LIHF-M* schedule. We obtain close to 80% precision in identifying the inopportune moments. We also report the recall rate of inopportune moments for every user in Fig. 17c. We observe that for 14% of the users, recall rate is greater than 75%, and for 60% of the user recall rate is greater than 50% For few users, none of the inopportune moments are detected correctly because they have very few sessions (less than 4% of overall sessions) tagged with inopportune moments. (R3,Q3) It is observed that users with high number of *No Response* (Fig. 10a) gets more benefit using the inopportune moment detection model. In summary, the proposed model combined with *LIHF* ESM performs best, while other ESM schedules also detect the inopportune moments well with this model.

8.4.1 Influence of Inopportune Moment Detection Features

We find the importance of every feature by ranking them based on the information gain (IG) achieved by adding it for predicting the inopportune moment. We use the *InfoGainAttributeEval* method from Weka [40] to obtain the information gain of each feature. Our results show that last ESM probe response is the most important feature followed by application category.

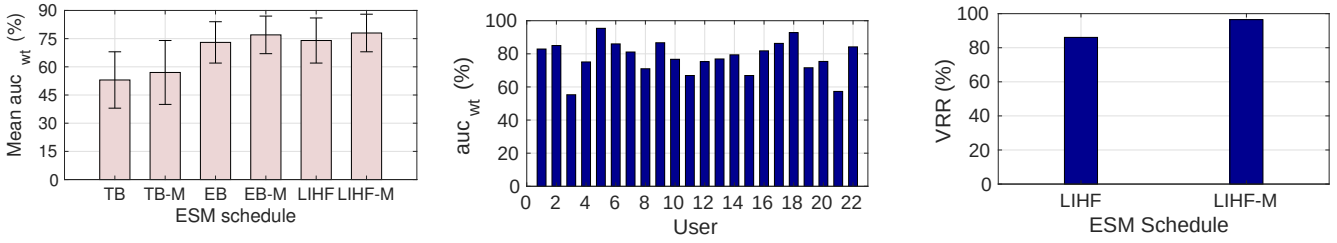
Feature Name	Rank	Average IG
Last ESM probe response	1	0.669
App category	2	0.053
Session length	3	0.019
Session duration	4	0.012

TABLE 6: Ranking inopportune moment detection features



(a) Performance comparison (b) Performance metrics in *LIHF-M* (c) User-wise recall in *LIHF-M*

Fig. 17: Inopportune moment detection performance. (a) Comparison of inopportune moment classification performance for different ESM schedules (b) Performance metrics in *LIHF-M* schedule in identifying inopportune moments (c) Recall rate of inopportune moments for every user in *LIHF-M* schedule



(a) Schedule-wise AUCROC (b) User-wise AUCROC for *LIHF-M* (c) Valid response rate comparison

Fig. 18: Survey response quality measurement. (a) Emotion classification accuracy (AUCROC) using different ESM schedules. Applying the inopportune moment detection model improves the policy-based schedules. (b) Emotion classification accuracy (auc_{wt}) for LIHF-M schedule for every user (c) Comparing number of valid response for *LIHF, LIHF-M* schedules.

8.5 Survey Response Quality

8.5.1 Emotion Classification Performance

The measured response quality in terms of emotion classification accuracy for different ESMs is shown in Fig. 18a. We observe that *LIHF-M* outperforms other schedules with a mean AUCROC of 78%. We also show the user-wise emotion detection AUCROC (auc_{wt}) corresponding to *LIHF-M* schedule in Fig. 18b. We observe that for more than 45% users the AUCROC is greater than 80% and for all but two users the AUCROC is greater than 70%. This helps to obtain superior classification performance for *LIHF-M* schedule (Fig. 18a). It returns maximum improvement of 24% with respect to *TB* and improvement of 5% with respect to *EB*. We also observe that after applying the inopportune moment detection model, mean AUCROC (auc_{wt}) improves (by $\approx$ 4%) for each corresponding schedule (*TB, EB, LIHF*).

As the number of probes are less in time-based schedules, if some of these are reponded randomly due to probing at inopportune moments, the prediction quality suffers a lot. On the contrary, since in case of event-based schedules there are large number of probes, if some of these are marked randomly, the overall prediction does not deteriorate that much, but event-based schedules issue high number of probes. However, in case of *LIHF*, since we already balance between probing rate and timely self-report collection, applying the inopportune moment detection model on top of it further increases the quality of self-reports as reflected by the overall emotion classification performance.

Influence of Emotion Detection Features

We find the importance of the features used for emotion detection using *InfoGainAttributeEval* method from Weka. We compute the average Information Gain (IG) of every feature and rank them in Table 7. We observe that last ESM response is the most discriminating feature, followed by features like typing speed, backspace percentage. All the features are found to have an effect on emotion detection.

Feature name	Rank	Average IG.
Last ESM probe response	1	0.468
Session typing speed	2	0.376
Backspace percentage	3	0.270
Session length	4	0.231
Special character percentage	5	0.203
Session duration	6	0.181

TABLE 7: Ranking emotion detection features

8.5.2 Number of Valid Responses

We compare the valid response rate (VRR) for *LIHF, LIHF-M* schedules in Fig. 18c. We do not consider other schedules as those labels were generated synthetically. The VRR for *LIHF* is 86% and the same for *LIHF-M* is 96%. This further proves the effectiveness of the inopportune moment detection model. As the model is in place for *LIHF-M*, it detects and skips probing at the inopportune moments, thereby improving the number of valid emotion responses.

8.6 Post-study Qualitative Assessment

We conducted a post-study participant survey to gauge the effectiveness of proposed ESM design approach. We asked the participants questions related to various aspects of ESM design (i.e. probing frequency, survey completion time, cognitive load, timely self-report collection and interruptibility) and obtained the rating in a scale of 1 (worst) to 5 (best). We compute the weighted average score as follows, $m_{wt} = \frac{\sum_{i=1}^{5} i*n_i}{N}$, where i denotes the rating provided by the user, n_i indicates the number of users provided the rating i and N indicates the total number of users.

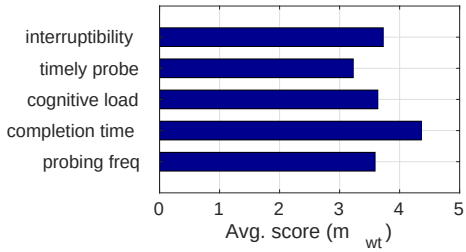


Fig. 19: Average score of different ESM design parameters based on post-study survey

We report the average scores of these parameters in Fig. 19. We asked (a) if the daily probing frequency is appropriate, to which 64% participants agreed (provided a score ≥ 4) and only 18% participants disagreed (provided a score ≤ 2); thus we obtained an average score of 3.6 (*probing freq*). (b) We checked if the time required to fill the survey is high, to which 91% participants disagreed (provided a score ≥ 4), which yielded a final score of 4.4 (*completion time*). (c) In terms of cognitive load exerted while filling out the survey, 64% participants agreed that the mental effort required is very small (rated at least 4), while 14% rated neutral; which finally resulted into a score of 3.6 (*cognitive load*). (d) We also investigated if the survey responses are collected timely i.e. close to the typing completion; to which 55% participants agreed and we obtained a final score of 3.2 (*timely probe*). The score is relatively less in this aspect because often the probing may be delayed by external factors like screenlock, imposed delay. (e) Finally, we checked if the *LIHF* ESM schedule interrupts the regular activity, to which 68% participants disagreed (provided a score ≥ 4). This returned an average score of 3.7 (*interruptibility*). This relatively less score can be attributed to the probing at inopportune moments (as inopportune moment detection model was not coupled with *LIHF* policy during data collection).

8.7 Limitations

It is important to be aware of the limitations of this work before adopting the techniques. First, the inopportune moment detection model may not perform well for some users if the number of *No Response* labels are very few (less than 4% of all sessions) as observed in Fig. 17c. Second, in order to reduce probing frequency, we have collected responses via LIHF ESM schedule instead of EB schedule. As a result, we had to duplicate ESM response for few EB probes. It is possible that had the labeling been done via event-based ESM schedules, it may improve accuracy, but at the cost of responding to significantly high number of probes. Another possible limitation could be the application keyboard. Since most of the participants are conversant with Google keyboard, use of a new keyboard may have disrupted their daily activities. However, we do not observe a significant effect in the app usage due to this as we record 86% valid emotion labels and on average 209 typing sessions per user (Table 4). Finally, during self-reporting if the participants have skipped the popup instead of selecting *No Response*, we could not capture those moments in our study.

9 Conclusion

This paper advocates mobile based ESM design to optimize probing rate, timely self-report collection and user interruptibility together for better survey response quality. We propose a two-phase ESM schedule design to improve survey response quality in terms of valid and accurate responses collected by considering these 3 parameters. In phase-1, we develop an ESM schedule named as *LIHF* schedule, which balances between probing rate and self-reports timeliness. In phase-2, it optimizes probing further by implementing a machine learning model, which ensures probing at opportune moments only. We validate the proposed ESM schedule design method using typing based emotion detection in smartphone in a 3-week in-the-wild study involving 22 participants. It reveals that proposed design reduces the average probing rate by 64% and collects self-reports more timely by reducing the average elapsed time by 9%. It also highlights that the proposed model of identifying inopportune moments detects these unfavorable moments with an average accuracy (AUCROC) of 89%. The combined effect of reduced probing rate, timely self-report collection and inopportune probe elimination is manifested in survey response quality, which results in 96% valid emotion label collection and a maximum improvement of (R3,Q1)24% in emotion classification accuracy.

References

[1] R. Larson and M. Csikszentmihalyi, "The experience sampling method." *New Directions for Methodology of Social & Behavioral Science*, 1983.

[2] J. M. Hektner, J. A. Schmidt, and M. Csikszentmihalyi, *Experience sampling method: Measuring the quality of everyday life*. Sage, 2007.

[3] V. Pejovic, N. Lathia, C. Mascolo, and M. Musolesi, "Mobile-based experience sampling for behaviour research," in *Emotions and Personality in Personalized Services*. Springer, 2016, pp. 141–161.

[4] J. Hernandez, D. McDuff, C. Infante, P. Maes, K. Quigley, and R. Picard, "Wearable esm: Differences in the experience sampling method across wearable devices," in *In Proceedings of ACM Mobile-HCI*, 2016, pp. 195–205.

[5] N. D. Lane, M. Mohammod, M. Lin, X. Yang, H. Lu, S. Ali, A. Doryab, E. Berke, T. Choudhury, and A. Campbell, "Bewell: A smartphone application to monitor, model and promote well-being," in *5th international ICST conference on pervasive computing technologies for healthcare*, 2011, pp. 23–26.

[6] R. LiKamWa, Y. Liu, N. D. Lane, and L. Zhong, "Moodscope: building a mood sensor from smartphone usage patterns," in *Proceeding of the ACM Mobisys*, 2013, pp. 389–402.

[7] M. Pielot, T. Dingler, J. S. Pedro, and N. Oliver, "When attention is not scarce-detecting boredom from mobile phone usage," in *Proceedings of the ACM UbiComp*, 2015, pp. 825–836.

[8] R. Wang, F. Chen, Z. Chen, T. Li, G. Harari, S. Tignor, X. Zhou, D. Ben-Zeev, and A. T. Campbell, "Studentlife: assessing mental health, academic performance and behavioral trends of college students using smartphones," in *Proceedings of the ACM UbiComp*, 2014, pp. 3–14.

[9] C. N. Scollon, C.-K. Prieto, and E. Diener, "Experience sampling: promises and pitfalls, strength and weaknesses," in *Assessing well-being*. Springer, 2009, pp. 157–180.

[10] S. Consolvo and M. Walker, "Using the experience sampling method to evaluate ubicomp applications," *IEEE Pervasive Computing*, vol. 2, no. 2, pp. 24–31, 2003.

[11] A. Mehrotra, J. Vermeulen, V. Pejovic, and M. Musolesi, "Ask, but don't interrupt: the case for interruptibility-aware mobile experience sampling," in *Adjunct Proceedings of the ACM Ubicomp*, 2015.

[12] N. V. Berkel, D. Ferreira, and V. Kostakos, "The experience sampling method on mobile devices," *ACM Computing Surveys (CSUR)*, vol. 50, no. 6, p. 93, 2017.

[13] J. Froehlich, M. Y. Chen, S. Consolvo, B. Harrison, and J. A. Landay, "Myexperience: a system for in situ tracing and capturing of user feedback on mobile phones," in *Proceedings of the ACM Mobisys*, 2007.

[14] K. K. Rachuri, M. Musolesi, C. Mascolo, P. J. Rentfrow, C. Longworth, and A. Aucinas, "Emotionsense: A mobile phones based adaptive platform for experimental social psychology research," in *Proceedings of ACM UbiComp*, 2010.

[15] S. Ghosh, N. Ganguly, B. Mitra, and P. De, "Towards designing an intelligent experience sampling method for emotion detection," in *Proceedings of the IEEE CCNC*, 2017.

[16] V. Pejovic and M. Musolesi, "Interruptme: designing intelligent prompting mechanisms for pervasive applications," in *Proceedings of ACM UbiComp*, 2014, pp. 897–908.

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

[18] A. Mathur, N. D. Lane, and F. Kawsar, "Engagement-aware computing: Modelling user engagement from mobile contexts," in *Proceedings of ACM UbiComp*, 2016.

[19] M. Pielot, B. Cardoso, K. Katevas, J. Serrà, A. Matic, and N. Oliver, "Beyond interruptibility: Predicting opportune moments to engage mobile phone users," *Proceedings of the ACM IMWUT*, vol. 1, no. 3, p. 91, 2017.

[20] J. E. Fischer, C. Greenhalgh, and S. Benford, "Investigating episodes of mobile phone activity as indicators of opportune moments to deliver notifications," in *Proceedings of ACM MobileHCI*, 2011, pp. 181–190.

[21] K. Kushlev, B. Cardoso, and M. Pielot, "Too tense for candy crush: affect influences user engagement with proactively suggested content," in *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI 17). ACM, New York, NY, USA*, 2017.

[22] D. Weber, A. Voit, P. Kratzer, and N. Henze, "In-situ investigation of notifications in multi-device environments," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 2016, pp. 1259–1264.

[23] L. D. Turner, S. M. Allen, and R. M. Whitaker, "Push or delay? decomposing smartphone notification response behaviour," in *Human Behavior Understanding: 6th International Workshop, HBU 2015*. Cham: Springer International Publishing, 2015.

[24] A. Raij, A. Ghosh, S. Kumar, and M. Srivastava, "Privacy risks emerging from the adoption of innocuous wearable sensors in the mobile environment," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2011, pp. 11–20.

[25] D. T. Wagner, A. Rice, and A. R. Beresford, "Device analyzer: Understanding smartphone usage," in *International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services*. Springer, 2013, pp. 195–208.

[26] R. Rawassizadeh, M. Tomitsch, K. Wac, and A. M. Tjoa, "Ubiqlog: a generic mobile phone-based life-log framework," *Personal and ubiquitous computing*, vol. 17, no. 4, pp. 621–637, 2013.

[27] R. Rawassizadeh, E. Momeni, C. Dobbins, J. Gharibshah, and M. Pazzani, "Scalable daily human behavioral pattern mining from multivariate temporal data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 11, pp. 3098–3112, 2016.

[28] D. Ferreira, V. Kostakos, and A. K. Dey, "Aware: mobile context instrumentation framework," *Frontiers in ICT*, vol. 2, p. 6, 2015.

[29] V. Srinivasan, S. Moghaddam, A. Mukherji, K. K. Rachuri, C. Xu, and E. M. Tapia, "Mobileminer: Mining your frequent patterns on your phone," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 2014, pp. 389–400.

[30] S. Nath, "Ace: exploiting correlation for energy-efficient and continuous context sensing," in *Proceedings of the 10th international conference on Mobile systems, applications, and services*. ACM, 2012, pp. 29–42.

[31] L. F. Barrett and D. J. Barrett, "An introduction to computerized experience sampling in psychology," *Social Science Computer Review*, vol. 19, no. 2, pp. 175–185, 2001.

[32] A. Gaggioli, G. Pioggia, G. Tartarisco, G. Baldus, D. Corda, P. Cipresso, and G. Riva, "A mobile data collection platform for mental health research," *Personal and Ubiquitous Computing*, vol. 17, no. 2, pp. 241–251, 2013.

[33] "Personal analytics companion," https://www.pacoapp.com/.

[34] A. Sahami Shirazi, N. Henze, T. Dingler, M. Pielot, D. Weber, and A. Schmidt, "Large-scale assessment of mobile notifications," in *Proceedings of the ACM SIGCHI*, 2014, pp. 3055–3064.

[35] J. Ho and S. S. Intille, "Using context-aware computing to reduce the perceived burden of interruptions from mobile devices," in *Proceedings of ACM SIGCHI*, 2005, pp. 909–918.

[36] M. Pielot, R. de Oliveira, H. Kwak, and N. Oliver, "Didn't you see my message?: predicting attentiveness to mobile instant messages," in *Proceedings of the ACM SIGCHI*, 2014, pp. 3319–3328.

[37] J. A. Russell, "A circumplex model of affect," *Journal of Personality and Social Psychology*, vol. 39, no. 6, pp. 1161–1178, 1980.

[38] I. B. Mauss and M. D. Robinson, "Measures of emotion: A review," *Cognition and emotion*, vol. 23, no. 2, pp. 209–237, 2009.

[39] http://developer.android.com/guide/topics/text/creating-input-method.html.

[40] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[41] P. Verduyn and S. Lavrijsen, "Which emotions last longest and why: The role of event importance and rumination," *Motivation and Emotion*, vol. 39, no. 1, pp. 119–127, 2015.

[42] S. Ghosh, N. Ganguly, B. Mitra, and P. De, "Tapsense: Combining self-report patterns and typing characteristics for smartphone based emotion detection," in *Proceedings of the ACM MobileHCI*, 2017.

[43] http://in.mathworks.com/help/stats/kstest.html.

[44] A. Tarasov, S. J. Delany, and C. Cullen, "Using crowdsourcing for labelling emotional speech assets," 2010.

[45] X. Zhu and X. Wu, "Class noise vs. attribute noise: A quantitative study," *Artificial intelligence review*, vol. 22, no. 3, pp. 177–210, 2004.

[46] B. Frénay and M. Verleysen, "Classification in the presence of label noise: a survey," *IEEE transactions on neural networks and learning systems*, vol. 25, no. 5, pp. 845–869, 2014.

Surjya Ghosh is a PhD student in the Department of Computer Science and Engineering in IIT Kharagpur, India. He received his BTech in Computer Science and Engineering from Haldia Institute of Technology and MTech in Information and Communication Technology from IIT Kharagpur, India. His research interests are in the area of affective computing and smartphone computing and applications.

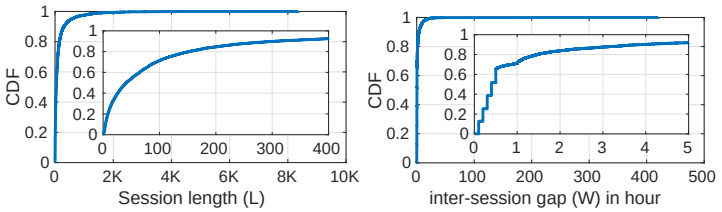
Niloy Ganguly is a Professor in the Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, India. He has received his B.Tech from IIT Kharagpur in 1992 and his PhD from BESU, Kolkata, India in 2004. He has spent two years as Post-Doctoral Fellow in Technical University, Dresden, Germany before joining IIT, Kharagpur in 2005. His research interests are in affective computing, mobile networks and online social networks.

Bivas Mitra is an Assistant Professor in the Department of Computer Science & Engineering at IIT Kharagpur, India. Prior to that, he worked with Samsung Electronics, Noida as a Chief Engineer. He received Ph.D in Computer Science & Engineering from IIT Kharagpur, India. He did his first postdoc (May 2010-June 2011) at the French National Centre for Scientific Research (CNRS), Paris, France and second postdoc (July 2011-July 2012) at the Universite catholique de Louvain (UCL), Belgium.

Pradipta De is an Assistant Professor at the Computer Sciences Department of Georgia Southern University. He is also affiliated to SUNY Korea, a remote campus of StonyBrook University in Songdo, South Korea, as a Research Assistant Professor. Pradipta worked as a Research Staff Member at IBM Research - India from 2005 to 2012. He received Ph.D. from StonyBrook University (SUNY) in 2007. He did his undergraduate in Computer Science and Engineering from Jadavpur University, India.

APPENDIX A
PARAMETER THRESHOLD VALUE

We use three parameters L, W, T as defined in section 5.1.1 to balance between probing frequency and timeliness in label collection.



(a) CDF for session length (L) (b) CDF for inter-session gap (W)

Fig. 20: CDFs for session length (L) and inter-session gap (W) reveal that the distribution is skewed. 66^{th} percentile value is chosen as threshold so that two-third values are less than threshold.

Based on our initial dataset, we observe the CDF of session length (L) in Fig. 20a, which reveals that frequency distribution of session length is highly skewed. So, we select 66^{th} percentile value as the threshold so that two-third values are less than this value. We observe similar CDF and frequency distribution (Fig. 20b) for inter-session gap (W) and use the 66^{th} percentile value as the threshold.

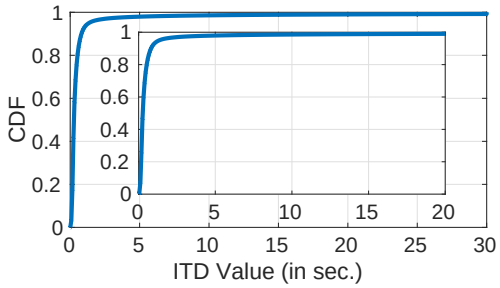
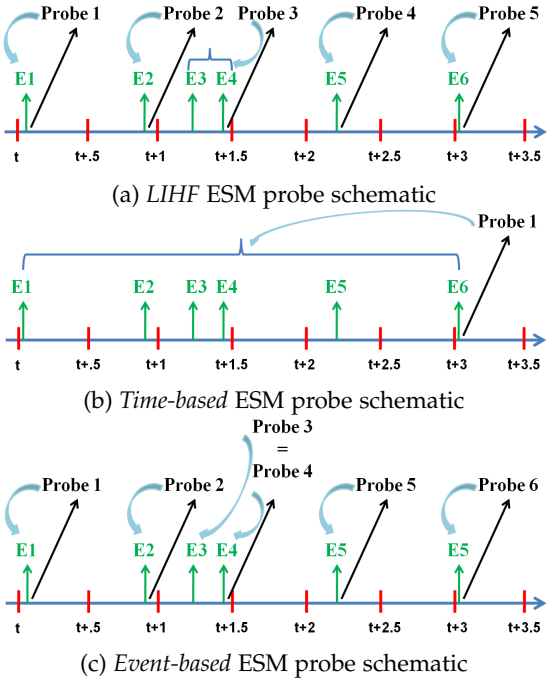


Fig. 21: CDF of polling interval (T). It reveals the distribution is skewed and 99% ITD values are close to 15 seconds.

However, polling interval (T) is to be chosen in such a way that for most of the sessions, the event of interest is captured within this interval. In this case, the event is change of application after typing in a session. For this purpose, we measure the elapsed time between two successive key pressing events (ITD) in a session. We note the CDF of all ITD values from all sessions in Fig. 21. We observe that 99% of the inter-tap duration (ITDs) are less than 15 seconds i.e. for most of the sessions the application change happens after 15 seconds. So, we decide to use 15 seconds as the threshold for T.

APPENDIX B
ESM TRACE GENERATION

In this section, we discuss in detail the steps followed to generate trace for *Time-based (TB)* ESM and *Event-based (EB)* ESM schedule from the data collected using *LIHF* ESM schedule. In Fig. 22, a schematic is given to depict the same. E_i denotes the application switching event after sufficient typing. In case of *LIHF* ESM, there are 6 such events, however only 5 probes were issued (Fig. 22a). No probe is issued after $E3$ because it occurs within time-window ($W = 30$ minutes) since last probe (`Probe 2`). In order to generate the corresponding *Time-based* trace, probes are considered at 3 hour interval. As a result, there will be only one probe `Probe 1` and all events $E1$ to $E6$ will be labeled with the single emotion response collected via it (Fig. 22b). But in case of conversion to *Event-based* ESM, all events are treated separately, as a result there will be in total 6 probes and the emotion labels will be assigned accordingly to the respective events (Fig. 22c). Next, we define the formal procedure for trace generation.



(a) *LIHF* ESM probe schematic

(b) *Time-based* ESM probe schematic

(c) *Event-based* ESM probe schematic

Fig. 22: Schematic to display how *Event-based* and *Time-based* traces are generated from *LIHF* ESM.

B.1 Generation of Time-based Trace

We take the trace collected from LIHF schedule $[\mathbb{C}]^{lihf}_{p\times 5}$ as $p \times 5$ matrix where p denotes the total number of key press events. We generate the respective Time based trace $[\mathbb{C}]^{time}_{p\times 5}$ following the Algorithm 3. We consider the sampling interval of *Time-based* ESM as 3 hours. We parse through (`line 5 - 12`) the LIHF trace $[\mathbb{C}]^{lihf}_{p\times 5}$ and all key press events. As in case of *LIHF*, two responses may be recorded less than 3 hour interval, we may need to down-sample, which is performed in following way. If two emotion responses for key press events are collected within 3 hours, both are considered as a part of single session and the later is labeled with the previous emotion. Otherwise, they belong to different session and the new emotion response is considered (`line 7 - 9`).

B.2 Generation of Event-based Trace

We design Algorithm 4 to generate the corresponding event-based trace $[\mathbb{C}]^{event}_{p\times 5}$ from the collected LIHF trace $[\mathbb{C}]^{lihf}_{p\times 5}$.

Algorithm 3: Time-based trace generation

Input: $[\mathbb{C}]^{lihf}_{p\times 5}$, key pressing details of a user as obtained from LIHF schedule; $[\mathbb{C}]^{lihf}_{:,1}$ contain session number, $[\mathbb{C}]^{lihf}_{:,2}$ contain the application names, $[\mathbb{C}]^{lihf}_{:,3}$ contain the timestamp of the key press, $[\mathbb{C}]^{lihf}_{:,4}$ contain emotion recording timestamp for the session, $[\mathbb{C}]^{lihf}_{:,5}$ contain the associated emotion,; p denotes the total number of key press performed by the user.
Output: $[\mathbb{C}]^{time}_{p\times 5}$; Corresponding time-based trace

```
/* Copy the first row from LIHF schedule (session
   number, application name, emotion and other)      */
```
1 $session^{time} \leftarrow [\mathbb{C}]^{lihf}(1,1)$
2 $app^{time} \leftarrow [\mathbb{C}]^{lihf}(1,2)$
3 $emotion^{time} \leftarrow [\mathbb{C}]^{lihf}(1,5)$
4 $[\mathbb{C}]^{time}(1,:) \leftarrow [\mathbb{C}]^{lihf}(1,:)$
5 **for** $i \leftarrow 2$ ***to*** p **do**
```
    /* Find elapsed time between two consecutive
       emotion recording timestamp                   */
```
6 $\delta \leftarrow$ time_difference($[\mathbb{C}]^{lihf}(i,5)$,$[\mathbb{C}]^{lihf}(i-1,5)$)
7 **if** $\delta \geq 3$ *hours* **then**
8 $session^{time} \leftarrow session^{time} + 1$
9 $emotion^{time} \leftarrow [\mathbb{C}]^{lihf}(i,5)$
10 $[\mathbb{C}]^{time}(i,1) \leftarrow session^{time}$
11 $[\mathbb{C}]^{time}(i,2:4) \leftarrow [\mathbb{C}]^{lihf}(i,2:4)$
12 $[\mathbb{C}]^{time}(i,5) \leftarrow emotion^{time}$
13 return $[\mathbb{C}]^{time}$

We consider changing application after typing as an event. We parse through (`line 5 - 12`) the trace obtained from LIHF schedule and all key press events. If two consecutive key press events are associated with different application, they belong to separate session (`line 6 - 7`). Otherwise, they are considered as part of the same session. In both these cases, no emotion response is dropped (unlike time-based), they are associated with different sessions. In case of *LIHF*, multiple sessions are grouped and tagged with single emotion, but in case of event-based schedule, this grouping is not done and every session is labeled with the same response. This is how the over-sampling is done in case of event-based schedule.

Algorithm 4: Event-based trace generation

Input: $[\mathbb{C}]^{lihf}_{p\times 5}$, key pressing details of a user as obtained from LIHF schedule; $[\mathbb{C}]^{lihf}_{:,1}$ contain session number, $[\mathbb{C}]^{lihf}_{:,2}$ contain the application names, $[\mathbb{C}]^{lihf}_{:,3}$ contain the timestamp of the key press, $[\mathbb{C}]^{lihf}_{:,4}$ contain emotion recording timestamp for the session, $[\mathbb{C}]^{lihf}_{:,5}$ contain the associated emotion,; p denotes the total number of key press performed by the user.
Output: $[\mathbb{C}]^{event}_{p\times 5}$; Corresponding event-based trace

```
/* Copy the first row from LIHF schedule (session
   number, application name, emotion and other)      */
```
1 $session^{event} \leftarrow [\mathbb{C}]^{lihf}(1,1)$
2 $old\_app^{event} \leftarrow [\mathbb{C}]^{lihf}(1,2)$
3 $[\mathbb{C}]^{event}(1,:) \leftarrow [\mathbb{C}]^{lihf}(1,:)$
4 **for** $i \leftarrow 2$ ***to*** p **do**
```
    /* Find current application name                 */
```
5 $curr\_app^{event} \leftarrow [\mathbb{C}]^{lihf}(i,2)$
6 **if** $curr\_app^{event} \neq old\_app^{event}$ **then**
7 $session^{event} \leftarrow session^{event} + 1$
8 $[\mathbb{C}]^{event}(i,1) \leftarrow session^{event}$
9 $[\mathbb{C}]^{event}(i,2:5) \leftarrow [\mathbb{C}]^{lihf}(i,2:5)$
10 return $[\mathbb{C}]^{event}$