# A Statistical Classifier for assessing the Level of Stress from the analysis of Interaction Patterns in a Touch Screen

Davide Carneiro, Paulo Novais, Marco Gomes, Paulo Moura Oliveira, José Neves

**Abstract** This paper describes an approach for assessing the level of stress of users of mobile devices with tactile screens by analysing their touch patterns. Two features are extracted from touches: duration and intensity. These features allow to analyse the intensity curve of each touch. We use decision trees (J48) and support vector machines (SMO) to train a stress detection classifier using additional data collected in previous experiments. This data includes the amount of movement, acceleration on the device, cognitive performance, among others. In previous work we have shown the co-relation between these parameters and stress. Both algorithms show around 80% of correctly classified instances. The decision tree can be used to classify, in real time, the touches of the users, serving as an input to the assessment of the stress level.

## 1 Introduction

There are many scenarios in which the use of stress-aware applications could be of interest to improve the performance and quality of work of organizations. In general there is an interest in the scientific community for applications that can acquire and

Davide Carneiro
University of Minho, Braga, Portugal, e-mail: dcarneiro@di.uminho.pt

Paulo Novais
University of Minho, Braga, Portugal, e-mail: pjon@di.uminho.pt

Marco Gomes
University of Minho, Braga, Portugal, e-mail: pg18373@alunos.uminho.pt

Paulo Moura Oliveira
University of Trás-os-Montes e Alto Douro, Portugal e-mail: oliveira@utad.pt

José Neves
University of Minho, Braga, Portugal, e-mail: jneves@di.uminho.pt

use meaningful information from the user's context. In [7], the authors provide a review of several context-aware applications published in conferences and journals between 2000 and 2007. Moreover, the authors also suggest a new classification framework of context-aware systems and explore each of its features. In this scope and given the nature of stress, soft-computing approaches can be very useful [11].

Stress is evidently part of this context information and can be quite important, depending on the scope of the application. In [8], a system to support tacit communication between fire-fighters with multiple levels of redundancy in both communication and user alerts is presented. This system supports decision and planning based on the level of stress of the fire-fighters, in real time, allowing a better management and security of the personal in the field. Applications for domestic environments also exist. In [9], a Conflict Manager to resolve conflicts for context-aware applications in smart home environments is presented. Conflicts arise when multiple users access an application or when various applications share limited resources to provide services. In order to resolve conflicts between users the Conflict Manager looks at parameters such as their levels of stress.

In this paper, we exploit the fact that tactile devices are nowadays relatively common and available. Moreover, many professions require or welcome their use, such as medical personnel, the military, fire-fighters, among many others. We propose a statistical classifier that is able to assess the level of stress of the users by analysing their touch patterns. The two features considered are the variation of the intensity and the duration of the touch.

## 2 Background

The word *stress* has many connotations and definitions based on various perspectives of the human condition. Many experts endorse the original definition of *stress* concept to the one proposed by Hans Selye [6]. He defined stress as a non-specific response of the body to any demand placed upon it. Selye defined external demands as *stressors* (the load or stimulus that triggers a response) and the internal body changes that they produce as the *stress response*.

However, specialists have expanded the previous concept of *stress*. Now, it is seen as the inability to cope with a perceived threat to one's mental, physical, or emotional well-being, which results in a series of physiological responses of adaptation. Researchers started to focus on the cognitive and behavioural causes of stress, and stress became viewed as a mind-body, psychosomatic, or psycho-physiologic phenomenon. A free interpretation of this phenomenon could refer stress as a physico-physiologic arousal response occurring in the body as result of stimuli by virtue of the cognitive interpretation of the individual.

Given the complexity of stress and its effects, a multi-modal approach is applied to obtain a more complete schematic description, that accounts for its known or inferred properties. These modalities include quantifiable measurements on the user's physical appearance, physiology, behaviours and performance. Figure 1 depicts the multi-modal approach used.

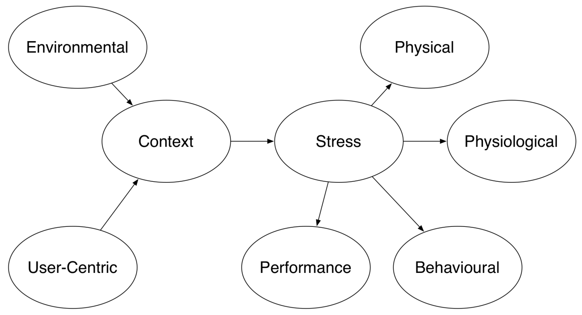

**Fig. 1** A high-level information model about stress.

The context node is divided in two types according to the source of contextual information, namely, the user-centric and the environmental context. User-centric information is composed of two categories: the background and the dynamic behaviour. The background is composed by several attributes that can be extracted from the users profile. These attributes are the age, gender, working area, social status, personality traits among others. The dynamic behaviour reflects the contextual attributes related to the users activity. The environmental information fuses the physical environment characteristics, social environment information and computational environment measurements. Physical environment includes attributes such as the time, temperature, location, noise level, and luminance. High levels of noise, extreme temperatures and low levels of luminance are well known potential stressors. The social environment includes issues such as the population density around the user. The computation environmental context can be characterized by the measurement of the electromagnetic field and the number of surrounding electronic devices.

Among the features that can reveal stress, those that can characterize the behavioural node are the user's interactions with the computer, the mouse/touch screen pressure from clicks/touches, his/her agitation level (through the sensory data from the accelerometer placed in mobile devices or by analyzing movement), as well as input frequency and speed. Also, the performance node is depicted in terms of accuracy and response, where the accuracy feature is related to the precision of the touch and the response feature corresponds to the analysis (qualitative and temporally) of the users responses to the platform demands. The physiological variables provide observable features about the users stress state [10]. These features include the Galvanic Skin Response (GSR), that assesses the electrical proprieties of the skin in response to different kinds of stimuli, the General Somatic Activity (GSA) that assesses the minute movement of human body and others such as respiratory rate or pupilographic activity. Physical appearance includes the visual features that characterize the users eyelid movement such as pupil movement (e.g. eye gaze, papillary response), facial expression or head movement.

## 3 A non-invasive environment for detecting stress

The experiment detailed in section 5 was undertaken in the Intelligent Systems Lab, in the University of Minho. In this lab, we built a closed environment with the main objective of monitoring several interaction parameters that could be related to stress: the stress lab. This stress lab allows for a user to interact with specific applications while being fully monitored in a non invasive way. It is composed of several devices that can acquire information from the user's context (Figure 2). Table 1 presents a brief description of the main functionalities of these devices.

**Fig. 2** Devices used to study the effects of stress in the interaction parameters.

**Table 1** Brief description of the devices that compose the stress detection environment

| Device | Brief description | Main features |
|---|---|---|
| HP Touchsmart | All-in-one PC | touchscreen, web cam, large screen |
| Samsung Galaxy Tab | Tablet PC | touchscreen, web cam, accelerometer, relatively large screen, mobile, Android OS |
| HTC PDAs | Smartphones | touchscreen, camera, accelerometer, mobile, Android OS |
| Sony FCB-EX780BP | 25x Super HAD PAL Color Block Camera with External Sync | 25x Optical Zoom, Image stabilizer, Day/Night Mode, Privacy Zone Masking |

We are interested in devices that can provide us with some information about the user but without interfering with the interaction or with the activities being performed. This environment allows us to capture information about the following parameters: Touch Accuracy, Touch Intensity, Touch Duration, Amount of Movement, Acceleration and Cognitive Performance. In previous work we have performed significance tests on these parameters to find differences at the level of the user due to stress (see for example [1, 2]). In this paper, we use this test environment and, supported by the results previously achieved, we train a classifier that can distinguish between ”stressed” and ”calm” touch patterns.

## 4 Feature Extraction

To assess the level of stress of each touch, out system relies on the event listeners provided by the Android framework. An event listener is an interface in the `View` class that contains a single callback method that will be called by the Android framework when the View to which the listener has been registered is triggered by user interaction with the item in the UI. For the purpose of this paper, our system uses the `onTouch()` callback method, which is called when the user performs an action qualified as a touch event, including a press, a release, or any movement gesture on the screen (within the bounds of the item). Thus, in each touch of the user on an item of the UI several touch events are fired: one when the finger of the user first touches the screen (identified by the action event `ACTION_DOWN`), several while the user is touching (depending on the duration of the touch) and one when the finger releases the screen (identified by the action event `ACTION_UP`).

Each of these events has information about the intensity of the touch (via the `getPressure()` method) quantifying the pressure exerted on the screen and about the position of the event. Moreover, when each event is fired our system registers it with the current time. This allows to visualize the evolution of a touch in terms of its intensity over time. Moreover, from this information we can also extract the duration and intensity features.

### Duration features

The duration of a touch is defined as the difference between the time-stamps of the action events `ACTION_UP` and `ACTION_DOWN`. One of the hypothesis being tested is that the stress of a user will have an influence on the duration of the touch, hence our interest. The duration of the touch can however be influenced by factors other than the stress. Namely, the type of item of the UI being touched. In that sense, we do not use for this purpose events fired by items such as sliders or by scrolling pages. For the purpose of this experiment, we are just interested in the standard touches used to interact with buttons, inputting text and similar actions.

### Intensity features

The intensity of a touch event depicts the force exerted by finger of the user when touching the device. Given that each touch event includes a pressure and that each touch fires several touch events (as described in section 4), it is possible to analyse the variation of the intensity throughout all the touch, from the moment the finger touches the screen to the moment it releases it.

Concerning this temporal evolution of the intensity of the touch, we are interested in the initial and final value of intensity of each touch as well as its maximum and mean values. First we will thus test the hypothesis that stress can influence these parameters and that there are significant differences in at least some of them between a stressed and a calm user.

## 5 Experiment design and methods

The main goal of this experiment is to investigate if it is possible to build a classifier for touch patterns that can be used in real-life applications to provide some information about the level of stress of the user. Our approach is to use two standard and well known machine learning tools: a decision tree constructor and a support vector machine. As the decision tree constructor we use the J48 algorithm - the `java` implementation of the C4.5 [3]. As support vector machine, we use the SMO function, which implements John Platt's sequential minimal optimization algorithm for training a support vector classifier [4]. These experiments were performed using the Weka workbench (Weka 3.6.3) [5].

The results of the two classifiers will be compared by looking at some performance measures such as the percentage of correctly classified instances, the Kappa statistic (which is a chance-corrected measure of agreement between the classifications and the true classes) and the ROC area.

We have a particular interest in using decision trees since a model of a decision tree can then be used to classify, in real time and in a real life application, the level of stress of a user, by following the explicit rules defined by the model.

### *5.1 Dataset*

The dataset used during this experiment was collected in the Intelligent Systems Lab of the Department of Informatics, at the University of Minho. To collect the data, a group of 18 users was asked to play a game that included performing mental calculations and could also include memorizing some intermediary results for posterior use. During the game, the users could be subject to stress in the form of unexpected repetitive and annoying sounds, vibration of the handheld device or a time limit.

With this setting it was possible, in previous experiments, to build the dataset depicted in table 2, that allowed us to determine how each user is affected by stress through significance tests, and develop personalized stress models. Moreover, a generic model was also developed that can be used, although with expected smaller accuracy, when no personalized data about a user is available.

**Table 2** Description of the dataset used as a basis for building the classifier

| data | brief description | size |
|---|---|---|
| Acceleration | Data concerning the acceleration felt on the handheld device while playing the game | 27291 |
| Maximum intensity of touch | Data about the maximum intensity of each touch in a touchscreen | 1825 |
| Mean intensity of touch | This dataset contains data about the mean intensity of each touch event in a touchscreen | 1825 |
| Amount of movement | A dataset containing information about the amount of movement during tests | 25416 |
| Touches on target | This dataset contains information about the accuracy of the touches | 1825 |
| Stressed touches | A dataset containing information that allows to classify each touch as stressed or not stressed | 1825 |
| Score | A dataset describing the performance of the user playing the game, during the tests | 321 |
| Touch duration | A dataset containing the duration of each touch event | 1825 |

## *5.2 Experiment design*

All the parameters in table 2 are correlated with stress, with some users showing more significant results than others. In previous work we have studied this relation. We will now focus on how we use this information to build a classifier.

As previously described, each touch in the screen results in several touch events that are fired during the time of the touch. This number varies according to the duration of the touch. In that sense, this data for each touch, as it is, cannot be used to build a classifier (each touch would have a potentially different list of values of intensity, one for each touch event). Figure 3 (a) highlights this by depicting different types of touches.

To tackle this problem we explored the fact that the intensity from all the touches follows a similar shape: a convex curve that grows to a maximum point and then decreases. Thus, the approach was to fit a second polynomial degree curve to each touch pattern. To perform this fitting in real-time the proposed system uses `J/Link`, the Mathematica's Java interface that allows for controlling Mathematica Kernels from Java programs. Specifically, we use the `Fit[`*`data, funs, vars`*`]` function which finds a least-squares fit to a list of data as a linear combination of the functions *funs* of variables *vars*. To implement this we are using Mathematica® v8.0. An example of this approach is depicted in Figure 3 (b). Given that the second degree polynomial curves are of the type $y = ax^2 + bx + c$ we can compare the parameters of the curve of each touch pattern: similar values of `a,b` and `c` indicate similar curves, thus similar touch patterns. Hence, the input for the classifier are three numeric attributes `a,b` and `c` (the independent variables) and a nominal attribute that describes the state of the user at the time of the touch as ”stressed” or ”not stressed” (the dependent variable). The classifiers were trained using this data, comprising a total of 349 instances.

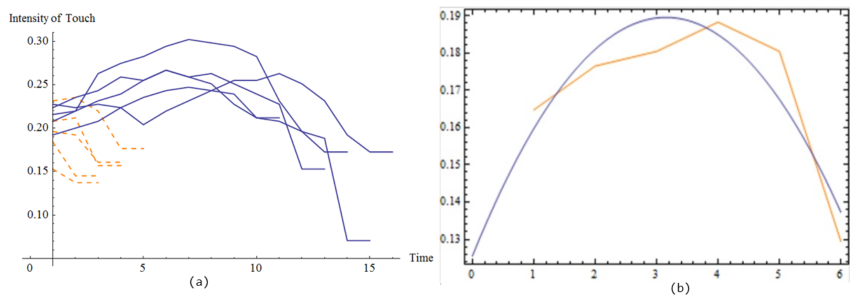

**Fig. 3** (a) 10 different touch patterns from users: touches can be composed of a different number of touch events. The orange lines depict touches classified as "calm" whereas the blue lines belong to touches classified as "stressed". (b) Fitting a polynomial curve (blue curve) to a given touch (orange line).

## 6 Results and Conclusions

Since selecting the optimal parameters for an algorithm may be a rather time-consuming process, to implement this experiments we used a meta-classifier provided by weka that allows to optimize a given base-classifier. Specifically, we used the `weka.classifiers.meta.CVParameterSelection`. After finding the best possible configuration of parameters, the meta-classifier then trains an instance of the base classifier with these parameters and uses it for subsequent predictions. The meta-classifier was used with lower bound 0.01, upper bound 0.5 and 10 optimization steps.

When using the J48 classification tree as the base classifier for the meta-classifier, the model is able to correctly classify 271 out of the 349 instances, which amounts to 77.6504%. The Kappa statistic for this model is 0.5434 and the value of the ROC area is 0.796. The constructed tree has a size of 15 nodes and a total of 8 leaves (Figure 4 (a)). In this tree, attributes `x0`, `x1` and `x2` correspond to the values of `a`, `b` and `c` of the polynomial curve, respectively. Given this, it is possible to use the rules of this tree to build a classifier for distinguishing between stressed and calm touch curves.

When the SMO function is used to build a classifier, the results achieved are similar. In fact, the correctly classified instances amount to 79.9427% (279 out of 349), the value of the Kappa statistics is 0.5809 and the value of the ROC area is 0.781. These results also show that a classifier can be trained with this data to distinguish between stressed and calm touches. Given that the results of both classifiers are similar, we decided on using the J48 tree since it can easily be used by our system to classify touches in real time.

To evaluate the performance of the tree, we used it to classify, in real-time, the touches of 16 users during one of the experiments performed. In short, each user had to perform mental calculations under different levels of stress ranging from 1 (with no stressors) to 5 (with maximum level of stress induced). While the touches were being classified in real time, the remaining of the parameters described in section 3 were also under monitoring. This allowed us to ensure that there were significant differences on other parameters due to stress as well. In the worst case only one parameter showed significant differences and, in the best case, 5 different parameters showed significant differences for the same user. Concerning all the data, in average each user shows significant differences in 3 out of 6 parameters, which allowed us to conclude, in previous work, that stress does have an effect on these behavioural parameters.

Thus, what we did in this experiment was to analyse the behaviour of the classifier for each user under each level of stress and determine if the results of the classifier were in line with the results of the remaining parameters. Concerning the data collected from the 16 users, 13 show an increase in the touches classified as stressed when comparing the data from level 1 with the data from level 5. The minimum value of increase detected was of 6%, the maximum value of increase was of 60% and the mean increase of touches classified as stressed, for all users, was of 32.3077%. The three users for which the classifier reported a decreasing percentage of stressed touches for increased levels of stress have shown relatively low values of decrease (-2.5%, -5% and -1%). This means that the results of the classifier are consistent with the ones previously achieved in 81.25% of the cases. Figure 4 (b) depicts the mean increase of the touches classified as "stressed" in each of the five levels of stress of the experiment.

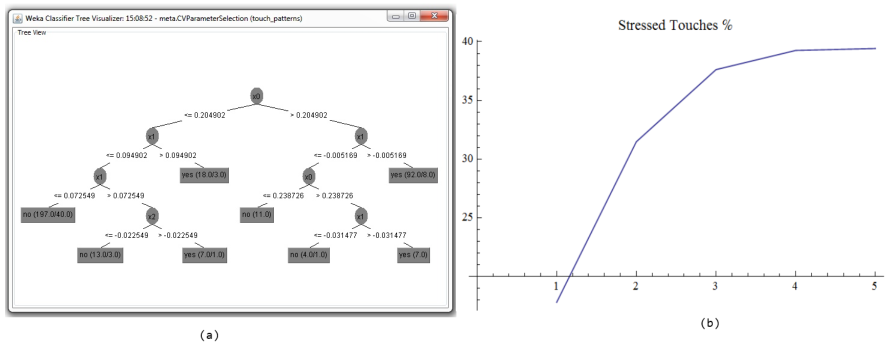

**Fig. 4** (a) J48 prunned tree generated by the algorithm. This tree can be used to classify touches in real time as stressed ("yes" leaves) or not stressed ("no" leaves). (b) Mean increase in the percentage of touches classified as "stressed" in each of the five levels of stress concerning all the users.

Moreover, we have to state that the classifier was built as a generic model, i.e., we used data from all the users. We believe that better results would still be achieved if we were to develop personalized classifiers. This, however, was not the objective of the paper. Given this, we can conclude that it is possible to build a classifier for assessing the touches of users in a touch screen, based on their shape, and classify them as ”stressed” or ”not stressed”. This, by itself is not enough to describe the level of stress of a user but can certainly be a significant input that can be used with that purpose, in conjunction with other inputs, as shown in this paper.

**Acknowledgements** This work is funded by National Funds through the FCT - Fundao para a Ciłncia e a Tecnologia (Portuguese Foundation for Science and Technology) within project PEst-OE/EEI/UI0752/2011. The work of Davide Carneiro is also supported by a doctoral grant by FCT (SFRH/BD/64890/2009).

## References

1. Carneiro D., Carlos Castillo J., Novais P., Fernndez-Caballero A., Neves J., Lpez M., Stress Monitoring in Conflict Resolution Situations, in Ambient Intelligence - Software and Applications - 3rd International Symposium on Ambient Intelligence (ISAmI 2012), Paulo Novais, Kasper Hallenborg, Dante I. Tapia, and Juan M. Corchado Rodrguez (Eds.), Springer - Series Advances in Intelligent and Soft Computing, vol. 153, pp 137-144, ISBN 978-3-642-28785-5, 2012.
2. Carneiro D., Novais P., Costa R., Neves J., Enhancing the Role of Multi-agent Systems in the Development of Intelligent Environments, in Trends in Practical Applications of Agents and Multiagent Systems, Yves Demazeau et al. (eds), in Advances in Intelligent and Soft Computing, Vol. 71, Springer- Verlag, ISBN 978-3-642-12432-7, pp. 123-130, (PAAMS 10 - The 8th International Conference on Practical Applications of Agents and Multi-Agent Systems, University of Salamanca, Spain, 26-28th April/2010), 2010.
3. Quinlan R(1993) C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo, CA.
4. Platt J. (1998) Fast Training of Support Vector Machines using Sequential Minimal Optimization. In: B. Schoelkopf and C. Burges and A. Smola, editors, Advances in Kernel Methods - Support Vector Learning.
5. Holmes G, Donkin A, Witten I H (1994) Weka: A machine learning workbench. In: Proc. Second Australia and New Zealand Conference on Intelligent Information Systems, Brisbane, Australia.
6. Selye, H (1956) The stress of life. No. vol. 5. In: McGraw-Hill paperbacks, McGraw-Hill
7. Jong-yi Hong, Eui-ho Suh, Sung-Jin Kim (2009) Context-aware systems: A literature review and classification. In: Expert Systems with Applications, Volume 36, Issue 4, pp. 8509?8522.
8. Jiang X, Chen N Y, Hong J I, Wang K, Takayama L, Landay J A (2004) Siren: Context-aware Computing for Firefighting. In: Lecture Notes in Computer Science, Volume 3001/2004, 87-105.
9. Shin C, Woo W (2005) Conflict Resolution Method Utilizing Context History for Context-Aware Applications. In: Cognitive Science Research, issue 577, pp. 105-110.
10. Picard, RW (1997)Affectivecomputing. MITPress, Cambridge, MA, USA.
11. Abraham, A (2009) Hybrid Soft Computing and Applications, International Journal of Computational Intelligence and Applications, World Scientific Press, Singapore, Vol. 8, No. 1, pp. 5-7.