Estimating User Interruptibility by Measuring Table-top Pressure

Takahisa TANI [*2] Seiji YAMADA[*1*2*3]

[*1] National Institute of Informatics
[*2] The Graduate University for Advanced Studies
[*3] Tokyo Institute of Technology

A user working with his/her desktop computer would benefit from notifications (e.g., e-mails, micro-blogs, and application updates) being given at adequate times when he/she is interruptible. To do so, a notification system needs to determine the user's state of activity. In this paper, we propose a novel method for estimating user states with a pressure sensor on a desk. We use a lattice-like pressure sensor sheet and distinguish between two simple user states: interruptible or not. The pressure can be measured without the user being aware of it, and changes in the pressure reflect useful information like typing, an arm resting on the desk, mouse operation, and so on. We carefully developed features which can be extracted from the sensed raw data and used a machine learning technique to identify the user's interruptibility. We conducted experiments for two different tasks to evaluate the accuracy of our proposed method and obtained promising results.

1. Introduction

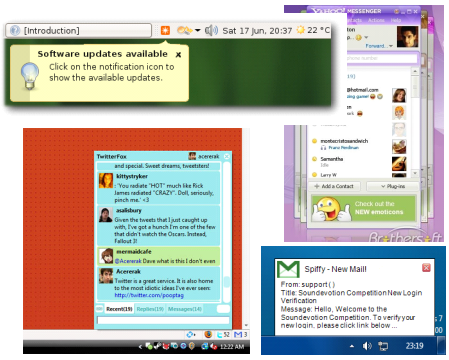


Figure 1: Various information notifications

In the current office environment connected to the Internet, a user tends to get a lot of *notifications*[6] in the form of e-mails, micro-blog, instant messages, application update alerts and so on like in Figure 1.

A significant problem with such notifications is that they arrive as they are sent, i.e., without the system being aware of whether the user has time to receive them or not. If messages arrive at inopportune times, they can cause serious stress and reduce the user's productivity [1]. One way of alleviating this problem would be to control the notification period in accordance with the user's state of activity. In other words, this means a system needs to estimate whether a user is interruptible or not, and send information only when he/she are interruptible [7].

There are other approaches that do not estimate whether the user is interruptible. A peripheral display [8] is one such approach. In such studies, notification is indicated in the user's peripheral field, such as in a sub-window in parallel with a main window used for a main task, and a user can unconsciously recognize the content of the notification while focusing on the main task. However, with this approach, it is hard to determine the positions, size, and color of the notification's window. Estimating a user's state has the advantage of being useful for other purposes besides notification, such as estimating emotional states.

Hence, a notification system needs to monitor user behaviors like typing, operating a mouse, and so on to estimate whether he/she is interruptible or not. There are a number of studies on systems that use the frequency of keyboard strokes and mouse operations [4]. However, these methods can not be applied to cases in which the frequency does not reflect the user's interruptibility or when the user does not use such input equipment. There are also estimation methods that use additional sensors like a web camera [5]. However, these methods need to monitor user behavior by watching their faces and bodies, and thus they could cause psychological stress on the user.

In this study, we developed a novel method for estimating a user's interruptibility by measuring tabletop pressure. At a desk with a PC, there are changes in pressure on the tabletop caused by the forces of various user behaviors including typing, resting one's arm on the desk, and so on. We considered that useful information for estimating the user's interruptibility can be extracted from such slight changes in tabletop pressure. However, there are only a few studies on estimating interruptibility by using tabletop pressure. First, we carefully identified features adequate for the estimation. Then, we compared three machine learning techniques to classify a user's interruptibility. Eventually, we conducted experiments to evaluate the accuracy of our method in two different tasks.

2. Estimating a user's interruptibility by measuring tabletop pressure

For measuring the tabletop pressure, we spread a pressure sensor sheet having measurement points in a reticular pattern on a tabletop. We assumed that a the user does all his/her work on the sheet and that all objects on the tabletop are placed on it. We investi-

Contact: Takahisa TANI, The Graduate University for Advanced Studies, 2-1-2 Hitotsubashi Chiyoda Tokyo, tani@nii.ac.jp

Figure 2: LL-Sensor.

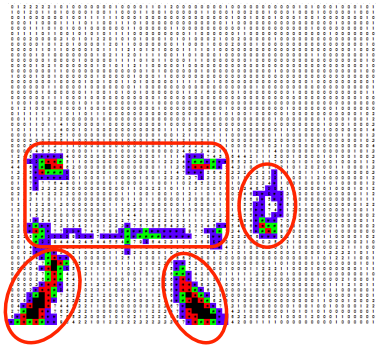


Figure 3: Output of sensor.

gated the forces involved in typing, resting one's arm, and placing objects on the tabletop, and we found that we need a sensor sheet about 1 meter square with pressure gradation ability of 10 grams.

Hence, we used the LL-sensor (Xiroku Co., Ltd) in Figure 2. This sensor leverages the feature of mutual induction. It is 600 mm square, and its resolution is 10 mm square. It outputs not a physical quantity but a unique value. Figure 3 shows the output of the LL-sensor in a heat-map fashion. The white unit means the lowest value and the blue, green, red, and black means higher values. A keyboard is placed on the square area, and the user's arms are placed on the elliptical areas.

2.1 Features Extraction for Interruptibility Estimation

We extract features from the raw pressure data. In particular, we use key pressing force weight, location, and changes in them as features. We assume that the objects on a tabletop are only the keyboard of the PC and the users' arms. However, we can relax this restriction by developing procedures to detect other objects like a mug, a book, and so on. We considered typical tasks on a tabletop to be *typing* and *mouse-operation* and that the features related to the two tasks should be extracted. Also, the information related to arm regions is useful because it might change due to a user's busyness. Thus, we introduce 24 features: the pressure, the region size, the x/y-coordinates of the center of gravity of a keyboard's left/right/front feet regions, the mouse region and the left/right arm regions.

We used a simple pattern matching algorithm to automatically extract the features from the raw pressure data. First, templates for the regions pressed on by the keyboard's feet and the mouse are manually obtained, and they are normalized. Then, by scanning the whole area with the templates, the regions corresponding to the templates are identified. Eventually the pressure, the region size, and the x/y-coordinates of the center of gravity for the regions are calculated.

To extract left/right-arm regions, the whole data are scanned from the left/right-bottom to the right/left-top and the regions having pressure values over a threshold are extracted. Then the pressure, the region size, and the x/y-coordinates of the center of gravity for the regions are calculated.

The raw pressure data are obtained from a LL-sensor every 30ms, and the features can be quickly computed anytime with an average of nine frames data 1 sec before.

2.2 Classifying User Interruptibility

After extracting features from the raw data, we needed to identify user's interruptibility from the data. We used classification learning to classify the state of the user into *interruptible* or *uninterruptible*.

In the experiments, we try to apply state-of-the-arts classification learning algorithms: random forests [2], SVM (Support Vector Machine) with a kernel [10], and a traditional algorithm, C4.5 [9]. We can expect the classification accuracy of the random forests and SVM to be higher than the C4.5. However, the C4.5 has human readability, so it is easily interpreted. We apply these classification learning algorithms and compare the results in the experiments.

3. Experiments

3.1 Experimental Environment and Tasks

We built a simplified desk work environment to eliminate complicated factors. Figure 4 is an overview of it. A participant sits down in front of a desk, and the monitor shows a task window.

We use two typical tabletop tasks: a *typing task* and a *mouse operation task*. In the typing task, an experimenter required participants to type the scrolling display of characters as correctly as possible (Figure 5(a)), and their typing was recorded. The scroll speed could be used to control the task difficulty. When typing, the system asked them whether they were interruptible or uninterruptible by displaying a notification dialog in the center (Figure 6). At that time, participants step on the foot switch of left side to accept it (interruptible), and step on the foot switch of right side to reject it (uninterruptible). These judgments were used as class labels for training data. The "accept" and "reject" corresponded to "interruptible" and "uninterruptible." The notification window closed after the answer. The participants were asked to suppose that the notifications provided them with small amounts of information like weather reports and news. These procedures were explained to participants in the instructions.

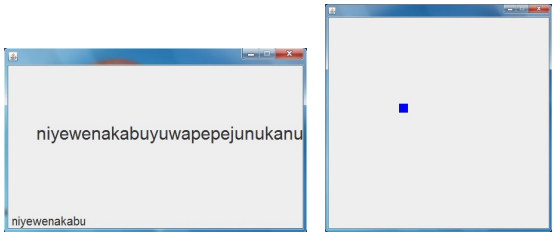
In the mouse operation task, the task window in Figure 5(b) was displayed, and a small blue square appeared and disappeared repeatedly at random positions. This appearance cycle could be used to control the task difficulty. Participants were required to repeatedly click the small blue squire during it appeared. The notification procedures were the same as the typing task.

3.2 Participants and Experimental Procedures

The participants were 20 students and staff members in the information science department (ages ranging from 23 to 51, mean 35.4, S.D. = 11.7), and they consisted of 10 males and 10 females.

To obtain both accept and reject data, we needed to set *hard* and *easy* phases in the two tasks. In the hard phase, participants have to achieve the task, so they tend to reject the notifications

Figure 4: Experimental environment.



(a) Typing. (b) Mouse operation.

Figure 5: Task windows for two tasks.

(uninterruptible), and in the easy phase, they tend to accept them (interruptible). Before trials of each participant, an experimenter manually set the both phase by investigating his/her typing and mouse operation ability.

For each participant, two trials of the typing task and the mouse operation task were given, and the order of the four trials was counterbalanced. In each trial, the hard and easy phases were alternately applied for one minute, and each trial was five minutes long, consisting of five phases. Eventually, the total time for a participant was about 20 minutes. In each trial, notifications were displayed every 20 sec as described before.

We used random forests, SVM (a RBF kernel, $\gamma = 0.0$) and C4.5 (confidence factor = 0.25) implemented on weka3.6.4 [*1] as classification learning algorithms.

4. Experimental Results

Table 1 shows part of the data obtained for the left arm in the typing task. The amount of all the data obtained from the two tasks was 1119 (interruptible: 722, uninterruptible: 477), and those of the typing and mouse operation tasks were respectively 600 (interruptible: 333, uninterruptible: 269) and 599 (interruptible: 389,

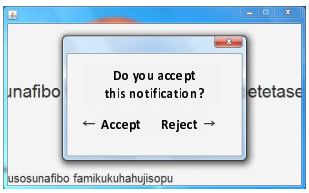


Figure 6: Notification window in the typing task.

∗1 http://www.cs.waikato.ac.nz/ml/weka/

Table 1: Data examples (left arm).

Area	Pressure	COG_x	COG_y	Class
7.6	20.9	2.8	20.1	*accept*
4.9	24.6	1.9	20.8	*reject*
3.7	8.8	3.0	39.0	*accept*
15.1	22.4	12.1	16.5	*reject*

Table 2: Accuracy of estimation of the three classification learning algorithms (The best in bold).

	TP Rate [%]		
CL algorithm	Typing	Mouse op.	Whole
SVM	**76.8**	**72.3**	**73.6**
Random forests	75.8	69.6	73.1
C4.5	71.7	71.6	71.3

uninterruptible: 210). Since the data were not significantly imbalanced, we did not apply under-sampling[3].

After giving the whole data to the three classification learning algorithms as the training data, the data were evaluated by using 10-fold cross validation. The experimental results on the accuracy of the three classification learning algorithms are shown in Table 2, where the "TP rate" means the rate of correctly classified data and the accuracy for the two tasks and the whole task is described.

From the figure, we find that the SVM outperforms other two CL algorithms in the typing task, the mouse operation task, and the whole task. Thus, SVM should be applied to this work.

5. Discussion

5.1 Evaluation of Estimating Interruptibility with Tabletop Pressure

The experimental results show our method's estimation of user interruptibility has about 77% accuracy for the typing task and about 72% for the mouse operation. We consider our main target to be the typing task because editing documents is the main desk work at an office. We consider this level of accuracy (about 77%) for the typing task to be sufficient as a first trial of using tabletop pressure and that it shows our approach to be promising. Furthermore, we can combine our method with conventional methods with other sensors.

As seen in Table 2, the accuracy of the mouse operation task is less than that of the typing task. In contrast with the keyboard's feet pressure, the mouse pressure might be hard to sense with the LL-sensor and might slightly change during the mouse operation. We have significant problems to overcome to improve the accuracy in the mouse operation.

5.2 Estimation of the State of a User Not Typing

Our method can estimate the user's state only when the user is typing on a keyboard on a desktop because a pressure sensor cannot sense any change in pressure when the user is not typing. Thus, in the experimental evaluation, we assumed that a user could be interrupted when s/he was not typing.

However, this assumption is not always valid in real environments. For example, a user might be thinking, reading web pages, and watching a movie on the display when they are not typing, and they would not want to be interrupted in such situations. To cope

Table 3: Accuracy of feature selection.

	TP Rate [%]		
Selected features	Typing	Mouse op.	Whole
Keyboard pressure	74.7	–	–
Mouse pressure	–	64.3	–
Keyboard + mouse	75.8	70.0	69.8
All	76.8	72.3	73.6

with this problem, we plan to extend the current features to cover no-typing situations. We will introduce additional features including sensing the pressure of a mug and the shape and area occupied by arms resting on a desktop, which the pressure sensor can sense. We consider these additional features to be promising because a user does not pick up a mug frequently and does not change the position of his/her arm much when concentrating on something.

5.3 Feature Selection for Practical Applications

Although we verified that our approach of estimating user's interruptibility by measuring tabletop pressure is promising, the LL-sensor with a lattice of small pressure sensors is unique and expensive, and this makes our method difficult to be applied to practical applications in a real environment. We consider a way to solve this problem is to use small and inexpensive pressure sensors. There are various such pressure sensors which can be easily attached to three feet of a keyboard and the bottom of a mouse. Thus, if a user's interruptibility can be estimated only with inexpensive sensors as accurately as with all the pressure sensors of a LL-sensor, our approach will be quite more practical.

Such pressure sensors can be simulated by detecting the data corresponding with the keyboard feet regions and the mouse region from all the data obtained from the LL-sensor. The results on accuracy are shown in Figure 3. As seen in the table, in the typing and the mouse operation tasks, the accuracy of only the keyboard pressure (+ a mouse pressure) is almost equal to that of all the sensors (LL-sensor). Thus, our approach can be made more practical by using small and inexpensive sensors. Meanwhile, there are significant difference in a whole tasks.

6. CONCLUSION

We developed a novel method for estimating a user's interruptibility by using tabletop pressure. We conducted experiments with participants in two typical tabletop tasks to show that our method could estimate when a user was too busy to receive typical messages. As a result, we confirmed our method could estimate the interruption. Then we discussed the accuracy of our method and a way to make our method more practical.

In the future, we will use richer features taken from real experimental environments. This will help to increase accuracy and make it possible to estimate activity states of users when they are not using the keyboard. For that purpose, we will try to determine the optimal number of the features. In addition, we will assess the utility of cost-sensitive learning.

References

[1] Bailey, B. P., Konstan, J. A., and Carlis, J. V. The effects of interruptions on task performance, annoyance, and anxiety in the user interface. In *Proceedings INTERACT ' 01*, IOS Press (2001), 593–601.

[2] Breiman, L. Random forests. *Machine Learning 45*, 1 (2001), 5–32.

[3] Drummond, C., and Holte, R. C4.5 and class imbalance and and cost sensitivity: why under-sampling beats over-sampling. In *Workshop on Learning from Imbalanced Datasets(ICML03)* (2003), 0–0.

[4] Epp, C., Lippold, M., and Mandryk, R. Identifying emotional states using keystroke dynamics. In *Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems (CHI 2011)* (Vancouver, BC, Canada, 2011), 715–724.

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

[6] Iqbal, S. T., and Bailey, B. P. Effects of intelligent notification management on users and their tasks. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, ACM (New York, NY, USA, 2008), 93–102.

[7] Iqbal, S. T., and Bailey, B. P. Oasis: A framework for linking notification delivery to the perceptual structure of goal-directed tasks. *ACM Transactions on Computer-Human Interaction 17* (2010), 15:1–15:28.

[8] McCrickard, D. S., Catrambone, R., and Stasko, J. T. Evaluating animation in the periphery as a mechanism for maintaining awareness, 2001.

[9] Quinlan, J. R. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[10] Vapnik, V. N. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.