GPU Power Prediction via Ensemble Machine Learning for DVFS Space Exploration

Bishwajit Dutta
Dept. of ECE, Virginia Tech
bdutta@vt.edu

Vignesh Adhinarayanan
Dept. of CS, Virginia Tech
avignesh@vt.edu

Wu-chun Feng
Dept. of CS and ECE, Virginia Tech
wfeng@vt.edu

ABSTRACT

A software-based approach to achieve high performance within a power budget often involves dynamic voltage and frequency scaling (DVFS). Thus, accurately predicting the power consumption of an application at different DVFS levels (or more generally, different processor configurations) is paramount for the energy-efficient functioning of a high-performance computing (HPC) system. The increasing prevalence of graphics processing units (GPUs) in HPC systems presents new challenges in power management, and machine learning presents an unique way to improve the software-based power management of these systems. As such, we explore the problem of GPU power prediction at different DVFS states via machine learning. Specifically, we propose a new ensemble technique that incorporates three machine-learning techniques — sequential minimal optimization regression, simple linear regression, and decision tree — to reduce the mean absolute error (MAE) to 3.5%.

1 INTRODUCTION

Power and energy efficiency have emerged as first-order design constraints in high-performance computing (HPC) systems. For the DOE, an exascale supercomputer needs to operate under 20 MW [4]. The increasing prevalence of *dark silicon* [10] and emerging *hardware-overprovisioned* supercomputers have made it harder to safely operate these systems under their respective power budgets. This has necessitated the introduction of *power-management* systems such as Intel's Running Average Power Limit (RAPL) [7] and research prototypes [6, 19] that are capable of enforcing strict *power caps*. Central to such a power-management system is the ability to predict the power consumption of an application at different processor configurations (e.g., DVFS states) so as to configure the system for best performance while ensuring that *power caps* are not violated.

While many models for DVFS-based power prediction have been proposed [9], two emerging trends motivate the need for our work. First, graphics processing units (GPUs) are increasingly common in HPC. The latest ranking of the Top500 supercomputers has 101 GPU-accelerated systems [1]. Second, recent advances in machine learning (ML) have necessitated a re-examination of data-driven modeling in many areas of computing, including system design. As such, we investigate the applicability of several machine-learning (ML) techniques in predicting the power consumed by a GPU at different voltage-frequency settings (or P-states). Our contributions in this paper include the following:

- *Accurate power prediction of a GPU at different DVFS states.* We explore eight (8) prediction techniques to predict the GPU's power consumption at different DVFS states.
- *Statistically rigorous comparison of different machine-learning techniques.* Unlike previous studies that *only* compare the *mean error* of a few modeling techniques, we use *Tukey's HSD* test to compare multiple approaches in a statistically rigorous manner.
- *Design of an ensemble approach for GPU power prediction.* We are among the first to propose and evaluate different ensemble designs of machine-learning (ML) techniques for DVFS prediction.

2 MOTIVATION

Our paper seeks to predict the power consumption of an application at different system configurations (i.e., DVFS states). As such, we motivate the importance of doing so through use cases, and we articulate the challenges to be tackled.

Power Capping. Modern computing systems can draw more power than they can safely sustain. To illustrate the need for power capping, we show the power profile of an application on a processor running at three different frequencies (*f_1*, *f_2*, *f_3*) in Fig. 1. With an enforced power cap of 120 W, a conservative approach would set the machine at *f_2* to guarantee that the power cap is *never* violated. However, if a power predictor can accurately predict the power at different frequencies, the power management system can operate at *f_3* for phase I, *f_2* for phase II, and back to *f_3* for phase III.

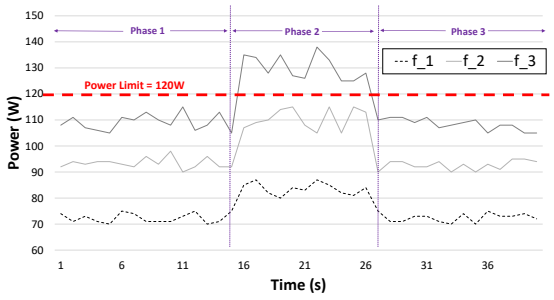

Figure 1: Power consumption at different frequencies

Energy-Performance Trade-off Analysis. The energy cost of operating today's HPC systems is about 25% of the acquisition cost [11]. Reliable prediction of power and performance can enable judicious trade-offs between energy and performance to lower the total cost of operation (TCO). An alternative approach is to exhaustively explore the configuration space for the ideal frequency for an

CF '18, Ischia, Italy

978-1-4503-5761-6/18/05…$15.00
DOI: 10.1145/3203217.3203273

application. However, with numerous configuration options available in today's HPC system, the energy cost of finding the correct configuration itself could exceed the energy cost of running the application. Thus, automatically predicting power consumption at different configuration points via modeling becomes indispensable.

Design Space Exploration. The ability to accurately estimate power and performance at different hardware configurations can accelerate design-space exploration, e.g., estimating power consumption when the L2 cache size is doubled (and hence, its utilization level is halved for an application) while keeping all other architectural aspects the same. Low-level power estimation techniques are generally too slow for fast design-space pruning. Thus, research into off-line high-level prediction techniques can lead to cheaper and faster design-space exploration.

Challenges. Previous work, which modeled GPU power consumption at different DVFS states via ML, operated *only* at the level of a GPU kernel [18]. While our ML approach differs along three different fronts: (1) modeling parameters, (2) breadth of techniques evaluated, and (3) target metrics modeled, the most fundamental difference is the level of problem formulation. While previous work tries to predict the power consumption of individual GPU kernels only, we tackle the more challenging problem of predicting the power consumption of an application as a whole. In [18], the individual kernels can exhibit irregular scaling properties with respect to frequencies. This scaling behavior gets more complex when an application has several kernels, each of which is scheduled in parallel and has different computational and memory bounds. Thus, this paper aims to capture the complex interaction of frequencies and resource utilization levels with respect to performance and power.

3 METHODOLOGY

Fig. 2 outlines our approach, which consists of a training phase and a testing phase, as described below, followed by details on our data collection, modeling techniques, and evaluation methods.

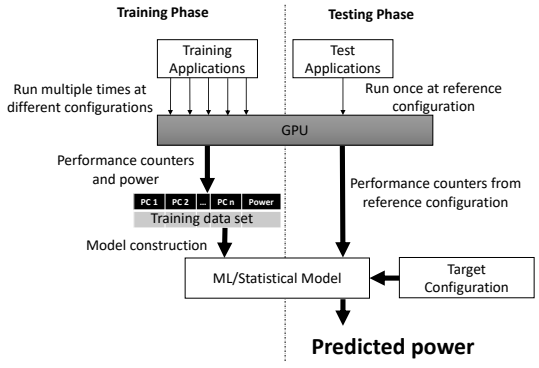

Figure 2: Overview of our approach

Training Phase. We run our training applications, shown in Table 1, on an NVIDIA Quadro P4000 GPU and collect the average power consumption and several performance counters at different configurations. For the Quadro P4000, there exist 288 possible combinations of GPU core frequency and memory frequency. Thus, in the training phase, each application must be run a few hundred times to collect the necessary data for modeling; this data collection is only a one-time cost. This data is then fed as input to the machine-learning model, which is initially tuned with the *training* data itself.

Table 1: Training Applications

Source	Applications
SHOC	Device Memory, Sort, Scan, SpMV, Reduction
Rodinia	Huffman, HotSpot
CUDA SDK	Matrix Multiplication, Transpose, Radix Sort, Interval, Line of Sight, Merge Sort

Testing Phase. After the model is constructed, it is tested using a different set of applications, as noted in Table 2. We collect performance counters for the test applications on a reference configuration and predict the power consumption at any target configuration using our machine-learning model.

Table 2: Test Applications

Source	Applications
SHOC	FFT, MaxFlops, BFS, Triad
Rodinia	Stream Cluster, Gaussian
CUDA SDK	Quad Tree, Scalar Product, Image Segment, Eigen Values

3.1 Data Collection

During the training phase, we profile the target GPU's power consumption every 100 ms using its built-in power meter via `nvprof`. During both the training and testing phases, we profile the utilization level of the following units using `nvprof`: (1) DRAM, (2) instruction issue slot, (3) L2 cache, (4) texture cache, (5) texture unit, (6) special functional unit (SFU), (7) load/store unit, (8) control unit, (9) single-precision (SP) unit, and (10) double-precision (DP) unit. These performance counters represent every major component within the GPU except the register file, which does not have an explicit performance counter, associated with it, but its utilization should track the utilization of other units.

In our experimental setup, while power is collected at the application level, the performance counters (i.e., utilization levels) are profiled at the kernel level by `nvprof`. Thus, we aggregate the kernel-level utilization metrics into an application-level metric:

$$\textit{Application level utilization of a resource} = \frac{\sum_{i=1}^{n}(\textit{Resource utilization level of } kernel_i * \textit{Time spent in } kernel_i)}{\sum_{i=1}^{n} \textit{Time spent in } kernel_i}$$

3.2 Modeling Techniques

Initially, we study eight different machine-learning techniques — `ZeroR`, simple linear regression (`SLR`), k-nearest neighbors (`KNN`), `bagging`, `random forest`, sequential minimal optimization regression (`SMOreg`), `decision tree`, and `neural networks` — to predict GPU power consumption at different frequencies. These techniques cover a breadth of approaches in predicting continuous variables and forms the fundamental building blocks in other prediction problems. We then use these blocks to construct ensemble models, as

described in §5. While we used R and weka software for evaluating the accuracy of the various models, we use terminology from weka for the sake of consistency. While the specific parameter values used for the various approaches are elided due to space constraints,[1] we empirically determined values so as to minimize error within the *training* dataset and *not* the *test* dataset.

4 SUMMARY OF RESULTS

Fig. 3 shows the mean absolute error (MAE) percentage and the error bars. SMOreg shows the best accuracy with a MAE at 4.5%, followed by REPTree and KNN at 5.5%, SLR at 6.0%, and RandomForest and Bagging at about 7.5%. NeuralNet performs the worst with a MAE of 15% due largely to the need for a huge training data set for it to be an effective method. Relative to maximum error, SLR performs the best with 15% MAE, followed by Bagging at 17% and SMOreg at 20%. KNN delivers the worst results as it only considers nearest points for prediction, which can be inaccurate due to the complex relationships between the predictors and response variables.

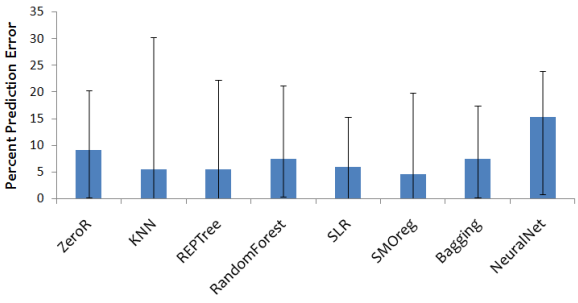

Figure 3: Percent prediction errors for different algorithms

Statistical Significance of Results. In order to statistically compare the prediction accuracy of the different ML algorithms, we use Tukey's HSD (honest significant difference) test. This test gives us the pairwise statistical difference between the mean error values predicted by different algorithms.

Fig. 4 presents the results of Tukey's HSD test. The bars show the 95% confidence interval between the lowest and highest estimate of the difference between percentage MAE of algorithm pairs. For example, for the comparison between ZeroR and SMOreg, we can say with 95% confidence that the MAE difference between ZeroR and SMOreg is 2% at the least and 7.5% at the most. When the lines in this graph cross the *zero* axis, it means that the difference in the prediction error is not significant enough to say that one method is better than the other (i.e., the result is not statistically significant). From Fig. 4, we also observe that SMOreg, SLR, KNN, and REPTree statistically exhibit better accuracy than the baseline ZeroR and that NeuralNet performs worse than every other ML technique.

5 AN ENSEMBLE METHOD FOR GPU POWER PREDICTION

Because solutions in other areas, e.g., traffic forecasting [17], deliver better accuracy and more stable results by creating an ensemble

[1]The parameter values can be found in the longer technical report [8].

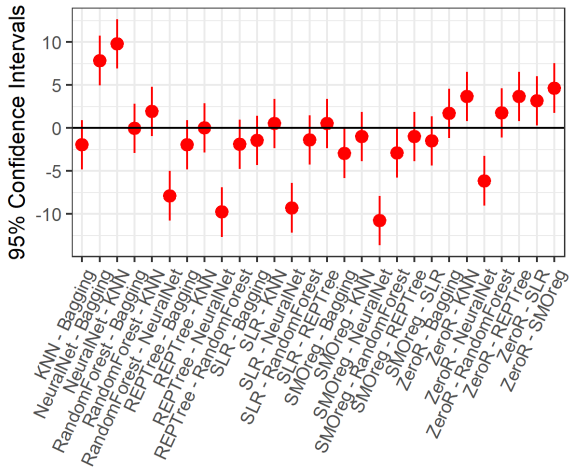

Figure 4: Tukey HSD confidence intervals for MAE difference between algorithm pairs

methodology, we propose and study such an approach to address the problem of GPU power prediction.

5.1 Methodology

To create our ensemble model, we calculate the weighted average of the "best" base methods. In spite of the simplicity of this approach, it is more accurate than complex voting mechanisms [17]. There exist many approaches to choose the base methods to construct the ensemble model (e.g., lowest mean error, lowest maximum error, and so on). In our approach, we rank the base methods in increasing order of MAE and progressively add more base methods to the ensemble. We also investigate unweighted and weighted averaging of the base methods. For the weighted averages, the weights are based on the reciprocal values of the MAE of the base methods. That is, *Ensemble's Prediction* =

$$\frac{\sum_{i=1}^{n}(Algorithm_i\ Prediction * \frac{1}{Algorithm_i Percentage MAE})}{\sum_{i=1}^{n}\frac{1}{Algorithm_i Percentage MAE}}$$

The six ensembles we created and studied are as follows:

- **top2:** Linear average of predictions made by the *two* most accurate individual methods – SMOreg and SLR.
- **top2 (weighted):** Weighted average of the *two* most accurate methods (SMOreg and SLR), where the weights are inversely proportional to their respective MAE.
- **top3:** Linear average of predictions made by the *three* most accurate individual methods – SMOreg, SLR, and REPTree.
- **top3 (weighted):** Weighted average of top *three* methods
- **all:** Linear average of all *seven* methods.
- **all (weighted):** Weighted average of all *seven* methods.

5.2 Experimental Results

Fig. 5 shows the prediction accuracy of the various ensembles we created and compares them against the most accurate base method (i.e., SMOReg with an average MAE of 4.5% and maximum of 20%).

We observe that for all the ensemble methods, the MAEs are much lower than with SMOReg and that the MAE decreases as the number of base methods increases but only up to a certain point. We obtain the most accurate results with three base methods, i.e., top3 with an MAE of 3.5% and maximum error of 10.5%. On the right-hand side of Fig. 5, we see the impact of weighting the base methods based on their individual accuracy. top3 (weighted) shows an MAE of 3.4% and maximum error of 11%. Weighting the base methods does not affect the results initially when the number of methods is small; the methods all show similar MAEs (and hence, similar weights). However, as we include more base methods in the ensemble and as the accuracy of those methods diverge, calculating a weighted average of the base methods improved the ensemble's accuracy.

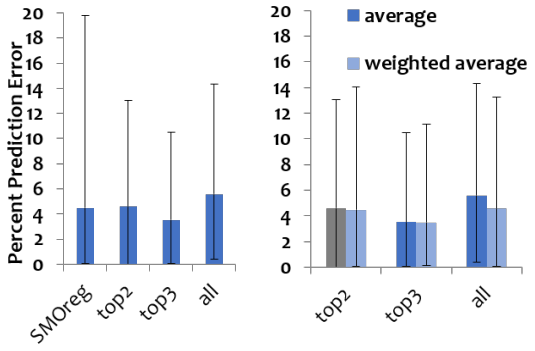

Figure 5: Percent prediction error for various ensembles. Note: top2 → SMOreg+SLR, top3 → SMOreg+SLR+REPTree

6 RELATED WORK

Many statistical and ML techniques have been explored in the past for power prediction (e.g., linear regression [5], decision trees [12], clustering techniques [5], support vector machines [13], evolutionary techniques [14], and neural networks [18]), but they have largely applied to CPUs only (see [9]). Most work on GPU power modeling [3, 15, 16] focuses on estimating the power consumption of a GPU in its reference configuration. Our work predicts the power consumption across *many* GPU machine configurations – a significantly harder problem.

Abe et al. [2] worked on a similar problem where they used a linear regression model to predict the power consumption of a GPU for different configurations. However, their approach only yielded an average error of over 20% compared to 3.5% for our ensemble method. Wu et al. [18] used the K-means algorithm and neural network to predict power consumption of a GPU while achieving an accuracy comparable to ours. Our work differs from theirs in the following ways: (1) Their model requires knowledge of power consumption at a reference point even for the test application, which we do *not* need. (2) We predict the power consumption of applications rather than GPU kernels, which have simpler scaling curves. (3) Our model requires only a *fraction* of the data points for modeling compared to theirs. In addition, we present a statistically rigorous study of different ML techniques, which is broader than any other previous study to date (including those done on CPUs).

7 CONCLUSION

We presented a rigorous comparison of different machine-learning (ML) techniques to predict the power consumption of an application at different GPU configurations. Our evaluation showed that SMOreg delivers the best results with a mean error of 4.5%, and RandomForest delivers the most consistent results at different frequencies. To further improve the accuracy of prediction, we designed several ensembles approaches from the base ML techniques. We found that the most accurate ensemble is a combination of the SMOreg, SLR, and REPTree methods, which reduced the mean absolute error (MAE) prediction from 4.5% to 3.5% and maximum error from 15% to 11%. In the future, we plan to use our model to improve the performance of a power-capped heterogeneous system.

Acknowledgements: This work was supported in part by DOE Office of Advanced Scientific Computing Research (ASCR) grant DE-SC0012637.

REFERENCES

[1] TOP500 Supercomputer Site, 2017. http://www.top500.org.
[2] Abe, Y., Sasaki, H., Kato, S., Inoue, K., Edahiro, M., and Peres, M. Power and Performance Characterization and Modeling of GPU-Accelerated Systems. In *IEEE Int'l Parallel and Distributed Processing Symposium (IPDPS)* (May 2014).
[3] Adhinarayanan, V., Subramaniam, B., and Feng, W. Online Power Estimation of Graphics Processing Units. In *IEEE/ACM Int'l Symposium on Cluster, Cloud and Grid Computing (CCGrid)* (May 2016).
[4] Ahern, S., et al. Scientific Discovery at the Exascale: Report from the DOE ASCR Workshop on Exascale Data Management, Analysis, and visualization. *DOE Office of Advanced Scientific Computing Research* (2011).
[5] Bailey, P. E., Lowenthal, D. K., Ravi, V., Rountree, B., Schulz, M., and de Supinski, B. R. Adaptive Configuration Selection for Power-Constrained Heterogeneous Systems. In *Int'l Conference on Parallel Processing (ICPP)* (2014).
[6] Cochran, R., Hankendi, C., Coskun, A. K., and Reda, S. Pack & Cap: Adaptive DVFS and Thread Packing Under Power Caps. In *IEEE/ACM Int'l Symp. on Microarchitecture (MICRO)* (2011).
[7] David, H., Gorbatov, E., Hanebutte, U. R., Khanna, R., and Le, C. Rapl: memory power estimation and capping. In *ACM/IEEE International Symposium on Low-Power Electronics and Design (ISLPED)* (2010).
[8] Dutta, B., Adhinarayanan, V., and Feng, W. GPU Power Prediction via Ensemble Machine Learning for DVFS Space Exploration. Tech. rep., Department of Computer Science, Virginia Polytechnic Institute & State University, 2018. https://vtechworks.lib.vt.edu/handle/10919/81997.
[9] Eeckhout, L. Computer architecture performance evaluation methods. *Synthesis Lectures on Computer Architecture 5*, 1 (2010).
[10] Esmaeilzadeh, H., Blem, E., St Amant, R., Sankaralingam, K., and Burger, D. Dark silicon and the end of multicore scaling. In *ACM SIGARCH Computer Architecture News* (2011), vol. 39, pp. 365–376.
[11] Gholkar, N., Mueller, F., and Rountree, B. A Power-aware Cost Model for HPC Procurement. In *IEEE IPDPS Workshops* (2016).
[12] Jia, W., Shaw, K. A., and Martonosi, M. Starchart: Hardware and Software Optimization Using Recursive Partitioning Regression Trees. In *IEEE Int'l Conference on Parallel Architectures and Compilation Techniques (PACT)* (2013).
[13] Ma, X., Dong, M., Zhong, L., and Deng, Z. Statistical Power Consumption Analysis and Modeling for GPU-based Computing. In *Workshop on Power Aware Computing and Systems (HotPower)* (2009).
[14] Maghazeh, A., Bordoloi, U. D., Eles, P., and Peng, Z. General Purpose Computing on Low-Power Embedded GPUs: Has it come of age? In *Int'l Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation* (2013).
[15] Nagasaka, H., Maruyama, N., Nukada, A., Endo, T., and Matsuoka, S. Statistical Power Modeling of GPU Kernels Using Performance Counters. In *Int'l Green Computing Conference (IGCC)* (2010).
[16] Song, S., Su, C., Rountree, B., and Cameron, K. W. A Simplified and Accurate Model of Power-Performance Efficiency on Emergent GPU Architectures. In *IEEE Int'l Symposium on Parallel and Distributed Processing (IPDPS)* (2013), IEEE.
[17] Sun, S. Traffic Flow Forecasting Based on Multitask Ensemble Learning. In *ACM/SIGEVO Summit on Genetic and Evolutionary Computation* (2009).
[18] Wu, G., Greathouse, J. L., Lyashevsky, A., Jayasena, N., and Chiou, D. GPGPU performance and power estimation using machine learning. In *IEEE Int'l Symposium on High Performance Computer Architecture (HPCA)* (2015).
[19] Zhang, H., and Hoffmann, H. Maximizing Performance Under a Power Cap: A Comparison of Hardware, Software, and Hybrid techniques. *ACM SIGPLAN Notices 51*, 4 (2016), 545–559.