Towards an API for User Attention Prediction in Mobile Notification Overload

Alberto Vianna Dias da Silva
Federal Institute of Bahia (IFBA)
Jacobina, Bahia, Brazil
Graduate Program in Computer Science (PGCOMP),
Federal University of Bahia (UFBA)
Salvador, Bahia, Brazil
profalbertovianna@gmail.com

Vaninha Vieira
Graduate Program in Computer Science (PGCOMP),
Department of Computer Science, Federal University of Bahia (UFBA)
Salvador, Bahia, Brazil
vaninha@ufba.br

ABSTRACT

Mobile device users receive a lot of notifications per day. Some important notifications may be lost because of notification overloading. For some user, losing a notification can generate serious problems. This work proposes an Application Programming Interface (API) that uses context-awareness and machine learning to predict user attention to support mobile application developers and increases the chances of a important notification to be attended.

KEYWORDS

Mobile notification, Mobile sensing, Context-awareness, User attention prediction, Attention awareness

1 INTRODUCTION

Mobile devices are now part of everyone's life. The enormous number of mobile subscribers can give an idea of that. From 1989 to 2015 this number increased from seven million to 7.1 billion of mobile subscribers [13]. In addition, most of these subscribers use a smart mobile device, such as: smartphone, tablet, smartwatch, etc.

The possibility of notifying a user and get his/her attention is an interesting feature that smart mobile devices have. Each application inside these devices can send a mobile notification to users and give an information to them.

A notification can be defined as a visual, audible, or tactile signal, generated by an application or service, that gives information to a user who is out of focus [6]. On mobile devices, notifications are usually sent instantly when there is some activity that may be relevant to the user when the application is not open, such as: a new email, a text message that has just arrived, or a new comment on their social networks. In some cases the user takes immediate action after the notification arrives, while in others it is simply ignored. These actions depend on the relevance of the notification and user context [10].

On mobile devices a notification is a message that can be displayed outside the application context. When the application issues a notification, it first appears as an icon in the notification area.

Mobile notifications have been used for different purposes. Some examples are: to tell the user that a new email has arrived, to alert the user a new instant message needs to be responded, or even to notify that the device is fully charged.

In a study published by Falaki *et al.* (2010) [4], they measure the average of applications installed in mobile devices from 255 users. This number varies from 10 to 90 applications and average was roughly 50. Hence, an average of 50 applications can notify the user. Yahoo Aviate (2014) [17] collected data from their application use and disclosed more recent information about Android users. They informed that users have an average of 95 applications installed and an average of 35 applications that are used through the day.

Iqbal and Horvitz (2010) [6] evaluated the use of notifications generated by an email tool in a desktop computer. The study mentioned that users found notifications very important for them, however inopportune and disruptive in some moments - imagine users receiving notification from 50 different applications.

Because of that, the number of notifications a user receives in mobile devices has increased a lot and some problems have emerged, such as: receiving notifications in inopportune moments; interrupting an important task at work; not realizing an urgent e-mail has arrived and not responding because of the big amount of notifications received before, etc. Thus, resulting in a scenario very different from Weiser and Brown (1997) [15] calm computing technologies.

One of the most significant scenarios is the notification considered important by the user. For example, a businessman who owns a company which sells products over the Internet, considers a notification that a product could not be delivered a very important one. Also, for an elderly, a notification warning it is time to take some medicine is very important. In current scenario, there is a big chance that a user does not see that notification because of the overload of notifications, and mobile application developers have limited resources to deal with this situation.

A usual way to handle with lots of notifications is to group them with some linkage process, as used by [7]. Our study uses a different approach. We used context-awareness and machine learning to predict user attention.

We believe that context information about the user could infer the user attention to incoming notifications. For instance, a physical activity can give suggestions if a user is paying attention to her/his phone.

Context is defined by Dey [3] as "any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves"

In: XVIII Workshop de Teses de Dissertações (WTD 2018), Salvador, Brasil. Anais do XXIV Simpósio Brasileiro de Sistemas Multimídia e Web: Workshops e Pôsteres. Porto Alegre: Sociedade Brasileira de Computação, 2018.

ISBN 978-85-7669-435-9.

Context information is important to define the user current state and the environment where it is inserted, but only this is insufficient. In order to use this information satisfactorily, it is ideal to build an adaptive system that meets the needs of the user in real time using context information. In short, a context-aware system.

Context-aware systems are able to adapt their operations to the current context, without explicit intervention of the user, aiming to increase their usability and effectiveness by taking into account context elements [1]. A system is context-aware if "it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task" [3].

A large-scale study was published by Shirazi *et al.* (2014) [10]. One of their proposed future work was the automatic assessment of notifications, to deliver notification when user appreciated and had the least negative impact. This can be achieved by using context information about user, network, and environment.

One motivation of this work has a real scenario. We developed a solution for monitoring beta-blocked cardiac patients [11], when we evaluated the application, some important notifications were lost by the user because of the overloading of notifications. The solution needed an intelligent mechanism to raise the chances for user seeing a notification and lead to the characterization of the research problem presented in next section.

1.1 Related Works and Research Problem

Recent studies in the field have already tried to improve notification process [2, 8, 9]. However, they used offline classifying algorithms from machine learning to try to improve notification process in predicting user attention.

We believe that offline classifying algorithms are not the best choice for this task, because notification context is very dynamic. For example, if user installs a new application, the learning model could give a wrong result, because it will know nothing about the new application and user habits. Thus, an online machine learning technique is more suitable for such task. Corroborating with this idea, Mehrotra *et al.* (2015) [8] attempted to use online learning. However, they still used offline learning, because the learning model would be re-trained once a day without further analysis.

Furthermore, these studies mentioned before conceived context models, analyzed them, but did not improve them in any way. Therefore, there is no context model as a pattern to be used in attention prediction to improve notification process.

In addition, none of the studies created any software component to facilitate the use of the proposed mechanisms. Developers, who want an intelligent mechanism of notification, could not use existing solutions. They need an Application Programming Interface (API), a service, a framework, or a library to incorporate those mechanisms in their applications.

The term API, in this work, is the same used in Stylos and Myers (2007) [12], can refer to either the abstract notion of the interfaces, the distribution suitable for use when programming (binaries with definition files), and the implementation source of the API. We choose to develop an API because we do not want to change the way of writing a program, as a framework do, and our proposal is collection of software components that differs our solution from a library.

The research gaps showed before lead to the research question presented in Table 1.

Table 1: Research Question.

How to improve the probability of a notification generated by a mobile application to be attended by the user?

Our study aims to present a solution that uses a mature context model and online machine learning to predict user attentiveness. We used the same concept for the word "attentiveness" as used by Pielot *et al.* (2014) [9] extended to notifications: "we define user attentiveness as the degree to which (s)he is paying attention to incoming instant messages".

1.2 Objectives

The research question that conducts this work is presented in Table 1. From the research question, the general objective emerged:

- Create an API to add intelligence in mobile notification process to predict user attentiveness, using context-awareness and online machine learning, to support mobile developers and increase the chances of a notification be attended.

To reach the general objective, the next steps were performed as specific objectives:

- Investigate the state-of-art of the area and conceive a discussion about topic;
- Create a context model with the contextual elements for the basis of the proposed intelligent mechanism;
- Investigate which machine learning algorithm is most suitable to the scenario;
- Validate whether the context model can be used to predict user attention using selected machine learning algorithms in an experimentation study;
- Design and develop a novel API to predict attention in mobile notifications using online machine learning;
- Perform an experiment to validate the developed solution.

2 METHODOLOGY

This section describes the methodology proposed for this work. A complete overview of it can be achieved in Figure 1.

The first task performed to reach the objectives of this research was to conduct a experimentation study to check the viability of the proposal. After the viability validation and promising results found, we went further on the definition of the methodology.

There are different artifacts proposed in this study. One of the main artifacts is a novel context model to be used as a reference in the area. To reach this goal a step called "Defined Context Model" is presented in Figure 1. The context model will be defined through literature review and related works analysis, during viability experimentation a first version of the model was defined (more information on Section 4). Other versions will be produced through the use of focus group, participants interviews, specialists surveys, and a systematic mapping study.

This model will be used to gather real data from different mobile devices users to construct a real data set. Machine learning algorithms will be applied to check which algorithm has the best

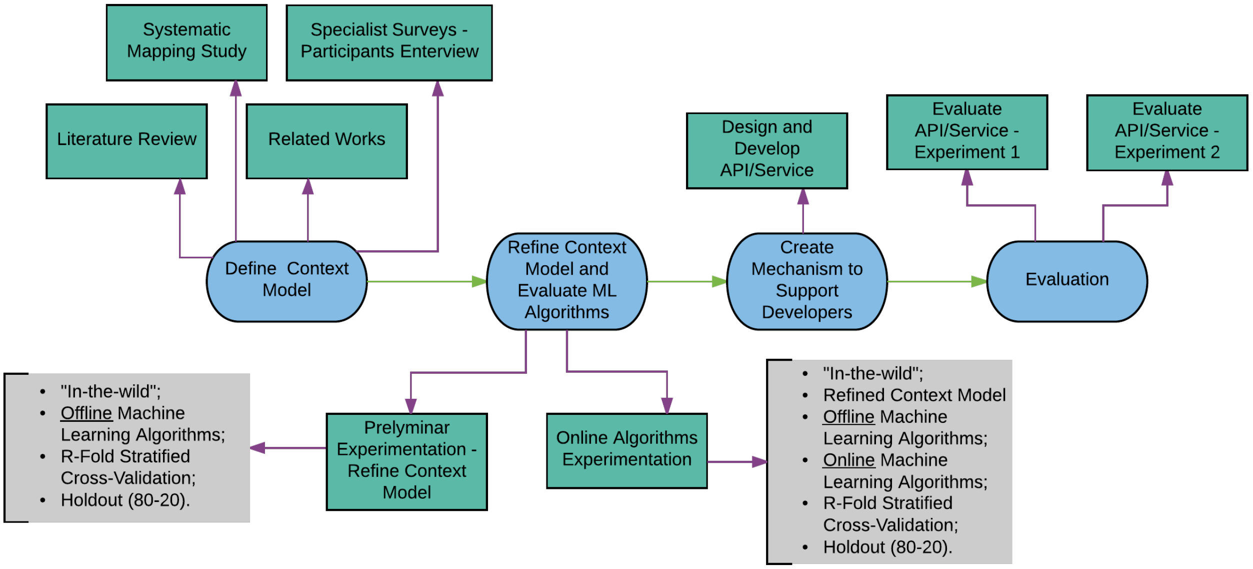
Systematic Mapping Study
Specialist Surveys - Participants Enterview
Literature Review
Related Works
Design and Develop API/Service
Evaluate API/Service - Experiment 1
Evaluate API/Service - Experiment 2
Define Context Model
Refine Context Model and Evaluate ML Algorithms
Create Mechanism to Support Developers
Evaluation
"In-the-wild";
Offline Machine Learning Algorithms;
R-Fold Stratified Cross-Validation;
Holdout (80-20).
Prelyminar Experimentation - Refine Context Model
Online Algorithms Experimentation
"In-the-wild";
Refined Context Model
Offline Machine Learning Algorithms;
Online Machine Learning Algorithms;
R-Fold Stratified Cross-Validation;
Holdout (80-20).

Figure 1: Methodology Overview.

results for this work scenario. This step is represented in Figure 1 as "Refine Context Model and Evaluate ML Algorithms". Two experimentations are expected, both will use as machine learning validation methods r-fold stratified cross-validation and holdout.

The step called "Create Mechanism to Support Developers" represents the development of the API/service using the context representation, the machine learning algorithm with the best result, and the architecture presented in Figure 2. **The main idea of the solution is to respond: will the user see the notification or not?**

The solution will be evaluated in two controlled experiments. At first, a preliminary experiment inside the academy. The second, a controlled experiment, with the evolutions and the resolution of unforeseen problems of the first experiment. In both experiments, two application that generates important notifications will be used as study cases and we will check whether the solution could predict user attentiveness.

3 CURRENT STATUS OF WORK

At the current moment, we are evolving the context model. We applied a focus group with 15 mobile device user to extract new context elements. Furthermore, we are conducting a systematic mapping to identify and characterize the body of knowledge of scientific literature reporting solutions that support the use of context-awareness in notification process in mobile and wearable devices.

The main research question of the study is **"How has context-awareness been used to support notification overloading in mobile and wearable devices?"** For now, we have already surpassed the selection/filtering phase and we are extracting the selected paper presented in Table 2.

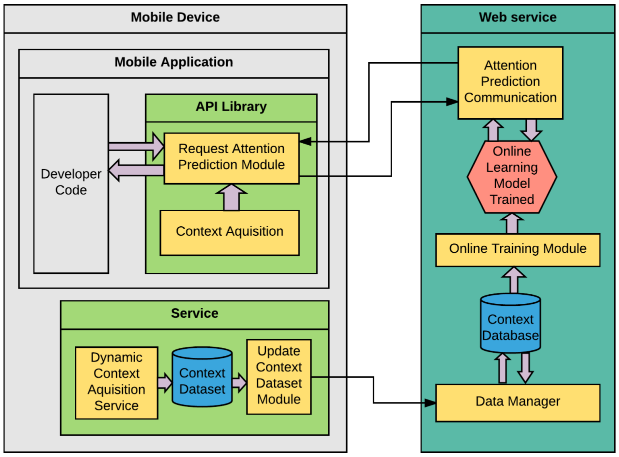
Mobile Device
Mobile Application
API Library
Developer Code
Request Attention Prediction Module
Context Aquisition
Service
Dynamic Context Aquisition Service
Context Dataset
Update Context Dataset Module
Web service
Attention Prediction Communication
Online Learning Model Trained
Online Training Module
Context Database
Data Manager

Figure 2: The proposed Architecture.

Table 2: Mapping study selection results.

Results	Amount of paper
Duplicated papers	445
Rejected papers	623
Accepted papers	38

4 PREVIOUS RESULTS

A preliminary study was performed to validate the viability of the project. A context model was constructed after collection of all different contextual elements from the studies that had a relevant impact on results. A few more contextual elements we considered important were also added and grouped in the first version of a new context model. The context model has contextual elements that can be related to prediction of user attention.

4.1 Context Model - First Version

In order to model a context-aware application, it is necessary to identify entities and contextual elements [14]. The context model used in this preliminary study is presented in Table 3. A contextual element can be static (does not change over time), dynamic (changes over time), explicit (informed by the user) and implicit (obtained without user interaction).

Table 3: First version of a High-level Context Model.

Entity	**Contextual Element**		**Data Source**
	Name	Data Type	
User	Location	Coordinate	GPS / WIFI
User	Physical Activity	Enum.	Google API
User	Last Time Near	Time	App, sensor and Clock
Smartphone	Last Time Unlocked	Time	App and Clock
Smartphone	Connection Type	Enum.	Smartphone Info
Smartphone	Ringer Mode	Enum.	Smartphone Info
Environment	Noise Level	Number	Microphone
Environment	Day of Week	Enum.	Calendar
Environment	Day-Shift	Enum.	Clock
Notification	Application name	Text	Smartphone Info
Notification	Title of notification	Text	Smartphone Info
Notification	Attended	Boolean	Smartphone Info

In context model presented in Table 3, four entities were identified: the user that received the notification; the smartphone that generated the notification; the environment the user was immerse; and data from the notification itself. All contextual elements were classified as dynamic and implicit.

The smartphone is the unique data source used in the project. In this way, Table 3 shows which element of smartphone was used to acquire context information.

After definition of context model, we performed a preliminary experimentation study and developed a smartphone application to gather information and contextual elements from different users. We named this application "DataCollector".

4.2 Preliminary Experimentation Study

DataCollector was developed in native Android SDK[1]. We implemented a NotificationListenerService[2] to monitor user notifications in an unobtrusively way. In favor of identifying the physical activity of the user, we used Activity Recognition API for Android from Google[3].

When a notification is presented, following information is collected: location latitude and longitude from global positioning system (GPS) sensor; surrounding sound amplitude using microphone; and last time user was close to the phone in seconds, we gather this information from proximity sensor.

Other information acquired are: last time user was interacting with her/his device, in seconds, determined by last screen lock moment; connection type (mobile, wifi); ringer mode (vibrating, ringing, etc.); day-shift and day of week were extracted from notification posted moment.

A Notification has several attributes, we used in this study the notification title. Notification message and notification ticker were not used due to privacy concerns.

To the best of our knowledge, there is no way to get directly the information that a user attend to a notification or not. To infer the user attended the notification, we implemented two features. First, we check for 30 seconds whether the application that generated the notification had opened after the notification was removed. If application opened, we consider the user clicked the notification. Second, clicking a notification is not the only way to respond a notification, user can unlock the smartphone device and attend or respond the notification without clicking the notification. Considering this scenario, we use the same strategy performed in literature [8, 9], if the user opens the application that generated the notification in 10 minutes, since after it arrival, notification is set to "attended".

4.2.1 Preliminary Predicting Attentiveness Model. We conducted a data collection and installed DataCollector application in 3 people different devices. They used the application for at least 15 days. All of them are Brazilians and volunteered for the study. DataCollector collects all kinds of notification: instant messages, e-mails, system notifications, etc. At the end, 33892 notification were collected. After pre-processing, 10584 notification were analyzed.

4.2.2 Evaluation. The evaluation was conducted using both methods R-Fold Stratified Cross-Validation and Holdout. For the first one, we use R=10 proportional folds to validate. The second one, we use the proportion of 80% as training data-set and 20% as test data-set. We perform the experiment using Weka machine learn software [5]. Tested machine learning algorithms were: (i) C4.5; (ii) Random Forest; (iii) AddaBoost; (iv) Naive Bayes; (v) KNN (K equal to 1, 2, and 3); (vi) SVM; and (vii) Voted Perceptron. Each algorithm and validation method combination was configured to test with the execution of 100 interactions. Table 4 shows results with correct classification rate average for each algorithm and validation method. Random forest had the best results in R-Fold Stratified Cross-Validation and Holdout.

[1] https://goo.gl/J5wb1X
[2] https://goo.gl/Oh6B5o
[3] https://goo.gl/214fMQ

Table 4: Algorithms average of correct classification rates.

Algorithm	R-Fold	Holdout
Random Forest	93.87%	93.83%
KNN (K=1)	93.84%	93.82%
KNN (K=2)	93.36%	93.24%
C4.5	93.01%	92.81%
KNN (K=3)	92.89%	92.68%
SVM	79.93%	80.02%
Voted Perceptron	77.94%	77.85%
Naive Bayes	74.43%	74.50%
AdaBoost	71.05%	71.27%

Some algorithms, such as KNN, SVM and Perceptrons, use attributes with numeric values in their logic. As Weka stores all attributes values as double floating point numbers internally [16], we decided to not transform all categorical attributes in numeric, allowing Weka transforms by its native implementation.

5 EXPECTED CONTRIBUTIONS

As expected contributions, we can mention:

- A novel and mature context model for notification attentiveness;
- An API that can be used to help mobile developers in attention prediction of the user;
- A smartphone application for collect raw data about notifications;
- A raw data-set of notifications that can be used in future studies.

ACKNOWLEDGMENTS

The authors would like to thank all the volunteers who participated in the experiment. The work is partially supported by the CAPES Agency under Grant No.:#1490744.

REFERENCES

[1] Matthias Baldauf, Schahram Dustdar, and Florian Rosenberg. 2007. A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing* 2, 4 (2007), 263.

[2] Fulvio Corno, Luigi De Russis, and Teodoro Montanaro. 2015. A context and user aware smart notification system. In *IEEE 2nd World Forum on Internet of Things (WF-IoT).* Milan, Italy, 645–651.

[3] Anind K. Dey. 2001. Understanding and using context. *Personal and ubiquitous computing* 5, 1 (2001), 4–7.

[4] Hossein Falaki, Ratul Mahajan, Srikanth Kandula, Dimitrios Lymberopoulos, Ramesh Govindan, and Deborah Estrin. 2010. Diversity in Smartphone Usage. In *MobiSys '10 Proceedings of the 8th international conference on Mobile systems, applications, and services.* San Francisco, 179–194.

[5] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. 2009. The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter* 11, 1 (2009), 10–18.

[6] Shamsi T Iqbal and Eric Horvitz. 2010. Notifications and Awareness : A Field Study of Alert Usage and Preferences. In *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work, CSCW.* New York, New York, USA, 27–30.

[7] Juan Antonio Lossio-Ventura, Hakim Hacid, Mathieu Roche, and Pascal Poncelet. 2016. Communication overload management through social interactions clustering. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing.* Pisa, Italy, 1166–1169.

[8] Abhinav Mehrotra, Mirco Musolesi, Robert Hendley, and Veljko Pejovic. 2015. Designing Content-driven Intelligent Notification Mechanisms for Mobile Applications. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing.* Osaka, Japan, 813–824.

[9] Martin Pielot, Rodrigo De Oliveira, Haewoon Kwak, and Nuria Oliver. 2014. Didn't You See My Message? Predicting Attentiveness to Mobile Instant Messages. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* Toronto, Ontario, Canada, 3319–3328.

[10] Alireza Sahami Shirazi, Niels Henze, Tilman Dingler, Martin Pielot, Dominik Weber, and Albrecht Schmidt. 2014. Large-Scale Assessment of Mobile Notifications. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* Toronto, Ontario, Canada, 3055–3064.

[11] Alberto Vianna Dias da Silva and Vaninha Vieira. 2016. BBAware - A Context-Aware Mobile and Wearable Architecture for Monitoring Beta-Blocked Cardiac Patients. In *Simpósio Brasileiro de Computação Ubíqua e Pervasiva.* Anais do XXXVI Congresso da Sociedade Brasileira de Computação, Porto Alegre.

[12] Jeffrey Stylos and Brad Myers. 2007. Mapping the space of API design decisions. In *Proceedings - IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC 2007.* Coeur d'Alene, Idaho, USA, 50–57.

[13] TeleGeography. 2015. Global Mobile Subscribers Surpass 7 Billion. https://www.telegeography.com/press/press-releases/2015/07/28/global-mobile-subscribers-surpass-7-billion/index.html. (2015). Online; accessed 08 February 2017.

[14] Vaninha Vieira, Patricia Tedesco, and Ana Carolina Salgado. 2011. Designing context-sensitive systems: An integrated approach. *Expert Systems with Applications* 38, 2 (2011), 1119–1138.

[15] Mark Weiser and John Seely Brown. 1997. The Coming Age of Calm Technology. In *Beyond Calculation.* Springer New York, New York, 75–85.

[16] Weka. Why am I missing certain nominal or string values from sparse instances? goo.gl/yJ9DmO. (????). Online; accessed 24 Semptember 2016.

[17] YahooAviate. 2014. How Android Users Interact with Their Phones. https://yahooaviate.tumblr.com/post/95795838933. (2014). Online; accessed 30 March 2017.