2016 Euromicro Conference on Digital System Design

Device Context Classification for Mobile Power Consumption Reduction

Ismat Chaib Draa*, Maroua Nouiri†, Smail Niar*, Bekrar Abdelghani*
*University of Valenciennes, Valenciennes, France
{firstname.lastname}@univ-valenciennes.com
†University of El Manar of Tunis, Sciences Faculty of Tunis, LIP2 Laboratory,2092 Tunis, Tunisia
{firstname.lastname}@gmail.com

***Abstract*—The diverse range of wireless interfaces, sensors, processing components added to the increasing popularity of power-hungry applications reduce the battery life of mobile devices. This paper proposes a tool for identifying the device context, understanding the user habits and preferences in order to adjust available resources and find trade-off between the power consumption and the user satisfaction. We use Machine Learning (ML) methods to identify and classify user/device contexts. On this basis, a software is developed to control at run-time system component activities. When applied only for the screen brightness level knob, the proposed solution can lower the power consumption by up to 20% vs. the out-of-the-box OS brightness manager with a negligible energy overhead.**

***Keywords*—Power consumption, classification, Machine Learning.**

I. Introduction

The capabilities and platform complexity of mobile devices are continuously improving, becoming truly system-on-chip (SoC) devices able to adapt their operation over three distinguishing dimensions, namely communication, sensing, and computation. In our modern societies, each person has several of these handled devices. Power consumption in mobile devices is crucial because applications running on current and future handled devices are very complex. The needs of these new applications in terms of computing, communication and storage have significantly exceeded the ability of the batteries. In addition, the state of the art in battery technology shows that the only alternative left at the moment is reducing the power consumption [11].
Our proposal is, using the computing power and the various sensors, to capture and store device context and user habits informations at run-time to reduce power consumption. The device context is defined by the device utilization manner, spatial and temporal data, environment informations and the user preferences in terms of applications and resources configuration. This context, user behaviors and habits informations are classified as *users patterns* and exploited to improve upon the default power management policies of the OS. The main contributions of this work can be summarized as follows:

1) We exploit rich sensor hubs to collect and explore a large set of data to search context usage patterns and to calibrate the user satisfaction.
2) We propose a new context classification method based on ML algorithms to find trade-off between power consumption reduction opportunities and user-experience satisfaction. On this basis, we can decrease or turn off some resources and HW components without impacting the user satisfaction.

The global architecture of our approach is given in [7]. In this paper, we focused on screen brightness because it is among the most power consuming components in a mobile system and the display remains the primary user interface on mobile devices. The overall user experience with these devices is highly determined by the display subsystem. However, our solution is generic and can be adapted for several contexts and components such as GPS or Wi-Fi. First, we collect data about user behavior, system's informations and device's environment like mentioned on our previous works [7]. The collected data are exploited thru three mechanisms:

1) **Offline detection and identification**: permits us to identify the most pertinent data that compose a specified context.
2) **Context Classification**: Selected data in the aforementioned mechanism are classified thru Machine Learning and data mining algorithms.
3) Depending on the second mechanism outputs, the **Optimizer Actuator** performs actions to reduce power consumption.

Sec. 2 presents the context identification. In Sec. 3, we present the classification phase with the machine learning algorithms. In Sec. 4, we present experimental results obtained with our approach and a comparison in terms of power consumption between our approach and the OS power management. Sec. 5 presents the related work and finally we conclude in Sec. 6.

II. Luminosity Context Identification

The aim of our work is to propose to every user a customized energy management depending on his/her preferences and behavior. This section presents the context recognition and the user satisfaction identification. The collected informations can be exploited in several ways. In [6], we focused on the context without taking into account the user needs, In [7], we proposed a technique to predict user action and reduce energy consumption. In this paper, we will exploit collected data to implement a user/device context classification based on user perception and satisfaction.
The existing methods of screen brightness management depend on the surrounding ambient light levels. Several analysis has shown the existing simplistic model currently used for display brightness control is not sufficient [8]. Indeed, the eyesight, the tolerance to light and the configuration preferences differ from

978-1-5090-2817-7/16 $31.00 © 2016 IEEE
DOI 10.1109/DSD.2016.102

user to another. This difference between the users requirements in terms of luminosity leads us to select more contextual data to decrease screen brightness and consequently power consumption. Data, like applications needs in terms of brightness and average brightness level preferences, are captured in order to determine a context. The context varies over time for the same user and varies also from a user to another. For each context, an adequate screen brightness level is determined. The proposed brightness level by our system must meet two requirements: must not impact negatively the user satisfaction and must reduce power consumption. Luminosity context is built over 3 selected parameters from the probes. They cover the context and user preferences:

- **Ambient luminosity (AL) [0-100]**: this parameter is recurrent in all the brightness management methods.
- **Per-User brightness preferences [0-100]**: the system probe proposes the preferred user screen brightness during all the day, then an average is calculated.
- **Application need in terms of brightness [0-100]**: this last parameter matches between the running applications categories and the selected screen brightness during their run.

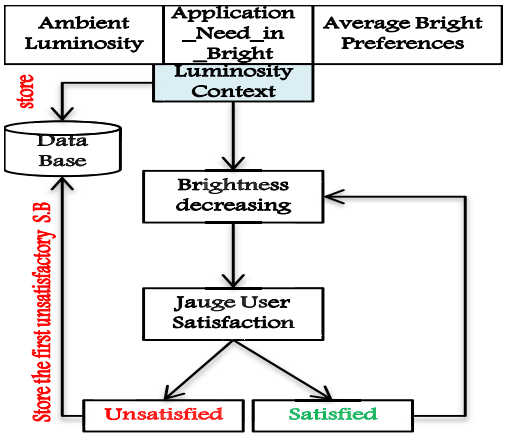

Figure 1: Luminosity Context identification

These parameters values are captured at the same time and stored into a database. The time-out between two collections is adjustable and varies depending on the application. For each context sample, we have an appropriate brightness level. This brightness level is determined trough the change blindness mechanism [10]. In this paper we exploit the change blindness concept by decreasing screen brightness level gradually until the user is bothered by the change. When the brightness is decreased automatically and still not modified manually by the user, we conclude that the user is satisfied and we store the appropriate brightness value corresponding to the luminosity context. When our automatic adjustment is not tolerated by the user, we readjust the decreasing at the last value which satisfied the user.

III. Luminosity Context Classification

In this section, we detail the classification mechanism. Unlike our previous works based on Bayesian approach, in this paper we are extending use of Artificial Neural Network (ANN) as a classifier method because of our type of data and the uncertain aspect of user behavior [5]. In our case, we classify each context in terms of screen brightness requirements.

A. Artificial neural networks (ANNs)

Artificial neural networks (ANNs) are networks of simple processing elements operating on their local data and communicating with other elements [2]. In this research, two neural network were constructed: Cascade back-propagation neural network and Feed-forward neural networks [5]. The network is trained on a set of paired data to determine input-output mapping. This phase is called Training, the paired data are composed by an input and for each input a desired output called target. The combination of weights which minimizes the error function is considered to be a solution of the luminosity context learning problem. After the training phase, the weights of the connections between neuron are then fixed. We test our neural network with new inputs (Luminosity Context) without target (desired outputs), this phase is called simulation. The simulation classifies the new inputs and permit us to have information about our network behavior.

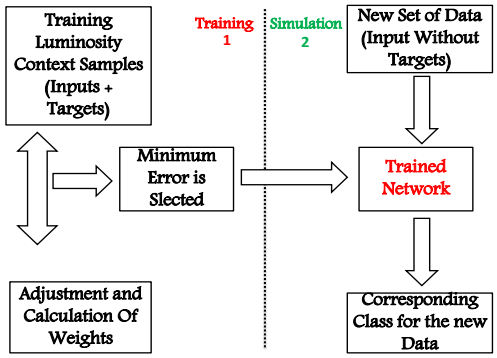

Figure 2: Training Pattern of ANN for Simulation

B. ANN For Screen Brightness Classification

Our network was trained with 100 epochs. Generally determining the size of the hidden layer is a problem. The number of neurons in each hidden layer varied from 1 to 10. Weights and biases were randomly initialized . As the number of hidden layers, the neurons in each hidden layer are function of expected intelligence, there is no generic method to determine the optimum values. The parameters which cover the context and the user preferences are Ambient Luminosity (ALS), Average Brightness and Application Need in terms of Brightness. These parameters represent the input of our Neural Network. The appropriate screen brightness level which was detected previously in the Suitable Brightness Detection (Figure 1) represents the target parameter. For the output, we have 3 classes:

- Class Low: This class corresponds to a low need in terms of screen brightness [0%-33%].
- Class Medium: This class is for a medium need in terms of screen brightness [33%-66%].

- Class High: This class represents the highest brightness values[66%- 100%].

Each context sample (3 inputs) is assigned to a screen brightness level range which corresponds to a class. The dataset in the experimental tests (120 samples) from the probes, were divided: 80% for training, and 20 % for testing. Cascade forward BP and feed forward BP were trained with up to 100 epochs. Weights and biases were randomly initialized.

C. Neural Network Accuracy

In this subsection, we present the training results of our ANNs to validate it and to choose the most accurate architecture. Figure 3 presents the Mean Square Error (MSE) and the iteration at which the validation performance reached a minimum (Epoch). It shows also the regression results. The MSE confirms us to choose the Feed Forward back-

ANNS	MSE	EPOCHS	TRAINING	VALIDATION	TEST
Cascade	0.073154	3	Out = 0.7 * Tar + 0.11	Out= 0.89* Tar +0.0058	Out= 0.59* Tar + 0.15
Feed Forward	4.5976 e – 08	19	Out= 0.89* Tar+ 0.0058	Out = 0.89* Tar+ 0.096	Out= 0.85* Tar + 0.047

Figure 3: Neural Networks Accuracy

Propagation for its accuracy and minimal MSE in comparison with the Cascade Forward Back-Propagation. After choosing **Feed Forward Back-Propagation** for its accuracy. The final weights of each input are collected to classify any other new luminosity sample. The next step is to adapt the screen brightness level depending on the result of the classification.

IV. Experimental Results

In this section we present the gain obtained in terms of power consumption and the solution's cost. The experimentations have been realized on an Ultrabook running Windows 8.1. Power measurement has been obtained using the Yokogawa WT210 power analyzer.

To demonstrate our solution efficiency, we made tests in different user scenarios with different luminosity context, current screen brightness and suitable brightness calculated by our classification. Recall that the classification output is a range of values. The optimizer actuator decreases screen brightness from the higher boundary to the smallest value that satisfies the user. The optimizer actuator decreases the screen brightness gradually by $delta$ units every $lambda$ seconds. To not impact the user satisfaction and converge as quickly as possible to the optimale value, $delta$ and $lambda$ have been fixed experimentally to 3 units and 4 seconds respectively. Figures 4 and 5 compare the OS brightness management with our solution in terms of % of brightness and power consumption.

Our 6 users are Yoda, Han, Luke, Blinks, Finn and Vador. Firstly, we give the allowed screen brightness by the OS which is based only on the ambient light. Secondly, the most appropriate screen brightness obtained by our approach is given.

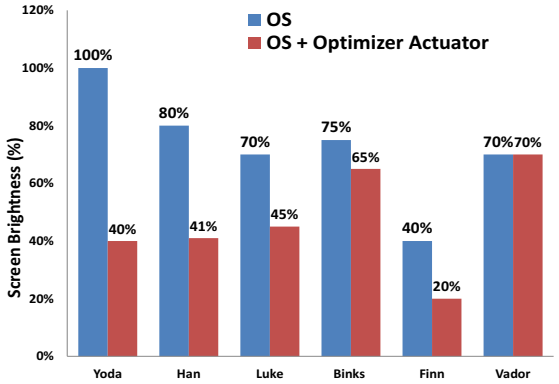

Figure 4: OS Vs (OS + Optimizer) Screen Brightness Adjustment

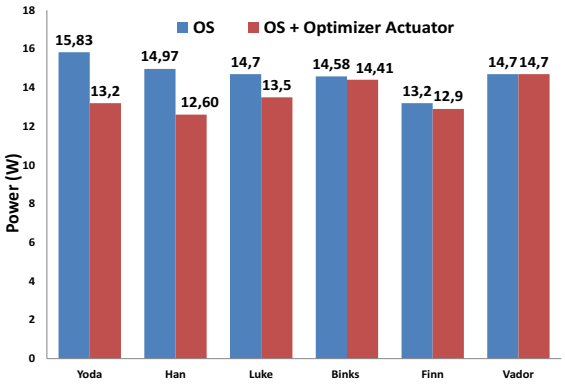

Figure 5: OS Vs (OS + Optimizer) Power Consumption

In figure 4, we compare the screen brightness calculated by the OS and the screen brightness level calculated by our solution. Recall that in our Ultrabook, the minimum allowable brightness level is 20%, unlike other mobile systems, the screen is not completely off even if 0% is given. Yoda, Han and Luke are the pseudonyms of the real users, Binks, Finn and Vador are the simulated ones. For each user, the screen brightness level has been decreased respectively by **60%, 39%, 25%, 10%, 20% and 0%**. This difference is due to the classification results which show the variations in terms of screen brightness from user to another as mentioned in Sections 2 and 3.

The power consumption reduction which results from our screen brightness management is respectively: **16.62%, 15.78 %, 8.16%, 1.16%, 18.5% and 0%** as depicted in figure 5. The power reduction opportunities with the real users are more important than the obtained power gain with the simulated users. The real users show us that with our solution it is possible to save more power than the OS and to propose configurations in terms of HW/SW which are less costly than the OS power management (Depending on each user). The simulated users highlight that at worst, our solution save as much power as the OS. We note that the power reduction is linearly correlated to the screen brightness level decreasing.

The training phase lasts 8 seconds for all the users because of the same dataset size (120 samples). The CPU utilization is at 12% before the training, it increases during the training to reach a peak estimated at 57%. After 8 seconds of training, the CPU utilization is stabilized to its initial value. The memory reaches a peak of 522 MB during the training and is stabilized at 462 MB after the training. Finally, we have a power consumption overhead of 12.3 W which is also stabilized after the 8 seconds of training. This cost represents the power consumption overhead for the whole training phase.

V. Related Works

In this section, we summarize some of the existing works in energy consumption optimization for mobile systems at the application level. We mainly focus on the approaches that take into account user behavior and experience in energy consumption reduction. We also present some works which focus only on the display power reduction. Finally, we highlight the main differences with our approach. Authors in [3] and [8] proposed the utilization of user activities and context information-based technics. Authors proposed also several management policies for each Hardware component. In their approach the CPU frequency, for example, was adjusted dynamically depending on the workload. They also proposed to reduce background process life time depending on the obtained patterns.
In [4], the authors presented the power monitor approach. Their architecture corresponds to a client-server architecture developed to collect usage logs from Android powered devices. Based on the utilization patterns, power saving profiles are generated and are personalized to match the needs of each device in the system. The experimental results show that the power monitor can increase the battery life by almost 20%. However this solution has some privacy issues due to the exploitation of usage pattern generation. In [1], authors improved AMOLED power analysis by taking into account dynamic factors those affect power consumption. These factors are linked to the user when he/she is using device's built-in camera for playing a video game and recording video.
At the opposite of all the previous works and papers, our project differs in the following points:

- Classifying a device context which takes into account the user, the system and the environment is very helpful and allows an efficient screen brightness.
- In our approach, we take into account user satisfaction to propose a customized solutions. This approach has been rarely used in current commercial solutions.
- Although studies in [9] and [1] are concerned with improvingthe power consumption for a given readability level, Our work focused on improving OS policies for screen brightness management.

VI. Conclusion and Perspective

In this paper, a new technique for energy consumption reduction in mobile systems has been proposed. Our approach is based on machine learning techniques. At the opposite of existing methods, where the users needs and behaviors are rarely taken into account, in our approach, we not only take these elements but, we also take into account informations from context and environment. In comparison to the advanced energy management provided by windows 8.1, for some situations the gain offered by our approach reaches 20% depending on the user's needs and habits.

This work will be extend to consider more possible user patterns and more applications. We will also improve the user satisfaction level to control our power saving technique. Finally, the offline classification phase will also be extended to include sound level, GPS, CPU and WiFi needs.

References

[1] X. Chen, Y. Chen, Z. Ma, and F. C. Fernandes. How is energy consumed in smartphone display applications? In *Proceedings of the 14th Workshop on Mobile Computing Systems and Applications*, page 3. ACM, 2013.

[2] j. P. Daniel Svozil, Vladimir Kvasnicka. introduction to multi layer feed-forward neural networks. *Chemometrics and Intelligent Laboratory Systems*, 1997.

[3] S. Datta, C. Bonnet, and N. Nikaein. Power monitor v2: Novel power saving android application. In *Consumer Electronics (ISCE), 2013 IEEE 17th International Symposium on*, pages 253–254, June 2013.

[4] S. Datta, C. Bonnet, and N. Nikaein. Personalized power saving profiles generation analyzing smart device usage patterns. In *Wireless and Mobile Networking Conference (WMNC), 2014 7th IFIP*, pages 1–8, May 2014.

[5] V. K. P. Dheeraj S. Badde, Anil k. Gupta. Cascade and feed forward back propagation artificial neural network models for prediction of compressive strength of ready mix concrete. *Journal of Mechanical and Civil Engineering*, 1997.

[6] I. C. Draa, J. Tayeb, S. Niar, and M. Desertot. User information analysis for energy consumption optimization in mobile systems. In *Proceedings of the Workshop on High Performance Energy Efficient Embedded Systems (HIP3ES) colocated with HIPEAC*, Amsterdam, Netherlands, 2015. ACM.

[7] I. C. Draa, J. Tayeb, S. Niar, and E. Grislin. Application sequence prediction for energy consumption reduction in mobile systems. In *Computer and Information Technology; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on*, pages 23–30. IEEE, 2015.

[8] M. Schuchhardt, S. Jha, R. Ayoub, M. Kishinevsky, and G. Memik. Caped: Context-aware personalized display brightness for mobile devices. In *Proceedings of the 2014 International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, CASES '14, pages 19:1–19:10, New York, NY, USA, 2014. ACM.

[9] D. Shin, Y. Kim, N. Chang, and M. Pedram. Dynamic voltage scaling of oled displays. In *Design Automation Conference (DAC), 2011 48th ACM/EDAC/IEEE*, pages 53–58. IEEE, 2011.

[10] D. J. Simons and D. T. Levin. Change blindness. *Trends in cognitive sciences*, 1(7):261–267, 1997.

[11] N. Vallina-Rodriguez and J. Crowcroft. Energy management techniques in modern mobile handsets. *Communications Surveys & Tutorials, IEEE*, 15(1):179–198, 2013.