Constructing an Accurate and a High-Performance Power Profiler for Embedded Systems and Smartphones

Oussama Djedidi, Mohand Djeziri, Nacer M'Sirdi, Aziz Naamane

Constructing an Accurate and a High-Performance Power Profiler for Embedded Systems and Smartphones

Oussama Djedidi, Mohand Djeziri, Nacer M'Sirdi, Aziz Naamane

Constructing an Accurate and a High-Performance Power Profiler for Embedded Systems and Smartphones[*]

Oussama Djedidi[†]
Mohand A. Djeziri
Nacer K. M'Sirdi
Aziz Naamane
oussama.djedidi@lis-lab.fr
mohand.djeziri@lis-lab.fr
nacer.msirdi@lis-lab.fr
aziz.naamane@lis-lab.fr
Aix Marseille University, Université de Toulon, CNRS, LIS, SASV, Marseille, France

ABSTRACT

The main objective of this paper is to present a new accurate power profiler for embedded systems and smartphones. The second objective is, for it, to be a tutorial explaining the main steps to build power profilers for embedded and mobile systems, in general.

We start our work by firstly describing the general methodology of building a power profiler. Then, we showcase how each step is undertaken to build a profiler with two power models. The first one was an artificial neural network (called N^2) that presented a lot of noise in its estimation. After debugging and improvement, the second model, a NARX neural network (we call N^3) was built. It eliminated all the drawback of the first model and had a mean absolute percentage error of 2.8%.

CCS CONCEPTS

• **General and reference** → **Measurement**; **Estimation**; • **Computer systems organization** → **System on a chip**; **Embedded systems**; • **Hardware** → **Power estimation and optimization**; **Safety critical systems**;

KEYWORDS

Data fitting, Embedded Systems, Modeling, NARX neural Networks, Power Consumption, Smartphone

1 INTRODUCTION

Smartphones come—naturally—with limited power supply, incinting the interest in power consumption, and battery life. Following this trend and interestn, here, we continue our previous works [5, 6] and create a model-based power profiler for smartphones and embedded system. Model-based Power profilers are profiler that generates power estimations via an incorporated model [7]. These tools allow the user to estimate the power consumption of the whole device [12], specific components [11], or the applications [10].

Researchers have been building power models and profilers for nearly two decades. The most prominent of which have been indexed in surveys of the specialized literature [1, 7]. More recently, several other models also found their place in the literature [2, 3, 12]. Some of these models offered easier construction and implementation [8], while others focused on the accuracy [13].

In the next section, we outline the main steps in building a power profiler. Then, in the following sections, we use these steps to design and build neural network-based power profilers. In the last section, the obtained results are discussed validating the profiler.

2 DESIGN AND CONSTRUCTION OF POWER PROFILERS

Power profilers are pieces of software, hardware, or a combination of both that have the main function of measuring or estimating and logging power consumption. They differ in the philosophies behind their design, as they also differ in their construction methods. Nevertheless, the process of building one can be summarized in several major steps. In figure 1, the general organigram of these steps are laid out. The figure also shows, in the case of unsatisfactory results, which steps should be revisited according to the constraints underhand. For instance, if the estimations of the profiler are not accurate enough, one should reiterate through the steps between the choice of modeling scheme and model training. For in general, low accuracy is indicative of an inadequate model for the process, inputs not correlating or not related to the output, or insufficient data for the training (in the case of black-box modeling).

In the following paragraphs, these steps are showcased in action through the building of an actual profiler.
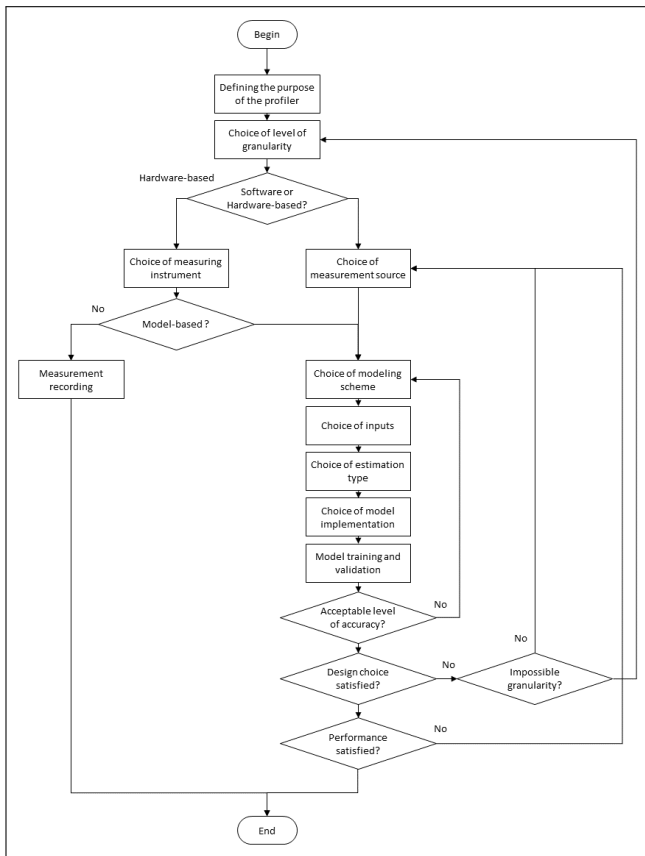
Figure 1: A general algorithm for constructing a power profiler for mobile devices.

2.1 Knowing the system : Presenting the studied device

Before we start building the profiler, we will first introduce the device on which all our tests were done. We chose to use an Android™ smartphone, for three main reasons. Firstly, these devices are available on a wide scale and easily programmable, making all the programs we develop easily portable between devices. Secondly, these devices run on the operating system (OS) used by most of the embedded systems in the world—Linux—extending the portability of our profiler to, virtually, all devices running it. Finally, these devices are well instrumented and need no external or invasive measurement schemes.

The phone is running Android™ 8.0 (Oreo), and has an Exynos 8895 SoC with an octa-core processor using ARM's big.Little architecture, a Mali-G71 MP20 GPU, and 6 GB of RAM. Communication-wise, it supports LTE bands, Wi-Fi 802.11, and Bluetooth 5.0. It is also equipped with a 6.2 inches QHD+ AMOLED screen, and two cameras—one front facing and a second rear facing.

2.2 Constructing the N^2 power profiler

In beginning of this section, we have outlined the steps to building a power profiler. Hereafter, they are briefly demonstrated.

2.3 Purpose of the profiler

The power profiler we are building in this work is meant to serve as a power monitor for the mobile device and its components. It should collect data, generate power estimation with its incorporated power model for the device as a whole, and its components individually, and log the said data. The granularity is to be set at components level. The monitored components are the major culprits when it comes to power consumption. While, the device contains other peripherals, their power draw is considered marginal and a part of the static power consumed by the device, such as the capacitive touch layer on the screen [9]. The following is a list of the monitored components:

- **The system on chip (SoC)**: CPU, GPU, and RAM.
- **The camera modules**
- **The LTE module**
- **The AMOLED touchscreen**
- **The GPS module**
- **Speakers**
- **The wireless (Wi-Fi and Bluetooth) module**

2.4 The profiling scheme and measurements source

The profiler built here is software-based and will rely on system files and traces for its power readings and inputs. This profiling scheme and measurement source are adequate for the chosen level of granularity. Moreover, the results obtained with these methods are on par with accuracy of hardware-based measurements [4].

2.5 The modelling scheme

SoCs are very complex and contain a large number of sub-modules. White-box techniques would require the modeling of all those components, making the process of constructing one arduous [1]. Moreover, the operating system (OS) generates traces and logs to show its current state and the status of the peripherals. We opted to use these system files.

The availability of these data points, alongside the complexity of the system, led us towards the use of black-box modeling techniques like artificial neural networks (ANN). ANN have the advantage of being able to be nonlinear which makes them more suitable to estimate nonlinear power changes in the devices [7].

In our case, we use data fitting ANN that we call N^2. N^2 is composed of two layers. Firstly, a hidden layer for that has a neuron for every input variable; the neuron has a sigmoid transfer function. Then, the output layer comes with only one neuron that has a linear transfer function. Since the output layer is a linear neuron, the estimate of total power consumed by the device is the sum of the power consumed by each individual component. Thus, N^2 gives estimations for both the system as a whole and the components individually.

2.6 The choice on inputs

Since black-box modeling is used, the inputs used are those which correlate most with power consumption and indicate the state of the peripheral.
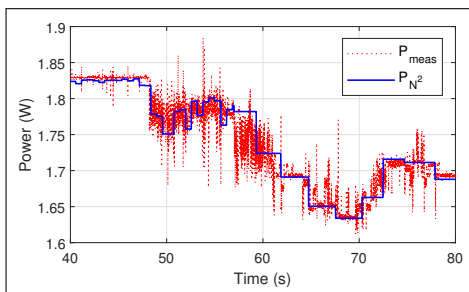
Figure 2: Evolution of the relative percentage estimation error for N^2.

2.6.1 The SoC. CPU power consumption is a function of its frequency and voltage [5]. The input used to estimate power consumption is the frequency f of each CPU core (or block of cores in the big.Little configuration). The same input—frequency—is used for the GPU. The frequency of the CPU also sets the sampling time of the profiler, since it is reevaluated every 20 ms [5]. Finally, for the RAM, we use for the occupied RAM devided by its maximum value; we call it Memory Occupation Rate (MOR).

2.6.2 The communication peripherals. They include the LTE module, Wi-Fi, Bluetooth and the GPS modules. For each of these chips, we use status (On/Off). For the Wireless chip and the LTE module, we also add the signal strength, and transfer rate (Download and upload). The LTE Chip has an additional input; the On-call status (Off, 2G, 3G).

2.6.3 The input and output peripherals. To account for the power drawn by the touchscreen, we used the status of the screen (On/Off) and its brightness as inputs, since for modern AMOLED screens, consumed power is a a quadratic function of the brightness [9]. As for the speakers, the status (On/Off) and the volume are used as inputs.

Finally, for the cameras and the LED flash light, we the status of the camera and the flash (On/Off) alongside its recording status (Camera recording or not).

2.6.4 Model construction and Implementation. ANN require relatively large amounts of data and resources to be trained into accuracy. Mobile CPUs are not yet capable of doing such training. The evident solution to these limitations is to do the training of a separate device. Once the training process finished, N^2 will be installed on a companion device that will generate online estimations.

3 PRELIMINARY RESULTS

After training the model, it was tested on the phone for validation. Figure 2 shows the evolution of the estimations results alongside the power measurements. While the estimations from N^2 follow the general shape of the measurements, they also present a lot of noise and deviate from them. During some instances, The relative percentage estimation error reached around 8%. The Mean Absolute Percentage Error (MAPE) is about 6%, and the Mean Squared Error (MSE) is 2.3896×10^{-4}.

These results are encouraging. Nevertheless, the noise present in the estimation and the MAPE indicate that there is still a room for improvement, and led us to the second version of the profiler.

4 ENHANCING THE POWER PROFILER, VERSION 2.0: N^3

The previous version of the model had unsatisfactory accuracy while satisfying the rest of the design choices. According to the algorithm in figure 1, we have to retrace our steps back to the choice of measurement scheme.

4.1 Updating the design choices

- **The measurement scheme:** It presented no issues, neither for the performance no for the accuracy. Moreover, the main point of using this measurement scheme is to avoid invasive procedures. Thus, it will not be changed.
- **Modelling :** Investigating the noise, we hypothesized that it might be due to either using unnecessary extra inputs or that the device's power control loop is causing changes that the model cannot predict. Experimental data, however, indicate the strong correlation between the chosen inputs and the output, which eliminate the extra inputs possibility. Moreover, the implementation of the model and its training presented caused no deterioration to the accuracy. Hence, by process of elimination, we need to update the modeling scheme.

4.2 NARX neural net

Nonlinear Autoregressive Network with eXogenous inputs (NARX) neural networks are a form of recursive dynamic neural networks. They are an established methods built to predict the output of time series [14]. Equation 1 gives their mathematical representation.

$$y(k + 1) = f \left[y(k), ..., y(k - d_y + 1) \right.$$
$$\left. u(k - n), u(k - n - 1), ..., u(k - n - d_u + 1) \right] \quad (1)$$

$y(k)$ and $u(k)$ are the output to be predicted and the input a the time step k, respectively. The term n is the process dead-time. While d_y and d_u are the orders of time delays for the input and output, also called input and output-memory.

Our model—N^3—is composed of two layers, like N^2. The only differences between the two in the design is the recursion of the output and the time delay line. Since the power changes reacts pretty quickly with changes in the characteristics (inputs), d_y, d_u, and n were all set to the value of 1.

5 EXPERIMENTAL RESULTS AND DISCUSSION

In figure 3, the measured power consumption of the smartphone is drawn against values estimated by N^3. The model accurately predict the power consumed by the phone with with minimal estimation errors. It has an estimation MAPE of only 2.82%, and the recorded Mean Absolute Error (MAE) is 0.0168 W, whereas the Mean Squared Error (MSE) is 7.04×10^{-4}.

This version of the model and profiler is accurate and does not present any of the accuracy issues shown by the previous model.
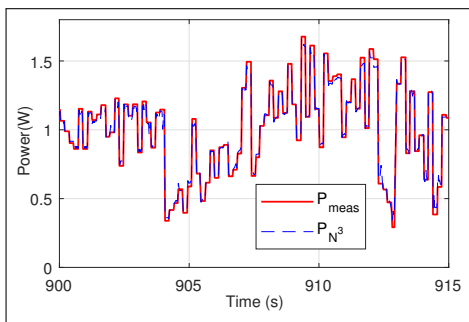
Figure 3: N^3 Power estimation against the measurements for the system.

N^3 is validated. However, to reach such accuracy, we had to collect more than 10^6 data points. The MAPE decreased from 5% to 2.88% by adding more samples, which came at the cost of longer training times, but beyond 1.2×10^6 samples, it no longer improved.

Performance-wise, N^3 caused no issues for the system. In fact, it had only a 3% load overhead in a 15 minutes test. As for the power overhead, it only amounted to a 5% increase in the same test.

Finally, we compared our results with the publicly available power models; *Trepn*, *Snapdragon Profiler*, *PowerBooter*[15]. Table 1 shows the accuracy of all these models as reported in the literature, compared to the power model we presented in this paper.

Table 1: Comparison of the accuracy of several power models and profilers.

Model	Accuracy (%)	Source
N^3	**2.88**	**Measured**
Trepn	5.5	Measured
Snapdragon Profiler	4.8	Measured
PowerBooter	4	[7]

While we are pleased that our model performed on par with established works, it is also worth noting that *PowerBooter* has a higher levels of granularity than our model does, was updated for several years. Furthermore, *Trepn* was trained using its own brands SoCs, and thus would probably achieve its better reported accuracy with them [1].

6 CONCLUSION

In this work, we have presented a comprehensive algorithm to build a power profiler for embedded systems and smartphones. Then, we proceeded to build and present a power profiler for mobile devices using that algorithm. The new model (N^3) is very simple to construct and train. It causes little overhead and performs at the speeds necessary to follow the power changes in embedded systems. Finally, it improves upon most of the existing models in terms of accuracy with a mean absolute percentage error of 2.8%, one of the lowest in the reported literature.

This model will be incorporated in the in the incremental modeling scheme previously built by Djedidi et al. [5] to improve the estimation results, and study the health status and operating state of the device.

REFERENCES

[1] Raja Wasim Ahmad, Abdullah Gani, Siti Hafizah Ab Hamid, Feng Xia, and Muhammad Shiraz. 2015. A Review on mobile application energy profiling: Taxonomy, state-of-the-art, and open research issues. *J. Netw. Comput. Appl.* 58 (dec 2015), 42–59. https://doi.org/10.1016/j.jnca.2015.09.002

[2] Shaiful Alam Chowdhury and Abram Hindle. 2016. GreenOracle: Estimating Software Energy Consumption with Energy Measurement Corpora. In *Proc. 13th Int. Work. Min. Softw. Repos. - MSR '16*. ACM Press, New York, New York, USA, 49–60. https://doi.org/10.1145/2901739.2901763

[3] Dario Di Nucci, Fabio Palomba, Antonio Prota, Annibale Panichella, Andy Zaidman, and Andrea De Lucia. 2017. PETrA: A Software-Based Tool for Estimating the Energy Profile of Android Applications. In *2017 IEEE/ACM 39th Int. Conf. Softw. Eng. Companion*. IEEE, 3–6. https://doi.org/10.1109/ICSE-C.2017.18

[4] Dario Di Nucci, Fabio Palomba, Antonio Prota, Annibale Panichella, Andy Zaidman, and Andrea De Lucia. 2017. Software-based energy profiling of Android apps: Simple, efficient and reliable?. In *SANER 2017 - 24th IEEE Int. Conf. Softw. Anal. Evol. Reengineering*, Vol. 15. IEEE, 103–114. https://doi.org/10.1109/SANER.2017.7884613

[5] Oussama Djedidi, Mohand Arab Djeziri, Nacer K. M'Sirdi, and Aziz Naamane. 2017. Modular Modelling of an Embedded Mobile CPU-GPU Chip for Feature Estimation. In *Proc. 14th Int. Conf. Informatics Control. Autom. Robot.*, Vol. 1. SCITEPRESS - Science and Technology Publications, Madrid, Spain, 338–345. https://doi.org/10.5220/0006470803380345

[6] Oussama Djedidi, Mohand Arab Djeziri, Nacer K M'Sirdi, and Aziz Naamane. 2018. A Novel Easy-to-Construct Power Model for Embedded and Mobile System. In *Proc. 15th Int. Conf. Informatics Control. Autom. Robot. (ICINCO 2018)*, Vol. 1. INSTICC, SciTePress, Porto, Portugal, 541–545. https://doi.org/10.5220/0006915805410545

[7] Mohammad Ashraful Hoque, Matti Siekkinen, Kashif Nizam Khan, Yu Xiao, and Sasu Tarkoma. 2015. Modeling, Profiling, and Debugging the Energy Consumption of Mobile Devices. *ACM Comput. Surv.* 48, 3 (2015), 1–40. https://doi.org/10.1145/2840723

[8] Kitae Kim, Donghwa Shin, Qing Xie, Yanzhi Wang, Massoud Pedram, and Naehyuck Chang. 2014. FEPMA: Fine-Grained Event-Driven Power Meter for Android Smartphones Based on Device Driver Layer Event Monitoring. In *Des. Autom. Test Eur. Conf. Exhib.* IEEE Conference Publications, New Jersey, 1–6. https://doi.org/10.7873/DATE.2014.380

[9] Ying Dar Lin, Ekarat Rattagan, Yuan Cheng Lai, Li Pin Chang, Yun Chien Yo, Cheng Yuan Ho, and Shun Lee Chang. 2014. Calibrating parameters and formulas for process-level energy consumption profiling in smartphones. *J. Netw. Comput. Appl.* 44 (sep 2014), 106–119. https://doi.org/10.1016/j.jnca.2014.04.014

[10] Radhika Mittal, Aman Kansal, and Ranveer Chandra. 2012. Empowering developers to estimate app energy consumption. In *Proc. 18th Annu. Int. Conf. Mob. Comput. Netw. - Mobicom '12*. ACM Press, New York, New York, USA, 317. https://doi.org/10.1145/2348543.2348583

[11] Ekarat Rattagan, Edward T.H. Chu, Ying Dar Lin, and Yuan Cheng Lai. 2015. Semi-online power estimation for smartphone hardware components. In *2015 10th IEEE Int. Symp. Ind. Embed. Syst. SIES 2015 - Proc.* IEEE, 174–177. https://doi.org/10.1109/SIES.2015.7185058

[12] Narendra Kumar Shukla, Rosarium Pila, and Saurabh Rawat. 2016. Utilization-based power consumption profiling in smartphones. In *Proc. 2016 2nd Int. Conf. Contemp. Comput. Informatics, IC3I 2016*. IEEE, 881–886. https://doi.org/10.1109/IC3I.2016.7919046

[13] Matthew J. Walker, Stephan Diestelhorst, Andreas Hansson, Anup K. Das, Sheng Yang, Bashir M. Al-Hashimi, and Geoff V. Merrett. 2017. Accurate and Stable Run-Time Power Modeling for Mobile and Embedded CPUs. *IEEE Trans. Comput. Des. Integr. Circuits Syst.* 36, 1 (jan 2017), 106–119. https://doi.org/10.1109/TCAD.2016.2562920

[14] Hang Xie, Hao Tang, and Yu He Liao. 2009. Time series prediction based on narx neural networks: An advanced approach. In *Proc. 2009 Int. Conf. Mach. Learn. Cybern.*, Vol. 3. IEEE, 1275–1279. https://doi.org/10.1109/ICMLC.2009.5212326

[15] Lide Zhang, Birjodh Tiwana, Zhiyun Qian, Zhaoguang Wang, Robert P. Dick, Zhuoqing Morley Mao, and Lei Yang. 2010. Accurate online power estimation and automatic battery behavior based power model generation for smartphones. In *Proc. eighth IEEE/ACM/IFIP Int. Conf. Hardware/software codesign Syst. Synth. - CODES/ISSS '10*. ACM Press, New York, New York, USA, 105. https://doi.org/10.1145/1878961.1878982