TapSense: Combining Self-Report Patterns and Typing Characteristics for Smartphone based Emotion Detection

Surjya Ghosh
IIT Kharagpur, India
surjya.ghosh@iitkgp.ac.in

Niloy Ganguly
IIT Kharagpur, India
niloy@cse.iitkgp.ernet.in

Bivas Mitra
IIT Kharagpur, India
bivas@cse.iitkgp.ernet.in

Pradipta De
Georgia Southern University,
Statesboro, USA
pde@georgiasouthern.edu

ABSTRACT
Typing based communication applications on smartphones, like WhatsApp, can induce emotional exchanges. The effects of an emotion in one session of communication can persist across sessions. In this work, we attempt automatic emotion detection by jointly modeling the typing characteristics, and the persistence of emotion. Typing characteristics, like speed, number of mistakes, special characters used, are inferred from typing sessions. Self reports recording emotion states after typing sessions capture persistence of emotion. We use this data to train a personalized machine learning model for multi-state emotion classification. We implemented an Android based smartphone application, called *TapSense*, that records typing related metadata, and uses a carefully designed Experience Sampling Method (ESM) to collect emotion self reports. We are able to classify four emotion states - *happy, sad, stressed,* and *relaxed,* with an average accuracy (AUCROC) of 84% for a group of 22 participants who installed and used *TapSense* for 3 weeks.

ACM Classification Keywords
H.1.2 User/Machine Systems, H.5.2 User Interfaces, D.2.2 Design Tools and Techniques; :

Author Keywords
Emotion Detection; Emotion Persistence; Smartphone Typing; Experience Sampling Method; SMOTE; Markov Chain

INTRODUCTION
Keystroke dynamics has been shown to be an effective modality for automatic detection of person's affective states [14]. Works using keystroke dynamics collect data related to user's typing pattern on a computer keyboard to infer different emotion states [8]. Nowadays we use several applications on smartphones that are typing based. Moreover, these typing based applications are often conversational in nature, especially communication applications, like WhatsApp. Thus keystroke dynamics on smartphones during these typing sessions can elicit clues about user's emotion. Using smartphone typing patterns also has the added advantage of conducting in-situ experiments.

Effects of different emotions can linger for different durations. As pointed out by Verduyn et al. various emotion states, like sad or happy, can persist differently for different individuals [29]. Considering typing sessions as the stimulus for the emotions, it is natural to assume that emotions from one session can persist into another session. If a user is asked to report the emotion state after typing sessions, the effect of emotion persistence can affect the self report. Therefore, it is important to incorporate the effect of emotion persistence while predicting emotion states.

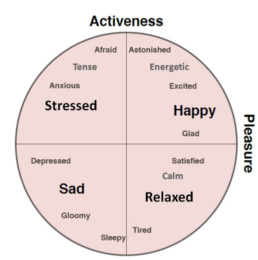


Figure 1: Circumplex emotion model
[23]

In this work, we present a personalized multi-state emotion prediction model that uses both typing patterns and the transitions between emotion states as recorded by users through self reports. Typing patterns are captured by the speed of typing, duration of typing, use of special characters, mistakes while typing denoted by the use of delete key. The persistence of emotion states is modeled assuming that a future emotion state is affected by the previous emotion states, and the emotion state transitions exhibited by the user in the past. We model the emotion transition as a Markov chain where the importance of an emotion state transition is based on the recency of recorded transition to the current self report. We implement

MobileHCI '17, September 04-07, 2017, Vienna, Austria
© 2017 ACM. ISBN 978-1-4503-5075-4/17/09…$15.00
DOI: http://dx.doi.org/10.1145/3098279.3098564

an Android based application, called *TapSense*, that collects emotion self-reports and typing details to drive the model. Self report collection is driven by a refined Experience Sampling Method (ESM), which decides to issue the probes timely and yet keeps the probing rate low. Typing events are captured by instrumenting the smartphone keyboard. However, we collect only meta information to avoid privacy concerns. The machine learning model constructed based on the data collected can predict up to four emotion states - *happy, sad, stressed, and relaxed* - chosen from different quadrants of the Circumplex Emotion Model (Figure 1).

We conducted a field study in-the-wild by installing *TapSense* in smartphones of 22 volunteers. *TapSense* transparently recorded the typing metadata as the users went about using various typing applications on their phone. The ESM generated pop-ups to record emotion states at moments that are close to the typing events, while ensuring that the number of such self report requests are limited to minimize survey fatigue among the respondents. We collected data for 3 weeks. As reported in other similar experiments [19], Relaxed states were recorded more often than the other three states leading to a data imbalance. We used the Synthetic Minority Over-sampling Technique (SMOTE) to overcome the data imbalance [6]. We trained a Random Forest based machine learning model to classify the four emotion states. We obtained an average accuracy (AUCROC) of 84% in classification. Precision and recall for Relaxed state reached close to 80%, while for the remaining three states it reached above 60%.

The main contributions of this paper are:

- A personalized machine learning model that uses both *typing characteristics* on a smartphone, as well as, the emotion *persistence* effect in self reports to infer multiple emotion states.
- A low overhead, non-invasive smartphone application, called *TapSense*, suitable for long running in-situ experiments for automatic emotion detection.

RELATED WORK

We present relevant works that use smartphone as a device for collecting data for emotion detection. Among these, studies where smartphone touch patterns are explored for emotion recognition are the closest to our approach. Our contributions lie in defining a new predictive model combining typing patterns and emotion persistence, supported by an effective ESM strategy.

Smartphone-based Emotion Recognition Techniques

Smartphone-based emotion recognition techniques can broadly be divided into following two categories.

Multi-state Emotion Prediction: A number of works exploit the smartphone usage details and build a multi-state emotion prediction model. MoodScope proposed to infer mood exploiting multiple information channels, such as SMS, email, phone call patterns, application usage, web browsing, and location [19]. In EmotionSense, Rachuri et al. used multiple features from the Emotional Prosody Speech and Transcripts library to train the emotion classifier [22].

Single-state Emotion Prediction: We also find that there are multiple works, which use different information sources to infer presence of a particular emotion state. For example, Pielot et al. tried to infer boredom from smartphone usage patterns like call details, sensor details, etc. [21]. In their work on detecting stress, Lu et al. built a stress classification model using a number of acoustic features [20]. Similarly, Bogomolov et al. showed that daily happiness [3] and daily stress [2] can be inferred from mobile phone usage, personality traits, and weather data.

ESM for smartphone-based Emotion Detection: Experience Sampling Method (ESM) [12] is used for collecting self-reports and *Time-based*, *Event-based* ESM scheduling policies are most commonly adopted by these techniques [7]. For example, in MoodScope [19], Boredom detector [21] authors used *Time-based* ESM. On the contrary, studies like EmotionSense [22] used *Event-based* ESM driven by contextual information.

Touch-based Emotion Recognition Techniques

Widespread availability of touch-based devices and steady increase [17] in the usage of instant messaging apps open a new possibility of inferring emotion from touch interactions. For example, Lee et al. designed a Twitter client app and collected data from various on-board sensors including typing to predict emotion [16]. The work targeted typing behavior on only one application, and was validated on a single user. On the contrary, our exploration goes significantly further in its methodologies, realization, and evaluation. Similarly, Gao et al. used multiple finger-stroke related features to identify different emotional states during a touch based game play [10]. Ciman et al. detected stress conditions by analyzing multiple features of swipe, scroll and text input interactions in a smartphone [30]. In [13], Kim et al. proposed an emotion recognition framework analyzing touch behavior during touch interaction using 12 attributes from 3 on-board sensors. Although focused on narrow application scenarios, all of these works point to the value of touch patterns in emotion detection.

In this work, we make use of typing characteristics in smartphone to enable emotion detection across all typing based applications. We propose a model to combine different typing features and persistence of affective states for automatic emotion prediction. We (a) do not use any privacy sensitive information like call details, SMS details or browsing history, (b) limit the apparatus to only a smartphone to make the usage non-invasive, and (c) avoid resource-intensive information sources and processing on the user device.

METHODOLOGY

We designed and implemented an Android based application, called *TapSense*, for data collection. The application was installed on smartphones of volunteers who used it without any intervention. The raw data from all the users is uploaded periodically to a server for data analysis, and constructing the personalized emotion prediction model. In this section, we present the details of the application, data processing steps, and the feature selection for model construction.

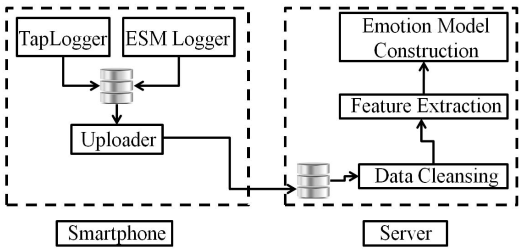


Figure 2: *TapSense* architecture

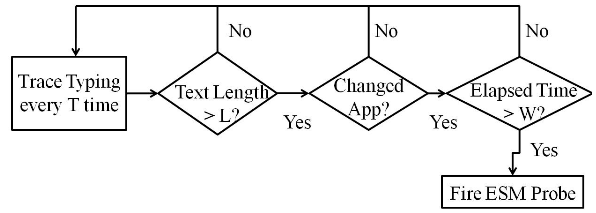


Figure 5: ESM for collecting self-reports

Experiment Apparatus

TapSense is designed as a client server application. The client component is an Android application focused on collecting (i) data related to typing characteristics, and (ii) emotion self-reports. The server component acts as a storage for the data uploaded from each user device. We perform all the data processing and model generation tasks on the server. Figure 2 shows the architecture of our experiment apparatus. *TapLogger* collects data when a user is typing, while *ESMLogger* is designed to trigger a popup questionnaire to collect user response about her current emotion state. Next, we present the details of the data collection components.

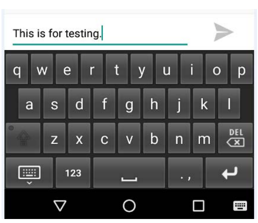


Figure 3: *TapSense* Keyboard

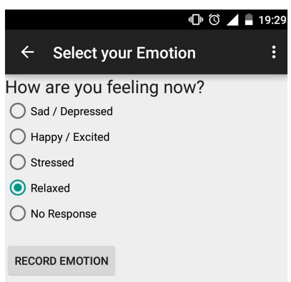


Figure 4: Self-reporting UI for emotion collection

Typing Data Collection

In order to collect typing related information, the *TapLogger* module is implemented as part of an instrumented keyboard. We take advantage of the Input Method Editor (IME) [1] facility in Android to implement the keyboard. When *TapSense* is installed by a volunteer, she is prompted that the default keyboard is being replaced with the instrumented keyboard. *TapSense* keyboard has the same functionalities as any QWERTY keyboard, as shown in Figure 3), thereby not affecting the user's natural usage. The information that is logged are timestamp of each tap event when a character is entered, the type of key input, such as alphanumeric keys, or delete keys. We also record the application used by the user during the typing activity. Actual text is never recorded to protect the privacy of the user. The data is uploaded to the server when the user is in a WiFi hotspot.

Self Report Collection

Self reports are used to collect data from the user about her emotion state soon after she has performed some typing activity. This allows us to correlate the emotion label recorded by user to the typing characteristics. There are two challenges in designing a reliable self report collection mechanism. First, the request for self reporting must be generated in a timely manner such that it is close to a typing event, but at the same time it should not disturb the user. Also, if the phone is locked immediately after a *typing session*, then triggering a self report request will go unnoticed for a long time. Second, the user may be busy and may decide not to respond to a probe. We must not allow default label to be captured, and give explicit option to the user to skip the question.

We design our self report collection based on Experience Sampling Methods (ESM). However, we tune the ESM to reduce survey fatigue by limiting the number of probes, while trying to make it more timely for high fidelity input from the user. The *ESMLogger* implements a quasi-regular scheduling policy where self report collection is triggered only (a) if the user has performed sufficient amount of typing (L) before changing the current application and (b) a minimum time interval (W) has elapsed since the last ESM probe. In order to collect self-reports close to the typing completion, we set a polling parameter, which checks at every time interval T and accordingly issues the probe if the previous two conditions are satisfied. The ESM is illustrated in Figure 5.

The self report collection user interface is a pop-up window as shown in Figure 4. The UI design is based on the following arguments.

- We select one dominant emotion state from each of the four different quadrants of the circumplex model (Figure 1).
- If the user is busy, and wants to skip a survey, there is a *No Response* option. This ensures that a user does not pick a label randomly.
- By default, whenever the UI is shown, *No Response* option is selected. In order to provide emotion label, user needs to select correct emotion label and record.

Tagging Typing Session with Emotion Self Report

As a user performs typing activity, we extract her typing session. We define *typing session* as the period user stays onto a single application without changing the same.

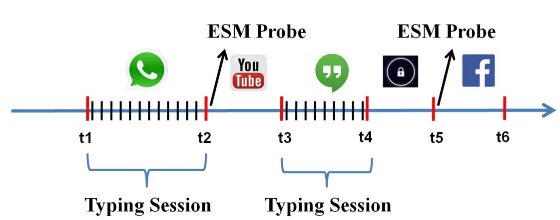


Figure 6: Schematic showing the process of associating user provided emotion label with *typing session*. For example, time interval between `t1` and `t2` is considered a *typing session*, where each small bar within this *session* is a typing event. Emotion label provided between `t2` and `t3` is associated with this *typing session*.

Figure 6 depicts how emotion labels are associated with a *typing session*. For example, when a user uses WhatsApp uninterrupted without switching to other application from `t1` till `t2`, then we define elapsed time between `t1` and `t2` as a *typing Session*. Each small bar within this session is a typing event and we calculate the elapsed time between two subsequent typing event as the *Inter-Tap Duration (ITD)*. ITD acts as a measure of the typing speed. Once the user switches the application, an ESM probe, based on the configuration parameters, is triggered to record an emotion label. Upon selecting an emotion label `l`, all preceding *typing sessions* are labeled with `l`. We use *typing session* and *session* interchangeably.

Data Processing

After the raw data is collected on the server, following data cleansing tasks are performed to filter out irrelevant information before model generation.

(i) Removal of No Response Labels: The first step is to remove all *session*s labeled with *No Response* labels, which do not indicate any emotion state.

(ii) Session Elimination: If time duration between the end of a *session* and the collection of the subsequent label is high, it may not reflect accurately the emotion experienced during the *session*. Therefore, we filter out all *session*s for which the interval between typing and emotion label collection is more than 3 hours.

(iii) Elimination of small sessions: In order to ensure that sufficient number of typing events are performed in every *session*, so that different typing cues can be extracted from the *session* and linked with the emotion state, we decide to discard *session*s with less than 50 typing events.

(iv) Outlier ITD Elimination: We consider an ITD value that is more than 3 times the standard deviation away from the mean value as an outlier. We remove the outliers from each *session* such that the mean session ITD value is not skewed. We try with different threshold values and finally use the above-mentioned values to ensure that we obtain sufficient amount of typing sessions for every user and there is enough typing details in every session.

Category	Feature Name
Keystroke Features	Mean Session ITD (MSI)
	Refined Mean Session ITD (RMSI)
	Number of special characters
	Number of backspaces (or delete)
	Session duration
Auxiliary Features	Persistent emotion (PRE)
	Working Hour Indicator

Table 1: Feature table

Feature Selection

We identify several features, related to typing characteristics, along with the self report patterns that can be used to train a model for emotion prediction. The features, as listed in Table 1, are explained further. Features extracted from typing activities are named as *keystroke features*. We name the other two features as *auxiliary features* as they do not require monitoring of any additional activity or sensor data.

Keystroke Features

We use typing speed as a feature, but use two different representations of it - *Mean Session ITD (MSI)* and *Refined Mean Session ITD (RMSI)*. In order to compute the *MSI*, we compute the average of all Inter-Tap Durations (ITDs) present in a *session*.

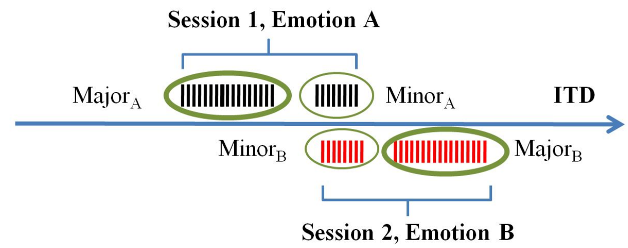


Figure 7: Schematic showing the intuition behind using *RMSI* as feature. Identification of dominant set of ITDs in a *session* and giving it preference to while computing *RMSI* provides it better distinguishing ability to identify two emotion states.

However, we find that it is possible to have overlapping ITD values in two different *typing sessions* tagged with different emotion states, if the emotion labels are captured within a short time span. This may be due to the effect of last emotion on the current one. Therefore *mean session ITD (MSI)* computed using all ITD values for an emotion label may not provide clear demarcation between the two emotion states and there is a need for additional sophisticated mechanism to trace typing speed. So, we introduce the feature *Refined Mean Session ITD (RMSI)*, which is calculated based on the dominant set of ITDs present in a *session*. Figure 7 describes the intuition behind selecting *RMSI* as the feature. We identify the major cluster and compute mean giving it the preference so that the difference in *RMSI* become more pronounced for two different emotion states. We implement the following clustering based approach to compute the value of *RMSI* as outlined in Algorithm 1.

Algorithm 1: *RMSI* Calculation Method

Input: Session ITDs(S)
Output: *RMSI*
$[C_{major}, C_{minor}] \leftarrow$ kmeans((S),2)
$itd_{major} \leftarrow$ mean(C_{major})
$itd_{minor} \leftarrow$ mean(C_{minor})
$S_{sorted} \leftarrow$ sort(S) in ascending order
`/* This step is to normalize ITD data usage */`
if $itd_{major} < itd_{minor}$ **then**
    RMSI $\leftarrow$ Compute mean using top 80% samples of S_{sorted}
else
    RMSI $\leftarrow$ Compute mean using bottom 80% samples of S_{sorted}

We also consider other keystroke features. We compute the fraction of backspace and delete keys present in a *session* and use it as a feature. We use this feature to get a general idea about the number of mistakes being made. Similarly, we use the fraction of special characters in a *session* and *session* duration as features to get an idea about how these characteristics vary with emotion states.

Persistent Emotion (PRE) based on Self Reports

Different emotion states persist for different durations, and can affect how a user responds emotionally to a new emotion stimulus [29]. This implies that when a user records her emotion state, it can get influenced by earlier emotion states. As a result, it is possible to estimate the next emotion state based on the previous emotion states information. The notion of using previous emotion state details to estimate current emotion represents *persistent emotion (PRE)* feature.

$$\underbrace{\begin{bmatrix} H & S & T & R \end{bmatrix}_n}_{e_n} = \underbrace{\begin{bmatrix} H & S & T & R \end{bmatrix}_{n-1}}_{e_{n-1}} \times \underbrace{\begin{pmatrix} p_{hh} & p_{hs} & p_{ht} & p_{hr} \\ p_{sh} & p_{ss} & p_{st} & p_{sr} \\ p_{th} & p_{ts} & p_{tt} & p_{tr} \\ p_{rh} & p_{rs} & p_{rt} & p_{rr} \end{pmatrix}}_{P}$$

Figure 8: Schematic showing the process of computing *PRE* of n^{th} *session* (e_n). We multiply the self-report of $(n-1)^{th}$ *session* with transition matrix P, which is computed by analyzing the transitions of previous $(n-1)$ *sessions*. $[e_i]_{1\times 4}$ is a vector with denoted position of different emotion states. For a given self-report, that position is set to 1, rest are 0. $[P]_{4\times 4}$ is the transition matrix; p_{xy} indicates transition probability of moving from state x to y. *H,S,T,R* denote *happy, sad, stressed, relaxed* states respectively.

We model the feature *PRE* using discrete-time Markov chain [25]. Figure 8 depicts the modeling of *PRE* for n^{th} *session*. Mathematically, we express the same as follows,

$$e_n = e_{n-1}.P \tag{1}$$

where P is the transition matrix containing the state transition probabilities and e_n denotes the *PRE* of n^{th} *session*, e_{n-1} denotes the self-report of $(n-1)^{th}$ *session*. The state space of e_i contains the set of recorded emotion states {*happy, sad, stressed, relaxed*}. In order to calculate the transition matrix (P), state-wise transition probabilities are calculated. Ideally, while calculating the transition probability (p_{xy}) of making a transition from state x to y, the total number of transitions (n_{xy}) made from x to y should be divided by the total number of transitions (n_x) possible from x, which can be expressed as,

$$p_{xy} = \frac{n_{xy}}{n_x} \tag{2}$$

where $x, y \in$ {*happy, sad, stressed, relaxed*}.

However, this scheme does not consider the elapsed time between two consecutive emotion self reports and assigns equal importance to all transitions. We differ here because if the elapsed time is low, then intuitively there will be high influence of last emotion on current emotion and vice-versa. In our approach, we consider the elapsed time between two emotion self-reports and accordingly calculate the transition probability. For every emotion state x, we define a matrix (T_x), which contains all the state transitions from x and the corresponding elapsed time between these two. As a result, if there are k transitions from state x, then

$$T_x = [y \quad elapsed\,time_{xy}]_{k\times 2} \tag{3}$$

where $y \in$ {*happy, sad, stressed, relaxed*}. Then for x, the value of the minimum elapsed time is found as,

$$\tau^x_{min} = minimum(T_x(:,2)) \tag{4}$$

The total number of transitions (n_{xy}) from x to y is redefined as

$$n_{xy} = \sum_{i=1}^{k} \frac{\tau^x_{min}}{T_x(T_x(i,1) = y, 2)} \tag{5}$$

and the total number of transitions (n_x) from x is redefined as

$$n_x = \sum_{\forall y \in \{happy, sad, stressed, relaxed\}} n_{xy} \tag{6}$$

Once we redefine n_{xy}, n_x the probability values (p_{xy}) are computed as per equation 2.

Lets consider the following example. If there is 1 transition from *relaxed (r)* state to *sad (s)* state, with elapsed time of 60 minutes and 1 transitions from *relaxed (r)* state to *happy (h)* state with elapsed time of 15 minutes; the un-weighted scheme assigns the transition probability 0.5 to p_{rh} and p_{rs}. But the proposed scheme would assign probability 0.8 and 0.2 respectively to p_{rh} and p_{rs}. In summary, since the *relaxed* to *happy* transition has lower elapsed time, it gets higher transition probability.

Algorithm 2: *PRE* Calculation Method for n^{th} *session*

Input: $[ET]_{(n-1)\times 3}$; details of previous $(n-1)$ *sessions*; ET_{i1} denotes the emotion label associated with *session* $(i-1)$, ET_{i2} denotes emotion label associated with *session* i, ET_{i3} denotes the elapsed time between recording emotion for *session* $(i-1)$ and i.

Output: *PRE* for *session n*

```
1  𝔼 ← [ happy, sad, stressed, relaxed ]
2  P ← []
3  foreach e_from ∈ 𝔼 do
       /* Extract transition details from each emotion                */
4      T_from ← ET(ET(:,1) = e_from, :)
       /* Check if transition details are empty                       */
5      if (!isempty(T_from)) then
           /* Find minimum elapsed time among all transitions         */
6          τ_min ← Find minimum (T_from(:,3))
           /* Initialize all possible transitions from e_from         */
7          [n_h, n_s, n_t, n_r] ← 0
8          for i ← 1 to length(T_from(:,2)) do
               /* Weigh each transition factoring in minimum
                  elapsed time                                        */
9              if T_from(i,2) = happy then
10                 n_h = n_h + τ_min / T_from(i,3)
11             else if T_from(i,2) = sad then
12                 n_s = n_s + τ_min / T_from(i,3)
13             else if T_from(i,2) = stressed then
14                 n_t = n_t + τ_min / T_from(i,3)
15             else if T_from(i,2) = relaxed then
16                 n_r = n_r + τ_min / T_from(i,3)
           /* Find total number of weighted transitions               */
17         sum = n_h + n_s + n_t + n_r
           /* Compute the transition probability and attach to
              the matrix                                              */
18         P = [P; n_h/sum  n_s/sum  n_t/sum  n_r/sum]
       /* No transition from a given emotion state                    */
19     else
20         P = [P; 0 0 0 0]
21 prev_emo ← ET(end, 2)
22 PRE = prev_emo.P
```

We describe the process of computing the *PRE* of n^{th} *session* in Algorithm 2. During training phase, when self-reports are collected, for every *session*, the proposed algorithm first computes P and then multiplies it with the self-report of previous *session* to find *PRE* for current *session*. In order to find transition probability from every emotion state (line number 3), it first extracts the number of transitions made from that particular state (line number 4). Then, it finds the transition with minimum elapsed time (line number 6) and accordingly for every transitions made from current emotion state, weigh the number of transitions (line number 9 to 16). Then it computes the probability by normalizing these number of transitions with total transitions and creates the transition matrix P (line number 17 to 18). Once P is computed, it multiplies previous self-report with P and returns the *PRE* (line number 22).

Working Hour Indicator

Similarly, we feel that emotion states may vary between working hour and non-working hour. We select working hour indicator also as feature since a significant amount of time is spent at work and such a setup can be stressful at times. We have set this indicator if the emotion recorded is within 10 AM to 5 PM on a week day (Monday to Friday).

Field Study

Survey Focus Group: We recruited 30 graduate students (25 male, 5 female, aged between $24-33$ years) to use *TapSense*. We installed the application on their smartphones and instructed them to use it for 3 weeks to record their emotion states. 3 participants left the study in between and 5 participants entered less than 50 labels during entire period. Finally, we collected data from the remaining 22 users (20 male, 2 female).

Instructions to the Focus Group: We instructed participants to select the *TapSense Keyboard* as the default keyboard. We informed the group members that when they switch from an application which involved typing, they may receive a survey questionnaire as a pop-up, where they can record their emotion state. We also advised participants to record *No Response* label if they are busy and do not want to record emotion state.

DATA ANALYSIS

We collected a total of 605362 typing events spanning across 3976 *typing sessions*. This adds up to 154 hours of typing. However, after the data cleansing operation, number of *typing sessions* reduced to 2705. In Table 2, we record the *sessions* removed at each of the data cleansing steps.

Data cleansing step	Eliminated sessions (%)
No Response Removal	2.5
Outlier session elimination	7.4
Small session elimination	22.0

Table 2: Amount of eliminated sessions

Final Dataset

Our final dataset comprises of 529698 typing events, which constitute close to 135 hours of typing. There are 2705 *sessions* with an average of 123 sessions per user. The median session length and median session duration are found to be 114 and 98 seconds respectively. Table 3 summarizes the final dataset.

Total typing events	529698
Total typing sessions	2705
Total typing duration (in Hr.)	135
Mean typing sessions (per user)	123
Minimum number of typing sessions for a user	40
Maximum number of typing sessions for a user	485

Table 3: Final dataset details

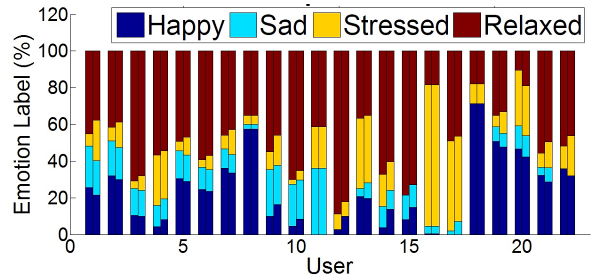


Figure 9: Emotion distribution of every user. All but 6 users have recorded all 4 emotion states. For every user, there are two bars, where the first bar indicates the distribution of emotion samples in original data as recorded by the participants. The corresponding second bar indicates the distribution of emotion samples after applying SMOTE.

Emotion Distribution

In Figure 9 we analyze the distribution of the four emotion states recorded by the users. Except 6 users $(11, 12, 15, 17, 18, 22)$, all the users recorded four emotion states. For most of the users *relaxed* is the dominant emotion state. We also observe that all the emotion states are not uniformly distributed creating data imbalance among the four emotion categories. Overall we have recorded $19\%, 9\%, 23\%, 49\%$ *sessions* tagged with *happy, sad, stressed* and *relaxed* emotion respectively from the participant provided self-reports.

User	Added Sample (%)	User	Added Sample (%)	User	Added Sample (%)
U1	19.71	U9	19.61	U17	5.56
U2	7.55	U10	7.27	U18	0.00
U3	4.17	U11	0.00	U19	6.25
U4	4.27	U12	8.04	U20	10.44
U5	5.06	U13	4.41	U21	12.17
U6	4.08	U14	11.54	U22	12.35
U7	7.38	U15	8.02	-	NA
U8	0.00	U16	0.00	-	NA

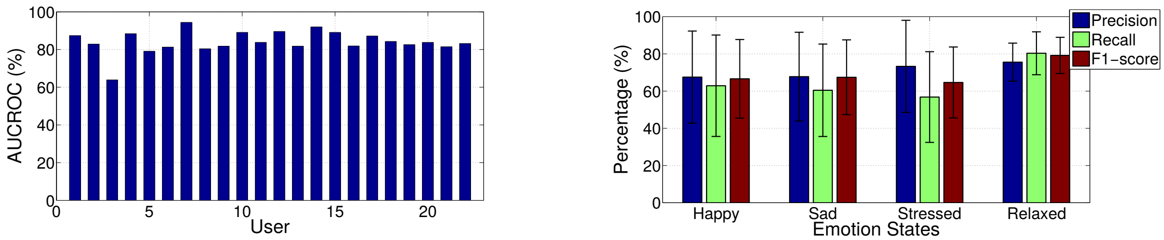
Table 4: User-wise percentage of newly added samples using SMOTE

Countering Data Imbalance using SMOTE

In order to overcome the problem of data imbalance in emotion samples, we use Synthetic Minority Over-sampling Technique (SMOTE) [6]. SMOTE is designed to re-sample the class with the least number of instances so that almost all classes are balanced. While using SMOTE we ensured that - (a) we do not include any new state i.e. if the user has not originally provided any emotion state, the same is not added after sampling and (b) we try to add as few records as possible, so that the emotion state with least number of samples is boosted to have approximately as many samples as the category with the next higher count. By applying SMOTE we add 8% new records. Additional data introduced per user is shown in Table 4. We also show the user-wise comparison of emotion sample distribution before and after applying SMOTE in Figure 9. All results reported later on are based on this data generated after applying SMOTE unless otherwise stated.

EVALUATION: EMOTION CLASSIFICATION

We tested three different models - L2-regularized Logistic Regression (LR) [9], Support Vector Machines with Radial Basis Functions kernel (SVM) [28], and Random Forests (RF) [4] using 10-fold cross validation. We report the results of Random Forests (RF) since it generates the best classification



(a) Accuracy (AUCROC) of predicting different emotions across all users **(b)** Mean value of Precision, Recall and F1-score for each emotion state. Error bar indicates standard deviation.

Figure 10: Performance evaluation of classification model using 10-fold cross validation

performance. AUCROC, area under the ROC (Receiver Operating Characteristic) curve, is used to report the classification accuracy as it is the ideal metric when the dataset has imbalance. We report the weighted average of AUCROC from four different emotion states.

Figure 10a shows the overall classification accuracy. We obtain an average accuracy (AUCROC) of 84% (standard deviation 6%) while the maximum AUCROC is 94%. The quality of prediction for each emotion category is presented in Figure 10b. The emotion states are identified with precision between 67% and 75%, and recall rate between 57% and 80%). We observe that *relaxed* state is identified with highest precision, followed by *stressed*, *happy* and *sad* states respectively. Similarly, we observe highest recall for *relaxed*, followed by *happy, sad* and *stressed* states. The recall rate for *stressed* state is low since for some users the model performed poorly due to scarcity of data. As data volume increases, as in the case of *relaxed* state, the performance metrics improve.

Dataset	Happy	Sad	Stressed	Relaxed
Original Data	0.519	0.557	0.359	0.776
Over-sampled Data	0.666	0.674	0.646	0.792

Table 5: Comparing average F-Score for different emotion states on original data and over-sampled data

Effect of SMOTE on Classification Performance

We compare the difference in classification performance for the two cases - data with imbalance, and data processed using SMOTE. The average accuracy (AUCROC) is 80% for the original dataset, while it is 84% after applying SMOTE. We also report the F-score for each emotion category for both datasets in Table 5. We find that state-wise performance is poor in case of original data, however it improves fairly when we perform over-sampling using only 8% data. This shows that the proposed model is robust and with adequate data it can attain high classification performance. Improvement in the SMOTE dataset can be attributed to adding more samples in underrepresented categories.

Comparison of Alternate Models

We explore the possibility of using only *PRE* or *Keystroke* as the feature to train a model, against use of the combined features. This brings out the efficacy of the proposed model. We also compare the proposed model with an aggregate model to check if the overhead of personalized training can be reduced.

- *Model A - Persistent Emotion (PRE) Model:* We construct a personalized model using *persistent emotion (PRE)* as the only feature. We compare the proposed model with this model to find, if *PRE* alone is sufficient to provide high emotion classification performance.
- *Model B - Keystroke only Model:* In this case, we construct a personalized model using only keystroke features. We select this model to understand the role of only keystroke features for emotion classification and if the auxiliary features used really help in boosting the classifier performance.
- *Model C - Aggregate Model:* While personalized models generally report high accuracy, they require individual training. We attempt to reduce the amount of training required by forming an aggregate model. If it is found to be working, this model can be used as an initial model for a new user, reducing the overhead of personalized training. The rationale behind aggregate model is that there exists similarity in typing pattern across users. The aggregate model is tested using leave-one-participant-out cross validation. For every user, we construct the model using remaining users' data and then test the model using this user's data and compute emotion classification accuracy.

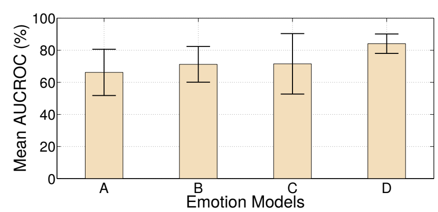


Figure 11: Mean AUCROC comparison with different models. Error bar indicates std. deviation. *PRE* based model (A) performs worst, *Keystroke* based model (B) performs moderately, but the combined proposed model (D) outperforms all other models.

In Figure 11, we compare the performance of the proposed personalized model (model D) with all the above-mentioned models. We observe that the proposed personalized model outperforms each of the other models in terms of average accuracy (AUCROC) as well as variations in AUCROC across participants (minimum standard deviation). It attains an average AUCROC of close to 84% (standard deviation 6%).

The personalized models based on *persistent emotion (PRE)* only (model A) and keystroke features only (model B) attain average AUCROC value of 66% (standard deviation 14%) and 71% (standard deviation 11%) respectively, which are far below than the accuracy of proposed model. We observe that in case of the aggregate model (model C), we attain an average AUCROC of 71% (standard deviation 19%). In case of aggregate model, we observe comparatively high AUCROC value for few participants, however for most of the participants it performs poorly, resulting in such high standard deviation.

Feature Analysis

In order to investigate the role played by different features, we rank each feature based on the information gain (IG) achieved by adding it for predicting different emotions. We use the *InfoGainAttributeEval* method from WEKA [11] to obtain the information gain each feature brings to the overall classification model. Table 6 shows the average ranking of the features. The feature evaluation used 10-fold cross validation. Our results show *PRE* and *RMSI* top the list, indicating that these two are having strong influence on predicted emotion.

Feature Name	Rank	Average IG
PRE	1	0.4226
RMSI	2	0.2324
Working hour indicator	3	0.1368
MSI	4	0.1257
Backspace percentage	5	0.0529
Session duration	6	0.0270
Special char percentage	7	0.0226

Table 6: Ranking features based on Information Gain

We also inspect the role of these features on emotion classification for each user separately. For every user, we compute the relative information gain (*RIG*) of each feature. Let F denotes the feature set, which comprises of 7 features as mentioned in Table 1. f_i denotes a feature belonging to F, $IG(f_i)$ denotes the information gain brought by f_i in overall classification. Then for a user, relative information gain of a feature (f_i) is denoted by $RIG(f_i)$ and defined as,

$$RIG(f_i) = \frac{IG(f_i)}{\sum_{j=1}^{7} IG(f_j)} \tag{7}$$

For every user and every feature, we compute this and plot the result in Figure 12. We observe that for every user, there is a contribution from *PRE*. It is also noted that for users like $(10, 11, 15, 16)$ who do not seem to have an effect on typing as per experienced emotion *PRE* becomes useful in emotion classification. Among different keystroke features *RMSI* is found to have a strong effect on the user population. We discuss in detail the role of these two features next.

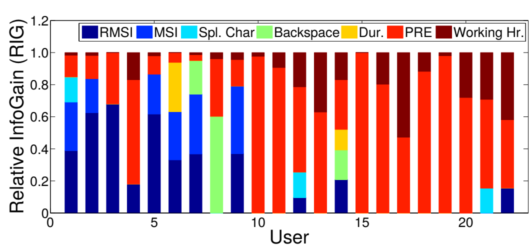


Figure 12: Feature importance based on Relative Information Gain (*RIG*). *PRE* is found to have an effect on all users, while *RMSI* is found to be the most influential among all keystroke features.

Role of PRE on Emotion Prediction

In this section, we investigate the role of *PRE* in determining different emotion states. We observe from Figure 13, that approximately 60% of users (13 out of 22) are having *RIG* of *PRE* more than 40% and close to 72% of users (16 out of 22) having the same more than 30%.

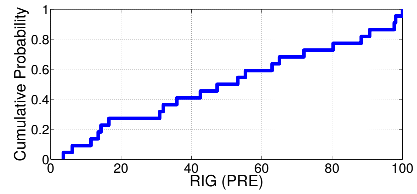


Figure 13: CDF of $RIG(PRE)$. Approximately 72% of the users have an $RIG(PRE)$ greater than 30%.

We investigate the possible reason for having high *RIG* for *PRE*. A high *RIG* for *PRE* indicates that there is one-to-one correspondence between *persistent emotion (PRE)* and actual emotion i.e. *PRE* matches with the actual emotion for large number of *sessions*. We validate the same by computing fraction of *sessions*, where *PRE* matches with actual emotion. We plot the same in Figure 14, which depicts this matching fraction for every user. We observe that users having high ($>=30\%$) *RIG* for *PRE* are also having high matching fraction and vice-versa. In summary, we find that *PRE* matches with actual emotion for large number of *session*; thus making it a strong discriminator for emotion prediction.

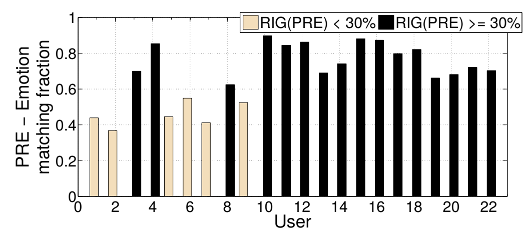


Figure 14: *PRE-emotion matching fraction* for different users. Users having high ($>=30\%$) $RIG(PRE)$ are also having high matching fraction, indicating for large number of *sessions*, *PRE* matches with actual emotion.

Role of RMSI on Emotion Prediction

We investigate the role of *RMSI* using statistical tests i.e. how good it is in discriminating different emotion states. We group the *sessions* with same emotion label and extract *RMSI* from every *session*. Next, we conduct a one-way ANOVA [24] test for each user to inspect whether at least one emotion state has significantly ($p < 0.05$) different *RMSI* than the other three. We identify 11 (50% of the total population) users for whom *RMSI* of one emotion state is significantly different than that of other emotion states. The same also can be observed in Figure 12 by *RIG* for *RMSI*.

R - H	S - H	T - H	S - R	T - R	S - T
36%	78%	45%	78%	36%	56%

Table 7: Tukey HSD test overview to identify the percentage (%) of users having significantly ($p < 0.05$) different *RMSI* for a given emotion pair. A user can have multiple distinguishable emotion pairs. R, H, S, T denote emotion states *relaxed*, *happy*, *sad*, *stressed* respectively.

Once we identify these users, we investigate further to identify the difference in *RMSI* across every emotion pairs for these users. We perform post hoc comparisons using the Tukey HSD [15] to identify which pair of emotion states have significantly ($p < 0.05$) different *RMSI* for these users. Table 7 summarizes the Tukey HSD test result. We observe that for 78% of the users, emotion pairs {*sad* and *relaxed*}, {*sad* and *happy*} are distinguishable, for 56% of users *sad* and *stressed* states are distinguishable and so on. In summary, we observe using *RMSI*, we not only identify 50% users, but also distinguish one emotion from other for these set of users.

EVALUATION: APPLICATION PERFORMANCE

We evaluate *TapSense* with respect to (a) device power consumption, (b) required training period, and (c) ESM effectiveness. We also report the findings from post experiment participant survey.

Energy Overhead

We also monitor the energy consumed by *TapSense* and report the result in Figure 15. *TapSense* monitors typing on smartphone and collects emotion self-reports from the user. Later based on availability of WiFi connection, these details are sent to the server for feature extraction and model building.

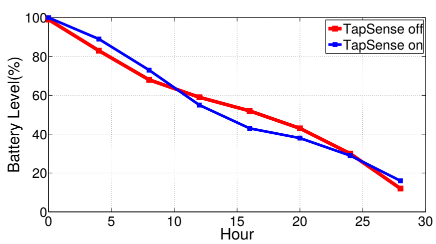


Figure 15: Comparison of battery depletion (keeping *TapSense* on and off) reveals that in both cases, not only the depletion time, but also the depletion rate is similar.

We run the application on a Moto G2 (Android version 6.0) once it is fully charged and monitor how quickly the battery is completely drained. During this period, the user uses an always-on WiFi connection and records 12 emotion labels using *TapSense* user interface along with her normal daily activities. We note that it takes approximately 29 hours to completely deplete the battery. We repeat the same experiment on the same device again once it is fully charged and measure the power consumption turning off *TapSense*. It is noted that complete depletion takes 29.5 hours and the hourly depletion rate is also similar, when *TapSense* is on. This ensures that *TapSense* does not consume noticeable amount of extra energy.

Training Duration

In order to deploy an usable system, it is required that the system attains a reasonable classification performance within a short time span. We verify how the classification accuracy (AUCROC) changes with training period in Figure 16. We accumulate the data at an interval of every 3 days and measure the AUCROC of the proposed personalized model for every user. As expected, the mean AUCROC increases with longer training period and the variation in AUCROC across participants also reduces. Within a period of 12 days, an average AUCROC of 71% is obtained, which touches 77% at the end of 18 days. We also plot the AUCROC for two representative users (*user 7, user 22*) with varying training period. We observe that for *user 7*, the accuracy does not change much with time and it remains more or less stable, whereas for *user 22* the accuracy improves with more training data. These results indicate that the system performs better with more training data and attains high accuracy. However, it may be possible that the performance improves even further with additional training data.

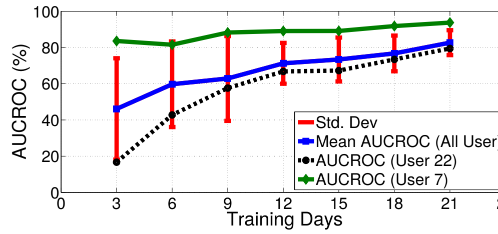


Figure 16: Classification accuracy (AUCROC) with varying training period. Mean AUCROC improves with time and the std dev reduces indicating less variability in AUCROC values across different users. For user like *user7*, AUCROC does not vary much with time, however for most of the users like *user 22* AUCROC improves with time.

ESM Effectiveness

TapSense runs on smartphone as an Android application to monitor typing and collect emotion self-reports. The ESM scheduling policy shown in Figure 5 trades off between timely label collection and survey fatigue. In this section, we validate the ESM in terms of (i) Average number of ESM probes issued daily, (ii) Elapsed time in collecting labels after typing, and (iii) Amount of *No Response* labels.

Number of ESM probes: During this 3-weeks study, we have issued on average 4.6 (standard deviation 2.6, median 3.9) probes per day for every user.

Elapsed time between typing and emotion recording: We also verify the elapsed time between typing completion and label collection. It is expected that if the emotion labels are collected close to the corresponding *typing sessions*, the chance of user forgetting the actual emotion state would be low. We plot the distribution of the elapsed time between typing completion and emotion labeling in Figure 17. We note a median elapsed time value is less than 5 minutes. Similarly 75^{th} and 90^{th} percentile value observed for elapsed time in our set is less than 30 minutes and 1 hour respectively.

Number of No Response sessions: We have instructed the participants during survey to select *No Response* while recording emotion if popup appears at an inopportune time and they do not wish to provide any label at that time. In Figure 18, we note the percentage of *No Response* sessions as recorded by the participants. We observe that 9 participants have not recorded any *No Response* sessions, while 18 participants have recorded less than 5% *No Response* sessions. Overall, we have tagged only 2.5% *session*s as *No Response*.

All these aspects emphasize that the self-reports are collected close to the typing sessions and the application does not cause major interruption in user activities.

Post-study Participant Feedback

We conducted a post study survey following the *Post-Study System Usability Questionnaire* (PSSUQ) [18] to gauge the system effectiveness from usability perspective. One of the first concerns is whether our system is intrusive due to the

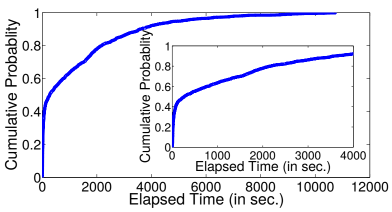


Figure 17: Distribution of elapsed time between typing and emotion recording for all *sessions* across all users.

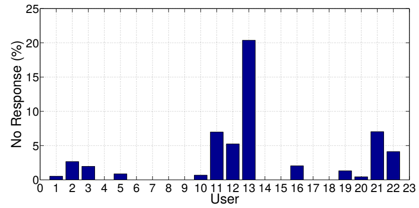


Figure 18: Amount of *No Response* sessions as recorded by different users.

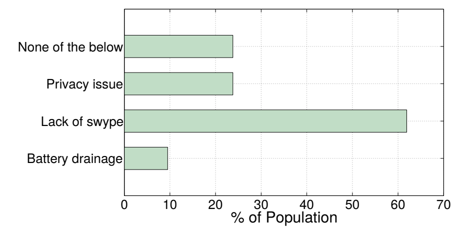


Figure 19: User preferences in *TapSense* usability survey

ESM probes. On rating intrusiveness, 20% of the participants gave a score of 2 and 75% gave a score 3 on a scale of $1-3$, where 1 indicates intrusive and 3 not intrusive. Second, source of disruption was the instrumented keyboard used in *TapSense*. Since we focused on typing, we had disabled swype feature. 61% of the participants reported lack of swype facility as an inconvenience (Figure 19). Finally, we were concerned how the users will perceive privacy issues with this application. Since we had explained that only typing metadata is stored, it assured users of privacy. Only 24% of the participants expressed discomfort with the level of privacy (Figure 19).

DISCUSSION

Our results show that typing features combined with emotion persistence can help in multi-state emotion prediction. In this section, we discuss the implications of our findings and challenges in designing and deploying *TapSense*.

Insights

An important question that this work addresses is how strongly is smartphone typing correlated to our perceived emotion. We notice that typing characteristics can vary across individuals significantly. The performance of an aggregate model based on keystroke features was poorer compared to personalized models. We also find that among different keystroke features, typing speed is a strong indicator of emotion states. It alone can predict emotions for about half of our experiment population, especially when it comes to detecting multiple emotion states. We find that even for 2-state classification just by using typing speed it is easier to classify emotions across the valence dimension, i.e. correctly identifying between pairs like {*happy, sad*}, {*relaxed, stressed*}, compared to detecting in the arousal dimension, i.e. distinguishing {*stressed, sad*}.

The next insight is that effects of an emotion state continue to persist over a period of time. If typing captures our emotion, then the effect of the past emotions should also influence typing. Thus, jointly modeling the typing characteristics and emotion persistence is found to be more accurate. In our dataset, we observe that when two consecutive emotion states are recorded by a user within 5 hours then they are same for 60% of the cases. We observed this effect among 18 out of the 22 of participants. However, finding the exact duration of persistence and its intensity for each emotion type and for different individuals will require further investigation.

Deployment Recommendations

While designing *TapSense* we paid close attention to minimize resource usage, and limit intrusiveness. Experiment results and user feedback both confirm that users used it without disruptions in their normal usage pattern. Therefore, we could have prolonged our data collection to collect more samples. But more samples would not help in overcoming the data imbalance problem since users tend to report *relaxed* state more often than the other states. Hence, we believe that alternative techniques to counter data imbalance is essential. We adopt the approach of over-sampling the minority category using SMOTE. Alternatively, one can explore more recent advances in machine learning techniques that can correctly handle unbalanced datasets [5],[27].

Second, in this work, we assume that during the entire typing session the emotion state of a user does not change for simplicity of design. However, this may not hold true always and more intelligent self-report collection procedure may be deployed to capture these within-session emotion variations. But fine grained self-report collection has the risk of user fatigue in the ESM design.

Finally, using supervised learning in modeling personality traits involves labeled data collection from the user. Emotion labels, especially when asked to report only a single emotion state, are tricky to capture with high fidelity [31]. One solution is to deploy automated emotion labeling technique, as reported in [26].

CONCLUSION

Keystroke dynamics on desktop computers is a known modality for automatic emotion detection. With numerous typing based communication applications on smartphone, typing characteristics provide a rich source to model user emotion. Past emotion states are also good indicators of how a person feels now since different emotions have different persistence influence on people. In this work, we jointly use the typing features, and persistence of self reported emotion states, to train a personalized Random Forest based model for automatic classification of multiple emotion states. The collection of self report is driven by an Experience Sampling Method (ESM), which focuses on collecting user responses close to the typing sessions. We designed an Android based application, called *TapSense*, which was installed on smartphones of 22 volunteers, and collected typing metadata and self reports for 3 weeks. The model trained using this data for each user could classify 4 emotion states, *happy, sad, stressed, relaxed*, with an average accuracy (AUCROC) of 84%. Experimental results indicate that typing speed alone can distinguish multiple emotion states for 50% of the population, while the effect of emotion persistence is observed across all participants. The evaluation also finds the application has low overhead, which paves the way for building light-weight, non-invasive emotion detection systems.

ACKNOWLEDGMENTS

We thank all the participants of the study. We also thank Microsoft Research India for the travel grant to attend the conference.

REFERENCES

1. Android IME 2016. (2016). http://developer.android.com/guide/topics/text/creating-input-method.html.
2. Andrey Bogomolov, Bruno Lepri, Michela Ferron, Fabio Pianesi, and Alex (Sandy) Pentland. 2014. Daily Stress Recognition from Mobile Phone Data, Weather Conditions and Individual Traits. In *Proceedings of the 22nd ACM International Conference on Multimedia*.
3. A. Bogomolov, B. Lepri, and F. Pianesi. 2013. Happiness Recognition from Mobile Phone Data. In *Proceedings of the IEEE International Conference on Social Computing (SocialCom)*.
4. Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.
5. L Breiman, C Chen, and A Liaw. 2004. Using random forest to learn imbalanced data. *J. of Machine Learning Research* 666 (2004).
6. Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research* 16 (2002), 321–357.
7. Sunny Consolvo and Miriam Walker. 2003. Using the experience sampling method to evaluate ubicomp applications. *IEEE Pervasive Computing* 2, 2 (2003), 24–31.
8. Clayton Epp, Michael Lippold, and Regan L Mandryk. 2011. Identifying emotional states using keystroke dynamics. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*. 715–724.
9. Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 2001. *The elements of statistical learning*. Vol. 1. Springer series in statistics Springer, Berlin.
10. Yuan Gao, Nadia Bianchi-Berthouze, and Hongying Meng. 2012. What Does Touch Tell Us About Emotions in Touchscreen-Based Gameplay? *ACM Trans. on Computer Human Interactions* 19, 4 (Dec. 2012).
11. Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA Data Mining Software: An Update. *SIGKDD Explor. Newsl.* 11, 1 (Nov. 2009), 10–18.
12. Joel M Hektner, Jennifer A Schmidt, and Mihaly Csikszentmihalyi. 2007. *Experience sampling method: Measuring the quality of everyday life*. Sage.
13. Hyun-Jun Kim and Young Sang Choi. 2012. Exploring emotional preference for smartphone applications. In *IEEE Consumer Communications and Networking Conference (CCNC)*.
14. Agata Kolakowska. 2013. A review of emotion recognition methods based on keystroke dynamics and mouse movements. In *Proceedings of the 6th IEEE International Conference on Human System Interaction (HSI)*. 548–555.
15. D Lane and N Salkin. 2010. Tukey's honestly significant difference (HSD). *N. Salkind der.), Encyclopedia of Research Design içinde, Thousand Oaks, CA: SAGE Publications* 1 (2010), 1566–1571.
16. Hosub Lee, Young Sang Choi, Sunjae Lee, and IP Park. 2012. Towards unobtrusive emotion recognition for affective social communication. In *IEEE Consumer Communications and Networking Conference (CCNC)*.
17. Uichin Lee, Joonwon Lee, Minsam Ko, Changhun Lee, Yuhwan Kim, Subin Yang, Koji Yatani, Gahgene Gweon, Kyong-Mee Chung, and Junehwa Song. 2014. Hooked on smartphones: an exploratory study on smartphone overuse among college students. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*. 2327–2336.
18. James R Lewis. 1991. Psychometric evaluation of an after-scenario questionnaire for computer usability studies: the ASQ. *ACM SIGCHI Bulletin* 23, 1 (1991), 78–81.
19. Robert LiKamWa, Yunxin Liu, Nicholas D Lane, and Lin Zhong. 2013. Moodscope: Building a mood sensor from smartphone usage patterns. In *Proceeding of the 11th ACM Mobisys*.
20. Hong Lu, Denise Frauendorfer, Mashfiqui Rabbi, Marianne Schmid Mast, Gokul T Chittaranjan, Andrew T Campbell, Daniel Gatica-Perez, and Tanzeem Choudhury. 2012. Stresssense: Detecting stress in unconstrained acoustic environments using smartphones. In *Proceedings of ACM UbiComp*.
21. 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 ACM UbiComp*.
22. Kiran K. Rachuri, Mirco Musolesi, Cecilia Mascolo, Peter J. Rentfrow, Chris Longworth, and Andrius Aucinas. 2010. EmotionSense: A Mobile Phones Based Adaptive Platform for Experimental Social Psychology Research. In *Proceedings of ACM UbiComp*.
23. James A Russell. 1980. A circumplex model of affect. *Journal of Personality and Social Psychology* 39, 6 (1980), 1161–1178.
24. Neil J Salkind. 2010. *Encyclopedia of research design*. Vol. 1. Sage.
25. Williams J Stewart. 1994. *Introduction to the numerical solutions of Markov chains*. Princeton Univ. Press.

26. Boyuan Sun, Qiang Ma, Shanfeng Zhang, Kebin Liu, and Yunhao Liu. 2015. iSelf: Towards cold-start emotion labeling using transfer learning with smartphones. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*.
27. Y. Tang, Y. Q. Zhang, N. V. Chawla, and S. Krasser. 2009. SVMs Modeling for Highly Imbalanced Classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39, 1 (Feb 2009), 281–288.
28. Vladimir Vapnik. 2013. *The nature of statistical learning theory*. Springer Science & Business Media.
29. Philippe Verduyn and Saskia Lavrijsen. 2015. Which emotions last longest and why: The role of event importance and rumination. *Motivation and Emotion* 39, 1 (2015), 119–127.
30. Katarzyna Wac, Matteo Ciman, and Ombretta Gaggi. 2015. iSenseStress: Assessing Stress Through Human-Smartphone Interaction Analysis. In *9th International Conference on Pervasive Computing Technologies for Healthcare-PervasiveHealth*. 8–p.
31. Biqiao Zhang, Georg Essl, and Emily Mower Provost. 2016. Automatic recognition of self-reported and perceived emotion: does joint modeling help?. In *Proceedings of the 18th ACM International Conference on Multimodal Interaction*. 217–224.